Cumulus Linux 4.2 User Guide

NVIDIA® Cumulus Linux is the first full-featured Linux operating system for the networking industry. The Debian Buster-based, networking-focused distribution runs on hardware produced by a broad partner ecosystem, ensuring unmatched customer choice regarding silicon, optics, cables, and systems.

This user guide provides in-depth documentation on the Cumulus Linux

installation process, system configuration and management, network solutions, and monitoring and troubleshooting recommendations. In addition, the quick start guide provides an end-to-end setup process to get you started.

Cumulus Linux 4.2 includes the NetQ agent and CLI, which is installed by default on the Cumulus Linux switch. Use NetQ to monitor and manage your data center network infrastructure and operational health. Refer to the NetQ documentation for details.

For a list of the new features in this release, see What's New. For bug fixes and known issues present in this release, refer to the Cumulus Linux 4.2 Release Notes.

Open Source Contributions

To implement various Cumulus Linux features, Cumulus Networks has forked various software projects, like CFEngine `Netdev` and some Puppet Labs packages. Some of the forked code resides in the Cumulus Networks GitHub repository and some is available as part of the Cumulus Linux repository as Debian source packages.

Cumulus Networks has also developed and released new applications as open source. The list of open source projects is on the open source software page.

Hardware Compatibility List

You can find the most up-to-date hardware compatibility list (HCL) here. Use the HCL to confirm that your switch model is supported by Cumulus Linux. The HCL is updated regularly, listing products by port configuration, manufacturer and SKU part number.

Stay up to Date

- Subscribe to our product bulletin mailing list to receive important announcements and updates about issues that arise in our products.
- Subscribe to our security announcement mailing list to receive alerts whenever we update our software for security issues.

Download the User Guide

You can download a PDF version of the complete Cumulus Linux 4.2 user guide here.

What's New

This document supports the Cumulus Linux 4.2 release, and lists new platforms and features.

- For a list of all the platforms supported in Cumulus Linux 4.2, see the Hardware Compatibility List (HCL).
- For a list of open and fixed issues in Cumulus Linux 4.2, see the Cumulus Linux 4.2 Release Notes.
- To upgrade to Cumulus Linux 4.2, follow the steps in Upgrading Cumulus Linux.

What's New in Cumulus Linux 4.2.1

Cumulus Linux 4.2.1 supports a new platform, provides bug fixes, and contains certain enhancements.

New Platforms

- Mellanox SN4700 (Spectrum-3)

Enhancements

- The Mellanox SN3700 Spectrum-2 switch now supports 200G (100G was supported previously)
- EVPN multihoming is now generally available on Mellanox switches
- Inter-VRF route leaking is now ASIC accelerated by default

What's New in Cumulus Linux 4.2.0

Cumulus Linux 4.2.0 supports new platforms, provides bug fixes, and contains several new features and improvements.

New Platforms

- Mellanox SN4600C (100G Spectrum-3)
- Mellanox SN3420 (25G Spectrum-2)

New Features and Enhancements

- EVPN multihoming, supported on Mellanox switches, is a standards-based replacement for MLAG in data centers deploying Clos topologies - this feature is available for Early Access
- Auto BGP, which automatically assigns ASNs to switches in a two-tier leaf and spine environment
- Mandatory cumulus user default password change upon first login
- New ONIE command line options to set the *cumulus* user default password, add a license, and provide initial network configuration
- Ability to edit the Cumulus Linux image file
- Ability to set the CPU as a SPAN destination interface
- ECMP and LAG custom hash parameters have been moved to the `/etc/cumulus/datapath/traffic.conf` file and no longer require a `switchd` restart
- DSCP-based packet matching in PBR rules
- Link pause and priority flow control are now supported on the Edgecore Minipack-AS8000

Unsupported Platforms

These platforms are not supported in Cumulus Linux 4.2. They are supported in Cumulus Linux 3.7, until that release reaches its end of life.

- Cumulus Express CX-10256-S/Edgecore OMP-800 (100G Tomahawk)
- Dell S6000-ON (40G Trident2)
- EdgeCore Wedge-100 (100G Tomahawk)
- Facebook Backpack (100G Tomahawk)
- Facebook Voyager (100G Tomahawk)
- Delta AG7648
- QCT QuantaMesh BMS T3048-LY8
- QCT QuantaMesh BMS T3048-LY9

Quick Start Guide

This quick start guide provides an end-to-end setup process for installing and running Cumulus Linux, as well as a collection of example commands for getting started after installation is complete.

Prerequisites

Intermediate-level Linux knowledge is assumed for this guide. You need to be familiar with basic text editing, Unix file permissions, and process monitoring. A variety of text editors are pre-installed, including `vi` and `nano`.

You must have access to a Linux or UNIX shell. If you are running Windows, use a Linux environment like Cygwin as your command line tool for interacting with Cumulus Linux.

If you are a networking engineer but are unfamiliar with Linux concepts, refer to this reference guide to compare the Cumulus Linux CLI and configuration options, and their equivalent Cisco Nexus 3000 NX-OS commands and settings. You can also watch a series of short videos introducing you to Linux and Cumulus Linux-specific concepts.

Install Cumulus Linux

To install Cumulus Linux, you use ONIE (Open Network Install Environment), an extension to the traditional U-Boot software that allows for automatic discovery of a network installer image. This facilitates the

ecosystem model of procuring switches with an operating system choice, such as Cumulus Linux. The easiest way to install Cumulus Linux with ONIE is with local HTTP discovery:

1. If your host (laptop or server) is IPv6-enabled, make sure it is running a web server. If the host is IPv4-enabled, make sure it is running DHCP in addition to a web server.

2. Download the Cumulus Linux installation file to the root directory of the web server. Rename this file `onie-installer`.

3. Connect your host using an Ethernet cable to the management Ethernet port of the switch.

4. Power on the switch. The switch downloads the ONIE image installer and boots. You can watch the progress of the install in your terminal. After the installation completes, the Cumulus Linux login prompt appears in the terminal window.

> **NOTE**
>
> These steps describe a flexible unattended installation method. You do not need a console cable. A fresh install with ONIE using a local web server typically completes in less than ten minutes.
> You have more options for installing Cumulus Linux with ONIE. Read Installing a New Cumulus Linux Image to install Cumulus Linux using ONIE in the following ways:

- DHCP/web server with and without DHCP options
- Web server without DHCP
- FTP or TFTP without a web server
- Local file
- USB

After installing Cumulus Linux, you are ready to:

- Log in to Cumulus Linux on the switch.
- Install the Cumulus Linux license.
- Configure Cumulus Linux. This quick start guide provides instructions on configuring switch ports and a loopback interface.

Get Started

When starting Cumulus Linux for the first time, the management port makes a DHCPv4 request. To determine the IP address of the switch, you can cross reference the MAC address of the switch with your DHCP server. The MAC address is typically located on the side of the switch or on the box in which the unit ships.

Login Credentials

The default installation includes the system account (root), with full system

privileges and the user account (cumulus), with `sudo` privileges. The *root* account password is locked by default (which prohibits login). The *cumulus* account is configured with this default password:

```
cumulus
```

When you log into Cumulus Linux for the first time with the cumulus account, you are prompted to change the default password. After you provide a new password, the SSH session disconnects and you have to reconnect with the new password.

> **NOTE**
>
> ONIE includes options that allow you to change the default password for the *cumulus* account automatically when you install a new Cumulus Linux image. Refer to ONIE Installation Options. You can also change the default password using a ZTP script.

In this quick start guide, you use the *cumulus* account to configure Cumulus Linux.

All accounts except `root` are permitted remote SSH login; you can use `sudo` to grant a non-root account root-level access. Commands that change the system configuration require this elevated level of access.

For more information about `sudo`, read Using sudo to Delegate Privileges.

Serial Console Management

You are encouraged to perform management and configuration over the network, either in band or out of band. A serial console is fully supported; however, you might prefer the convenience of network-based management.

Typically, switches ship from the manufacturer with a mating DB9 serial cable. Switches with ONIE are always set to a 115200 baud rate.

Wired Ethernet Management

Switches supported in Cumulus Linux always contain at least one dedicated Ethernet management port, which is named eth0. This interface is geared specifically for out-of-band management use. The management interface uses DHCPv4 for addressing by default. You can set a static IP address with the Network Command Line Utility (NCLU) or by editing the `/etc/network/interfaces` file (Linux).

NCLU Commands | Linux Commands

Set the static IP address with the `interface address` and `interface gateway` NCLU commands:

```
cumulus@switch:~$ net add interface eth0 ip address
192.0.2.42/24
cumulus@switch:~$ net add interface eth0 ip gateway
192.0.2.1
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Configure the Hostname and Timezone

Configure the hostname and timezone for your switch. The hostname identifies the switch; make sure you configure the hostname to be unique and descriptive.

NOTE

- Do not use an underscore (_) in the hostname; underscores are not permitted.

- Avoid using apostrophes or non-ASCII characters in the hostname. Cumulus Linux does not parse these characters.

To change the hostname:

NCLU Commands | Linux Commands

Run the `net add hostname` command, which modifies both the `/etc/hostname` and `/etc/hosts` files with the desired hostname.

```
cumulus@switch:~$ net add hostname <hostname>
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

NOTE

- The command prompt in the terminal does not reflect the new hostname until you either log out of the switch or start

a new shell.

- When you use the NCLU command to set the hostname, DHCP **does not** override the hostname when you reboot the switch. However, if you disable the hostname setting with NCLU, DHCP **does** override the hostname the next time you reboot the switch.

The default timezone on the switch is (Coordinated Universal Time) UTC. Change the timezone on your switch to be the timezone for your location.

To update the timezone, use NTP interactive mode:

1. Run the following command in a terminal.

```
cumulus@switch:~$ sudo dpkg-reconfigure tzdata
```

2. Follow the on screen menu options to select the geographic area and region.

> **NOTE**
>
> Programs that are already running (including log files) and users currently logged in, do not see timezone changes made with interactive mode. To set the timezone for all services and daemons, reboot the switch.

Verify the System Time

Before you install the license, verify that the date and time on the switch are correct, and correct the date and time if necessary. If the date and time is incorrect, the switch might not be able to synchronize with Puppet or might return errors after you restart `switchd`:

```
Warning: Unit file of switchd.service changed on disk,
'systemctl daemon-reload' recommended.
```

Install the License

Cumulus Linux is licensed on a per-instance basis. Each network system is fully operational, enabling any capability to be utilized on the switch with the exception of forwarding on switch panel ports. Only eth0 and console ports are activated on an unlicensed instance of Cumulus Linux. Enabling front panel ports requires a license.

You receive a license key from NVIDIA or an authorized reseller. Here is a sample license key:

```
user@company.com|thequickbrownfoxjumpsoverthelazydog312
```

There are three ways to install the license onto the switch:

- Copy the license from a local server. Create a text file with the license and copy it to a server accessible from the switch. On the switch, use the following command to transfer the file directly on the switch, then install the license file:

```
cumulus@switch:~$ scp user@my_server:/home/user/
my_license_file.txt .
cumulus@switch:~$ sudo cl-license -i my_license_file.txt
```

- Copy the file to an HTTP server (not HTTPS), then reference the URL when you run `cl-license`:

```
cumulus@switch:~$ sudo cl-license -i <URL>
```

- Copy and paste the license key into the `cl-license` command:

```
cumulus@switch:~$ sudo cl-license -i
<paste license key>
^+d
```

Check that your license is installed with the `cl-license` command.

```
cumulus@switch:~$ cl-license
user@example.com|$ampleL1cen$et3xt
```

NOTE

It is not necessary to reboot the switch to activate the switch ports. After you install the license, restart the `switchd` service. All front panel ports become active and show up as swp1, swp2, and so on.

```
cumulus@switch:~$ sudo systemctl restart
switchd.service
```

WARNING

Restarting the `switchd` service causes all network ports to reset, interrupting network services, in addition to resetting the switch hardware configuration.

If a license is not installed on a Cumulus Linux switch, the `switchd` service does not start. After you install the license, start `switchd` as described above.

Configure Breakout Ports with Splitter Cables

If you are using 4x10G DAC or AOC cables, or want to break out 100G or 40G switch ports, configure the breakout ports. For more details, see Switch Port Attributes.

Test Cable Connectivity

By default, all data plane ports (every Ethernet port except the management interface, eth0) are disabled.

To test cable connectivity:

NCLU Commands | Linux Commands

To administratively enable a port:

```
cumulus@switch:~$ net add interface swp1
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

To administratively enable all physical ports, run the following command, where swp1-52 represents a switch with switch ports numbered from swp1 to swp52:

```
cumulus@switch:~$ net add interface swp1-52
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

To view link status, use the `net show interface all` command. The following examples show the output of ports in `admin down`, `down`, and `up` modes:

```
cumulus@switch:~$ net show interface all
State  Name           Spd  MTU    Mode
LLDP                    Summary
-----  -------------  ---  -----  -------------
----------------------  -------------------------
UP     lo             N/A  65536
Loopback                IP: 127.0.0.1/8
```

Configure Switch Ports

Layer 2 Port Configuration

Cumulus Linux does not put all ports into a bridge by default. To create a bridge and configure one or more front panel ports as members of the bridge, use the following examples as a guide.

NCLU Commands | Linux Commands

In the following configuration example, the front panel port swp1 is placed into a bridge called *bridge*.

```
cumulus@switch:~$ net add bridge bridge ports swp1
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

You can add a range of ports in one command. For example, to add swp1 through swp10, swp12, and swp14 through swp20 to bridge:

```
cumulus@switch:~$ net add bridge bridge ports
swp1-10,12,14-20
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

To view the changes in the kernel, use the `brctl` command:

```
cumulus@switch:~$ brctl show
bridge name     bridge id               STP enabled
```

```
interfaces
br0             8000.089e01cedcc2       yes             swp1
```

Layer 3 Port Configuration

You can also configure a front panel port or bridge interface as a layer 3 port.

NCLU Commands | Linux Commands

In the following configuration example, the front panel port swp1 is configured as a layer 3 access port:

```
cumulus@switch:~$ net add interface swp1 ip address
10.1.1.1/30
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

To add an IP address to a bridge interface, you must put it into a VLAN interface. If you want to use a VLAN other than the native one, set the bridge PVID:

```
cumulus@switch:~$ net add vlan 100 ip address 10.2.2.1/24
cumulus@switch:~$ net add bridge bridge pvid 100
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

To view the changes in the kernel, use the `ip addr show` command:

```
cumulus@switch:~$ ip addr show
...
4. swp1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
pfifo_fast master bridge state UP group default qlen 1000
        link/ether 44:38:39:00:6e:fe brd ff:ff:ff:ff:ff:ff
...
14: bridge: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
noqueue state UP group default
    link/ether 44:38:39:00:00:04 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::4638:39ff:fe00:4/64 scope link
        valid_lft forever preferred_lft forever
...
```

Configure a Loopback Interface

Cumulus Linux has a loopback interface preconfigured in the `/etc/network/interfaces` file. When the switch boots up, it has a loopback interface, called *lo*, which is up and assigned an IP address of 127.0.0.1.

TIP

The loopback interface *lo* must always be specified in the `/etc/`

`network/interfaces` file and must always be up.

To see the status of the loopback interface (lo):

NCLU Commands | Linux Commands

Use the `net show interface lo` command.

```
cumulus@switch:~$ net show interface lo
    Name    MAC                Speed    MTU    Mode
--  ------  -----------------  -------  -----  --------
UP  lo      00:00:00:00:00:00  N/A      65536  Loopback


Alias
-----
loopback interface
IP Details
-------------------------  --------------------
IP:                        127.0.0.1/8, ::1/128
IP Neighbor(ARP) Entries:  0
```

The loopback is up and is assigned an IP address of 127.0.0.1.

To add an IP address to a loopback interface, configure the *lo* interface:

```
cumulus@switch:~$ net add loopback lo ip address 10.1.1.1/32
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Multiple Loopbacks

You can add multiple loopback addresses. See Configure Multiple Loopbacks for details.

Reboot the Switch

After you complete the configuration in this section, reboot the switch:

```
cumulus@switch:~$ sudo reboot
```

Installation Management

This section describes how to manage, install, and upgrade Cumulus Linux on your switch.

Managing Cumulus Linux Disk Images

The Cumulus Linux operating system resides on a switch as a *disk image*. This section discusses how to manage the image.

To install a new Cumulus Linux image, refer to Installing a New Cumulus Linux Image. To upgrade Cumulus Linux, refer to Upgrading Cumulus Linux.

Determine the Switch Platform

To determine if your switch is on an x86 or ARM platform, run the `uname -m` command.

For example, on an x86 platform, `uname -m` outputs *x86_64*:

```
cumulus@switch:~$ uname -m
 x86_64
```

On an ARM platform, `uname -m` outputs *armv7l*:

```
cumulus@switch:~$ uname -m
 armv7l
```

You can also visit the HCL (hardware compatibility list) to look at your hardware and determine the processor type.

Reprovision the System (Restart the Installer)

Reprovisioning the system deletes all system data from the switch.

To stage an ONIE installer from the network (where ONIE automatically locates the installer), run the `onie-select -i` command. A reboot is required for the reinstall to begin.

```
cumulus@switch:~$ sudo onie-select -i
WARNING:
WARNING: Operating System install requested.
WARNING: This will wipe out all system data.
WARNING:
Are you sure (y/N)? y
Enabling install at next reboot...done.
Reboot required to take effect.
```

To cancel a pending reinstall operation, run the `onie-select -c` command:

```
cumulus@switch:~$ sudo onie-select -c
Cancelling pending install at next reboot...done.
```

To stage an installer located in a specific location, run the `onie-install -i` command. You can specify a local, absolute or relative path, an HTTP or HTTPS server, SCP or FTP server. You can also stage a Zero Touch Provisioning (ZTP) script along with the installer. The `onie-install` command is typically used with the `-a` option to activate installation. If you do not specify the `-a` option, a reboot is required for the reinstall to begin.

The following example stages the installer located at `http://203.0.113.10/image-installer` together with the ZTP script located at `http://203.0.113.10/ztp-script` and activates installation and ZTP:

```
cumulus@switch:~$ sudo onie-install -i http://203.0.113.10/
image-installer
cumulus@switch:~$ sudo onie-install -z http://203.0.113.10/ztp-
script
cumulus@switch:~$ sudo onie-install -a
```

You can also specify these options together in the same command. For example:

```
cumulus@switch:~$ sudo onie-install -i http://203.0.113.10/
image-installer -z http://203.0.113.10/ztp-script -a
```

To see more `onie-install` options, run `man onie-install.`

Uninstall All Images and Remove the Configuration

To remove all installed images and configurations, and return the switch to its factory defaults, run the `onie-select -k` command.

WARNING

The `onie-select -k` command takes a long time to run as it overwrites the entire NOS section of the flash. Only use this command if you want to erase all NOS data and take the switch out of service.

```
cumulus@switch:~$ sudo onie-select -k
WARNING:
WARNING: Operating System uninstall requested.
WARNING: This will wipe out all system data.
WARNING:
Are you sure (y/N)? y
Enabling uninstall at next reboot...done.
Reboot required to take effect.
```

A reboot is required for the uninstallation process to begin.

TIP

To cancel a pending uninstall operation, run the `onie-select -c` command:

```
cumulus@switch:~$ sudo onie-select -c
Cancelling pending uninstall at next reboot...done.
```

Boot into Rescue Mode

If your system becomes unresponsive is some way, you can correct certain issues by booting into ONIE rescue mode. In rescue mode, the file systems are unmounted and you can use various Cumulus Linux utilities to try and resolve a problem.

To reboot the system into ONIE rescue mode, run the `onie-select -r` command:

```
cumulus@switch:~$ sudo onie-select -r
```

```
WARNING:
WARNING: Rescue boot requested.
WARNING:
Are you sure (y/N)? y
Enabling rescue at next reboot...done.
Reboot required to take effect.
```

NOTE

A reboot is required to boot into rescue mode.

TIP

To cancel a pending rescue boot operation, run the `onie-select -c` command:

```
cumulus@switch:~$ sudo onie-select -c
Cancelling pending rescue at next reboot...done.
```

Inspect the Image File

The Cumulus Linux image file is executable. From a running switch, you can display, extract, and verify the contents of the image file.

To display the contents of the Cumulus Linux image file, pass the `info` option to the image file. For example, to display the contents of an image file called `onie-installer` located in the `/var/lib/cumulus/installer` directory:

```
cumulus@switch:~$ sudo /var/lib/cumulus/installer/onie-
installer info
Verifying image checksum ...OK.
Preparing image archive ... OK.
Control File Contents
=====================
Description: Cumulus Linux 4.1.0
Release: 4.1.0
Architecture: amd64
Switch-Architecture: bcm-amd64
Build-Id: dirtyz224615f
Build-Date: 2019-05-17T16:34:22+00:00
Build-User: clbuilder
Homepage: http://www.cumulusnetworks.com/
Min-Disk-Size: 1073741824
```

```
Min-Ram-Size: 536870912
mkimage-version: 0.11.111_gbcf0
```

To extract the contents of the image file, use with the `extract <path>` option. For example, to extract an image file called `onie-installer` located in the `/var/lib/cumulus/installer` directory to the `mypath` directory:

```
cumulus@switch:~$ sudo /var/lib/cumulus/installer/onie-
installer extract mypath
total 181860
-rw-r--r-- 1 4000 4000       308 May 16 19:04 control
drwxr-xr-x 5 4000 4000      4096 Apr 26 21:28 embedded-installer
-rw-r--r-- 1 4000 4000  13273936 May 16 19:04 initrd
-rw-r--r-- 1 4000 4000   4239088 May 16 19:04 kernel
-rw-r--r-- 1 4000 4000 168701528 May 16 19:04 sysroot.tar
```

To verify the contents of the image file, use with the `verify` option. For example, to verify the contents of an image file called `onie-installer` located in the `/var/lib/cumulus/installer` directory:

```
cumulus@switch:~$ sudo /var/lib/cumulus/installer/onie-
```

```
installer verify
Verifying image checksum ...OK.
Preparing image archive ... OK.
./cumulus-linux-bcm-amd64.bin.1: 161: ./cumulus-linux-bcm-
amd64.bin.1: onie-sysinfo: not found
Verifying image compatibility ...OK.
Verifying system ram ...OK.
```

Related Information

Open Network Install Environment (ONIE) Home Page

Installing a New Cumulus Linux Image

WARNING

In Cumulus Linux 4.2.0, the default password for the *cumulus* user account has changed to `cumulus`. The first time you log into Cumulus Linux, you are **required** to change this default password. Be sure to update any automation scripts before installing a new image. Cumulus Linux provides command line options to change the default password automatically during the installation process. Refer to ONIE Installation Options.

You can install a new Cumulus Linux image using ONIE, an open source project (equivalent to PXE on servers) that enables the installation of network operating systems (NOS) on bare metal switches.

Before you install Cumulus Linux, the switch can be in two different states:

- No image is installed on the switch (the switch is only running ONIE).
- Cumulus Linux is already installed on the switch but you want to use ONIE to reinstall Cumulus Linux or upgrade to a newer version.

The sections below describe some of the different ways you can install the

Cumulus Linux image, such as using a DHCP/web server, FTP, a local file, or a USB drive. Steps are provided for both installing directly from ONIE (if no image is installed on the switch) and from Cumulus Linux (if the image is already installed on the switch), where applicable. For additional methods to find and install the Cumulus Linux image, see the ONIE Design Specification.

You can download a Cumulus Linux image from the Cumulus Linux Downloads page.

> **WARNING**
>
> Installing the Cumulus Linux image is destructive; configuration files on the switch are not saved; copy them to a different server before installing.

In the following procedures:

- You can name your Cumulus Linux image using any of the ONIE naming schemes mentioned here.
- In the example commands, `[PLATFORM]` can be any supported Cumulus Linux platform, such as *x86_64*, or *arm*.
- Run the `sudo onie-install -h` command to show the ONIE installer options.
- After you install the Cumulus Linux image, you need to install the license

file. Refer to Install the License.

Install Using a DHCP/Web Server with DHCP Options

To install Cumulus Linux using a DHCP/web server *with* DHCP options, set up a DHCP/web server on your laptop and connect the eth0 management port of the switch to your laptop. After you connect the cable, the installation proceeds as follows:

1. The bare metal switch boots up and requests an IP address (DHCP request).
2. The DHCP server acknowledges and responds with DHCP option 114 and the location of the installation image.
3. ONIE downloads the Cumulus Linux image, installs, and reboots.
4. Success! You are now running Cumulus Linux.

> **NOTE**
>
> The most common method is to send DHCP option 114 with the entire URL to the web server (this can be the same system).

However, there are many other ways to use DHCP even if you do not have full control over DHCP. See the ONIE user guide for help with partial installer URLs and advanced DHCP options; both articles list more supported DHCP options.

Here is an example DHCP configuration with an ISC DHCP server:

```
subnet 172.0.24.0 netmask 255.255.255.0 {
  range 172.0.24.20 172.0.24.200;
  option default-url = "http://172.0.24.14/onie-installer-
[PLATFORM]";
}
```

Here is an example DHCP configuration with dnsmasq (static address assignment):

```
dhcp-host=sw4,192.168.100.14,6c:64:1a:00:03:ba,set:sw4
dhcp-option=tag:sw4,114,"http://roz.rtplab.test/onie-installer-
[PLATFORM]"
```

If you do not have a web server, you can use this free Apache example.

Install Using a DHCP/Web Server without DHCP Options

Follow the steps below if you can log into the switch on a serial console (ONIE), or log in on the console or with ssh (Install from Cumulus Linux).

Install from ONIE | Install from Cumulus Linux

1. Place the Cumulus Linux image in a directory on the web server.
2. Run the `onie-nos-install` command:

```
ONIE:/ #onie-nos-install http://10.0.1.251/path/to/
cumulus-install-[PLATFORM].bin
```

Install Using a Web Server with no DHCP

Follow the steps below if you can log into the switch on a serial console (ONIE), or log in on the console or with ssh (Install from Cumulus Linux) but *no* DHCP server is available.

> **NOTE**
>
> You need a console connection to access the switch; you cannot perform this procedure remotely.

Install from ONIE | Install from Cumulus Linux

1. ONIE is in *discovery mode*. You must disable discovery mode with the following command:

```
onie# onie-discovery-stop
```

   On older ONIE versions, if the `onie-discovery-stop` command is not supported, run:

```
onie# /etc/init.d/discover.sh stop
```

2. Assign a static address to eth0 with the `ip addr add` command:

```
ONIE:/ #ip addr add 10.0.1.252/24 dev eth0
```

3. Place the Cumulus Linux image in a directory on your web server.

4. Run the installer manually (because there are no DHCP options):

```
ONIE:/ #onie-nos-install http://10.0.1.251/path/to/
cumulus-install-[PLATFORM].bin
```

Install Using FTP Without a Web Server

Follow the steps below if your laptop is on the same network as the switch eth0 interface but *no* DHCP server is available.

> **NOTE**
>
> Installing the Cumulus Linux image using FTP from ONIE is not supported on the Dell N3048EP-ON switch. Use one of the other installation methods, such as a USB, or install the image from Cumulus Linux.

Install from ONIE | Install from Cumulus Linux

1. Set up DHCP or static addressing for eth0. The following example assigns a static address to eth0:

```
ONIE:/ #ip addr add 10.0.1.252/24 dev eth0
```

2. If you are using static addressing, disable ONIE discovery mode:

```
onie# onie-discovery-stop
```

On older ONIE versions, if the `onie-discovery-stop` command is not supported, run:

```
onie# /etc/init.d/discover.sh stop
```

3. Place the Cumulus Linux image into a TFTP or FTP directory.

4. If you are not using DHCP options, run one of the following commands (`tftp` for TFTP or `ftp` for FTP):

```
ONIE# onie-nos-install ftp://local-ftp-server/cumulus-
install-[PLATFORM].bin

ONIE# onie-nos-install tftp://local-tftp-server/cumulus-
install-[PLATFORM].bin
```

Install Using a Local File

Follow the steps below to install the Cumulus Linux image referencing a local file.

> **NOTE**
>
> Installing the Cumulus Linux image referencing a local file from ONIE is not supported on the Dell N3048EP-ON switch. Use one of the other installation methods, such as a web server or USB, or install the image from Cumulus Linux.

Install from ONIE | Install from Cumulus Linux

1. Set up DHCP or static addressing for eth0. The following example assigns a static address to eth0:

```
ONIE:/ #ip addr add 10.0.1.252/24 dev eth0
```

2. If you are using static addressing, disable ONIE discovery mode.

```
onie# onie-discovery-stop
```

On older ONIE versions, if the `onie-discovery-stop` command is not supported, run:

```
onie# /etc/init.d/discover.sh stop
```

3. Use scp to copy the Cumulus Linux image to the switch.

4. Run the installer manually from ONIE:

```
ONIE:/ #onie-nos-install /path/to/local/file/cumulus-
install-[PLATFORM].bin
```

Install Using a USB Drive

Follow the steps below to install the Cumulus Linux image using a USB drive. Instructions are provided for x86 and ARM platforms.

> **TIP**
>
> Installing Cumulus Linux using a USB drive is fine for a single switch here and there but is not scalable. DHCP can scale to hundreds of switch installs with zero manual input unlike USB installs.

Prepare for USB Installation

1. From the Cumulus Linux Downloads page, download the appropriate Cumulus Linux image for your x86 or ARM platform.

2. From a computer, prepare your USB drive by formatting it using one of the supported formats: FAT32, vFAT or EXT2.

   ▼ Optional: Prepare a USB Drive inside Cumulus Linux

3. Copy the Cumulus Linux image to the USB drive, then rename the image file to:

   - `onie-installer-x86_64`, if installing on an x86 platform

- `onie-installer-arm`, if installing on an ARM platform

You can also use any of the ONIE naming schemes mentioned here.

When using a Mac or Windows computer to rename the installation file, the file extension might still be present. Make sure to remove the file extension otherwise ONIE is not able to detect the file.

4. Insert the USB drive into the switch, then continue with the appropriate instructions below for your x86 or ARM platform.

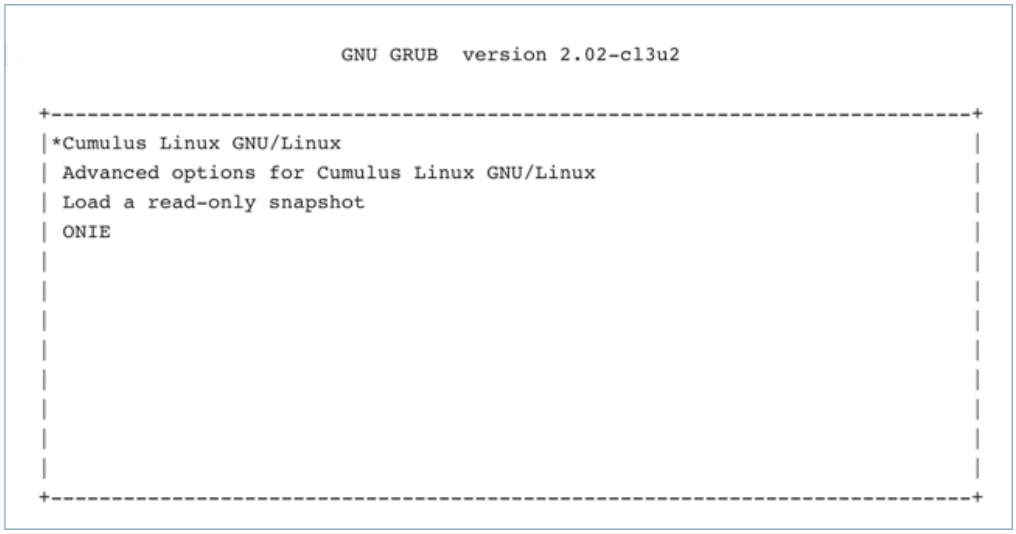
x86 Platforms ARM Platforms

1. Prepare the switch for installation:

   - If the switch is offline, connect to the console and power on the switch.
   - If the switch is already online in ONIE, use the `reboot` command.

   SSH sessions to the switch get dropped after this step. To complete the remaining instructions, connect to the console of the switch. Cumulus Linux switches display their boot process to the console; you need to monitor the console specifically to complete the next step.

2. Monitor the console and select the ONIE option from the first GRUB screen shown below.



3. Cumulus Linux on x86 uses GRUB chainloading to present a second GRUB menu specific to the ONIE partition. No action is necessary in this menu to select the default option *ONIE: Install OS*.

ONIE Installation Options

You can run several installer command line options from ONIE to perform basic switch configuration automatically after installation completes and Cumulus Linux boots for the first time. These options enable you to:

- Set a unique password for the *cumulus* user
- Apply a Cumulus Linux license
- Provide an initial network configuration
- Execute a ZTP script to perform necessary configuration

NOTE

The `onie-nos-install` command does *not* allow you specify command line parameters. You must access the switch from the console and transfer a disk image to the switch. You must then make the disk image executable and install the image directly from the ONIE command line with the options you want to use.
The following example commands transfer a disk image to the switch, make the image executable, and install the image with the `--password` option to change the default cumulus user password:

```
ONIE:/ # wget http://myserver.datacenter.com/cumulus-
linux-4.2.0-bcm-amd64.bin
```

```
ONIE:/ # chmod 755 cumulus-linux-4.2.0-bcm-amd64.bin
ONIE:/ # ./cumulus-linux-4.2.0-bcm-amd64.bin --password
'MyP4$$word'
```

You can run more than one option in the same command.

Set the cumulus User Password

The default *cumulus* user account password is `cumulus`. When you log into Cumulus Linux for the first time, you must provide a new password for the *cumulus* account, then log back into the system. This password change is **required** in Cumulus Linux 4.2 and later.

To automate this process, you can specify a new password from the command line of the installer with the `--password '<clear text-password>'` option. For example, to change the default *cumulus* user password to `MyP4$$word`:

```
ONIE:/ # ./cumulus-linux-4.2.0-bcm-amd64.bin --password
'MyP4$$word'
```

To provide a hashed password instead of a clear text password, use the `--hashed-password '<hash>'` option. Using an encrypted hash is recommended to maintain a secure management network.

1. Generate a sha-512 password hash with the following python command. The example command generates a sha-512 password hash for the password `MyP4$$word`.

```
user@host:~$ python3 -c "import crypt;
print(crypt.crypt('MyP4$$word',salt=crypt.mksalt()))"
$6$hs7OPmnrfvLNKfoZ$iB3hy5N6Vv6koqDmxixpTO6lej6VaoKGvs5E8p5zNo4tPec0KKqyQnrFMII3
```

2. Specify the new password from the command line of the installer with the `--hashed-password '<hash>'` command:

```
ONIE:/ # ./cumulus-linux-4.2.0-bcm-amd64.bin  --hashed-
password
'$6$hs7OPmnrfvLNKfoZ$iB3hy5N6Vv6koqDmxixpTO6lej6VaoKGvs5E8p5zNo4tPec0KKqyQnrFMII
```

NOTE

If you specify both the `--password` and `--hashed-password` options, the `--hashed-password` option takes precedence and the `--`

`password` option is ignored.

Apply a Cumulus Linux License

To apply a license and start the `switchd` service automatically after Cumulus Linux boots for the first time after installation, use the `--license <license-string>` option. For example:

```
ONIE:/ # ./cumulus-linux-4.2.0-bcm-amd64.bin  --license
'customer@datacenter.com|4C3YMCACDiK0D/
EnrxlXpj71FBBNAg4Yrq+brza4ZtJFCInvalid'
```

Provide Initial Network Configuration

To provide initial network configuration automatically when Cumulus Linux boots for the first time after installation, use the `--interfaces-file <filename>` option. For example, to copy the contents of a file called `network.intf` into the `/etc/network/interfaces` file and run the `ifreload -a` command:

```
ONIE:/ # ./cumulus-linux-4.2.0-bcm-amd64.bin  --interfaces-file
```

```
network.intf
```

Execute a ZTP Script

To run a ZTP script that contains commands to execute after Cumulus Linux boots for the first time after installation, use the `--ztp <filename>` option. For example, to run a ZTP script called `initial-conf.ztp`:

```
ONIE:/ # ./cumulus-linux-4.2.0-bcm-amd64.bin --ztp initial-
conf.ztp
```

The ZTP script must contain the `CUMULUS-AUTOPROVISIONING` string near the beginning of the file and must reside on the ONIE filesystem. Refer to Zero Touch Provisioning - ZTP.

If you use the `--ztp` option together with any of the other command line options, the ZTP script takes precedence and the other command line options are ignored.

Edit the Cumulus Linux Image (Advanced)

The Cumulus Linux disk image file contains a BASH script that includes a set of variables. You can set these variables to be able to install a fully-configured system with a single image file.

▼ To edit the image

Related Information

- ONIE Design Specification
- Cumulus Linux Downloads page
- Cumulus on a Stick
- Managing Cumulus Linux Disk Images

Upgrading Cumulus Linux

WARNING

In Cumulus Linux 4.2.0, the default password for the *cumulus* user account has changed to `cumulus`. The first time you log into Cumulus Linux, you are **required** to change this default password. Be sure to update any automation scripts before you upgrade. You can use ONIE command line options to change the default password automatically during the Cumulus Linux image installation process. Refer to ONIE Installation Options.

This topic describes how to upgrade Cumulus Linux on your switch.

Deploying, provisioning, configuring, and upgrading switches using automation is highly recommended, even with small networks or test labs. During the upgrade process, you can quickly upgrade dozens of devices in a repeatable manner. Using tools like Ansible, Chef, or Puppet for configuration management greatly increases the speed and accuracy of the next major upgrade; these tools also enable the quick swap of failed switch hardware.

Before You Upgrade

> **TIP**
>
> Be sure to read the knowledge base article Upgrades: Network Device and Linux Host Worldview Comparison, which provides a detailed comparison between the network device and Linux host worldview of upgrade and installation.

Understanding the location of configuration data is required for successful upgrades, migrations, and backup. As with other Linux distributions, the `/etc` directory is the primary location for all configuration data in Cumulus Linux. The following list is a likely set of files that you need to back up and migrate to a new release. Make sure you examine any file that has been changed. Consider making the following files and directories part of a backup strategy.

Network Configuration Files | Commonly-Used Files | Never Migrate Files

File Name and Location	Explanation	Cumulus Linux Documentation	Debian Documentation
`/etc/network/`	Network configuration files, most notably `/etc/network/interfaces` and `/etc/network/interfaces.d/`	Switch Port Attributes	N/A
`/etc/resolv.conf`	DNS resolution	Not unique to Cumulus Linux: wiki.debian.org/NetworkConfiguration	https://www.debian.org/doc/manuals/debian-reference/ch05.en.html
`/etc/frr/`	Routing application (responsible for BGP and OSPF)	FRRouting	N/A
`/etc/hostname`	Configuration file for the hostname of the switch	Quick Start Guide	https://wiki.debian.org/HowTo/ChangeHostname
`/etc/hosts`	Configuration file for the hostname of the switch	Quick Start Guide	https://wiki.debian.org/HowTo/ChangeHostname

If you are using certain forms of network virtualization, including VMware NSX-V or Midokura MidoNet, you might have updated the `/usr/share/openvswitch/scripts/ovs-ctl-vtep` file. This file is not marked as a configuration file; therefore, if the file contents change in a newer release of Cumulus Linux, they overwrite any changes you made to the file. Be sure to back up this file and the database file `conf.db` before upgrading.

> **NOTE**
>
> You can check which files have changed since the last Cumulus Linux image install with the following commands. Be sure to back up any changed files:
>
> - Run the `sudo dpkg --verify` command to show a list of changed files.
> - Run the `egrep -v '^$|^#|=""$' /etc/default/isc-dhcp-*` command to see if any of the generated `/etc/default/isc-*` files have changed.

Upgrade Cumulus Linux

You can upgrade Cumulus Linux in one of two ways:

- Install a Cumulus Linux image of the new release, using ONIE.
- Upgrade only the changed packages using the `sudo -E apt-get update` and `sudo -E apt-get upgrade` command.

> **NOTE**
>
> Upgrading an MLAG pair requires additional steps. If you are using MLAG to dual connect two Cumulus Linux switches in your environment, follow the steps in Upgrade Switches in an MLAG Pair below to ensure a smooth upgrade.

Should I Install a Cumulus Linux Image or Upgrade Packages?

The decision to upgrade Cumulus Linux by either installing a Cumulus Linux image or upgrading packages depends on your environment and your preferences. Here are some recommendations for each upgrade method.

Installing a Cumulus Linux image is recommended if you are performing a rolling upgrade in a production environment and if are using up-to-date and comprehensive automation scripts. This upgrade method enables you to choose the exact release to which you want to upgrade and is the *only* method available to upgrade your switch to a new release train (for example, from 3.7.12 to 4.1.0).

Be aware of the following when installing the Cumulus Linux image:

- Installing a Cumulus Linux image is destructive; any configuration files on the switch are not saved; copy them to a different server before you start the Cumulus Linux image install.
- You must move configuration data to the new OS using ZTP or

automation while the OS is first booted, or soon afterwards using out-of-band management.

- Moving a configuration file might cause issues;
- Identifying all the locations of configuration data is not always an easy task. See Before You Upgrade Cumulus Linux above.
- Merge conflicts with configuration file changes in the new release might go undetected.
- If configuration files are not restored correctly, you might be unable to ssh to the switch from in-band management. Out-of-band connectivity (eth0 or console) is recommended.
- You *must* reinstall and reconfigure third-party applications after upgrade.

Package upgrade is recommended if you are upgrading from Cumulus Linux 4.0, or if you use third-party applications (package upgrade does not replace or remove third-party applications, unlike the Cumulus Linux image install).

Be aware of the following when upgrading packages:

- You cannot upgrade the switch to a new release train. For example, you **cannot** upgrade the switch from **3**.7.x to **4**.1.0.
- The `sudo -E apt-get upgrade` command might result in services being restarted or stopped as part of the upgrade process.
- The `sudo -E apt-get upgrade` command might disrupt core services by changing core service dependency packages.
- After you upgrade, account UIDs and GIDs created by packages might be different on different switches, depending on the configuration and package installation history.

Cumulus Linux Image Install (ONIE)

ONIE is an open source project (equivalent to PXE on servers) that enables the installation of network operating systems (NOS) on a bare metal switch.

> **NOTE**
>
> Lightweight network virtualization (LNV) is deprecated in Cumulus Linux 4.0 in favor of Ethernet virtual private networks (EVPN. If your network is configured for LNV, you need to migrate your network configuration to a BGP EVPN configuration that is functionally equivalent ***before*** you upgrade. Refer to Migrating from LNV to EVPN.

To upgrade the switch:

1. Back up the configurations off the switch.
2. Download the Cumulus Linux image.
3. Install the Cumulus Linux image with the `onie-install -a -i <image-location>` command, which boots the switch into ONIE. The following example command installs the image from a web server, then reboots the switch. There are additional ways to install the Cumulus Linux image, such as using FTP, a local file, or a USB drive. For more information, see Installing a New Cumulus Linux Image.

```
cumulus@switch:~$ sudo onie-install -a -i http://10.0.1.251/
cumulus-linux-4.1.0-mlx-amd64.bin && sudo reboot
```

4. Restore the configuration files to the new release - ideally with automation.
5. Verify correct operation with the old configurations on the new release.
6. Reinstall third party applications and associated configurations.

Package Upgrade

Cumulus Linux completely embraces the Linux and Debian upgrade workflow, where you use an installer to install a base image, then perform any upgrades within that release train with `sudo -E apt-get update` and `sudo -E apt-get upgrade` commands. Any packages that have been changed since the base install get upgraded in place from the repository. All switch configuration files remain untouched, or in rare cases merged (using the Debian merge function) during the package upgrade.

When you use package upgrade to upgrade your switch, configuration data stays in place while the packages are upgraded. If the new release updates a configuration file that you changed previously, you are prompted for the version you want to use or if you want to evaluate the differences.

To upgrade the switch using package upgrade:

1. Back up the configurations from the switch.

2. Fetch the latest update metadata from the repository.

```
cumulus@switch:~$ sudo -E apt-get update
```

3. Review potential upgrade issues (in some cases, upgrading new packages might also upgrade additional existing packages due to dependencies). Run the following command to see the additional packages that will be installed or upgraded.

```
cumulus@switch:~$ sudo -E apt-get upgrade --dry-run
```

4. Upgrade all the packages to the latest distribution.

```
cumulus@switch:~$ sudo -E apt-get upgrade
```

If no reboot is required after the upgrade completes, the upgrade ends, restarts all upgraded services, and log messages in the `/var/log/syslog` file similar to the ones shown below. In the examples below, only the `frr` package is upgraded.

```
Policy: Service frr.service action stop postponed
Policy: Service frr.service action start postponed
Policy: Restarting services: frr.service
Policy: Finished restarting services
Policy: Removed /usr/sbin/policy-rc.d
Policy: Upgrade is finished
```

If the upgrade process encounters changed configuration files that have new versions in the release to which you are upgrading, you see a message similar to this:

```
Configuration file '/etc/frr/daemons'
==> Modified (by you or by a script) since installation.
==> Package distributor has shipped an updated version.
What would you like to do about it ? Your options are:
Y or I : install the package maintainer's version
N or O : keep your currently-installed version
D : show the differences between the versions
Z : start a shell to examine the situation
The default action is to keep your current version.
*** daemons (Y/I/N/O/D/Z) [default=N] ?

- To see the differences between the currently installed
```

```
version and the
new version, type `D`- To keep the currently installed
version, type `N`.
The new package version is installed with the suffix `_.dpkg-
dist`
(for example, `/etc/frr/daemons.dpkg-dist`). When upgrade is
complete and
**before** you reboot, merge your changes with the changes
from the newly
installed file.
- To install the new version, type `I`. Your currently
installed version is
saved with the suffix `.dpkg-old`.
When the upgrade is complete, you can search for the files
with the
`sudo find / -mount -type f -name '*.dpkg-*'` command.
```

If you see errors for expired GPG keys that prevent you from upgrading packages, follow the steps in Upgrading Expired GPG Keys.

5. Reboot the switch if the upgrade messages indicate that a system restart is required.

```
cumulus@switch:~$ sudo -E apt-get upgrade
... upgrade messages here ...

*** Caution: Service restart prior to reboot could cause
unpredictable behavior
*** System reboot required ***
cumulus@switch:~$ sudo reboot
```

6. Verify correct operation with the old configurations on the new version.

Upgrade Notes

Package upgrade always updates to the latest available release in the Cumulus Linux repository. For example, if you are currently running Cumulus Linux 4.0.0 and run the `sudo -E apt-get upgrade` command on that switch, the packages are upgraded to the latest releases contained in the latest 4.y.z release.

Because Cumulus Linux is a collection of different Debian Linux packages, be aware of the following:

- The `/etc/os-release` and `/etc/lsb-release` files are updated to the currently installed Cumulus Linux release when you upgrade the switch using either *package upgrade* or *Cumulus Linux image install*. For example, if you run `sudo -E apt-get upgrade` and the latest Cumulus Linux release on the repository is 4.1.0, these two files display the release

as 4.1.0 after the upgrade.

- The `/etc/image-release` file is updated **only** when you run a Cumulus Linux image install. Therefore, if you run a Cumulus Linux image install of Cumulus Linux 4.0.0, followed by a package upgrade to 4.1.0 using `sudo -E apt-get upgrade`, the `/etc/image-release` file continues to display Cumulus Linux 4.0.0, which is the originally installed base image.

Upgrade Switches in an MLAG Pair

If you are using MLAG to dual connect two switches in your environment, follow the steps below to upgrade the switches.

You must upgrade both switches in the MLAG pair to the same release of Cumulus Linux.

WARNING

For networks with MLAG deployments, you can only upgrade to Cumulus Linux 4.2 from version 3.7.10 or later. If you are using a version of Cumulus Linux earlier than 3.7.10, you must upgrade to version 3.7.10 first, then upgrade to version 4.2. Version 3.7.10 is available on the downloads page on our website.

> ⚠ **IMPORTANT**
>
> During upgrade, MLAG bonds stay single-connected while the switches are running different major releases; for example, while leaf01 is running 3.7.12 and leaf02 is running 4.2.0.
>
> This is due to a change in the bonding driver regarding how the *actor port key* is derived, which causes the port key to have a different value for links with the same speed/duplex settings across different major releases. The port key received from the LACP partner must remain consistent between all bond members in order for all bonds to be synchronized. When each MLAG switch sends LACPDUs with different port keys, only links to one MLAG switch are in sync.

1. Verify the switch is in the secondary role:

```
cumulus@switch:~$ clagctl status
```

2. Shut down the core uplink layer 3 interfaces:

```
cumulus@switch:~$ sudo ip link set swpX down
```

3. Shut down the peer link:

```
cumulus@switch:~$ sudo ip link set peerlink down
```

4. Run the `onie-install -a -i <image-location>` command to boot the switch into ONIE. The following example command installs the image from a web server. There are additional ways to install the Cumulus Linux image, such as using FTP, a local file, or a USB drive. For more information, see Installing a New Cumulus Linux Image.

```
cumulus@switch:~$ sudo onie-install -a -i http://10.0.1.251/
downloads/cumulus-linux-4.1.0-mlx-amd64.bin
```

5. Reboot the switch:

```
cumulus@switch:~$ sudo reboot
```

6. Verify STP convergence across both switches:

```
cumulus@switch:~$ mstpctl showall
```

7. Verify core uplinks and peer links are UP:

```
cumulus@switch:~$ net show interface
```

8. Verify MLAG convergence:

```
cumulus@switch:~$ clagctl status
```

9. Make this secondary switch the primary:

```
cumulus@switch:~$ clagctl priority 2048
```

10. Verify the other switch is now in the secondary role.

11. Repeat steps 2-8 on the new secondary switch.

12. Remove the priority 2048 and restore the priority back to 32768 on the current primary switch:

```
cumulus@switch:~$ clagctl priority 32768
```

Roll Back a Cumulus Linux Installation

Even the most well planned and tested upgrades can result in unforeseen problems; sometimes the best solution is to roll back to the previous state. There are three main strategies; all require detailed planning and execution:

- Flatten and rebuild: If the OS becomes unusable, you can use orchestration tools to reinstall the previous OS release from scratch and then rebuild the configuration automatically.
- Backup and restore: Another common strategy is to restore to a previous state using a backup captured before the upgrade. See Back up and Restore.

The method you employ is specific to your deployment strategy, so providing detailed steps for each scenario is outside the scope of this document.

Third Party Packages

Third party packages in the *Linux host* world often use the same package system as the distribution into which it is to be installed (for example, Debian uses `apt-get`). Or, the package might be compiled and installed by the system administrator. Configuration and executable files generally

follow the same filesystem hierarchy standards as other applications.

If you install any third party applications on a Cumulus Linux switch, configuration data is typically installed into the `/etc` directory, but it is not guaranteed. It is your responsibility to understand the behavior and configuration file information of any third party packages installed on the switch.

After you upgrade using a full Cumulus Linux image install, you need to reinstall any third party packages or any Cumulus Linux add-on packages.

Related Information

- Upgrades: Network Device Worldview and Linux Host Worldview Comparison
- Automation Solutions
- ONIE Design Specification
- Multi-Chassis Link Aggregation - MLAG
- Zero Touch Provisioning - ZTP

Migrating from LNV to EVPN

Lightweight network virtualization (LNV) is deprecated in Cumulus Linux 4.0 in favor of Ethernet virtual private networks (EVPN) to enable interoperability with switches from other manufacturers, to commit to industry standards, and because the benefits of EVPN outweigh those of LNV.

If your network is configured for LNV, you need to migrate your network configuration to a BGP EVPN configuration that is functionally equivalent **before** you upgrade to Cumulus Linux 4.0 or later.

Migration Considerations

You *cannot* run LNV and EVPN at the same time for the following reasons:

- It is *not* possible to reconcile the bridge-learning configuration on all of the VTEP interfaces if both LNV and EVPN are enabled at the same time. LNV requires MAC learning to be enabled on the VXLAN VTEP interfaces. EVPN requires MAC learning to be *disabled* on the VXLAN VTEP interfaces.
- The Linux bridge installs MAC address entries differently when LNV is enabled than when EVPN is enabled. Different flags are set on the MAC addresses in the Linux kernel depending on how the address is learned. Duplicate and/or conflicting bridge entries and race conditions become a possibility when both are enabled at the same time. Because the kernel bridging table is the basis for programming the forwarding ASICs, this

might lead to downstream inconsistencies in the hardware forwarding tables.
- The standard IPv4 unicast address family is commonly used to route inside the fabric for spine and leaf Clos networks. Because FRRouting does not currently support BGP dynamic capability negotiation, enabling the EVPN address family requires all of the neighbors to restart for the changes to take effect. This results in a brief disruption to traffic forwarding.

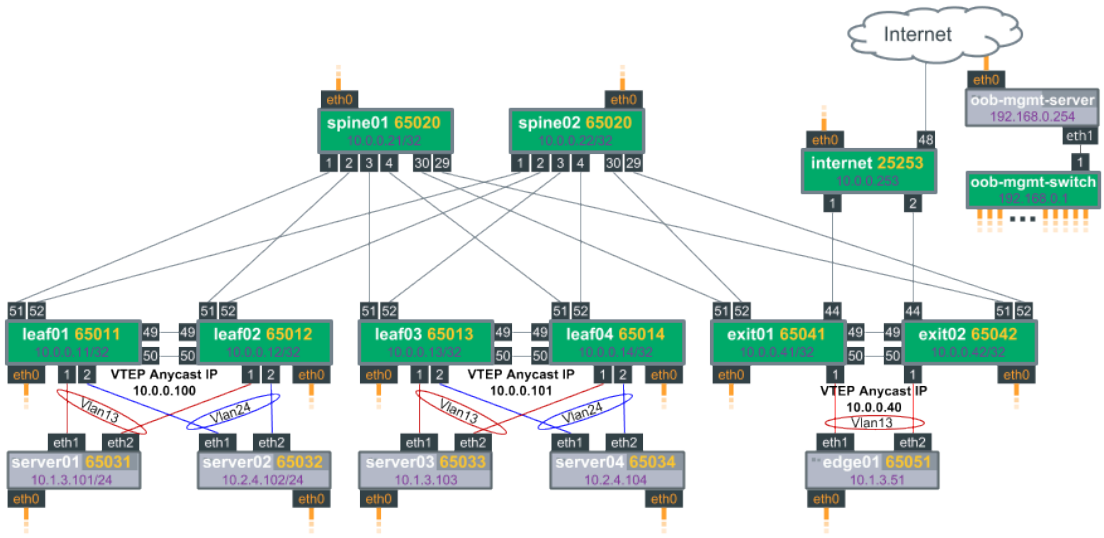
Upgrade to EVPN

Using automation, such as Ansible to upgrade to EVPN is highly recommended. Automation ensures minimal downtime, reduces human error, and is useful at almost any scale.

Using NCLU to update the configuration is also recommended for the following reasons:

- NCLU restarts services and reloads interfaces automatically so the changes can take effect.
- With the transactional commit model of NCLU, the order in which the NCLU commands are entered is of no consequence. This further reduces complexity and hidden dependencies.

The upgrade steps described here are based on the following example topology (based on the Cumulus Linux Reference Topology):



This topology:

- Uses MLAG to accommodate dual-attached servers
- Uses active-active (anycast) mode for VXLAN VTEPs (leaf and exit nodes)
- LNV service nodes (`vxsnd`) run in anycast mode on all spines
- Vlan13 and Vlan24 are extended between the two racks (leaf01 and leaf02) and (leaf03 and leaf04)
- VXLAN routing uses centralized routing at the exit nodes

The BGP EVPN configuration for a centralized routing topology is slightly different on the exit/routing leafs compared to the other ToR leaf switches.

1. Run the following NCLU commands on each type of device shown (leaf, exit, spine):

   Leaf node NCLU commands

```
# BGP changes
cumulus@switch:~$ net add bgp l2vpn evpn neighbor swp51-52
activate
cumulus@switch:~$ net add bgp l2vpn evpn advertise-all-vni

# Disable MAC learning on VNI
cumulus@switch:~$ net add vxlan vni-13 bridge learning off
cumulus@switch:~$ net add vxlan vni-24 bridge learning off

# Remove LNV (vxrd) configuration
cumulus@switch:~$ net del loopback lo vxrd-src-ip
cumulus@switch:~$ net del loopback lo vxrd-svcnode-ip
```

Exit node NCLU commands

```
# BGP changes
cumulus@switch:~$ net add bgp l2vpn evpn neighbor swp51-52
activate
cumulus@switch:~$ net add bgp l2vpn evpn advertise-all-vni
cumulus@switch:~$ net add bgp l2vpn evpn advertise-default-gw

# Disable MAC learning on VNI
cumulus@switch:~$ net add vxlan vni-13 bridge learning off
```

```
cumulus@switch:~$ net add vxlan vni-24 bridge learning off


# Remove LNV (vxrd) configuration
cumulus@switch:~$ net del loopback lo vxrd-src-ip
cumulus@switch:~$ net del loopback lo vxrd-svcnode-ip
```

Spine node NCLU commands

```
# BGP changes
cumulus@switch:~$ net add bgp l2vpn evpn neighbor swp1-4
activate


# Remove LNV service node (vxsnd) configuration
cumulus@switch:~$ net del lnv service-node anycast-ip
10.0.0.200
cumulus@switch:~$ net del lnv service-node peers 10.0.0.21
10.0.0.22
cumulus@switch:~$ net del lnv service-node source [primary-
loopback-ip]


# Remove unused LNV anycast address 10.0.0.200
cumulus@switch:~$ net del loopback lo ip address 10.0.0.200/32
cumulus@switch:~$ net del bgp ipv4 unicast network 10.0.0.200/
```

```
32
```

2. Manually disable and stop the LNV daemons. NCLU can remove the LNV configuration from the configuration files, but you must manually stop and disable these daemons before you commit the NCLU changes. After you commit the NCLU changes, NCLU restarts the BGP daemon, which enables the EVPN address family.

> **NOTE**
>
> Traffic loss can start to occur at this point.

3. To disable and stop the LNV registration daemon, run the following commands on the leaf and exit nodes:

```
cumulus@switch:~$ sudo systemctl disable vxrd
cumulus@switch:~$ sudo systemctl stop vxrd
```

4. To disable and stop the LNV service node daemon, run the following commands on the spine nodes:

```
cumulus@switch:~$ sudo systemctl disable vxsnd
cumulus@switch:~$ sudo systemctl stop vxsnd
```

5. To commit and apply the pending NCLU changes, run the following command on all the nodes:

```
cumulus@switch:~$ net commit
```

Verify the Upgrade

To check that LNV is disabled, run the `net show lnv` command on any node. This command returns no output when LNV is disabled.

> **NOTE**
>
> This command is for verification on Cumulus Linux 3.x only. This command has been removed in Cumulus Linux 4.0 and does not work after you upgrade.

```
cumulus@switch:~$ net show lnv
```

To ensure that EVPN BGP neighbors are up, run the `net show bgp l2vpn summary` command:

```
cumulus@switch:~$ net show bgp l2vpn evpn summary
BGP router identifier 10.0.0.11, local AS number 65011 vrf-id 0
BGP table version 0
RIB entries 23, using 3496 bytes of memory
Peers 2, using 39 KiB of memory
Neighbor        V         AS MsgRcvd MsgSent   TblVer  InQ
OutQ  Up/Down State/PfxRcd
spine01(swp51)  4      65020   10932   11064        0    0    0
00:14:28           48
spine02(swp52)  4      65020   10938   11068        0    0    0
00:14:27           48
Total number of neighbors 2
```

To examine the EVPN routes, run the `net show bgp l2vpn evpn route` command. Because a MAC address only appears as a type-2 route if the host has generated traffic and its MAC is learned by the local EVPN-enabled switch, a host that does not send any traffic does not create a type-2 EVPN route until it sends a frame that ingresses the EVPN-enabled local switch.

```
cumulus@switch:~$ net show bgp l2vpn evpn route
BGP table version is 45, local router ID is 10.0.0.11
Status codes: s suppressed, d damped, h history, * valid, >
best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete
EVPN type-2 prefix:
[2]:[ESI]:[EthTag]:[MAClen]:[MAC]:[IPlen]:[IP]
EVPN type-3 prefix: [3]:[EthTag]:[IPlen]:[OrigIP]
EVPN type-5 prefix: [5]:[ESI]:[EthTag]:[IPlen]:[IP]
    Network          Next Hop            Metric LocPrf Weight
Path
Route Distinguisher: 10.0.0.11:2
*> [2]:[0]:[0]:[48]:[00:03:00:11:11:01]
                    10.0.0.100                         32768 i
*> [2]:[0]:[0]:[48]:[02:03:00:11:11:01]
                    10.0.0.100                         32768 i
*> [2]:[0]:[0]:[48]:[02:03:00:11:11:02]
                    10.0.0.100                         32768 i
*> [3]:[0]:[32]:[10.0.0.100]
                    10.0.0.100                         32768 i
Route Distinguisher: 10.0.0.11:3
*> [2]:[0]:[0]:[48]:[00:03:00:22:22:02]
                    10.0.0.100                         32768 i
*> [2]:[0]:[0]:[48]:[02:03:00:22:22:01]
```

```
                    10.0.0.100                          32768 i
*> [2]:[0]:[0]:[48]:[02:03:00:22:22:02]
                    10.0.0.100                          32768 i
*> [3]:[0]:[32]:[10.0.0.100]
                    10.0.0.100                          32768 i
Route Distinguisher: 10.0.0.13:2
*  [2]:[0]:[0]:[48]:[00:03:00:33:33:01]
                    10.0.0.101                              0
65020 65013 i
*> [2]:[0]:[0]:[48]:[00:03:00:33:33:01]
                    10.0.0.101                              0
65020 65013 i
*  [2]:[0]:[0]:[48]:[02:03:00:33:33:01]
                    10.0.0.101                              0
65020 65013 i
*> [2]:[0]:[0]:[48]:[02:03:00:33:33:01]
                    10.0.0.101                              0
65020 65013 i
*  [2]:[0]:[0]:[48]:[02:03:00:33:33:02]
                    10.0.0.101                              0
65020 65013 i
*> [2]:[0]:[0]:[48]:[02:03:00:33:33:02]
                    10.0.0.101                              0
65020 65013 i
```

```
*  [3]:[0]:[32]:[10.0.0.101]
                    10.0.0.101                              0
65020 65013 i
*> [3]:[0]:[32]:[10.0.0.101]
                    10.0.0.101                              0
65020 65013 i
...
```

TIP

You can filter the EVPN route output by route type. The multicast route type corresponds to type-3. The prefix route type is type-5 (but is not used here).

```
cumulus@switch:~$ net show bgp l2vpn evpn route type
   macip      : MAC-IP (Type-2) route
   multicast  : Multicast
   prefix     : An IPv4 or IPv6 prefix
```

In the EVPN route output below, Cumulus Linux learned 00:03:00:33:33:01

with a next-hop (VTEP IP address) of 10.0.0.101. The MAC address of server03 is 00:03:00:33:33:01.

```
cumulus@leaf01:~$ net show bgp l2vpn evpn route
...

Route Distinguisher: 10.0.0.13:2
*  [2]:[0]:[0]:[48]:[00:03:00:33:33:01]
                    10.0.0.101                              0
65020 65013 i
...
```

To ensure the type-2 route is installed in the bridge table, run the `net show bridge macs <mac-address>` command on leaf01:

```
cumulus@leaf01:~$ net show bridge macs 00:03:00:33:33:01
VLAN      Master  Interface  MAC                TunnelDest
State  Flags          LastSeen
--------  ------  ---------  -----------------  ----------
-----  -------------  --------
13        bridge  vni-13
00:03:00:33:33:01                    offload        00:01:49
untagged          vni-13     00:03:00:33:33:01
10.0.0.101         self, offload  00:01:49
```

Back up and Restore

You can back up the current configuration on a switch and restore the configuration on the **same switch** or on another Cumulus Linux switch of the **same type and release**. The backup is a compressed tar file that includes all configuration files installed by Debian packages and marked as configuration files. In addition, the backup contains files in the `/etc` directory that are not installed by a Debian package but are modified when you install a new image or enable/disable certain services (such as the Cumulus license file).

Cumulus Linux automatically creates a backup of the configuration files on the switch after you install the Cumulus Linux image, in case you want to return to the initial switch configuration. NCLU automatically creates a backup of the configuration files when you run the `net commit` command and restores a previous configuration when you run the `net rollback` command.

Back up Configuration Files

To back up the current configuration files on the switch, run the `config-backup` command:

```
cumulus@switch:~$ sudo config-backup
```

If you run this command without any options, Cumulus Linux creates a backup of the current configuration and stores the backup file in the `/var/lib/config-backup/backups` directory. The filename includes the date and time you run the backup, and the switch name; for example, `config_backup-2019-04-23-21.30.47_leaf01`. You can restore the backup with the `config-restore` command, described below.

The switch can store up to 30 *non-permanent* backup files (or can allocate a maximum of 25 MB of disc space) in addition to the permanent backup files (see the `-p` option below). When this limit is reached, Cumulus Linux keeps the oldest and the newest backup files, then starts removing the second oldest file up to the second newest file.

> **NOTE**
>
> Cumulus Linux recommends you copy the backup file off the switch after backup is complete.

The `config-backup` command includes the following options:

Option	Description
`-h`	Displays this list of command options.
`-d`	Enables debugging output, which shows status messages during the backup process.

Option	Description
`-D <description>`	Adds a description, which is shown in the archive file list when you run the `config-restore -l` command.
`-p`	Adds -perm to the end of the backup filename to mark it as permanent. For example, `config_backup-2019-04-23-21.30.47_leaf01-pe` Be careful when using this option. Permanent backup files are not removed.
`-q`	Runs the command in quiet mode. No status messages are shown, only errors.
`-t <type>`	Specifies the type of configuration, which is shown in the archive file list when you run the `config-restore -l` command. You can provide any short text. For example, you can specify `pre`, `post`, or `pre-restore`.
`-v`	Enables verbose mode to show messages during the backup process.
`-X <pattern>`	Excludes certain files that match a specified pattern. For example, to exclude all backup files ending with a tilde (~), use the

Option	Description
	`-X .*~$` option.

config-backup Command Examples

The following command example creates a backup file in debugging mode and provides the description `myconfig`, which shows in the backup archive list.

```
cumulus@switch:~$ sudo config-backup -d -D myconfig
```

The following command example creates a backup file in quiet mode and excludes files that end in a tilde (~).

```
cumulus@switch:~$ sudo config-backup -q -X .*~$
```

The following command example creates a backup file in verbose mode and marks the file as permanent.

```
cumulus@switch:~$ sudo config-backup -pv
```

Restore Backup Files

You can restore a backup to the same switch or to a different switch. When restoring to a different switch, the switch must be of the **same type and release**. For example, you can restore a backup from a Broadcom Trident3 switch to a Broadcom Trident3 switch; however, you cannot restore a backup from a Broadcom Trident3 switch to a Mellanox Spectrum or to a Broadcom Tomahawk2 switch.

To restore a backup file, run the `config-restore` command with a specific filename (`-b <filename>`), file number (`-n <number>`), *or* the `-N` option, which restores the most recent backup file.

You can run the `config-restore -l` command to list the archived backup files by filename and number (see config-restore Command Examples below).

```
cumulus@switch:~$ sudo config-restore -b
config_backup-2019-04-23-21.30.47_leaf01
cumulus@switch:~$ sudo config-restore -n 10
cumulus@switch:~$ sudo config-restore -N
```

After the backup file is restored successfully, you are prompted to restart any affected services or reboot the switch if necessary.

Cumulus Linux reports any issues encountered during restore and prompts

you to continue or stop.

> **NOTE**
>
> - The `config-restore` command *requires* a filename, file number, or the most recent file option (`-N`).
> - You can only run one `config-backup` or `config-restore` command instance at the same time.

The `config-restore` command includes the following options:

Option	Description
`-h`	Displays this list of command options.
`-a <directory>`	Restores the backup to the directory specified.
`-B`	Runs *no* backup before restoring the configuration. If you do *not* specify this option, Cumulus Linux runs a backup to save the current configuration before the restore so that you can do a rollback if needed.
`-b <filename>`	Specifies the name of the backup file you want to restore

Option	Description
	(shown by `-l`).
`-D`	Shows the differences between the current configuration and the configuration in the backup file.
`-d`	Displays debugging output, which provides status messages during the restore process.
`-f`	Forces the restore; does not prompt for confirmations.
`-F <filename>`	Shows differences for only this file (used with -D).
`-i`	Displays information about the current backup file.
`-L`	Lists the configuration files in the backup file.
`-l`	Lists all backup files archived on the switch and includes the file number, type, and description.
`-N`	Restores the newest (most recent) backup file.
`-n <number>`	Specifies the backup file by number (shown by `-l`).
`-q`	Runs the command in quiet mode. No status messages are

Option	Description
	displayed, only errors.
`-T`	Runs the command in test mode; does not restore the configuration but shows what would be restored.
`-v`	Enables verbose mode to display status messages during restore.

config-restore Command Examples

The following command example lists the backup files available on the switch. The list includes the file number (#), type, description, and filename. Type is the text specified with the `config-backup -t` option.

```
cumulus@switch:~$ sudo config-restore -l
# Type       Description               Name
1 Initial    First system boot
config_backup-2019-04-23-00.42.11_cumulus-perm
2 Initial    First system boot
config_backup-2019-04-23-00.47.43_cumulus-perm
3 Initial    First system boot
config_backup-2019-04-23-18.12.26_cumulus-perm
4 pre nclu "net commit" (user cumulus)
config_backup-2019-04-23-19.55.13_leaf01
```

```
5 post-4     nclu "net commit" (user cumulus)
config_backup-2019-04-23-19.55.26_leaf01
6            config_backup-2019-04-23-21.20.41_leaf01
7            config_backup-2019-04-23-21.30.47_leaf01-perm
...
```

The following command example runs in verbose mode to restore the backup file `config_backup-2019-04-23-21.30.47_leaf01`.

```
cumulus@switch:~$ sudo config-restore -v -b
config_backup-2019-04-23-21.30.47_leaf01
```

The following command example runs test mode to restore the most recent backup file (no configuration is actually restored).

```
cumulus@switch:~$ sudo config-restore -T -N
```

The following command example lists the files in the most recent backup file.

```
cumulus@switch:~$ sudo config-restore -L -N
```

Adding and Updating Packages

You use the Advanced Packaging Tool (`apt`) to manage additional applications (in the form of packages) and to install the latest updates.

WARNING

Updating, upgrading, and installing packages with `apt` causes disruptions to network services:

- Upgrading a package might result in services being restarted or stopped as part of the upgrade process.
- Installing a package might disrupt core services by changing core service dependency packages. In some cases, installing new packages might also upgrade additional existing packages due to dependencies.

If services are stopped, you might need to reboot the switch for those services to restart.

Update the Package Cache

To work properly, `apt` relies on a local cache listing of the available packages. You must populate the cache initially, then periodically update it with `sudo -E apt-get update`:

```
cumulus@switch:~$ sudo -E apt-get update
Get:1 http://apt.cumulusnetworks.com CumulusLinux-4-latest
InRelease [7,624 B]
Get:2 http://apt.cumulusnetworks.com CumulusLinux-4-security-
updates-latest InRelease [7,555 B]
Get:3 http://apt.cumulusnetworks.com CumulusLinux-4-latest-
updates InRelease [7,660 B]
Get:4 http://apt.cumulusnetworks.com CumulusLinux-4-latest/
cumulus Sources [20 B]
Get:5 http://apt.cumulusnetworks.com CumulusLinux-4-latest/
upstream Sources [20 B]
Get:6 http://apt.cumulusnetworks.com CumulusLinux-4-latest/
cumulus amd64 Packages [38.4 kB]
Get:7 http://apt.cumulusnetworks.com CumulusLinux-4--latest/
upstream amd64 Packages [445 kB]
Get:8 http://apt.cumulusnetworks.com CumulusLinux-4-security-
updates-latest/cumulus Sources [20 B]
Get:9 http://apt.cumulusnetworks.com CumulusLinux-4-security-
```

```
updates-latest/upstream Sources [11.8 kB]
Get:10 http://apt.cumulusnetworks.com CumulusLinux-4-security-
updates-latest/cumulus amd64 Packages [20 B]
Get:11 http://apt.cumulusnetworks.com CumulusLinux-4-security-
updates-latest/upstream amd64 Packages [8,941 B]
Get:12 http://apt.cumulusnetworks.com CumulusLinux-4-updates-
latest/cumulus Sources [20 B]
Get:13 http://apt.cumulusnetworks.com CumulusLinux-4-updates-
latest/upstream Sources [776 B]
Get:14 http://apt.cumulusnetworks.com CumulusLinux-4-updates-
latest/cumulus amd64 Packages [38.4 kB]
Get:15 http://apt.cumulusnetworks.com CumulusLinux-4-updates-
latest/upstream amd64 Packages [444 kB]
Ign http://apt.cumulusnetworks.com CumulusLinux-4-latest/
cumulus Translation-en_US
Ign http://apt.cumulusnetworks.com CumulusLinux-4-latest/
cumulus Translation-en
Ign http://apt.cumulusnetworks.com CumulusLinux-4-latest/
upstream Translation-en_US
Ign http://apt.cumulusnetworks.com CumulusLinux-4-latest/
upstream Translation-en
Ign http://apt.cumulusnetworks.com CumulusLinux-4-security-
updates-latest/cumulus Translation-en_US
Ign http://apt.cumulusnetworks.com CumulusLinux-4-security-
```

```
updates-latest/cumulus Translation-en
Ign http://apt.cumulusnetworks.com CumulusLinux-4-security-
updates-latest/upstream Translation-en_US
Ign http://apt.cumulusnetworks.com CumulusLinux-4-security-
updates-latest/upstream Translation-en
Ign http://apt.cumulusnetworks.com CumulusLinux-4-updates-
latest/cumulus Translation-en_US
Ign http://apt.cumulusnetworks.com CumulusLinux-4-updates-
latest/cumulus Translation-en
Ign http://apt.cumulusnetworks.com CumulusLinux-4-updates-
latest/upstream Translation-en_US
Ign http://apt.cumulusnetworks.com CumulusLinux-4-updates-
latest/upstream Translation-en
Fetched 1,011 kB in 1s (797 kB/s)
Reading package lists... Done
```

TIP

Use the `-E` option with `sudo` whenever you run any `apt-get` command. This option preserves your environment variables (such as HTTP proxies) before you install new packages or upgrade your distribution.

List Available Packages

After the cache is populated, use the `apt-cache` command to search the cache and find the packages in which you are interested or to get information about an available package.

Here are examples of the `search` and `show` sub-commands:

```
cumulus@switch:~$ apt-cache search tcp
collectd-core - statistics collection and monitoring daemon
(core system)
fakeroot - tool for simulating superuser privileges
iperf - Internet Protocol bandwidth measuring tool
iptraf-ng - Next Generation Interactive Colorful IP LAN Monitor
libfakeroot - tool for simulating superuser privileges - shared
libraries
libfstrm0 - Frame Streams (fstrm) library
libibverbs1 - Library for direct userspace use of RDMA
(InfiniBand/iWARP)
libnginx-mod-stream - Stream module for Nginx
```

```
libqt4-network - Qt 4 network module
librtr-dev - Small extensible RPKI-RTR-Client C library -
development files
librtr0 - Small extensible RPKI-RTR-Client C library
libwiretap8 - network packet capture library -- shared library
libwrap0 - Wietse Venema's TCP wrappers library
libwrap0-dev - Wietse Venema's TCP wrappers library,
development files
netbase - Basic TCP/IP networking system
nmap-common - Architecture independent files for nmap
nuttcp - network performance measurement tool
openssh-client - secure shell (SSH) client, for secure access
to remote machines
openssh-server - secure shell (SSH) server, for secure access
from remote machines
openssh-sftp-server - secure shell (SSH) sftp server module,
for SFTP access from remote machines
python-dpkt - Python 2 packet creation / parsing module for
basic TCP/IP protocols
rsyslog - reliable system and kernel logging daemon
socat - multipurpose relay for bidirectional data transfer
tcpdump - command-line network traffic analyzer
```

```
cumulus@switch:~$ apt-cache show tcpdump
Package: tcpdump
Version: 4.9.3-1~deb10u1
Installed-Size: 1109
Maintainer: Romain Francoise <rfrancoise@debian.org>
Architecture: amd64
Replaces: apparmor-profiles-extra (<< 1.12~)
Depends: libc6 (>= 2.14), libpcap0.8 (>= 1.5.1), libssl1.1 (>=
1.1.0)
Suggests: apparmor (>= 2.3)
Breaks: apparmor-profiles-extra (<< 1.12~)
Size: 400060
SHA256:
3a63be16f96004bdf8848056f2621fbd863fadc0baf44bdcbc5d75dd98331fd3
SHA1: 2ab9f0d2673f49da466f5164ecec8836350aed42
MD5sum: 603baaf914de63f62a9f8055709257f3
Description: command-line network traffic analyzer
 This program allows you to dump the traffic on a network.
tcpdump
 is able to examine IPv4, ICMPv4, IPv6, ICMPv6, UDP, TCP, SNMP,
AFS
 BGP, RIP, PIM, DVMRP, IGMP, SMB, OSPF, NFS and many other
packet
 types.
```

```
 .
 It can be used to print out the headers of packets on a network
 interface, filter packets that match a certain expression. You
can
 use this tool to track down network problems, to detect attacks
 or to monitor network activities.
Description-md5: f01841bfda357d116d7ff7b7a47e8782
Homepage: http://www.tcpdump.org/
Multi-Arch: foreign
Section: net
Priority: optional
Filename: pool/upstream/t/tcpdump/
tcpdump_4.9.3-1~deb10u1_amd64.deb
```

NOTE

The search commands look for the search terms not only in the package name but in other parts of the package information; the search matches on more packages than you might expect.

List Packages Installed on the System

`apt-cache` command shows information about all the packages available in the repository. To see which packages are actually installed on your system

with their versions, run the following commands.

NCLU Commands | Linux Commands

Run the `net show package version` command:

```
cumulus@switch:~$ net show package version
Package                              Installed Version(s)
---------------------------------
-------------------------------------------------------------------------
acpi                                 1.7-1.1
acpi-support-base                    0.142-8
acpid                                1:2.0.31-1
adduser                              3.118
apt                                  1.8.2
arping                               2.19-6
arptables                            0.0.4+snapshot20181021-4
atftp                                0.7.git20120829-3.1
atftpd                               0.7.git20120829-3.1
...
```

NOTE

The apps repository was removed in Cumulus Linux 4.0.0.

Show the Version of a Package

To show the version of a specific package installed on the system:

NCLU Commands | Linux Commands

Run the `net show package version <package>` command. For example, the following command shows which version of the `vrf` package is installed on the system:

```
cumulus@switch:~$ net show package version vrf
1.0-cl4u2
```

Upgrade Packages

To upgrade all the packages installed on the system to their latest versions, run the following commands:

```
cumulus@switch:~$ sudo -E apt-get update
```

```
cumulus@switch:~$ sudo -E apt-get upgrade
```

A list of packages that will be upgraded is displayed and you are prompted to continue.

The above commands upgrade all installed versions with their latest versions but do not install any new packages.

Add New Packages

To add a new package, first ensure the package is not already installed on the system:

```
cumulus@switch:~$ dpkg -l | grep <name of package>
```

- If the package is installed already, you can update the package from the Cumulus Linux repository as part of the package upgrade process, which upgrades all packages on the system. See Upgrade Packages above.
- If the package is *not* already installed, add it by running `sudo -E apt-get install <name of package>`. This retrieves the package from the Cumulus Linux repository and installs it on your system together with any other packages on which this package might depend. The following example adds the `tcpreplay` package to the system:

```
cumulus@switch:~$ sudo -E apt-get update
cumulus@switch:~$ sudo -E apt-get install tcpreplay
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
tcpreplay
0 upgraded, 1 newly installed, 0 to remove and 1 not upgraded.
Need to get 436 kB of archives.
After this operation, 1008 kB of additional disk space will be
used
...
```

You can install several packages at the same time:

```
cumulus@switch:~$ sudo -E apt-get install <package 1> <package
2> <package 3>
```

TIP

In some cases, installing a new package might also upgrade

additional existing packages due to dependencies. To view these additional packages before you install, run the `apt-get install --dry-run` command.

Add Packages from Another Repository

As shipped, Cumulus Linux searches the Cumulus Linux repository for available packages. You can add additional repositories to search by adding them to the list of sources that `apt-get` consults. See `man sources.list` for more information.

TIP

NVIDIA has added features or made bug fixes to certain packages; do not replace these packages with versions from other repositories. Cumulus Linux is configured to ensure that the packages from the Cumulus Linux repository are always preferred over packages from other repositories.

If you want to install packages that are not in the Cumulus Linux repository, the procedure is the same as above, but with one additional step.

> **NOTE**
>
> Packages that are not part of the Cumulus Linux Repository are not typically tested and might not be supported by Cumulus Linux Technical Support.

Installing packages outside of the Cumulus Linux repository requires the use of `sudo -E apt-get`; however, depending on the package, you can use `easy-install` and other commands.

To install a new package, complete the following steps:

1. Run the `dpkg` command to ensure that the package is not already installed on the system:

```
cumulus@switch:~$ dpkg -l | grep <name of package>
```

2. If the package is installed already, ensure it is the version you need. If it is an older version, update the package from the Cumulus Linux repository:

```
cumulus@switch:~$ sudo -E apt-get update
cumulus@switch:~$ sudo -E apt-get install <name of package>
cumulus@switch:~$ sudo -E apt-get upgrade
```

3. If the package is not on the system, the package source location is most likely **not** in the `/etc/apt/sources.list` file. If the source for the new package is **not** in `sources.list`, edit and add the appropriate source to the file. For example, add the following if you want a package from the Debian repository that is **not** in the Cumulus Linux repository:

```
deb http://http.us.debian.org/debian buster main
deb http://security.debian.org/ buster/updates main
```

   Otherwise, the repository might be listed in `/etc/apt/sources.list` but is commented out. To uncomment the repository, remove the `#` at the start of the line, then save the file.

4. Run `sudo -E apt-get update`, then install the package and upgrade:

```
cumulus@switch:~$ sudo -E apt-get update
cumulus@switch:~$ sudo -E apt-get install <name of package>
```

```
cumulus@switch:~$ sudo -E apt-get upgrade
```

Add Packages from the Cumulus Linux Local Archive

Cumulus Linux contains a local archive embedded in the Cumulus Linux image. This archive contains the packages needed to install `ifplugd`, LDAP, RADIUS or TACACS+ without needing a network connection.

The archive is called `cumulus-local-apt-archive` and is referenced in the `/etc/apt/cumulus-local-apt-archive-sources.list` file. It contains the following packages:

- audisp-tacplus
- ifplugd
- libdaemon0
- libnss-ldapd
- libnss-mapuser
- libnss-tacplus
- libpam-ldapd
- libpam-radius-auth
- libpam-tacplus
- libtac2
- libtacplus-map1
- nslcd

You add these packages normally with `apt-get update && apt-get install`, as described above.

Related Information

- Debian GNU/Linux FAQ, Ch 8 Package management tools
- man pages for `apt-get`, `dpkg`, `sources.list`, `apt_preferences`

Considerations

At this time, you cannot directly browse the contents of the apt.cumulusnetworks.com repository using HTTP.

Zero Touch Provisioning - ZTP

Zero touch provisioning (ZTP) enables you to deploy network devices quickly in large-scale environments. On first boot, Cumulus Linux invokes ZTP, which executes the provisioning automation used to deploy the device for its intended role in the network.

The provisioning framework allows for a one-time, user-provided script to be executed. You can develop this script using a variety of automation tools and scripting languages, providing ample flexibility for you to design the provisioning scheme to meet your needs. You can also use it to add the switch to a configuration management (CM) platform such as Puppet, Chef, CFEngine or possibly a custom, proprietary tool.

While developing and testing the provisioning logic, you can use the `ztp` command in Cumulus Linux to manually invoke your provisioning script on a device.

ZTP in Cumulus Linux can occur automatically in one of the following ways, in this order:

1. Through a local file
2. Using a USB drive inserted into the switch (ZTP-USB)
3. Through DHCP

Each method is discussed in greater detail below.

Use a Local File

ZTP only looks once for a ZTP script on the local file system when the switch boots. ZTP searches for an install script that matches an ONIE-style waterfall in `/var/lib/cumulus/ztp`, looking for the most specific name first, and ending at the most generic:

- `'cumulus-ztp-' + architecture + '-' + vendor + '_' + model + '-r' + revision`
- `'cumulus-ztp-' + architecture + '-' + vendor + '_' + model`
- `'cumulus-ztp-' + vendor + '_' + model`
- `'cumulus-ztp-' + architecture`
- `'cumulus-ztp'`

For example:

```
cumulus-ztp-amd64-cel_pebble-rUNKNOWN
cumulus-ztp-amd64-cel_pebble
cumulus-ztp-cel_pebble
cumulus-ztp-amd64
cumulus-ztp
```

You can also trigger the ZTP process manually by running the `ztp --run <URL>` command, where the URL is the path to the ZTP script.

Use a USB Drive

> **NOTE**
>
> This feature has been tested only with *thumb* drives, not an actual external large USB hard drive.

If the `ztp` process does not discover a local script, it tries once to locate an inserted but unmounted USB drive. If it discovers one, it begins the ZTP process.

Cumulus Linux supports the use of a FAT32, FAT16, or VFAT-formatted USB drive as an installation source for ZTP scripts. You must plug in the USB drive **before** you power up the switch.

At minimum, the script must:

- Install the Cumulus Linux operating system and license.
- Copy over a basic configuration to the switch.
- Restart the switch or the relevant services to get `switchd` up and running with that configuration.

Follow these steps to perform ZTP using a USB drive:

1. Copy the Cumulus Linux license and installation image to the USB drive.
2. The `ztp` process searches the root filesystem of the newly mounted drive

for filenames matching an ONIE-style waterfall (see the patterns and examples above), looking for the most specific name first, and ending at the most generic.

3. The contents of the script are parsed to ensure it contains the `CUMULUS-AUTOPROVISIONING` flag (see example scripts).

> **NOTE**
>
> The USB drive is mounted to a temporary directory under `/tmp` (for example, `/tmp/tmpigGgjf/`). To reference files on the USB drive, use the environment variable `ZTP_USB_MOUNTPOINT` to refer to the USB root partition.

ZTP over DHCP

If the `ztp` process does not discover a local/ONIE script or applicable USB drive, it checks DHCP every ten seconds for up to five minutes for the presence of a ZTP URL specified in `/var/run/ztp.dhcp`. The URL can be any of HTTP, HTTPS, FTP or TFTP.

For ZTP using DHCP, provisioning initially takes place over the management network and is initiated through a DHCP hook. A DHCP option is used to specify a configuration script. This script is then requested from the Web server and executed locally on the switch.

The ZTP process over DHCP follows these steps:

1. The first time you boot Cumulus Linux, eth0 is configured for DHCP and makes a DHCP request.
2. The DHCP server offers a lease to the switch.
3. If option 239 is present in the response, the ZTP process starts.
4. The ZTP process requests the contents of the script from the URL, sending additional HTTP headers containing details about the switch.
5. The contents of the script are parsed to ensure it contains the `CUMULUS-AUTOPROVISIONING` flag (see example scripts).
6. If provisioning is necessary, the script executes locally on the switch with root privileges.
7. The return code of the script is examined. If it is 0, the provisioning state is marked as complete in the autoprovisioning configuration file.

Trigger ZTP over DHCP

If provisioning has not already occurred, it is possible to trigger the ZTP process over DHCP when eth0 is set to use DHCP and one of the following events occur:

- The switch boots.
- You plug a cable into or unplug a cable from the eth0 port.
- You disconnect, then reconnect the switch power cord.

You can also run the `ztp --run <URL>` command, where the `URL` is the path to the ZTP script.

Configure the DHCP Server

During the DHCP process over eth0, Cumulus Linux requests DHCP option

239. This option is used to specify the custom provisioning script.

For example, the `/etc/dhcp/dhcpd.conf` file for an ISC DHCP server looks like:

```
option cumulus-provision-url code 239 = text;


  subnet 192.0.2.0 netmask 255.255.255.0 {
  range 192.0.2.100 192.168.0.200;
  option cumulus-provision-url "http://192.0.2.1/demo.sh";
}
```

Additionally, you can specify the hostname of the switch with the `host-name` option:

```
subnet 192.168.0.0 netmask 255.255.255.0 {
  range 192.168.0.100 192.168.0.200;
  option cumulus-provision-url "http://192.0.2.1/demo.sh";
  host dc1-tor-sw1 { hardware ethernet 44:38:39:00:1a:6b; fixed-
address 192.168.0.101; option host-name "dc1-tor-sw1"; }
}
```

> **NOTE**
>
> Do not use an underscore (_) in the hostname; underscores are not permitted in hostnames.

Inspect HTTP Headers

The following HTTP headers are sent in the request to the webserver to retrieve the provisioning script:

```
Header                          Value                 Example
------                          -----                 -------
User-Agent
CumulusLinux-AutoProvision/0.4
CUMULUS-ARCH                    CPU architecture      x86_64
CUMULUS-BUILD                                         4.1.0
CUMULUS-LICENSE-INSTALLED       Either 0 or 1         1
CUMULUS-MANUFACTURER                                  odm
CUMULUS-PRODUCTNAME                                   switch_model
CUMULUS-SERIAL                                        XYZ123004
CUMULUS-BASE-MAC
44:38:39:FF:40:94
CUMULUS-MGMT-MAC
```

```
44:38:39:FF:00:00
CUMULUS-VERSION                                 4.1.0
CUMULUS-PROV-COUNT                              0
CUMULUS-PROV-MAX                                32
```

Write ZTP Scripts

> **NOTE**
>
> Remember to include the following line in any of the supported scripts that you expect to run using the autoprovisioning framework.
>
> ```
> # CUMULUS-AUTOPROVISIONING
> ```
>
> This line is required somewhere in the script file for execution to occur.

The script must contain the `CUMULUS-AUTOPROVISIONING` flag. You can include this flag in a comment or remark; the flag does not need to be echoed or written to `stdout`.

You can write the script in any language currently supported by Cumulus

Linux, such as:

- Perl
- Python
- Ruby
- Shell

The script must return an exit code of 0 upon success, as this triggers the autoprovisioning process to be marked as complete in the autoprovisioning configuration file.

The following script installs Cumulus Linux and its license from a USB drive and applies a configuration:

```
#!/bin/bash
function error() {
  echo -e "\e[0;33mERROR: The ZTP script failed while running
the command $BASH_COMMAND at line $BASH_LINENO.\e[0m" >&2
  exit 1
}

# Log all output from this script
exec >> /var/log/autoprovision 2>&1
date "+%FT%T ztp starting script $0"

trap error ERR
```

```
#Add Debian Repositories
echo "deb http://http.us.debian.org/debian buster main" >> /etc/
apt/sources.list
echo "deb http://security.debian.org/ buster/updates main" >>
/etc/apt/sources.list

#Update Package Cache
apt-get update -y

#Load interface config from usb
cp ${ZTP_USB_MOUNTPOINT}/interfaces /etc/network/interfaces

#Load port config from usb
#   (if breakout cables are used for certain interfaces)
cp ${ZTP_USB_MOUNTPOINT}/ports.conf /etc/cumulus/ports.conf

#Install a License from usb and restart switchd
/usr/cumulus/bin/cl-license -i
${ZTP_USB_MOUNTPOINT}/license.txt && systemctl restart
switchd.service

#Reload interfaces to apply loaded config
ifreload -a
```

```
#Output state of interfaces
net show interface

# CUMULUS-AUTOPROVISIONING
exit 0
```

Several ZTP example scripts are available in the Cumulus GitHub repository.

Best Practices

ZTP scripts come in different forms and frequently perform many of the same tasks. As BASH is the most common language used for ZTP scripts, the following BASH snippets are provided to accelerate your ability to perform common tasks with robust error checking.

Set the Default Cumulus User Password

The default *cumulus* user account password is `cumulus`. When you log into Cumulus Linux for the first time, you must provide a new password for the *cumulus* account, then log back into the system. This password change at first login is **required** in Cumulus Linux 4.2 and later.

Add the following function to your ZTP script to change the default *cumulus* user account password to a clear-text password. The example changes the password `cumulus` to `MyP4$$word`.

```
function set_password(){
     # Unexpire the cumulus account
     passwd -x 99999 cumulus
     # Set the password
     echo 'cumulus:MyP4$$word' | chpasswd
}
set_password
```

If you have an insecure management network, set the password with an encrypted hash instead of a clear-text password. Using an encrypted hash is recommended.

- First, generate a sha-512 password hash with the following python commands. The example commands generate a sha-512 password hash for the password `MyP4$$word`.

```
user@host:~$ python3 -c "import crypt;
print(crypt.crypt('MyP4$$word',salt=crypt.mksalt()))"
$6$hs7OPmnrfvLNKfoZ$iB3hy5N6Vv6koqDmxixpTO61ej6VaoKGvs5E8p5zNo4tPec0KKqyQnrFMII3
```

- Then, add the following function to the ZTP script to change the default *cumulus* user account password:

```
function set_password(){
     # Unexpire the cumulus account
     passwd -x 99999 cumulus
     # Set the password
     usermod -p
'$6$hs7OPmnrfvLNKfoZ$iB3hy5N6Vv6koqDmxixpTO6lej6VaoKGvs5E8p5zNo4tPec0KKqyQnrFMI
cumulus
}
set_password
```

Install a License

Use the following function to include error checking for license file installation.

```
function install_license(){
     # Install license
     echo "$(date) INFO: Installing License..."
     echo $1 | /usr/cumulus/bin/cl-license -i
     return_code=$?
     if [ "$return_code" == "0" ]; then
         echo "$(date) INFO: License Installed."
     else
```

```
        echo "$(date) ERROR: License not installed. Return
code was: $return_code"
        /usr/cumulus/bin/cl-license
        exit 1
    fi
}
```

Test DNS Name Resolution

DNS names are frequently used in ZTP scripts. The `ping_until_reachable` function tests that each DNS name resolves into a reachable IP address. Call this function with each DNS target used in your script before you use the DNS name elsewhere in your script.

The following example shows how to call the `ping_until_reachable` function in the context of a larger task.

```
function ping_until_reachable(){
    last_code=1
    max_tries=30
    tries=0
    while [ "0" != "$last_code" ] && [ "$tries" -lt
"$max_tries" ]; do
```

```
        tries=$((tries+1))
        echo "$(date) INFO: ( Attempt $tries of $max_tries )
Pinging $1 Target Until Reachable."
        ping $1 -c2 &> /dev/null
        last_code=$?
            sleep 1
    done
    if [ "$tries" -eq "$max_tries" ] && [ "$last_code" -ne "0"
]; then
        echo "$(date) ERROR: Reached maximum number of attempts
to ping the target $1 ."
        exit 1
    fi
}
```

Check the Cumulus Linux Release

The following script segment demonstrates how to check which Cumulus Linux release is running currently and upgrades the node if the release is not the *target release*. If the release *is* the target release, normal ZTP tasks execute. This script calls the `ping_until_reachable` script (described above) to make sure the server holding the image server and the ZTP script is reachable.

```
function init_ztp(){
    #do normal ZTP tasks
}

CUMULUS_TARGET_RELEASE=3.5.3
CUMULUS_CURRENT_RELEASE=$(cat /etc/lsb-release  | grep RELEASE
| cut -d "=" -f2)
IMAGE_SERVER_HOSTNAME=webserver.example.com
IMAGE_SERVER= "http:// "$IMAGE_SERVER_HOSTNAME "/
"$CUMULUS_TARGET_RELEASE ".bin "
ZTP_URL= "http:// "$IMAGE_SERVER_HOSTNAME "/ztp.sh "

if [ "$CUMULUS_TARGET_RELEASE" != "$CUMULUS_CURRENT_RELEASE" ];
then
    ping_until_reachable $IMAGE_SERVER_HOSTNAME
    /usr/cumulus/bin/onie-install -fa -i $IMAGE_SERVER -z
$ZTP_URL && reboot
else
    init_ztp && reboot
fi
exit 0
```

Apply Management VRF Configuration

If you apply a management VRF in your script, either apply it last or reboot

instead. If you do *not* apply a management VRF last, you need to prepend any commands that require `eth0` to communicate out with `/usr/bin/ip vrf exec mgmt`; for example, `/usr/bin/ip vrf exec mgmt apt-get update -y`.

Perform Ansible Provisioning Callbacks

After initially configuring a node with ZTP, use Provisioning Callbacks to inform Ansible Tower or AWX that the node is ready for more detailed provisioning. The following example demonstrates how to use a provisioning callback:

```
/usr/bin/curl -H "Content-Type:application/json" -k -X POST --
data '{"host_config_key":"'somekey'"}' -u username:password
http://ansible.example.com/api/v2/job_templates/1111/callback/
```

Disable the DHCP Hostname Override Setting

Make sure to disable the DHCP hostname override setting in your script (NCLU does this automatically).

```
function set_hostname(){
    # Remove DHCP Setting of Hostname
    sed s/'SETHOSTNAME="yes"'/'SETHOSTNAME="no"'/g -i /etc/dhcp/
dhclient-exit-hooks.d/dhcp-sethostname
    hostnamectl set-hostname $1
```

```
}
```

NCLU in ZTP Scripts

> **NOTE**
>
> Not all aspects of NCLU are supported when running during ZTP. Use traditional Linux methods of providing configuration to the switch during ZTP.

When you use NCLU in ZTP scripts, add the following loop to make sure NCLU has time to start up before being called.

```
# Waiting for NCLU to finish starting up
last_code=1
while [ "1" == "$last_code" ]; do
    net show interface &> /dev/null
    last_code=$?
done

net add vrf mgmt
net add time zone Etc/UTC
```

```
net add time ntp server 192.168.0.254 iburst
net commit
```

Test ZTP Scripts

There are a few commands you can use to test and debug your ZTP scripts.

You can use verbose mode to debug your script and see where your script failed. Include the `-v` option when you run ZTP:

```
cumulus@switch:~$ sudo ztp -v -r http://192.0.2.1/demo.sh
Attempting to provision via ZTP Manual from http://192.0.2.1/
demo.sh

Broadcast message from root@dell-s6010-01 (ttyS0) (Tue May 10
22:44:17 2016):

ZTP: Attempting to provision via ZTP Manual from
http://192.0.2.1/demo.sh
ZTP Manual: URL response code 200
ZTP Manual: Found Marker CUMULUS-AUTOPROVISIONING
ZTP Manual: Executing http://192.0.2.1/demo.sh
error: ZTP Manual: Payload returned code 1
```

```
error: Script returned failure
```

To see if ZTP is enabled and to see results of the most recent execution, you can run the `ztp -s` command.

```
cumulus@switch:~$ ztp -s
ZTP INFO:

State               enabled
Version             1.0
Result              Script Failure
Date                Mon 20 May 2019 09:31:27 PM UTC
Method              ZTP DHCP
URL                 http://192.0.2.1/demo.sh
```

If ZTP runs when the switch boots and not manually, you can run the `systemctl -l status ztp.service` then `journalctl -l -u ztp.service` to see if any failures occur:

```
cumulus@switch:~$ sudo systemctl -l status ztp.service
● ztp.service - Cumulus Linux ZTP
```

```
    Loaded: loaded (/lib/systemd/system/ztp.service; enabled)
    Active: failed (Result: exit-code) since Wed 2016-05-11
16:38:45 UTC; 1min 47s ago
        Docs: man:ztp(8)
    Process: 400 ExecStart=/usr/sbin/ztp -b (code=exited,
status=1/FAILURE)
    Main PID: 400 (code=exited, status=1/FAILURE)

May 11 16:37:45 cumulus ztp[400]: ztp [400]: ZTP USB: Device
not found
May 11 16:38:45 dell-s6010-01 ztp[400]: ztp [400]: ZTP DHCP:
Looking for ZTP Script provided by DHCP
May 11 16:38:45 dell-s6010-01 ztp[400]: ztp [400]: Attempting
to provision via ZTP DHCP from http://192.0.2.1/demo.sh
May 11 16:38:45 dell-s6010-01 ztp[400]: ztp [400]: ZTP DHCP:
URL response code 200
May 11 16:38:45 dell-s6010-01 ztp[400]: ztp [400]: ZTP DHCP:
Found Marker CUMULUS-AUTOPROVISIONING
May 11 16:38:45 dell-s6010-01 ztp[400]: ztp [400]: ZTP DHCP:
Executing http://192.0.2.1/demo.sh
May 11 16:38:45 dell-s6010-01 ztp[400]: ztp [400]: ZTP DHCP:
Payload returned code 1
May 11 16:38:45 dell-s6010-01 ztp[400]: ztp [400]: Script
returned failure
```

```
May 11 16:38:45 dell-s6010-01 systemd[1]: ztp.service: main
process exited, code=exited, status=1/FAILURE
May 11 16:38:45 dell-s6010-01 systemd[1]: Unit ztp.service
entered failed state.
cumulus@switch:~$
cumulus@switch:~$ sudo journalctl -l -u ztp.service --no-pager
-- Logs begin at Wed 2016-05-11 16:37:42 UTC, end at Wed
2016-05-11 16:40:39 UTC. --
May 11 16:37:45 cumulus ztp[400]: ztp [400]: /var/lib/cumulus/
ztp: Sate Directory does not exist. Creating it...
May 11 16:37:45 cumulus ztp[400]: ztp [400]: /var/run/ztp.lock:
Lock File does not exist. Creating it...
May 11 16:37:45 cumulus ztp[400]: ztp [400]: /var/lib/cumulus/
ztp/ztp_state.log: State File does not exist. Creating it...
May 11 16:37:45 cumulus ztp[400]: ztp [400]: ZTP LOCAL: Looking
for ZTP local Script
May 11 16:37:45 cumulus ztp[400]: ztp [400]: ZTP LOCAL:
Waterfall search for /var/lib/cumulus/ztp/cumulus-ztp-
x86_64-dell_s6010_s1220-rUNKNOWN
May 11 16:37:45 cumulus ztp[400]: ztp [400]: ZTP LOCAL:
Waterfall search for /var/lib/cumulus/ztp/cumulus-ztp-
x86_64-dell_s6010_s1220
May 11 16:37:45 cumulus ztp[400]: ztp [400]: ZTP LOCAL:
Waterfall search for /var/lib/cumulus/ztp/cumulus-ztp-
```

```
x86_64-dell
May 11 16:37:45 cumulus ztp[400]: ztp [400]: ZTP LOCAL:
Waterfall search for /var/lib/cumulus/ztp/cumulus-ztp-x86_64
May 11 16:37:45 cumulus ztp[400]: ztp [400]: ZTP LOCAL:
Waterfall search for /var/lib/cumulus/ztp/cumulus-ztp
May 11 16:37:45 cumulus ztp[400]: ztp [400]: ZTP USB: Looking
for unmounted USB devices
May 11 16:37:45 cumulus ztp[400]: ztp [400]: ZTP USB: Parsing
partitions
May 11 16:37:45 cumulus ztp[400]: ztp [400]: ZTP USB: Device
not found
May 11 16:38:45 dell-s6010-01 ztp[400]: ztp [400]: ZTP DHCP:
Looking for ZTP Script provided by DHCP
May 11 16:38:45 dell-s6010-01 ztp[400]: ztp [400]: Attempting
to provision via ZTP DHCP from http://192.0.2.1/demo.sh
May 11 16:38:45 dell-s6010-01 ztp[400]: ztp [400]: ZTP DHCP:
URL response code 200
May 11 16:38:45 dell-s6010-01 ztp[400]: ztp [400]: ZTP DHCP:
Found Marker CUMULUS-AUTOPROVISIONING
May 11 16:38:45 dell-s6010-01 ztp[400]: ztp [400]: ZTP DHCP:
Executing http://192.0.2.1/demo.sh
May 11 16:38:45 dell-s6010-01 ztp[400]: ztp [400]: ZTP DHCP:
Payload returned code 1
May 11 16:38:45 dell-s6010-01 ztp[400]: ztp [400]: Script
```

```
returned failure
May 11 16:38:45 dell-s6010-01 systemd[1]: ztp.service: main
process exited, code=exited, status=1/FAILURE
May 11 16:38:45 dell-s6010-01 systemd[1]: Unit ztp.service
entered failed state.
```

Instead of running `journalctl`, you can see the log history by running:

```
cumulus@switch:~$ cat /var/log/syslog | grep ztp
2016-05-11T16:37:45.132583+00:00 cumulus ztp [400]: /var/lib/
cumulus/ztp: State Directory does not exist. Creating it...
2016-05-11T16:37:45.134081+00:00 cumulus ztp [400]: /var/run/
ztp.lock: Lock File does not exist. Creating it...
2016-05-11T16:37:45.135360+00:00 cumulus ztp [400]: /var/lib/
cumulus/ztp/ztp_state.log: State File does not exist. Creating
it...
2016-05-11T16:37:45.185598+00:00 cumulus ztp [400]: ZTP LOCAL:
Looking for ZTP local Script
2016-05-11T16:37:45.485084+00:00 cumulus ztp [400]: ZTP LOCAL:
Waterfall search for /var/lib/cumulus/ztp/cumulus-ztp-
x86_64-dell_s6010_s1220-rUNKNOWN
2016-05-11T16:37:45.486394+00:00 cumulus ztp [400]: ZTP LOCAL:
Waterfall search for /var/lib/cumulus/ztp/cumulus-ztp-
```

```
x86_64-dell_s6010_s1220
2016-05-11T16:37:45.488385+00:00 cumulus ztp [400]: ZTP LOCAL:
Waterfall search for /var/lib/cumulus/ztp/cumulus-ztp-
x86_64-dell
2016-05-11T16:37:45.489665+00:00 cumulus ztp [400]: ZTP LOCAL:
Waterfall search for /var/lib/cumulus/ztp/cumulus-ztp-x86_64
2016-05-11T16:37:45.490854+00:00 cumulus ztp [400]: ZTP LOCAL:
Waterfall search for /var/lib/cumulus/ztp/cumulus-ztp
2016-05-11T16:37:45.492296+00:00 cumulus ztp [400]: ZTP USB:
Looking for unmounted USB devices
2016-05-11T16:37:45.493525+00:00 cumulus ztp [400]: ZTP USB:
Parsing partitions
2016-05-11T16:37:45.636422+00:00 cumulus ztp [400]: ZTP USB:
Device not found
2016-05-11T16:38:43.372857+00:00 cumulus ztp [1805]: Found ZTP
DHCP Request
2016-05-11T16:38:45.696562+00:00 cumulus ztp [400]: ZTP DHCP:
Looking for ZTP Script provided by DHCP
2016-05-11T16:38:45.698598+00:00 cumulus ztp [400]: Attempting
to provision via ZTP DHCP from http://192.0.2.1/demo.sh
2016-05-11T16:38:45.816275+00:00 cumulus ztp [400]: ZTP DHCP:
URL response code 200
2016-05-11T16:38:45.817446+00:00 cumulus ztp [400]: ZTP DHCP:
Found Marker CUMULUS-AUTOPROVISIONING
```

```
2016-05-11T16:38:45.818402+00:00 cumulus ztp [400]: ZTP DHCP:
Executing http://192.0.2.1/demo.sh
2016-05-11T16:38:45.834240+00:00 cumulus ztp [400]: ZTP DHCP:
Payload returned code 1
2016-05-11T16:38:45.835488+00:00 cumulus ztp [400]: Script
returned failure
2016-05-11T16:38:45.876334+00:00 cumulus systemd[1]:
ztp.service: main process exited, code=exited, status=1/FAILURE
2016-05-11T16:38:45.879410+00:00 cumulus systemd[1]: Unit
ztp.service entered failed state.
```

If you see that the issue is a script failure, you can modify the script and then run ZTP manually using `ztp -v -r <URL/path to that script>`, as above.

```
cumulus@switch:~$ sudo ztp -v -r http://192.0.2.1/demo.sh
Attempting to provision via ZTP Manual from http://192.0.2.1/
demo.sh

Broadcast message from root@dell-s6010-01 (ttyS0) (Tue May 10
22:44:17 2019):

ZTP: Attempting to provision via ZTP Manual from
```

```
http://192.0.2.1/demo.sh
ZTP Manual: URL response code 200
ZTP Manual: Found Marker CUMULUS-AUTOPROVISIONING
ZTP Manual: Executing http://192.0.2.1/demo.sh
error: ZTP Manual: Payload returned code 1
error: Script returned failure
cumulus@switch:~$ sudo ztp -s
State         enabled
Version       1.0
Result        Script Failure
Date          Mon 20 May 2019 09:31:27 PM UTC
Method        ZTP Manual
URL           http://192.0.2.1/demo.sh
```

Use the following command to check `syslog` for information about ZTP:

```
cumulus@switch:~$ sudo grep -i ztp /var/log/syslog
```

Common ZTP Script Errors

Could not find referenced script/interpreter in downloaded payload

```
cumulus@leaf01:~$ sudo cat /var/log/syslog | grep ztp
2018-04-24T15:06:08.887041+00:00 leaf01 ztp [13404]: Attempting
to provision via ZTP Manual from http://192.168.0.254/
ztp_oob_windows.sh
2018-04-24T15:06:09.106633+00:00 leaf01 ztp [13404]: ZTP
Manual: URL response code 200
2018-04-24T15:06:09.107327+00:00 leaf01 ztp [13404]: ZTP
Manual: Found Marker CUMULUS-AUTOPROVISIONING
2018-04-24T15:06:09.107635+00:00 leaf01 ztp [13404]: ZTP
Manual: Executing http://192.168.0.254/ztp_oob_windows.sh
2018-04-24T15:06:09.132651+00:00 leaf01 ztp [13404]: ZTP
Manual: Could not find referenced script/interpreter in
downloaded payload.
2018-04-24T15:06:14.135521+00:00 leaf01 ztp [13404]: ZTP
Manual: Retrying
2018-04-24T15:06:14.138915+00:00 leaf01 ztp [13404]: ZTP
Manual: URL response code 200
2018-04-24T15:06:14.139162+00:00 leaf01 ztp [13404]: ZTP
Manual: Found Marker CUMULUS-AUTOPROVISIONING
```

```
2018-04-24T15:06:14.139448+00:00 leaf01 ztp [13404]: ZTP
Manual: Executing http://192.168.0.254/ztp_oob_windows.sh
2018-04-24T15:06:14.143261+00:00 leaf01 ztp [13404]: ZTP
Manual: Could not find referenced script/interpreter in
downloaded payload.
2018-04-24T15:06:24.147580+00:00 leaf01 ztp [13404]: ZTP
Manual: Retrying
2018-04-24T15:06:24.150945+00:00 leaf01 ztp [13404]: ZTP
Manual: URL response code 200
2018-04-24T15:06:24.151177+00:00 leaf01 ztp [13404]: ZTP
Manual: Found Marker CUMULUS-AUTOPROVISIONING
2018-04-24T15:06:24.151374+00:00 leaf01 ztp [13404]: ZTP
Manual: Executing http://192.168.0.254/ztp_oob_windows.sh
2018-04-24T15:06:24.155026+00:00 leaf01 ztp [13404]: ZTP
Manual: Could not find referenced script/interpreter in
downloaded payload.
2018-04-24T15:06:39.164957+00:00 leaf01 ztp [13404]: ZTP
Manual: Retrying
2018-04-24T15:06:39.165425+00:00 leaf01 ztp [13404]: Script
returned failure
2018-04-24T15:06:39.175959+00:00 leaf01 ztp [13404]: ZTP script
failed. Exiting...
```

Errors in syslog for ZTP like those shown above often occur if the script is

created (or edited as some point) on a Windows machine. Check to make sure that the `\r\n` characters are *not* present in the end-of-line encodings.

Use the `cat -v ztp.sh` command to view the contents of the script and search for any hidden characters.

```
root@oob-mgmt-server:/var/www/html# cat -v ./ztp_oob_windows.sh
#!/bin/bash^M
^M
####################^M
#   ZTP Script^M
####################^M
^M
/usr/cumulus/bin/cl-license -i http://192.168.0.254/
license.txt^M
^M
# Clean method of performing a Reboot^M
nohup bash -c 'sleep 2; shutdown now -r "Rebooting to Complete
ZTP"' &^M
^M
exit 0^M
^M
# The line below is required to be a valid ZTP script^M
#CUMULUS-AUTOPROVISIONING^M
root@oob-mgmt-server:/var/www/html#
```

The `^M` characters in the output of your ZTP script, as shown above, indicate the presence of Windows end-of-line encodings that you need to remove.

Use the translate (`tr`) command on any Linux system to remove the `'\r'` characters from the file.

```
root@oob-mgmt-server:/var/www/html# tr -d '\r' <
ztp_oob_windows.sh > ztp_oob_unix.sh
root@oob-mgmt-server:/var/www/html# cat -v ./ztp_oob_unix.sh
#!/bin/bash
###################
#   ZTP Script
###################
/usr/cumulus/bin/cl-license -i http://192.168.0.254/license.txt
# Clean method of performing a Reboot
nohup bash -c 'sleep 2; shutdown now -r "Rebooting to Complete
ZTP"' &
exit 0
# The line below is required to be a valid ZTP script
#CUMULUS-AUTOPROVISIONING
root@oob-mgmt-server:/var/www/html#
```

Manually Use the ztp Command

To enable ZTP, use the `-e` option:

```
cumulus@switch:~$ sudo ztp -e
```

NOTE

Enabling ZTP means that ZTP tries to run the next time the switch boots. However, if ZTP already ran on a previous boot up or if a manual configuration has been found, ZTP will just exit without trying to look for any script.

ZTP checks for these manual configurations during bootup:

- Password changes
- Users and groups changes
- Packages changes
- Interfaces changes
- The presence of an installed license

When the switch is booted for the very first time, ZTP records the state of important files that are most likely going to be modified after that the switch is configured. If ZTP is still enabled after a reboot, ZTP compares the recorded state to the current state of these files. If they do not match, ZTP considers that the switch has already been provisioned and exits. These files are only erased after a reset.

To reset ZTP to its original state, use the `-R` option. This removes the `ztp` directory and ZTP runs the next time the switch reboots.

```
cumulus@switch:~$ sudo ztp -R
```

To disable ZTP, use the `-d` option:

```
cumulus@switch:~$ sudo ztp -d
```

To force provisioning to occur and ignore the status listed in the configuration file, use the `-r` option:

```
cumulus@switch:~$ sudo ztp -r cumulus-ztp.sh
```

To see the current ZTP state, use the `-s` option:

```
cumulus@switch:~$ sudo ztp -s
ZTP INFO:
State           disabled
Version         1.0
Result          success
```

```
Date            Mon May 20 21:51:04 2019 UTC
Method          Switch manually configured
URL             None
```

You can run the NCLU `net show system ztp script` or `net show system ztp json` command to see the current `ztp` state.

Considerations

- During the development of a provisioning script, the switch might need to be rebooted.
- You can use the Cumulus Linux `onie-select -i` command to cause the switch to reprovision itself and install a network operating system again using ONIE.

System Configuration

This section describes how to configure your Cumulus Linux switch. You can set the date and time, configure authentication, authorization, and accounting and configure access control lists (ACLs), which control the traffic entering your network.

This section also describes the services and daemons that Cumulus Linux uses, and describes how to configure `switchd`, the daemon at the heart of Cumulus Linux.

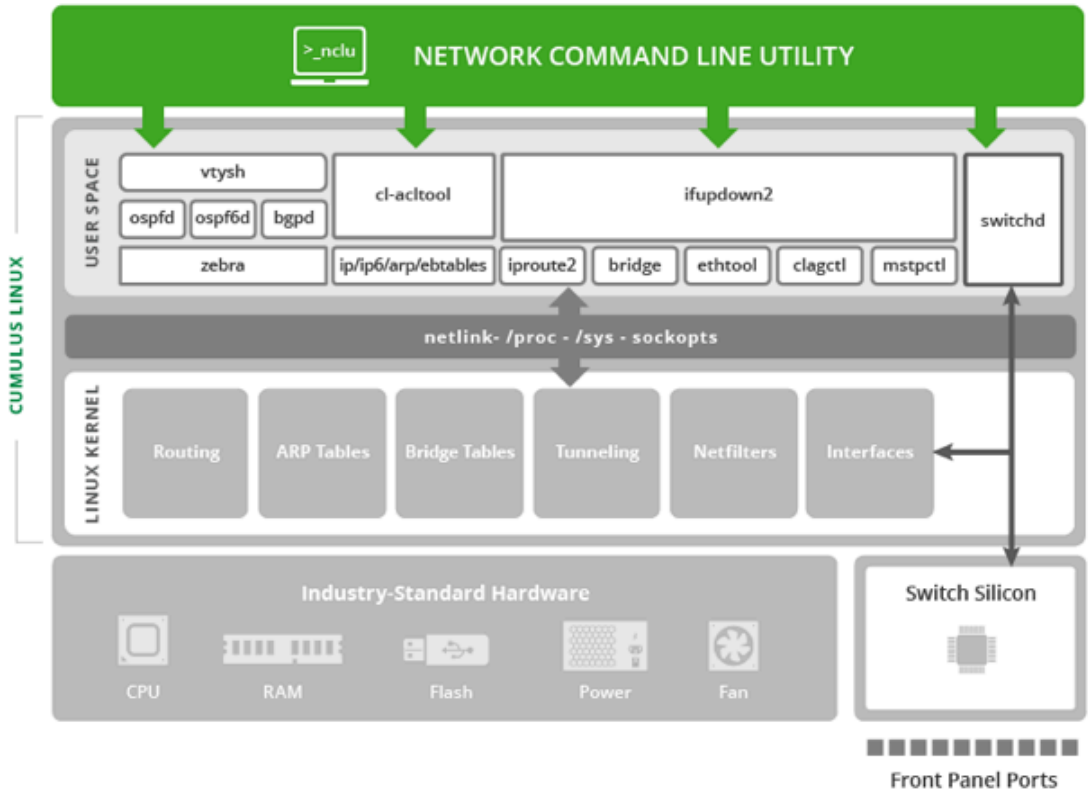
An overview of the Network Command Line Utility (NCLU) is also provided.

Network Command Line Utility - NCLU

The Network Command Line Utility (NCLU) is a command line interface for Cumulus Linux that simplifies the networking configuration process for all users.

NCLU resides in the Linux user space and provides consistent access to networking commands directly through bash, making configuration and troubleshooting simple and easy; no need to edit files or enter modes and sub-modes. NCLU provides these benefits:

- Embeds help, examples, and automatic command checking with suggestions in case you enter a typo.
- Runs directly from and integrates with bash, while being interoperable with the regular way of accessing underlying configuration files and automation.
- Configures dependent features automatically so that you don’t have to.



The NCLU wrapper utility called `net` is capable of configuring layer 2 and layer 3 features of the networking stack, installing ACLs and VXLANs, restoring configuration files, as well as providing monitoring and troubleshooting functionality for these features. You can configure both the `/etc/network/interfaces` and `/etc/frr/frr.conf` files with `net`, in addition to running show and clear commands related to `ifupdown2` and FRRouting.

NCLU Basics

Use the following workflow to stage and commit changes to Cumulus Linux with NCLU:

1. Use the `net add` and `net del` commands to stage and remove configuration changes.
2. Use the `net pending` command to review staged changes.
3. Use `net commit` and `net abort` to commit and delete staged changes.

> **NOTE**
>
> `net commit` applies the changes to the relevant configuration files, such as `/etc/network/interfaces`, then runs necessary follow on commands to enable the configuration, such as `ifreload -a`.
> If two different users try to commit a change at the same time, NCLU displays a warning but implements the change according to the first commit received. The second user will need to abort the commit.

When you have a running configuration, you can review and update the configuration with the following commands:

- `net show` is a series of commands for viewing various parts of the network configuration. For example, use `net show configuration` to view the complete network configuration, `net show commit history` to view a history of commits using NCLU, and `net show bgp` to view BGP status.
- `net clear` provides a way to clear `net show` counters, BGP and OSPF neighbor content, and more.

- `net rollback` provides a mechanism to revert back to an earlier configuration.
- `net commit confirm` requires you to press *Enter* to commit changes using NCLU. If you run `net commit confirm` but do not press *Enter* within 10 seconds, the commit automatically reverts and no changes are made.
- `net commit description <description>` enables you to provide a descriptive summary of the changes you are about to commit.
- `net commit permanent` retains the backup file taken when committing the change. Otherwise, the backup files created from NCLU commands are cleaned up periodically.
- `net del all` deletes all configurations.

> **NOTE**
>
> The `net del all` command does not remove management VRF configurations; NCLU does not interact with eth0 interfaces and management VRF.

Tab Completion, Verification, and Inline Help

In addition to tab completion and partial keyword command identification, NCLU includes verification checks to ensure you use the correct syntax. The examples below show the output for incorrect commands:

```
cumulus@switch:~$ net add bgp router-id 1.1.1.1/32
ERROR: Command not found

Did you mean one of the following?
    net add bgp router-id <ipv4>
        This command is looking for an IP address, not an IP/
prefixlen

cumulus@switch:~$ net add bgp router-id 1.1.1.1
cumulus@switch:~$ net add int swp10 mtu <TAB>
    <552-9216> :
cumulus@switch:~$ net add int swp10 mtu 9300
ERROR: Command not found

Did you mean one of the following?
    net add interface <interface> mtu <552-9216>
```

NCLU has a comprehensive built in help system. In addition to the net man page, you can use `?`and `help` to display available commands:

```
cumulus@switch:~$ net help

Usage:
```

```
    # net <COMMAND> [<ARGS>] [help]
    #
    # net is a command line utility for networking on Cumulus
Linux switches.
    #
    # COMMANDS are listed below and have context specific
arguments which can
    # be explored by typing "<TAB>" or "help" anytime while
using net.
    #
    # Use 'man net' for a more comprehensive overview.


    net abort
    net commit [verbose] [confirm [<number-seconds>]]
[description <wildcard>]
    net commit permanent <wildcard>
    net del all
    net help [verbose]
    net pending [json]
    net rollback (<number>|last)
    net rollback description <wildcard-snapshot>
    net show commit (history|<number>|last)
    net show rollback (<number>|last)
    net show rollback description <wildcard-snapshot>
```

```
    net show configuration
[commands|files|acl|bgp|multicast|ospf|ospf6]
    net show configuration interface [<interface>] [json]

Options:

    # Help commands
    help     : context sensitive information; see section below
    example  : detailed examples of common workflows

    # Configuration commands
    add      : add/modify configuration
    del      : remove configuration


    # Commit buffer commands
    abort    : abandon changes in the commit buffer
    commit   : apply the commit buffer to the system
    pending  : show changes staged in the commit buffer
    rollback : revert to a previous configuration state

    # Status commands
    show     : show command output
    clear    : clear counters, BGP neighbors, etc
```

```
cumulus@switch:~$ net help bestpath
The following commands contain keyword(s) 'bestpath'

    net (add|del) bgp bestpath as-path multipath-relax [as-
set|no-as-set]
    net (add|del) bgp bestpath compare-routerid
    net (add|del) bgp bestpath med missing-as-worst
    net (add|del) bgp ipv4 labeled-unicast neighbor <bgppeer>
addpath-tx-bestpath-per-AS
    net (add|del) bgp ipv4 unicast neighbor <bgppeer> addpath-
tx-bestpath-per-AS
    net (add|del) bgp ipv6 labeled-unicast neighbor <bgppeer>
addpath-tx-bestpath-per-AS
    net (add|del) bgp ipv6 unicast neighbor <bgppeer> addpath-
tx-bestpath-per-AS
    net (add|del) bgp neighbor <bgppeer> addpath-tx-bestpath-
per-AS
    net (add|del) bgp vrf <text> bestpath as-path multipath-
relax [as-set|no-as-set]
    net (add|del) bgp vrf <text> bestpath compare-routerid
    net (add|del) bgp vrf <text> bestpath med missing-as-worst
    net (add|del) bgp vrf <text> ipv4 labeled-unicast neighbor
<bgppeer> addpath-tx-bestpath-per-AS
```

```
    net (add|del) bgp vrf <text> ipv4 unicast neighbor
<bgppeer> addpath-tx-bestpath-per-AS
    net (add|del) bgp vrf <text> ipv6 labeled-unicast neighbor
<bgppeer> addpath-tx-bestpath-per-AS
    net (add|del) bgp vrf <text> ipv6 unicast neighbor
<bgppeer> addpath-tx-bestpath-per-AS
    net (add|del) bgp vrf <text> neighbor <bgppeer> addpath-tx-
bestpath-per-AS
    net add bgp debug bestpath <ip/prefixlen>
    net del bgp debug bestpath [<ip/prefixlen>]
    net show bgp (<ipv4>|<ipv4/prefixlen>|<ipv6>|<ipv6/
prefixlen>) [bestpath|multipath] [json]
    net show bgp vrf <text> (<ipv4>|<ipv4/
prefixlen>|<ipv6>|<ipv6/prefixlen>) [bestpath|multipath] [json]
```

NOTE

You can configure multiple interfaces at once:

```
cumulus@switch:~$ net add int swp7-9,12,15-17,22 mtu 9216
```

Search for Specific Commands

To search for specific NCLU commands so that you can identify the correct syntax to use, run the `net help verbose | <term>` command. For example, to show only commands that include `clag` (for MLAG):

```
cumulus@leaf01:mgmt:~$ net help verbose | grep clag
    net example clag basic-clag
    net example clag l2-with-server-vlan-trunks
    net example clag l3-uplinks-virtual-address
    net add clag peer sys-mac <mac-clag> interface <interface>
(primary|secondary) [backup-ip <ipv4>]
    net add clag peer sys-mac <mac-clag> interface <interface>
(primary|secondary) [backup-ip <ipv4> vrf <text>]
    net del clag peer
    net add clag port bond <interface> interface <interface>
clag-id <0-65535>
    net del clag port bond <interface>
    net show clag [our-macs|our-multicast-entries|our-multicast-
route|our-multicast-router-ports|peer-macs|peer-multicast-
entries|peer-multicast-route|peer-multicast-router-
ports|params|backup-ip|id] [verbose] [json]
    net show clag macs [<mac>] [json]
    net show clag neighbors [verbose]
    net show clag peer-lacp-rate
```

```
    net show clag verify-vlans [verbose]
    net show clag status [verbose] [json]
    net add bond <interface> clag id <0-65535>
    net add interface <interface> clag args <wildcard>
    net add interface <interface> clag backup-ip (<ipv4>|<ipv4>
vrf <text>)
    net add interface <interface> clag enable (yes|no)
    net add interface <interface> clag peer-ip
(<ipv4>|<ipv6>|linklocal)
    net add interface <interface> clag priority <0-65535>
    net add interface <interface> clag sys-mac <mac>
    net add loopback lo clag vxlan-anycast-ip <ipv4>
    net del bond <interface> clag id [<0-65535>]
    net del interface <interface> clag args [<wildcard>]
    ...
```

Add ? (Question Mark) Ability to NCLU

While tab completion is enabled by default, you can also configure NCLU to use the **?** (question mark character) to look at available commands. To enable this feature for the *cumulus* user, open the following file:

```
cumulus@switch:~$ sudo nano ~/.inputrc
```

Uncomment the very last line in the `.inputrc` file so that the file changes from this:

```
# Uncomment to use ? as an alternative to
# ?: complete
```

to this:

```
# Uncomment to use ? as an alternative to
?: complete
```

Save the file and reconnect to the switch. The ? (question mark) abilitywill work on all subsequent sessions on the switch.

```
cumulus@switch:~$ net
    abort     :  abandon changes in the commit buffer
    add       :  add/modify configuration
    clear     :  clear counters, BGP neighbors, etc
    commit    :  apply the commit buffer to the system
    del       :  remove configuration
    example   :  detailed examples of common workflows
    help      :  Show this screen and exit
```

```
    pending    :  show changes staged in the commit buffer
    rollback   :  revert to a previous configuration state
    show       :  show command output
```

> **NOTE**
>
> When the question mark is typed, NCLU autocompletes and shows all available options, but the question mark does not actually appear on the terminal. This is normal, expected behavior.

Built-In Examples

NCLU has a number of built in examples to guide you through basic configuration setup:

```
cumulus@switch:~$ net example
    acl               :  access-list
    bgp               :  Border Gateway Protocol
    bond              :  bond, port-channel, etc
    bridge            :  a layer2 bridge
    clag              :  Multi-Chassis Link Aggregation
    dhcp              :  Dynamic Host Configuration Protocol
```

```
    dot1x            :  Configure, Enable, Delete or Show IEEE
802.1X EAPOL
    evpn             :  Ethernet VPN
    link-settings    :  Physical link parameters
    management-vrf   :  Management VRF
    mlag             :  Multi-Chassis Link Aggregation
    ospf             :  Open Shortest Path First (OSPFv2)
    snmp-server      :  Configure the SNMP server
    syslog           :  Set syslog logging
    vlan-interfaces  :  IP interfaces for VLANs
    voice-vlan       :  VLAN used for IP Phones
    vrr              :  add help text
cumulus@switch:~$ net example bridge


Scenario
========
We are configuring switch1 and would like to configure the
following
- configure switch1 as an L2 switch for host-11 and host-12
- enable vlans 10-20
- place host-11 in vlan 10
- place host-12 in vlan 20
- create an SVI interface for vlan 10
- create an SVI interface for vlan 20
```

```
- assign IP 10.0.0.1/24 to the SVI for vlan 10
- assign IP 20.0.0.1/24 to the SVI for vlan 20
- configure swp3 as a trunk for vlans 10, 11, 12 and 20
                  swp3
         *switch1 --------- switch2
            /\
      swp1 /  \ swp2
          /    \
         /      \
     host-11   host-12

switch1 net commands
====================
- enable vlans 10-20
switch1# net add vlan 10-20
- place host-11 in vlan 10
- place host-12 in vlan 20
switch1# net add int swp1 bridge access 10
switch1# net add int swp2 bridge access 20
- create an SVI interface for vlan 10
- create an SVI interface for vlan 20
- assign IP 10.0.0.1/24 to the SVI for vlan 10
- assign IP 20.0.0.1/24 to the SVI for vlan 20
switch1# net add vlan 10 ip address 10.0.0.1/24
```

```
switch1# net add vlan 20 ip address 20.0.0.1/24
- configure swp3 as a trunk for vlans 10, 11, 12 and 20
switch1# net add int swp3 bridge trunk vlans 10-12,20
switch1# net pending
switch1# net commit

Verification
============
switch1# net show interface
switch1# net show bridge macs
```

Configure User Accounts

You can configure user accounts in Cumulus Linux with read-only or edit permissions for NCLU:

- You create user accounts with **read-only** permissions for NCLU by adding them to the `netshow` group. A user in the `netshow` group can run NCLU `net show` commands, such as `net show interface` or `net show config`, and certain general Linux commands, such as `ls`, `cd` or `man`, but cannot run `net add`, `net del` or `net commit` commands.
- You create user accounts with **edit** permissions for NCLU by adding them to the `netedit` group. A user in the `netedit` group can run NCLU configuration commands, such `net add`, `net del` or `net commit` in addition to NCLU `net show` commands.

The examples below demonstrate how to add a new user account or modify an existing user account called *myuser*.

To add a new user account with NCLU show permissions:

```
cumulus@switch:~$ sudo adduser --ingroup netshow myuser
Adding user `myuser' ...
Adding new user `myuser' (1001) with group `netshow'...
...
```

To add NCLU show permissions to a user account that already exists:

```
cumulus@switch:~$ sudo addgroup myuser netshow
Adding user `myuser' to group `netshow' ...
Adding user myuser to group netshow
Done
```

To add a new user account with NCLU edit permissions:

```
cumulus@switch:~$ sudo adduser --ingroup netedit myuser
Adding user `myuser' ...
Adding new user `myuser' (1001) with group `netedit'
...
```

To add NCLU edit permissions to a user account that already exists:

```
cumulus@switch:~$ sudo addgroup myuser netedit
Adding user `myuser' to group `netedit' ...
Adding user myuser to group netedit
Done
```

> **NOTE**
>
> You can use the `adduser` command for local user accounts only. You can use the `addgroup` command for both local and remote user accounts. For a remote user account, you must use the mapping username, such as `tacacs3` or `radius_user`, not the TACACS or RADIUS account name.

If the user tries to run commands that are not allowed, the following error displays:

```
myuser@switch:~$ net add hostname host01
ERROR: User username does not have permission to make
networking changes.
```

Edit the netd.conf File

Instead of using the NCLU commands described above, you can manually configure users and groups to be able to run NCLU commands.

Edit the `/etc/netd.conf` file to add users to the *users_with_edit* and *users_with_show* lines in the file, then save the file.

For example, if you want the user *netoperator* to be able to run both edit and show commands, add the user to the `users_with_edit` and `users_with_show` lines in the `/etc/netd.conf` file:

```
cumulus@switch:~$ sudo nano /etc/netd.conf


# Control which users/groups are allowed to run 'add', 'del',
# 'clear', 'net abort', 'net commit' and restart services
# to apply those changes
users_with_edit = root, cumulus, netoperator
groups_with_edit = netedit


# Control which users/groups are allowed to run 'show' commands
users_with_show = root, cumulus, netoperator
groups_with_show = netshow, netedit
```

To configure a new user group to use NCLU, add that group to the `groups_with_edit` and `groups_with_show` lines in the file.

WARNING

Use caution giving edit permissions to groups. For example, do not give edit permissions to the *tacacs* group.

Restart the netd Service

Whenever you modify `netd.conf` or when NSS services change, you must restart the `netd` service for the changes to take effect:

```
cumulus@switch:~$ sudo systemctl restart netd.service
```

Back Up the Configuration to a Single File

You can easily back up your NCLU configuration to a file by outputting the results of `net show configuration commands` to a file, then retrieving the contents of the file using the `source` command. You can then view the configuration at any time or copy it to other switches and use the `source` command to apply that configuration to those switches.

For example, to copy the configuration of a leaf switch called leaf01, run the following command:

```
cumulus@leaf01:~$ net show configuration commands >> leaf01.txt
```

With the commands all stored in a single file, you can now copy this file to another ToR switch in your network called leaf01 and apply the configuration by running:

```
cumulus@leaf01:~$ source leaf01.txt
```

Advanced Configuration

NCLU needs no initial configuration; however, if you need to modify certain configuration, you must manually update the `/etc/netd.conf` file. You can configure this file to allow different permission levels for users to edit configurations and run `show` commands. The file also contains a blacklist that hides less frequently used terms from the tabbed autocomplete.

After you edit the `netd.conf` file, restart the `netd` service for the changes to take effect.

```
cumulus@switch:~$ sudo nano /etc/netd.conf
cumulus@switch:~$ sudo systemctl restart netd.service
```

Configuration Variable	Default Setting	Description
show_linux_command	False	When true, displays the Linux command running in the background.
color_diffs	True	When true, the diffs shown in net pending and net commit use colors.
enable_<component>	True	When true, enables you to configure the component with NCLU. For example, when enable_frr is true, you can use NCLU to configure FRR.
users_with_edit	root, cumulus	Sets the Linux users with root edit privileges.
groups_with_edit	root, cumulus	Sets the Linux groups with root edit privileges.
users_with_show	root, cumulus	Controls which users are allowed to run show commands.
groups_with_show	root, cumulus	Controls which

Configuration Variable	Default Setting	Description
		groups are allowed to run show commands.
ifupdown_blacklist	address-purge, bond-ad-actor-sys-prio, bond-ad-actor-system, bond-num-grat-arp,bond-num-unsol-na, bond-use-carrier, bond-xmit-hash-policy, bridge-bridgeprio, bridge-fd, bridge-hashel, bridge-hashmax, bridge-hello, bridge-igmp-querier-src, bridge-maxage, bridge-maxwait, bridge-mclmc, bridge-mclmi bridge-mcmi, bridge-mcqi, bridge-mcqpi, bridge-mcqri, bridge-mcrouter, bridge-mcsqc, bridge-mcsqi, bridge-pathcosts, bridge-port-pvids, bridge-port-vids, bridge-portprios,	Hides corner case command options from tab complete, to simplify and streamline output.

Configuration Variable	Default Setting	Description
	bridge-waitport, broadcast, link-type, mstpctl-ageing, mstpctl-fdelay, mstpctl-forcevers, mstpctl-hello, mstpctl-maxage, mstpctl-maxhops, mstpctl-portp2p, mstpctl-portpathcost, mstpctl-portrestrtcn, mstpctl-treeportcost, mstpctl-treeportprio, mstpctl-txholdcount, netmask, preferred-lifetime, scope, vxlan-ageing, vxlan-learning, vxlan-port, up, down, bridge-gcint, bridge-mcqifaddr, bridge-mcqv4src	

> ⚠ **IMPORTANT**
>
> `net` provides an environment variable to set where the `net` output is directed. To only use `stdout`, set the `NCLU_TAB_STDOUT` environment variable to *true*. The value is not case sensitive.

Considerations

Unsupported Interface Names

NCLU does not support interfaces named `dev`.

Bonds With No Configured Members

If a bond interface is configured and it contains no members NCLU will report the interace does not exist.

Setting Date and Time

Setting the time zone, date and time requires root privileges; use `sudo`.

Set the Time Zone

You can use one of two methods to set the time zone on the switch:

- Edit the `/etc/timezone` file.
- Use the guided wizard.

Edit the /etc/timezone File

To see the current time zone, list the contents of `/etc/timezone`:

```
cumulus@switch:~$ cat /etc/timezone
US/Eastern
```

Edit the file to add your desired time zone. A list of valid time zones can be found here.

Use the following command to apply the new time zone immediately.

```
cumulus@switch:~$ sudo dpkg-reconfigure --frontend
noninteractive tzdata
```

Use the following command to change the /etc/localtime to reflect your current timezone. Use the same value as the previous step.

```
sudo ln -sf /usr/share/zoneinfo/US/Eastern /etc/localtime
```

Use the Guided Wizard

To set the time zone using the guided wizard, run `dpkg-reconfigure tzdata` as root:

```
cumulus@switch:~$ sudo dpkg-reconfigure tzdata
```

For more information, see the Debian System Administrator's Manual - Time.

Set the Date and Time

The switch contains a battery backed hardware clock that maintains the time while the switch is powered off and in between reboots. When the switch is running, the Cumulus Linux operating system maintains its own software clock.

During boot up, the time from the hardware clock is copied into the operating system's software clock. The software clock is then used for all timekeeping responsibilities. During system shutdown, the software clock is copied back to the battery backed hardware clock.

You can set the date and time on the software clock using the `date` command. First, determine your current time zone:

```
cumulus@switch:~$ date +%Z
```

> **NOTE**
>
> If you need to reconfigure the current time zone, refer to the instructions above.

Then, to set the system clock according to the time zone configured:

```
cumulus@switch:~$ sudo date -s "Tue Jan 12 00:37:13 2016"
```

See `man date(1)` for more information.

You can write the current value of the system (software) clock to the hardware clock using the `hwclock` command:

```
cumulus@switch:~$ sudo hwclock -w
```

See `man hwclock(8)` for more information.

Use NTP

The `ntpd` daemon running on the switch implements the NTP protocol. It synchronizes the system time with time servers listed in the `/etc/ntp.conf` file. The `ntpd` daemon is started at boot by default. See `man ntpd(8)` for details.

> **NOTE**
>
> If you intend to run this service within a VRF, including the

management VRF, follow these steps for configuring the service.

Configure NTP Servers

The default NTP configuration comprises the following servers, which are listed in the `/etc/ntpd.conf` file:

- server 0.cumulusnetworks.pool.ntp.org iburst
- server 1.cumulusnetworks.pool.ntp.org iburst
- server 2.cumulusnetworks.pool.ntp.org iburst
- server 3.cumulusnetworks.pool.ntp.org iburst

To add the NTP server or servers you want to use:

NCLU Commands | Linux Commands

Run the following commands. Include the `iburst` option to increase the sync speed.

```
cumulus@switch:~$ net add time ntp server
4.cumulusnetworks.pool.ntp.org iburst
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands add the NTP server to the list of servers in the `/etc/ntp.conf` file:

```
# pool.ntp.org maps to about 1000 low-stratum NTP servers.
Your server will
# pick a different set every time it starts up.  Please
consider joining the
# pool: <http://www.pool.ntp.org/join.html>
server 0.cumulusnetworks.pool.ntp.org iburst
server 1.cumulusnetworks.pool.ntp.org iburst
server 2.cumulusnetworks.pool.ntp.org iburst
server 3.cumulusnetworks.pool.ntp.org iburst
server 4.cumulusnetworks.pool.ntp.org iburst
```

> **NOTE**
>
> To set the initial date and time with NTP before starting the `ntpd` daemon, run the `ntpd -q` command. This command is the same as `ntpdate`, which is to be retired and no longer available.
>
> Be aware that `ntpd -q` can hang if the time servers are not reachable.

To verify that `ntpd` is running on the system:

```
cumulus@switch:~$ ps -ef | grep ntp
ntp       4074     1  0 Jun20 ?        00:00:33 /usr/sbin/ntpd
-p /var/run/ntpd.pid -g -u 101:102
```

To check the NTP peer status:

NCLU Commands | Linux Commands

Run the `net show time ntp servers` command:

```
cumulus@switch:~$ net show time ntp servers
      remote           refid      st t when poll reach
delay   offset  jitter
==============================================================================
+minime.fdf.net  58.180.158.150   3 u  140 1024  377
55.659    0.339   1.464
+69.195.159.158  128.138.140.44   2 u  259 1024  377
41.587    1.011   1.677
*chl.la          216.218.192.202  2 u  210 1024  377
4.008    1.277   1.628
+vps3.drown.org  17.253.2.125     2 u  743 1024  377
39.319   -0.316   1.384
```

To remove one or more NTP servers:

NCLU Commands | Linux Commands

Run the `net del time ntp <server>` command. The following example commands remove some of the default NTP servers.

```
cumulus@switch:~$ net del time ntp server
0.cumulusnetworks.pool.ntp.org
cumulus@switch:~$ net del time ntp server
1.cumulusnetworks.pool.ntp.org
cumulus@switch:~$ net del time ntp server
2.cumulusnetworks.pool.ntp.org
cumulus@switch:~$ net del time ntp server
3.cumulusnetworks.pool.ntp.org
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Specify the NTP Source Interface

By default, the source interface that NTP uses is eth0. To change the source interface:

NCLU Commands | Linux Commands

Run the `net add time ntp source <interface>` command. The following command example changes the NTP source interface to swp10.

```
cumulus@switch:~$ net add time ntp source swp10
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following configuration snippet in the `ntp.conf` file:

```
...
# Specify interfaces
interface listen swp10
...
```

Use NTP in a DHCP Environment

You can use DHCP to specify your NTP servers. Ensure that the DHCP-generated configuration file named `/run/ntp.conf.dhcp` exists. This file is generated by the `/etc/dhcp/dhclient-exit-hooks.d/ntp` script and is a

copy of the default `/etc/ntp.conf` with a modified server list from the DHCP server. If this file does not exist and you plan on using DHCP in the future, you can copy your current `/etc/ntp.conf` file to the location of the DHCP file.

To use DHCP to specify your NTP servers, run the `sudo -E systemctl edit ntp.service` command and add the `ExecStart=` line:

```
cumulus@switch:~$ sudo -E systemctl edit ntp.service
[Service]
ExecStart=
ExecStart=/usr/sbin/ntpd -n -u ntp:ntp -g -c /run/ntp.conf.dhcp
```

> **NOTE**
>
> The `sudo -E systemctl edit ntp.service` command always updates the base `ntp.service` even if `ntp@mgmt.service` is used. The `ntp@mgmt.service` is re-generated automatically.

To validate that your configuration, run these commands:

```
cumulus@switch:~$ sudo systemctl restart ntp
cumulus@switch:~$ sudo systemctl status -n0 ntp.service
```

If the state is not *Active*, or the alternate configuration file does not appear in the `ntp` command line, it is likely that a mistake was made. In this case, correct the mistake and rerun the three commands above to verify.

> **NOTE**
>
> When you use the above procedure to specify your NTP servers, the NCLU commands for changing NTP settings do not take effect.

Configure NTP with Authorization Keys

For added security, you can configure NTP to use authorization keys.

Configure the NTP Server

1. Create a `.keys` file, such as `/etc/ntp.keys`. Specify a key identifier (a number from 1-65535), an encryption method (M for MD5), and the password. The following provides an example:

```
#
# PLEASE DO NOT USE THE DEFAULT VALUES HERE.
#
#65535  M  akey
#1      M  pass
```

```
1  M  CumulusLinux!
```

2. In the `/etc/ntp/ntp.conf` file, add a pointer to the `/etc/ntp.keys` file you created above and specify the key identifier. For example:

```
keys /etc/ntp/ntp.keys
trustedkey 1
controlkey 1
requestkey 1
```

3. Restart NTP with the `sudo systemctl restart ntp` command.

Configure the NTP Client

The NTP client is the Cumulus Linux switch.

1. Create the same `.keys` file you created on the NTP server (`/etc/ntp.keys`). For example:

```
cumulus@switch:~$  sudo nano /etc/ntp.keys
#
# PLEASE DO NOT USE THE DEFAULT VALUES HERE.
```

```
#
#65535  M  akey
#1      M  pass


1  M  CumulusLinux!
```

2. Edit the `/etc/ntp.conf` file to specify the server you want to use, the key identifier, and a pointer to the `/etc/ntp.keys` file you created in step 1. For example:

```
cumulus@switch:~$ sudo nano /etc/ntp.conf
...
# You do need to talk to an NTP server or two (or three).
#pool ntp.your-provider.example
# OR
#server ntp.your-provider.example


# pool.ntp.org maps to about 1000 low-stratum NTP servers.
Your server will
# pick a different set every time it starts up.  Please
consider joining the
# pool: <http://www.pool.ntp.org/join.html>
#server 0.cumulusnetworks.pool.ntp.org iburst
```

```
#server 1.cumulusnetworks.pool.ntp.org iburst
#server 2.cumulusnetworks.pool.ntp.org iburst
#server 3.cumulusnetworks.pool.ntp.org iburst
server 10.50.23.121 key 1

#keys
keys /etc/ntp.keys
trustedkey 1
controlkey 1
requestkey 1
...
```

3. Restart NTP in the active VRF (default or management). For example:

```
cumulus@switch:~$ systemctl restart ntp@mgmt.service
```

4. Wait a few minutes, then run the `ntpq -c as` command to verify the configuration:

```
cumulus@switch:~$ ntpq -c as
```

```
ind assid status  conf reach auth condition  last_event cnt
===========================================================
  1 40828  f014   yes   yes   ok     reject   reachable  1
```

After authorization is accepted, you see the following command output:

```
cumulus@switch:~$ ntpq -c as

ind assid status  conf reach auth condition  last_event cnt
===========================================================
  1 40828  f61a   yes   yes   ok   sys.peer    sys_peer  1
```

Precision Time Protocol (PTP) Boundary Clock

With the growth of low latency and high performance applications, precision timing has become increasingly important. Precision Time Protocol (PTP) is used to synchronize clocks in a network and is capable of sub-microsecond accuracy. The clocks are organized in a master-slave hierarchy. The slaves are synchronized to their masters, which can be slaves to their own masters. The hierarchy is created and updated automatically by the best master clock (BMC) algorithm, which runs on every clock. The grandmaster clock is the top-level master and is typically synchronized by using a Global Positioning System (GPS) time source to provide a high-

degree of accuracy.

A boundary clock has multiple ports; one or more master ports and one or more slave ports. The master ports provide time (the time can originate from other masters further up the hierarchy) and the slave ports receive time. The boundary clock absorbs sync messages in the slave port, uses that port to set its clock, then generates new sync messages from this clock out of all of its master ports.

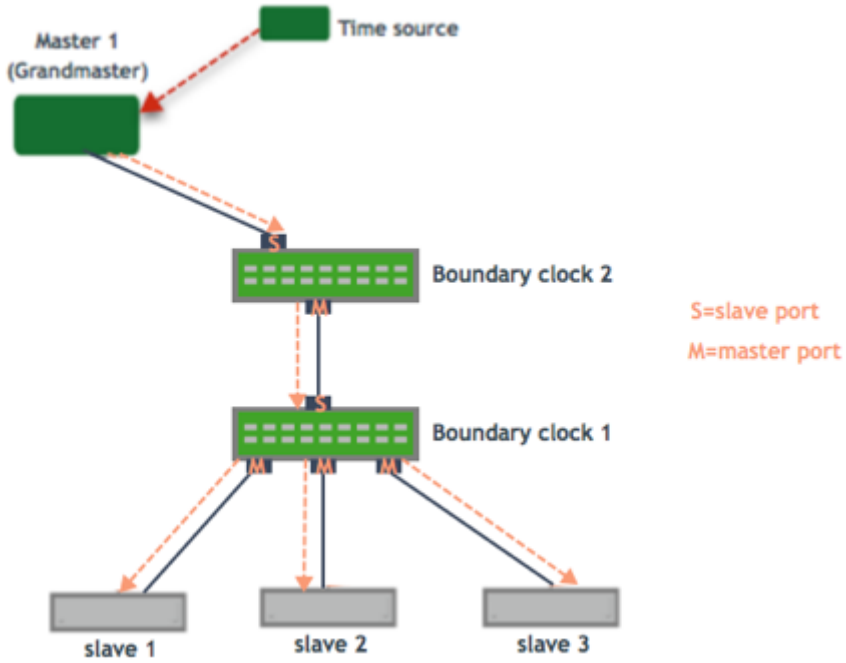
Cumulus Linux includes the `linuxptp` package for PTP, which uses the `phc2sys` daemon to synchronize the PTP clock with the system clock.

NOTE

- Cumulus Linux currently supports PTP on the Mellanox Spectrum ASIC only.
- PTP is supported in boundary clock mode only (the switch provides timing to downstream servers; it is a slave to a higher-level clock and a master to downstream clocks).
- The switch uses hardware time stamping to capture timestamps from an Ethernet frame at the physical layer. This allows PTP to account for delays in message transfer and greatly improves the accuracy of time synchronization.
- Only IPv4/UDP PTP packets are supported.
- Only a single PTP domain per network is supported. A PTP

domain is a network or a portion of a network within which all the clocks are synchronized.

In the following example, boundary clock 2 receives time from Master 1 (the grandmaster) on a PTP slave port, sets its clock and passes the time down from the PTP master port to boundary clock 1. Boundary clock 1 receives the time on a PTP slave port, sets its clock and passes the time down the hierarchy through the PTP master ports to the hosts that receive the time.



Enable the PTP Boundary Clock on the Switch

To enable the PTP boundary clock on the switch:

1. Open the `/etc/cumulus/switchd.conf` file in a text editor and add the following line:

```
ptp.timestamping = TRUE
```

2. Restart `switchd`:

```
cumulus@switch:~$ sudo systemctl restart switchd.service
```

WARNING

Restarting the `switchd` service causes all network ports to reset, interrupting network services, in addition to resetting the switch hardware configuration.

Configure the PTP Boundary Clock

To configure a boundary clock:

1. Configure the interfaces on the switch that you want to use for PTP. Each interface must be configured as a layer 3 routed interface with an IP address.

   > **NOTE**
   >
   > PTP *is* supported on BGP unnumbered interfaces.
   >
   > PTP is *not* supported on switched virtual interfaces (SVIs).

   ```
   cumulus@switch:~$ net add interface swp13s0 ip address
   10.0.0.9/32
   cumulus@switch:~$ net add interface swp13s1 ip address
   10.0.0.10/32
   ```

2. Configure PTP options on the switch:

   - Set the `gm-capable` option to `no` to configure the switch to be a boundary clock.
   - Set the priority, which selects the best master clock. You can set priority 1 or 2. For each priority, you can use a number between 0 and 255. The default priority is 255. For the boundary clock, use a number above 128. The lower priority is applied first.
   - Add the `time-stamping` parameter. The switch automatically enables

hardware time-stamping to capture timestamps from an Ethernet frame at the physical layer. If you are testing PTP in a virtual environment, hardware time-stamping is not available; however the `time-stamping` parameter is still required.

- Add the PTP master and slave interfaces. You do not specify which is a master interface and which is a slave interface; this is determined by the PTP packet received. The following commands provide an example configuration:

```
cumulus@switch:~$ net add ptp global gm-capable no
cumulus@switch:~$ net add ptp global priority2 254
cumulus@switch:~$ net add ptp global priority1 254
cumulus@switch:~$ net add ptp global time-stamping
cumulus@switch:~$ net add ptp interface swp13s0
cumulus@switch:~$ net add ptp interface swp13s1
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The `ptp4l` man page describes all the configuration parameters.

3. Restart the `ptp4l` and `phc2sys` daemons:

```
cumulus@switch:~$ sudo systemctl restart ptp4l.service
```

```
phc2sys.service
```

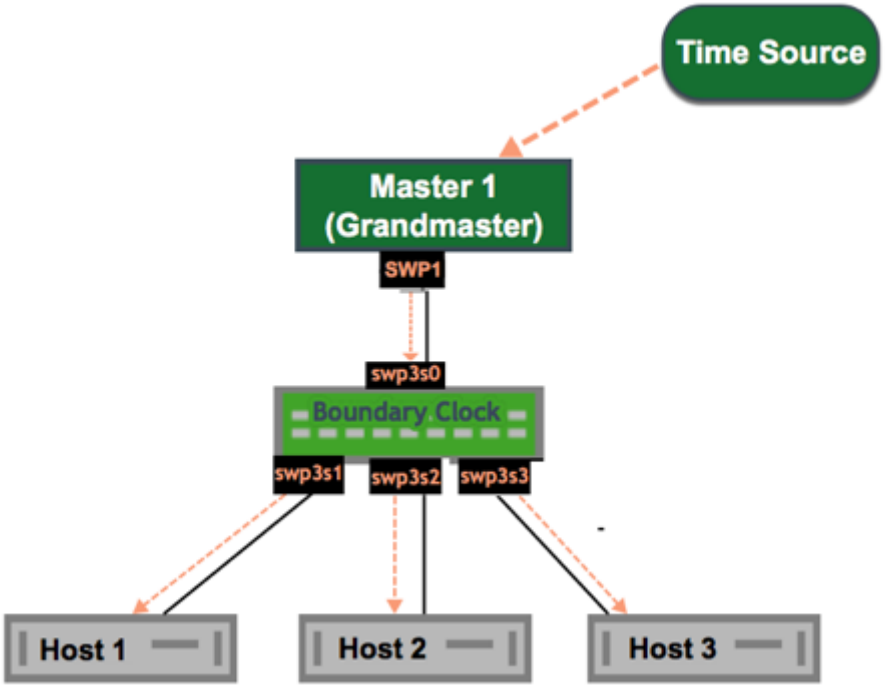
The configuration is saved in the `/etc/ptp4l.conf` file.

4. Enable the services to start at boot time:

```
cumulus@switch:~$ sudo systemctl enable ptp4l.service
phc2sys.service
```

Example Configuration

In the following example, the boundary clock on the switch receives time from Master 1 (the grandmaster) on PTP slave port swp3s0, sets its clock and passes the time down through PTP master ports swp3s1, swp3s2, and swp3s3 to the hosts that receive the time.



The configuration for the above example is shown below. The example assumes that you have already configured the layer 3 routed interfaces (`swp3s0`, `swp3s1`, `swp3s2`, and `swp3s3`) you want to use for PTP.

```
cumulus@switch:~$ net add ptp global gm-capable no
cumulus@switch:~$ net add ptp global priority2 254
cumulus@switch:~$ net add ptp global priority1 254
cumulus@switch:~$ net add ptp global time-stamping
cumulus@switch:~$ net add ptp interface swp3s0
cumulus@switch:~$ net add ptp interface swp3s1
cumulus@switch:~$ net add ptp interface swp3s2
cumulus@switch:~$ net add ptp interface swp3s3
```

```
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Verify PTP Boundary Clock Configuration

To view a summary of the PTP configuration on the switch, run the `net show configuration ptp` command:

```
cumulus@switch:~$ net show configuration ptp


ptp
  global


    slaveOnly
      0


    priority1
      255


    priority2
      255


    domainNumber
```

```
      0

    logging_level
      5

    path_trace_enabled
      0

    use_syslog
      1

    verbose
      0

    summary_interval
      0

    time_stamping
      hardware

    gmCapable
      0
  swp15s0
  swp15s1
```

```
...
```

View PTP Status Information

To view PTP status information, run the `net show ptp parent_data_set` command:

```
cumulus@switch:~$ net show ptp parent_data_set
parent_data_set
===============
parentPortIdentity                    000200.fffe.000001-1
parentStats                           0
observedParentOffsetScaledLogVariance 0xffff
observedParentClockPhaseChangeRate    0x7fffffff
grandmasterPriority1                  127
gm.ClockClass                         248
gm.ClockAccuracy                      0xfe
gm.OffsetScaledLogVariance            0xffff
grandmasterPriority2                  127
grandmasterIdentity                   000200.fffe.000001
```

To view the additional PTP status information, including the delta in nanoseconds from the master clock, run the `sudo pmc -u -b 0 'GET`

`TIME_STATUS_NP'` command:

```
cumulus@switch:~$ sudo pmc -u -b 0 'GET TIME_STATUS_NP'
sending: GET TIME_STATUS_NP
    7cfe90.fffe.f56dfc-0 seq 0 RESPONSE MANAGEMENT
TIME_STATUS_NP
        master_offset              12610
        ingress_time               1525717806521177336
        cumulativeScaledRateOffset +0.000000000
        scaledLastGmPhaseChange    0
        gmTimeBaseIndicator        0
        lastGmPhaseChange          0x0000'0000000000000000.0000
        gmPresent                  true
        gmIdentity                 000200.fffe.000005
    000200.fffe.000005-1 seq 0 RESPONSE MANAGEMENT
TIME_STATUS_NP
        master_offset              0
        ingress_time               0
        cumulativeScaledRateOffset +0.000000000
        scaledLastGmPhaseChange    0
        gmTimeBaseIndicator        0
        lastGmPhaseChange          0x0000'0000000000000000.0000
        gmPresent                  false
        gmIdentity                 000200.fffe.000005
    000200.fffe.000006-1 seq 0 RESPONSE MANAGEMENT
```

```
TIME_STATUS_NP
        master_offset              5544033534
        ingress_time               1525717812106811842
        cumulativeScaledRateOffset +0.000000000
        scaledLastGmPhaseChange    0
        gmTimeBaseIndicator        0
        lastGmPhaseChange          0x0000'0000000000000000.0000
        gmPresent                  true
        gmIdentity                 000200.fffe.000005
```

Delete PTP Boundary Clock Configuration

To delete PTP configuration, delete the PTP master and slave interfaces. The following example commands delete the PTP interfaces `swp3s0`, `swp3s1`, and `swp3s2`.

```
cumulus@switch:~$ net del ptp interface swp3s0
cumulus@switch:~$ net del ptp interface swp3s1
cumulus@switch:~$ net del ptp interface swp3s2
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Related Information

- Debian System Administrator's Manual - Time
- NTP website
- Wikipedia - Network Time Protocol
- Debian wiki - NTP

Authentication, Authorization and Accounting

This section describes how to set up user accounts, ssh for remote access, LDAP authentication, TACACS+, and RADIUS AAA.

SSH for Remote Access

You can generate authentication keys to access a Cumulus Linux switch securely with the `ssh-keygen` component of the Secure Shell (SSH) protocol. Cumulus Linux uses the OpenSSH package to provide this functionality. This section describes how to generate an SSH key pair.

Generate an SSH Key Pair

1. To generate the SSH key pair, run the `ssh-keygen` command and follow the prompts:

> ⚠ **IMPORTANT**
>
> To configure a completely passwordless system, do not enter a passphrase when prompted in the following step.

```
cumulus@leaf01:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/cumulus/.ssh/
id_rsa):
Enter passphrase (empty for no passphrase):
```

```
Enter same passphrase again:
Your identification has been saved in /home/cumulus/.ssh/
id_rsa.
Your public key has been saved in /home/cumulus/.ssh/
id_rsa.pub.
The key fingerprint is:
5a:b4:16:a0:f9:14:6b:51:f6:f6:c0:76:1a:35:2b:bb cumulus@leaf04
The key's randomart image is:
+---[RSA 2048]----+
|      +.o   o    |
|     o * o . o   |
|    o + o O o    |
|     + . = O     |
|      . S o .    |
|       +   .     |
|      .   E      |
|                 |
|                 |
+-----------------+
```

2. To copy the generated public key to the desired location, run the `ssh-copy-id` command and follow the prompts:

```
cumulus@leaf01:~$ ssh-copy-id -i /home/cumulus/.ssh/
id_rsa.pub cumulus@leaf02
The authenticity of host 'leaf02 (192.168.0.11)' can't be
established.
ECDSA key fingerprint is
b1:ce:b7:6a:20:f4:06:3a:09:3c:d9:42:de:99:66:6e.
Are you sure you want to continue connecting (yes/no)? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new
key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed
-- if you are prompted now it is to install the new keys
cumulus@leaf01's password:

Number of key(s) added: 1
```

`ssh-copy-id` does not work if the username on the remote switch is different from the username on the local switch. To work around this issue, use the `scp` command instead:

```
cumulus@leaf01:~$ scp .ssh/id_rsa.pub cumulus@leaf02:.ssh/
authorized_keys
Enter passphrase for key '/home/cumulus/.ssh/id_rsa':
id_rsa.pub
```

3. Connect to the remote switch to confirm that the authentication keys are in place:

```
cumulus@leaf01:~$ ssh cumulus@leaf02

Welcome to Cumulus VX (TM)

Cumulus VX (TM) is a community supported virtual appliance
designed for
experiencing, testing and prototyping the latest technology.
For any questions or technical support, visit our community
site at:
http://community.cumulusnetworks.com

The registered trademark Linux (R) is used pursuant to a
sublicense from LMI,
the exclusive licensee of Linus Torvalds, owner of the mark
on a world-wide basis.
Last login: Thu Sep 29 16:56:54 2016
```

Related Information

- Debian Documentation - Password-less logins with OpenSSH
- Wikipedia - Secure Shell (SSH)

User Accounts

By default, Cumulus Linux has two user accounts: *cumulus* and *root*.

The *cumulus* account:

- Uses the default password `cumulus`. You are required to change the default password when you log into Cumulus Linux for the first time.
- Is a user account in the *sudo* group with sudo privileges.
- Can log in to the system through all the usual channels, such as console and SSH.
- Along with the cumulus group, has both show and edit rights for NCLU.

The *root* account:

- Has the default password disabled by default
- Has the standard Linux root user access to everything on the switch
- Disabled password prohibits login to the switch by SSH, telnet, FTP, and so on

You can add additional user accounts as needed. Like the *cumulus* account, these accounts must use `sudo` to execute privileged commands; be sure to include them in the *sudo* group. For example:

```
cumulus@switch:~$ sudo adduser NEWUSERNAME sudo
```

To access the switch without a password, you need to boot into a single

shell/user mode.

You can add and configure user accounts in Cumulus Linux with read-only or edit permissions for NCLU. For more information, see Configure User Accounts.

Enable Remote Access for the root User

The root user does not have a password and cannot log into a switch using SSH. This default account behavior is consistent with Debian. To connect to a switch using the root account, you can do one of the following:

- Generate an SSH key
- Set a password

Generate an SSH Key for the root Account

1. In a terminal on your host system (not the switch), check to see if a key already exists:

   ```
   root@host:~# ls -al ~/.ssh/
   ```

   The name of the key is similar to `id_dsa.pub`, `id_rsa.pub`, or `id_ecdsa.pub`.

2. If a key does not exist, generate a new one by first creating the RSA key pair:

```
root@host:~# ssh-keygen -t rsa
```

3. You are prompted to enter a file in which to save the key (/root/.ssh/id_rsa). Press Enter to use the home directory of the root user or provide a different destination.

4. You are prompted to enter a passphrase (empty for no passphrase). This is optional but it does provide an extra layer of security.

5. The public key is now located in `/root/.ssh/id_rsa.pub`. The private key (identification) is now located in `/root/.ssh/id_rsa`.

6. Copy the public key to the switch. SSH to the switch as the cumulus user, then run:

```
cumulus@switch:~$ sudo mkdir -p /root/.ssh
cumulus@switch:~$ echo <SSH public key string> | sudo tee -a
/root/.ssh/authorized_keys
```

Set the root User Password

1. Run the following command:

```
cumulus@switch:~$ sudo passwd root
```

2. Change the `PermitRootLogin` setting in the `/etc/ssh/sshd_config` file from *without-password* to *yes*.

```
cumulus@switch:~$ sudo nano /etc/ssh/sshd_config
...
# Authentication:
LoginGraceTime 120
PermitRootLogin yes
StrictModes yes
...
```

3. Restart the `ssh` service:

```
cumulus@switch:~$ sudo systemctl reload ssh.service
```

Using sudo to Delegate Privileges

By default, Cumulus Linux has two user accounts: *root* and *cumulus*. The *cumulus* account is a normal user and is in the group *sudo*.

You can add more user accounts as needed. Like the *cumulus* account, these accounts must use `sudo` to execute privileged commands.

sudo Basics

`sudo` allows you to execute a command as superuser or another user as specified by the security policy. See `man sudo(8)` for details.

The default security policy is *sudoers*, which is configured using `/etc/sudoers`. Use `/etc/sudoers.d/` to add to the default sudoers policy. See `man sudoers(5)` for details.

WARNING

Use `visudo` only to edit the `sudoers` file; do not use another editor like `vi` or `emacs`. See `man visudo(8)` for details.

When creating a new file in `/etc/sudoers.d`, use `visudo -f`. This

option performs sanity checks before writing the file to avoid errors that prevent sudo from working.

Errors in the `sudoers` file can result in losing the ability to elevate privileges to root. You can fix this issue only by power cycling the switch and booting into single user mode. Before modifying `sudoers`, enable the root user by setting a password for the root user.

By default, users in the *sudo* group can use `sudo` to execute privileged commands. To add users to the sudo group, use the `useradd(8)` or `usermod(8)` command. To see which users belong to the sudo group, see `/etc/group` (`man group(5)`).

You can run any command as `sudo`, including `su`. A password is required.

The example below shows how to use `sudo` as a non-privileged user *cumulus* to bring up an interface:

```
cumulus@switch:~$ ip link show dev swp1
```

```
3: swp1: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast master
br0 state DOWN mode DEFAULT qlen 500
link/ether 44:38:39:00:27:9f brd ff:ff:ff:ff:ff:ff

cumulus@switch:~$ ip link set dev swp1 up
RTNETLINK answers: Operation not permitted

cumulus@switch:~$ sudo ip link set dev swp1 up
Password:

umulus@switch:~$ ip link show dev swp1
3: swp1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
pfifo_fast master br0 state UP mode DEFAULT qlen 500
link/ether 44:38:39:00:27:9f brd ff:ff:ff:ff:ff:ff
```

sudoers Examples

The following examples show how you grant as few privileges as necessary to a user or group of users to allow them to perform the required task. For each example, the system group *noc* is used; groups are prefixed with an %.

When executed by an unprivileged user, the example commands below must be prefixed with `sudo`.

Category	Privilege	Example Command	sudoers Entry
Monitoring	Switch port information	ethtool -m swp1	%noc ALL=(ALL) NOPASSWD:/sbin/ethtool
Monitoring	System diagnostics	cl-support	%noc ALL=(ALL) NOPASSWD:/usr/cumulus/bin/cl-support
Monitoring	Routing diagnostics	cl-resource-query	%noc ALL=(ALL) NOPASSWD:/usr/cumulus/bin/cl-resource-query
Image management	Install images	onie-select http://lab/install.bin	%noc ALL=(ALL) NOPASSWD:/usr/cumulus/bin/onie-select
Package management	Any apt-get command	apt-get update or apt-get install	%noc ALL=(ALL) NOPASSWD:/usr/bin/apt-get
Package management	Just apt-get update	apt-get update	%noc ALL=(ALL) NOPASSWD:/usr/bin/apt-get

Category	Privilege	Example Command	sudoers Entry
			update
Package management	Install packages	apt-get install vim	%noc ALL=(ALL) NOPASSWD:/usr/bin/apt-get install *
Package management	Upgrading	apt-get upgrade	%noc ALL=(ALL) NOPASSWD:/usr/bin/apt-get upgrade
Netfilter	Install ACL policies	cl-acltool -i	%noc ALL=(ALL) NOPASSWD:/usr/cumulus/bin/cl-acltool
Netfilter	List iptables rules	iptables -L	%noc ALL=(ALL) NOPASSWD:/sbin/iptables
L1 + 2 features	Any LLDP command	lldpcli show neighbors / configure	%noc ALL=(ALL) NOPASSWD:/usr/sbin/lldpcli
L1 + 2 features	Just show neighbors	lldpcli show neighbors	%noc ALL=(ALL) NOPASSWD:/usr/sbin/lldpcli

Category	Privilege	Example Command	sudoers Entry
			show neighbors*
Interfaces	Modify any interface	ip link set dev swp1 {up	down}
Interfaces	Up any interface	ifup swp1	%noc ALL=(ALL) NOPASSWD:/sbin/ifup
Interfaces	Down any interface	ifdown swp1	%noc ALL=(ALL) NOPASSWD:/sbin/ifdown
Interfaces	Up/down only swp2	ifup swp2 / ifdown swp2	%noc ALL=(ALL) NOPASSWD:/sbin/ifup swp2,/sbin/ifdown swp2
Interfaces	Any IP address change	ip addr {add	del} 192.0.2.1/30 dev swp1
Interfaces	Only set IP address	ip addr add 192.0.2.1/30 dev swp1	%noc ALL=(ALL) NOPASSWD:/sbin/ip addr add *
Ethernet bridging	Any bridge command	brctl addbr br0 / brctl	%noc ALL=(ALL)

Category	Privilege	Example Command	sudoers Entry
		delif br0 swp1	NOPASSWD:/sbin/brctl
Ethernet bridging	Add bridges and interfaces	brctl addbr br0 / brctl addif br0 swp1	%noc ALL=(ALL) NOPASSWD:/sbin/brctl addbr *,/sbin/brctl addif *
Spanning tree	Set STP properties	mstpctl setmaxage br2 20	%noc ALL=(ALL) NOPASSWD:/sbin/mstpctl
Troubleshooting	Restart switchd	systemctl restart switchd.service	%noc ALL=(ALL) NOPASSWD:/usr/sbin/service switchd *
Troubleshooting	Restart any service	systemctl cron switchd.service	%noc ALL=(ALL) NOPASSWD:/usr/sbin/service
Troubleshooting	Packet capture	tcpdump	%noc ALL=(ALL) NOPASSWD:/usr/sbin/tcpdump
L3	Add static routes	ip route add 10.2.0.0/16	%noc ALL=(ALL)

Category	Privilege	Example Command	sudoers Entry
		via 10.0.0.1	NOPASSWD:/bin/ip route add *
L3	Delete static routes	ip route del 10.2.0.0/16 via 10.0.0.1	%noc ALL=(ALL) NOPASSWD:/bin/ip route del *
L3	Any static route change	ip route *	%noc ALL=(ALL) NOPASSWD:/bin/ip route *
L3	Any iproute command	ip *	%noc ALL=(ALL) NOPASSWD:/bin/ip
L3	Non-modal OSPF	cl-ospf area 0.0.0.1 range 10.0.0.0/24	%noc ALL=(ALL) NOPASSWD:/usr/bin/cl-ospf

Related Information

- Debian wiki - sudo
- Adding Yourself to sudoers

LDAP Authentication and Authorization

Cumulus Linux uses Pluggable Authentication Modules (PAM) and Name Service Switch (NSS) for user authentication. NSS enables PAM to use LDAP to provide user authentication, group mapping, and information for other services on the system.

- NSS specifies the order of the information sources that are used to resolve names for each service. Using NSS with authentication and authorization provides the order and location for user lookup and group mapping on the system.
- PAM handles the interaction between the user and the system, providing login handling, session setup, authentication of users, and authorization of user actions.

> **NOTE**
>
> There are three common ways to configure LDAP authentication on Linux: you can use `libnss-ldap`, `libnss-ldapd`, or `libnss-sss`. This chapter describes `libnss-ldapd` only. From internal testing, this library worked best with Cumulus Linux and is the easiest to configure, automate, and troubleshoot.

Install libnss-ldapd

The `libldap-2.4-2` and `libldap-common` LDAP packages are already installed on the Cumulus Linux image; however you need to install these additional packages to use LDAP authentication:

- `libnss-ldapd`
- `libpam-ldapd`
- `ldap-utils`

To install the additional packages, run the following command:

```
cumulus@switch:~$ sudo apt-get install libnss-ldapd libpam-
ldapd ldap-utils nslcd
```

You can also install these packages even if the switch is not connected to the internet, as they are contained in the `cumulus-local-apt-archive` repository that is embedded in the Cumulus Linux image.

Follow the interactive prompts to specify the LDAP URI, search base distinguished name (DN), and services that must have LDAP lookups enabled. You need to select at least the `passwd`, `group`, and `shadow` services (press space to select a service). When done, click OK. This creates a very basic LDAP configuration using anonymous bind and initiates user search under the base DN specified.

After the dialog closes, the install process prints information similar to the following:

```
/etc/nsswitch.conf: enable LDAP lookups for group
/etc/nsswitch.conf: enable LDAP lookups for passwd
/etc/nsswitch.conf: enable LDAP lookups for shadow
```

After the installation is complete, the *name service caching daemon* (`nslcd`) runs. This service handles all the LDAP protocol interactions and caches information returned from the LDAP server. `ldap` is appended in the `/etc/nsswitch.conf` file, as is the secondary information source for *passwd*, *group*, and *shadow*. The local files (`/etc/passwd`, `/etc/groups` and `/etc/shadow`) are used first, as specified by the `compat` source.

```
passwd: compat ldap
group: compat ldap
shadow: compat ldap
```

WARNING

Keep `compat` as the first source in NSS for *passwd*, *group*, and

shadow. This prevents you from getting locked out of the system.

Entering incorrect information during the installation process might produce configuration errors. You can correct the information after installation by editing certain configuration files.

- Edit the `/etc/nslcd.conf` file to update the LDAP URI and search base DN (see Update the nslcd.conf File, below).
- Edit the `/etc/nssswitch.conf` file to update the service selections.

Be sure to restart `netd` after editing the files.

```
cumulus@switch:~$ sudo systemctl restart netd.service
```

▼ Alternative Installation Method Using debconf-utils

Update the nslcd.conf File

After installation, update the main configuration file (`/etc/nslcd.conf`) to accommodate the expected LDAP server settings.

This section documents some of the more important options that relate to

security and how queries are handled. For details on all the available configuration options, read the nslcd.conf man page.

> **NOTE**
>
> After first editing the `/etc/nslcd.conf` file and/or enabling LDAP in the `/etc/nsswitch.conf` file, you must restart `netd` with the `sudo systemctl restart netd` command. If you disable LDAP, you need to restart the `netd` service.

Connection

The LDAP client starts a session by connecting to the LDAP server on TCP and UDP port 389 or on port 636 for LDAPS. Depending on the configuration, this connection might be unauthenticated (anonymous bind); otherwise, the client must provide a bind user and password. The variables used to define the connection to the LDAP server are the URI and bind credentials.

The URI is mandatory and specifies the LDAP server location using the FQDN or IP address. The URI also designates whether to use `ldap://` for clear text transport, or `ldaps://` for SSL/TLS encrypted transport. You can also specify an alternate port in the URI. In production environments, the LDAPS protocol is recommended so that all communications are secure.

After the connection to the server is complete, the BIND operation authenticates the session. The BIND credentials are optional, and if not

specified, an anonymous bind is assumed. This is typically not allowed in most production environments. Configure authenticated (Simple) BIND by specifying the user (`binddn`) and password (`bindpw`) in the configuration. Another option is to use SASL (Simple Authentication and Security Layer) BIND, which provides authentication services using other mechanisms, like Kerberos. Contact your LDAP server administrator for this information as it depends on the configuration of the LDAP server and the credentials that are created for the client device.

```
# The location at which the LDAP server(s) should be reachable.
uri ldaps://ldap.example.com
# The DN to bind with for normal lookups.
binddn cn=CLswitch,ou=infra,dc=example,dc=com
bindpw CuMuLuS
```

Search Function

When an LDAP client requests information about a resource, it must connect and bind to the server. Then, it performs one or more resource queries depending on the lookup. All search queries sent to the LDAP server are created using the configured search *base*, *filter*, and the desired entry (*uid=myuser*) being searched. If the LDAP directory is large, this search might take a significant amount of time. It is a good idea to define a more specific search base for the common *maps* (*passwd* and *group*).

```
# The search base that will be used for all queries.
base dc=example,dc=com
# Mapped search bases to speed up common queries.
base passwd ou=people,dc=example,dc=com
base group ou=groups,dc=example,dc=com
```

Search Filters

It is also common to use search filters to specify criteria used when searching for objects within the directory. This is used to limit the search scope when authenticating users. The default filters applied are:

```
filter passwd (objectClass=posixAccount)
filter group (objectClass=posixGroup)
```

Attribute Mapping

The *map* configuration allows you to override the attributes pushed from LDAP. To override an attribute for a given *map*, specify the attribute name and the new value. This is useful to ensure that the shell is *bash* and the home directory is `/home/cumulus`:

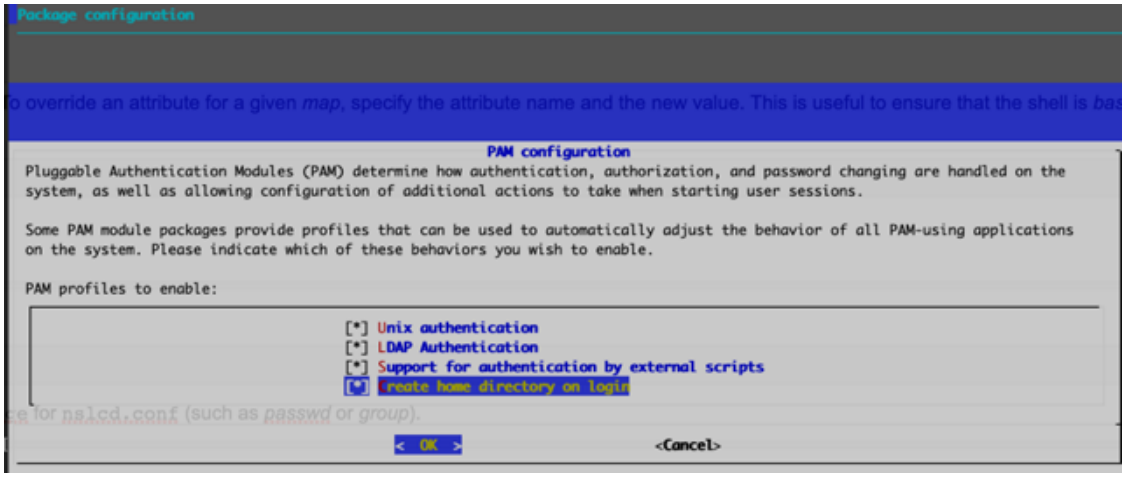
```
map    passwd homeDirectory "/home/cumulus"
map    passwd shell "/bin/bash"
```

> **NOTE**
>
> In LDAP, the ***map*** refers to one of the supported maps specified in the `manpage` for `nslcd.conf` (such as *passwd* or *group*).

Create Home Directory on Login

If you want to use unique home directories, run the `sudo pam-auth-update` command and select `Create home directory on login` in the PAM configuration dialog (press the space bar to select the option). Select OK, then press Enter to save the update and close the dialog.

```
cumulus@switch:~$ sudo pam-auth-update
```



The home directory for any user that logs in (using LDAP or not) is created and populated with the standard dotfiles from `/etc/skel` if it does not already exist.

NOTE

When `nslcd` starts, you might see an error message similar to the following (where 5816 is the `nslcd` PID):

```
nslcd[5816]: unable to dlopen /usr/lib/x86_64-linux-gnu/
sasl2/libsasldb.so: libdb-5.3.so: cannot open
shared object file: No such file or directory
```

You can safely ignore this message. The `libdb` package and resulting log messages from `nslcd` do not cause any issues when you use LDAP as a client for login and authentication.

Example Configuration

Here is an example configuration using Cumulus Linux.

```
# /etc/nslcd.conf
# nslcd configuration file. See nslcd.conf(5)
# for details.

# The user and group nslcd should run as.
uid nslcd
gid nslcd

# The location at which the LDAP server(s) should be reachable.
uri ldaps://myadserver.rtp.example.test

# The search base that will be used for all queries.
base ou=support,dc=rtp,dc=example,dc=test

# The LDAP protocol version to use.
#ldap_version 3

# The DN to bind with for normal lookups.
# defconf-set-selections doesn't seem to set this. so have to
manually set this.
binddn CN=cumulus admin,CN=Users,DC=rtp,DC=example,DC=test
```

```
bindpw 1Q2w3e4r!


# The DN used for password modifications by root.
#rootpwmoddn cn=admin,dc=example,dc=com


# SSL options
#ssl off (default)
# Not good does not prevent man in the middle attacks
#tls_reqcert demand(default)
tls_cacertfile /etc/ssl/certs/rtp-example-ca.crt


# The search scope.
#scope sub


# Add nested group support
# Supported in nslcd 0.9 and higher.
# default wheezy install of nslcd supports on 0.8. wheezy-
backports has 0.9
nss_nested_groups yes


# Mappings for Active Directory
# (replace the SIDs in the objectSid mappings with the value
for your domain)
# "dsquery * -filter (samaccountname=testuser1) -attr
```

```
ObjectSID" where cn == 'testuser1'
pagesize 1000
referrals off
idle_timelimit 1000

# Do not allow uids lower than 100 to login (aka Administrator)
# not needed as pam already has this support
# nss_min_uid 1000

# This filter says to get all users who are part of the
cumuluslnxadm group. Supports nested groups.
# Example, mary is part of the snrnetworkadm group which is
part of cumuluslnxadm group
# Ref: http://msdn.microsoft.com/en-us/library/
aa746475%28VS.85%29.aspx (LDAP_MATCHING_RULE_IN_CHAIN)
filter passwd
(&(Objectclass=user)(!(objectClass=computer))(memberOf:1.2.840.113556.1.4.1941:=cn
map    passwd uid           sAMAccountName
map    passwd uidNumber
objectSid:S-1-5-21-1391733952-3059161487-1245441232
map    passwd gidNumber
objectSid:S-1-5-21-1391733952-3059161487-1245441232
map    passwd homeDirectory "/home/$sAMAccountName"
map    passwd gecos         displayName
```

```
map    passwd loginShell    "/bin/bash"

# Filter for any AD group or user in the baseDN. the reason for
filtering for the
# user to make sure group listing for user files don't say
'<user> <gid>'. instead will say '<user> <user>'
# So for cosmetic reasons..nothing more.
filter group
(&(|(objectClass=group)(Objectclass=user))(!(objectClass=computer)))
map    group gidNumber
objectSid:S-1-5-21-1391733952-3059161487-1245441232
map    group cn            sAMAccountName
```

Configure LDAP Authorization

Linux uses the *sudo* command to allow non-administrator users (such as the default *cumulus* user account) to perform privileged operations. To control the users authorized to use sudo, the `/etc/sudoers` file and files located in the `/etc/sudoers.d/` directory define a series of rules. Typically, the rules are based on groups, but can also be defined for specific users. You can add sudo rules using the group names from LDAP. For example, if a group of users are associated with the group *netadmin*, you can add a rule to give those users sudo privileges. Refer to the sudoers manual (`man sudoers`) for a complete usage description. The following shows an example

in the `/etc/sudoers` file:

```
# The basic structure of a user specification is "who where =
(as_whom) what ".
%sudo ALL=(ALL:ALL) ALL
%netadmin ALL=(ALL:ALL) ALL
```

Active Directory Configuration

Active Directory (AD) is a fully featured LDAP-based NIS server create by Microsoft. It offers unique features that classic OpenLDAP servers do not have. AD can be more complicated to configure on the client and each version works a little differently with Linux-based LDAP clients. Some more advanced configuration examples, from testing LDAP clients on Cumulus Linux with Active Directory (AD/LDAP), are available in our knowledge base.

LDAP Verification Tools

Typically, password and group information is retrieved from LDAP and cached by the LDAP client daemon. To test the LDAP interaction, you can use these command-line tools to trigger an LDAP query from the device. This helps to create the best filters and verify the information sent back from the LDAP server.

Identify a User with the id Command

The `id` command performs a username lookup by following the lookup information sources in NSS for the *passwd* service. This simply returns the user ID, group ID and the group list retrieved from the information source. In the following example, the user *cumulus* is locally defined in `/etc/passwd`, and *myuser* is on LDAP. The NSS configuration has the `passwd` map configured with the sources `compat ldap`:

```
cumulus@switch:~$ id cumulus
uid=1000(cumulus) gid=1000(cumulus)
groups=1000(cumulus),24(cdrom),25(floppy),27(sudo),29(audio),30(dip),44(video),46(
cumulus@switch:~$ id myuser
uid=1230(myuser) gid=3000(Development)
groups=3000(Development),500(Employees),27(sudo)
```

getent

The `getent` command retrieves all records found with NSS for a given map. It can also retrieve a specific entry under that map. You can perform tests with the `passwd`, `group`, `shadow`, or any other map configured in the `/etc/nsswitch.conf` file. The output from this command is formatted according to the map requested. For the `passwd` service, the structure of the output is the same as the entries in `/etc/passwd`. The group map outputs the same structure as `/etc/group`.

In this example, looking up a specific user in the `passwd` map, the user *cumulus* is locally defined in `/etc/passwd`, and *myuser* is only in LDAP.

```
cumulus@switch:~$ getent passwd cumulus
cumulus:x:1000:1000::/home/cumulus:/bin/bash
cumulus@switch:~$ getent passwd myuser
myuser:x:1230:3000:My Test User:/home/myuser:/bin/bash
```

In the next example, looking up a specific group in the group service, the group *cumulus* is locally defined in `/etc/groups`, and *netadmin* is on LDAP.

```
cumulus@switch:~$ getent group cumulus
cumulus:x:1000:
cumulus@switch:~$ getent group netadmin
netadmin:*:502:larry,moe,curly,shemp
```

Running the command `getent passwd` or `getent group` without a specific request returns **all** local and LDAP entries for the *passwd* and *group* maps.

LDAP search

The `ldapsearch` command performs LDAP operations directly on the LDAP server. This does not interact with NSS. This command helps display what the LDAP daemon process is receiving back from the server. The command has many options. The simplest option uses anonymous bind to the host

and specifies the search DN and the attribute to look up.

```
cumulus@switch:~$ ldapsearch -H ldap://ldap.example.com -b
dc=example,dc=com -x uid=myuser
```

▼ Click to expand the command output ...

NCLU

To use NCLU, a user must be in either the `netshow` or `netedit` NCLU group in the LDAP database. You can either:

- Add a user or one of their groups to the `/etc/netd.conf` file manually.
- Add a user to the local `/etc/group` file as a member of the `netshow` or `netedit` groups.

In the following example, a user that is *not* in the `netshow` or `netedit` NCLU group in the LDAP database runs the NCLU `net show version` command, which produces an error:

```
hsolo@switch:~$ net show version
ERROR: 'getpwuid(): uid not found: 0922'
See /var/log/netd.log for more details
```

To add user to the `netshow` or `netedit` NCLU group in the LDAP database,

either edit the `/etc/group` file manually or use the `sudo adduser USERNAME netshow` command, then restart `netd`. For example, to add the user bill to the `netshow` group:

```
cumulus@switch:~$ sudo adduser hsolo netshow
Adding user `hsolo' to group `netshow' ...
Adding user hsolo to group netshow
Done.

cumulus@switch:~$ sudo systemctl restart netd
```

Now, the user can run the NCLU `net show` commands successfully:

```
hsolo@switch:~$ net show version
NCLU_VERSION=1.0-cl4u5
DISTRIB_ID="Cumulus Linux"
DISTRIB_RELEASE=4.1.0
DISTRIB_DESCRIPTION="Cumulus Linux 4.1.0"
```

LDAP Browsers

There are several GUI LDAP clients available that help you work with LDAP servers. These are free tools that show the structure of the LDAP database graphically.

- Apache Directory Studio
- LDAPManager

Troubleshooting

nslcd Debug Mode

When setting up LDAP authentication for the first time, turn off the `nslcd` service using the `systemctl stop nslcd.service` command (or the `systemctl stop nslcd@mgmt.service` if you are running the service in a management VRF) and run it in debug mode. Debug mode works whether you are using LDAP over SSL (port 636) or an unencrypted LDAP connection (port 389).

```
cumulus@switch:~$ sudo systemctl stop nslcd.service
cumulus@switch:~$ sudo nslcd -d
```

After you enable debug mode, run the following command to test LDAP queries:

```
cumulus@switch:~$ getent passwd
```

If LDAP is configured correctly, the following messages appear after you run the `getent` command:

```
nslcd: DEBUG: accept() failed (ignored): Resource temporarily
unavailable
nslcd: [8e1f29] DEBUG: connection from pid=11766 uid=0 gid=0
nslcd: [8e1f29] <passwd(all)> DEBUG:
myldap_search(base="dc=example,dc=com",
filter="(objectClass=posixAccount)")
nslcd: [8e1f29] <passwd(all)> DEBUG: ldap_result():
uid=myuser,ou=people,dc=example,dc=com
nslcd: [8e1f29] <passwd(all)> DEBUG: ldap_result(): ... 152
more results
nslcd: [8e1f29] <passwd(all)> DEBUG: ldap_result(): end of
results (162 total)
```

In the output above, `<passwd(all)>` indicates that the entire directory structure is queried.

You can query a specific user with the following command:

```
cumulus@switch:~$ getent passwd myuser
```

You can replace `myuser` with any username on the switch. The following debug output indicates that user `myuser` exists:

```
nslcd: DEBUG: add_uri(ldap://10.50.21.101)
nslcd: version 0.8.10 starting
nslcd: DEBUG: unlink() of /var/run/nslcd/socket failed
(ignored): No such file or directory
nslcd: DEBUG: setgroups(0,NULL) done
nslcd: DEBUG: setgid(110) done
nslcd: DEBUG: setuid(107) done
nslcd: accepting connections
nslcd: DEBUG: accept() failed (ignored): Resource temporarily
unavailable
nslcd: [8b4567] DEBUG: connection from pid=11369 uid=0 gid=0
nslcd: [8b4567] <passwd="myuser"> DEBUG:
myldap_search(base="dc=cumulusnetworks,dc=com",
filter="(&(objectClass=posixAccount)(uid=myuser))")
nslcd: [8b4567] <passwd="myuser"> DEBUG:
ldap_initialize(ldap://<ip_address>)
nslcd: [8b4567] <passwd="myuser"> DEBUG: ldap_set_rebind_proc()
nslcd: [8b4567] <passwd="myuser"> DEBUG:
ldap_set_option(LDAP_OPT_PROTOCOL_VERSION,3)
nslcd: [8b4567] <passwd="myuser"> DEBUG:
ldap_set_option(LDAP_OPT_DEREF,0)
nslcd: [8b4567] <passwd="myuser"> DEBUG:
ldap_set_option(LDAP_OPT_TIMELIMIT,0)
nslcd: [8b4567] <passwd="myuser"> DEBUG:
```

```
ldap_set_option(LDAP_OPT_TIMEOUT,0)
nslcd: [8b4567] <passwd="myuser"> DEBUG:
ldap_set_option(LDAP_OPT_NETWORK_TIMEOUT,0)
nslcd: [8b4567] <passwd="myuser"> DEBUG:
ldap_set_option(LDAP_OPT_REFERRALS,LDAP_OPT_ON)
nslcd: [8b4567] <passwd="myuser"> DEBUG:
ldap_set_option(LDAP_OPT_RESTART,LDAP_OPT_ON)
nslcd: [8b4567] <passwd="myuser"> DEBUG:
ldap_simple_bind_s(NULL,NULL) (uri="ldap://<ip_address>")
nslcd: [8b4567] <passwd="myuser"> DEBUG: ldap_result(): end of
results (0 total)
```

Common Problems

SSL/TLS

- The FQDN of the LDAP server URI does not match the FQDN in the CA-signed server certificate exactly.
- `nslcd` cannot read the SSL certificate and reports a *Permission denied* error in the debug during server connection negotiation. Check the permission on each directory in the path of the root SSL certificate. Ensure that it is readable by the `nslcd` user.

NSCD

- If the `nscd cache` daemon is also enabled and you make some changes to

the user from LDAP, you can clear the cache using the following commands:

```
nscd --invalidate = passwd
nscd --invalidate = group
```

- The `nscd` package works with `nslcd` to cache name entries returned from the LDAP server. This might cause authentication failures. To work around these issues, disable `nscd`, restart the `nslcd` service, then retry authentication:

```
cumulus@switch:~$ sudo nscd -K
cumulus@switch:~$ sudo systemctl restart nslcd.service
```

NOTE

If you are running the `nslcd` service in a management VRF, you need to run the `systemctl restart nslcd@mgmt.service` command instead of the `systemctl restart nslcd.service` command. For example:

```
cumulus@switch:~$ sudo nscd -K
cumulus@switch:~$ sudo systemctl restart
nslcd@mgmt.service
```

LDAP

- The search filter returns incorrect results. Check for typos in the search filter. Use `ldapsearch` to test your filter.
- Optionally, configure the basic LDAP connection and search parameters in `/etc/ldap/ldap.conf`.

```
# ldapsearch -D 'cn=CLadmin' -w 'CuMuLuS'
"(&(ObjectClass=inetOrgUser)(uid=myuser))"
```

- When a local username also exists in the LDAP database, the order of the information sources in `/etc/nsswitch` can be updated to query LDAP before the local user database. This is generally not recommended. For example, the configuration below ensures that LDAP is queried before the local database.

```
# /etc/nsswitch.conf

passwd:       ldap compat
```

Related Information

- Debian - configuring LDAP authentication
- Debian - configuring PAM to use LDAP
- GitHub - Arthur de Jong nslcd.conf file
- Debian backports

TACACS+

Cumulus Linux implements TACACS+ client AAA (Accounting, Authentication, and Authorization) in a transparent way with minimal configuration. The client implements the TACACS+ protocol as described in this IETF document. There is no need to create accounts or directories on the switch. Accounting records are sent to all configured TACACS+ servers by default. Use of per-command authorization requires additional setup on the switch.

Supported Features

- Authentication using PAM; includes `login`, `ssh`, `sudo` and `su`
- Runs over the eth0 management interface
- Ability to run in the management VRF
- TACACS+ privilege 15 users can run any command with sudo using the `/etc/sudoers.d/tacplus` file that is installed by the `libtacplus-map1` package
- Up to seven TACACS+ servers

Install the TACACS+ Client Packages

You can install the TACACS+ packages even if the switch is not connected to the internet, as they are contained in the `cumulus-local-apt-archive` repository that is embedded in the Cumulus Linux image.

To install all required packages, run these commands:

```
cumulus@switch:~$ sudo -E apt-get update
cumulus@switch:~$ sudo -E apt-get install tacplus-client
```

Configure the TACACS+ Client

After installing TACACS+, edit the `/etc/tacplus_servers` file to add at least one server and one shared secret (key). You can specify the server and secret parameters in any order anywhere in the file. Whitespace (spaces or tabs) are not allowed. For example, if your TACACS+ server IP address is `192.168.0.30` and your shared secret is `tacacskey`, add these parameters to the `/etc/tacplus_servers` file:

```
secret=tacacskey
server=192.168.0.30
```

Cumulus Linux supports a maximum of seven TACACS+ servers. To specify multiple servers, add one per line to the `/etc/tacplus_servers` file.

Connections are made in the order in which they are listed in this file. In most cases, you do not need to change any other parameters. You can add parameters used by any of the packages to this file, which affects all the TACACS+ client software. For example, the timeout value for NSS lookups (see description below) is set to 5 seconds by default in the `/etc/`

`tacplus_nss.conf` file, whereas the timeout value for other packages is 10 seconds and is set in the `/etc/tacplus_servers` file. The timeout value is per connection to the TACACS+ servers. (If authorization is configured per command, the timeout occurs for *each* command.) There are several (typically four) connections to the server per login attempt from PAM, as well as two or more through NSS. Therefore, with the default timeout values, a TACACS+ server that is not reachable can delay logins by a minute or more per unreachable server. If you must list unreachable TACACS+ servers, place them at the end of the server list and consider reducing the timeout values.

When you add or remove TACACS+ servers, you must restart `auditd` (with the `systemctl restart auditd` command) or you must send a signal (with `killall -HUP audisp-tacplus`) before `audisp-tacplus` rereads the configuration to see the changed server list.

You can also configure the IP address used as the source IP address when communicating with the TACACS+ server. See TACACS Configuration Parameters below for the full list of TACACS+ parameters.

Following is the complete list of the TACACS+ client configuration files, and their use.

Filename	Description
`/etc/tacplus_servers`	This is the primary file that requires configuration after installation. The file is used by all packages with `include=/etc/`

Filename	Description
	`tacplus_servers` parameters in the other configuration files that are installed. Typically, this file contains the shared secrets; make sure that the Linux file mode is 600.
`/etc/nsswitch.conf`	When the `libnss_tacplus` package is installed, this file is configured to enable tacplus lookups via `libnss_tacplus`. If you replace this file by automation or other means, you need to add tacplus as the first lookup method for the *passwd* database line.
`/etc/tacplus_nss.conf`	This file sets the basic parameters for `libnss_tacplus`. It includes a debug variable for debugging NSS lookups separately from other client packages.
`/usr/share/pam-configs/tacplus`	This is the configuration file for `pam-auth-update` to generate the files in the next row. These configurations are used at `login`, by `su`, and by `ssh`.
`/etc/pam.d/common-*`	The `/etc/pam.d/common-*` files are updated for `tacplus` authentication. The files are updated with `pam-auth-update`,

Filename	Description
	when `libpam-tacplus` is installed or removed.
`/etc/sudoers.d/tacplus`	This file allows TACACS+ privilege level 15 users to run commands with `sudo`. The file includes an example (commented out) of how to enable privilege level 15 TACACS users to use `sudo` without having to enter a password and provides an example of how to enable all TACACS users to run specific commands with sudo. Only edit this file with the `visudo -f /etc/sudoers.d/tacplus` command.
`/etc/audisp/plugins.d/audisp-tacplus.conf`	This is the `audisp` plugin configuration file. Typically, no modifications are required.
`/etc/audisp/audisp-tac_plus.conf`	This is the TACACS+ server configuration file for accounting. Typically, no modifications are required. You can use this configuration file when you only want to debug TACACS+ accounting issues, not all TACACS+ users.
`/etc/audit/rules.d/audisp-tacplus.rules`	The `auditd` rules for TACACS+ accounting. The `augenrules` command uses all rule files to

Filename	Description
	generate the rules file (described below).
`/etc/audit/audit.rules`	This is the audit rules file generated when `auditd` is installed.

WARNING

You can edit the `/etc/pam.d/common-*` files manually. However, if you run `pam-auth-update` again after making the changes, the update fails. Only perform configuration in `/usr/share/pam-configs/tacplus`, then run `pam-auth-update`.

TACACS+ Authentication (login)

The initial authentication configuration is done through the PAM modules and an updated version of the `libpam-tacplus` package. When the package is installed, the PAM configuration is updated in `/etc/pam.d` with the `pam-auth-update` command. If you have made changes to your PAM configuration, you need to integrate these changes yourself. If you are also using LDAP with the `libpam-ldap` package, you might need to edit the PAM configuration to ensure the LDAP and TACACS ordering that you prefer.

The `libpam-tacplus` are configured to skip over rules and the values in the `success=2` might require adjustments to skip over LDAP rules.

A user privilege level is determined by the TACACS+ privilege attribute `priv_lvl` for the user that is returned by the TACACS+ server during the user authorization exchange. The client accepts the attribute in either the mandatory or optional forms and also accepts `priv-lvl` as the attribute name. The attribute value must be a numeric string in the range 0 to 15, with 15 the most privileged level.

> **NOTE**
>
> By default, TACACS+ users at privilege levels other than 15 are not allowed to run `sudo` commands and are limited to commands that can be run with standard Linux user permissions.

TACACS+ Client Sequencing

Due to SSH and login processing mechanisms, Cumulus Linux needs to know the following very early in the AAA sequence:

- Whether the user is a valid TACACS+ user
- The user's privilege level

The only way to do this for non-local users — that is, users not present in the local password file — is to send a TACACS+ authorization request as the first communication with the TACACS+ server, prior to the authentication

and before a password is requested from the user logging in.

Some TACACS+ servers need special configuration to allow authorization requests prior to authentication. Contact your TACACS+ server vendor for the proper configuration if your TACACS+ server does not allow the initial authorization request.

Local Fallback Authentication

If a site wants to allow local fallback authentication for a user when none of the TACACS servers can be reached you can add a privileged user account as a local account on the switch.

To configure local fallback authentication:

1. Edit the `/etc/nsswitch.conf` file to remove the keyword `tacplus` from the line starting with `passwd`. (You need to add the keyword back in step 3.)

   An example of the `/etc/nsswitch.conf` file with the keyword `tacplus` removed from the line starting with `passwd` is shown below.

```
cumulus@switch:~$ sudo vi /etc/nsswitch.conf
#
# Example configuration of GNU Name Service Switch
functionality.
# If you have the `glibc-doc-reference' and `info' packages
```

```
installed, try:
# `info libc "Name Service Switch"' for information about
this file.

passwd:         files
group:          tacplus files
shadow:         files
gshadow:        files
...
```

2. To enable the local privileged user to run `sudo` and NCLU commands, run the `adduser` commands shown below. In the example commands, the TACACS account name is tacadmin.

> **NOTE**
>
> The first `adduser` command prompts for information and a password. You can skip most of the requested information by pressing ENTER.

```
cumulus@switch:~$ sudo adduser --ingroup tacacs tacadmin
```

```
cumulus@switch:~$ sudo adduser tacadmin netedit
cumulus@switch:~$ sudo adduser tacadmin sudo
```

3. Edit the `/etc/nsswitch.conf` file to add the keyword `tacplus` back to the line starting with `passwd` (the keyword you removed in the first step).

4. Restart the `netd` service with the following command:

```
cumulus@switch:~$ sudo systemctl restart netd
```

TACACS+ Accounting

TACACS+ accounting is implemented with the `audisp` module, with an additional plugin for `auditd`/`audisp`. The plugin maps the auid in the accounting record to a TACACS login, based on the `auid` and `sessionid`. The `audisp` module requires `libnss_tacplus` and uses the `libtacplus_map.so` library interfaces as part of the modified lipam_tacplus` package.

Communication with the TACACS+ servers is done with the `libsimple-tacact1` library, through `dlopen()`. A maximum of 240 bytes of command name and arguments are sent in the accounting record, due to the TACACS+ field length limitation of 255 bytes.

> **NOTE**
>
> All Linux commands result in an accounting record, including commands run as part of the login process or as sub-processes of other commands. This can sometimes generate a large number of accounting records.

Configure the IP address and encryption key of the server in the `/etc/tacplus_servers` file. Minimal configuration to `auditd` and `audisp` is necessary to enable the audit records necessary for accounting. These records are installed as part of the package.

`audisp-tacplus` installs the audit rules for command accounting. Modifying the configuration files is not usually necessary. However, when a management VRF is configured, the accounting configuration does need special modification because the `auditd` service starts prior to networking. It is necessary to add the *vrf* parameter and to signal the `audisp-tacplus` process to reread the configuration. The example below shows that the management VRF is named *mgmt*. You can place the *vrf* parameter in either the `/etc/tacplus_servers` file or in the `/etc/audisp/audisp-tac_plus.conf` file.

```
vrf=mgmt
```

After editing the configuration file, send the **HUP** signal `killall -HUP audisp-tacplus` to notify the accounting process to reread the file.

> **NOTE**
>
> All `sudo` commands run by TACACS+ users generate accounting records against the original TACACS+ login name.

For more information, refer to the `audisp.8` and `auditd.8` man pages.

Configure NCLU for TACACS+ Users

When you install or upgrade TACACS+ packages, mapped user accounts are created automatically. All *tacacs0* through *tacacs15* users are added to the *netshow* group.

For any TACACS+ users to execute `net add`, `net del`, and `net commit` commands and to restart services with NCLU, you need to add those users to the `users_with_edit` variable in the `/etc/netd.conf` file. Add the *tacacs15* user and, depending upon your policies, other users (*tacacs1* through *tacacs14*) to this variable.

To give a TACACS+ user access to the show commands, add the *tacacs* group to the `groups_with_show` variable.

WARNING

Do not add the *tacacs* group to the `groups_with_edit` variable; this is dangerous and can potentially enable any user to log into the switch as the root user.

To add the users, edit the `/etc/netd.conf` file:

```
cumulus@switch:~$ sudo nano /etc/netd.conf
...
# Control which users/groups are allowed to run "add", "del",
# "clear", "abort", and "commit" commands.
users_with_edit = root, cumulus, tacacs15
groups_with_edit = netedit

# Control which users/groups are allowed to run "show" commands
users_with_show = root, cumulus
groups_with_show = netshow, netedit, tacacs
...
```

After you save and exit the `netd.conf` file, restart the `netd` service. Run:

```
cumulus@switch:~$ sudo systemctl restart netd
```

TACACS+ Per-command Authorization

The `tacplus-auth` command handles the per-command authorization. To make this an enforced authorization, you must change the TACACS+ login to use a restricted shell, with a very limited executable search path. Otherwise, the user can bypass the authorization. The `tacplus-restrict` utility simplifies the setup of the restricted environment. The example below initializes the environment for the *tacacs0* user account. This is the account used for TACACS+ users at privilege level `0`.

```
tacuser0@switch:~$ sudo tacplus-restrict -i -u tacacs0 -a
command1 command2 ... commandN
```

If the user/command combination is not authorized by the TACACS+ server, a message similar to the following displays:

```
tacuser0@switch:~$ net show version
net not authorized by TACACS+ with given arguments, not
executing
```

The following table provides the command options:

Option	Description
`-i`	Initializes the environment. You only need to issue this option once per username.
`-a`	You can invoke the utility with the -a option as many times as desired. For each command in the -a list, a symbolic link is created from tacplus-auth to the relative portion of the command name in the local bin subdirectory. You also need to enable these commands on the TACACS+ server (refer to the TACACS+ server documentation). It is common to have the server allow some options to a command, but not others.
`-f`	Re-initializes the environment. If you need to restart, issue the -f option with -i to force the re-initialization; otherwise, repeated use of -i is ignored. As part of the initialization: - The user's shell is changed to /bin/rbash. - Any existing dot files are saved. - A limited environment is set up that does not allow general command execution, but instead

Option	Description
	allows only commands from the user's local bin subdirectory.

For example, if you want to allow the user to be able to run the `net` and `ip` commands (if authorized by the TACACS+ server), use the command:

```
cumulus@switch:~$ sudo tacplus-restrict -i -u tacacs0 -a ip net
```

After running this command, examine the `tacacs0` directory::

```
cumulus@switch:~$ sudo ls -lR ~tacacs0
total 12
lrwxrwxrwx 1 root root 22 Nov 21 22:07 ip -> /usr/sbin/tacplus-
auth
lrwxrwxrwx 1 root root 22 Nov 21 22:07 net -> /usr/sbin/tacplus-
auth
```

Other than shell built-ins, the only two commands the privilege level 0 TACACS users can run are the `ip` and `net` commands.

If you mistakenly add potential commands with the `-a` option, you can remove them. The example below shows how to remove the `net` command:

```
cumulus@switch:~$ sudo rm ~tacacs0/bin/net
```

You can remove all commands as follows:

```
cumulus@switch:~$ sudo rm ~tacacs0/bin/*
```

Use the `man` command on the switch for more information on `tacplus-auth` and `tacplus-restrict`.

```
cumulus@switch:~$ man tacplus-auth tacplus-restrict
```

NSS Plugin

When used with `pam_tacplus`, TACACS+ authenticated users can log in without a local account on the system using the NSS plugin that comes with the `tacplus_nss` package. The plugin uses the mapped `tacplus` information if the user is not found in the local password file, provides the `getpwnam()` and `getpwuid()`entry point,s and uses the TACACS+ authentication functions.

The plugin asks the TACACS+ server if the user is known, and then for relevant attributes to determine the privilege level of the user. When the

`libnss_tacplus` package is installed, `nsswitch.conf` is modified to set `tacplus` as the first lookup method for `passwd`. If the order is changed, lookups return the local accounts, such as `tacacs0`

If the user is not found, a mapped lookup is performed using the `libtacplus.so` exported functions. The privilege level is appended to `tacacs` and the lookup searches for the name in the local password file. For example, privilege level 15 searches for the tacacs15 user. If the user is found, the password structure is filled in with information for the user.

If the user is not found, the privilege level is decremented and checked again until privilege level 0 (user `tacacs0`) is reached. This allows use of only the two local users `tacacs0` and `tacacs15`, if minimal configuration is desired.

TACACS Configuration Parameters

The recognized configuration options are the same as the `libpam_tacplus` command line arguments; however, not all `pam_tacplus` options are supported. These configuration parameters are documented in the `tacplus_servers.5` man page, which is part of the `libpam-tacplus` package.

The table below describes the configuration options available:

Configuration Option	Description
debug	The output debugging information through syslog(3). **Note**: Debugging is heavy,

Configuration Option	Description
	including passwords. Do not leave debugging enabled on a production switch after you have completed troubleshooting.
secret=STRING	The secret key used to encrypt and decrypt packets sent to and received from the server. You can specify the secret key more than once in any order with respect to the server= parameter. When fewer secret= parameters are specified, the last secret given is used for the remaining servers. Only use this parameter in files such as /etc/tacplus_servers that are not world readable.
server=hostname server=ip-address	Adds a TACACS+ server to the servers list. Servers are queried in turn until a match is found, or no servers remain in the list. Can be specified up to 7 times. An IP address can be optionally followed by a port number, preceded by a ":". The default port is 49. **Note**: When sending accounting records, the record is sent to all servers in the list if acct_all=1, which is the default.
source_ip=ipv4-address	Sets the IP address used as the

Configuration Option	Description
	source IP address when communicating with the TACACS+ server. You must specify an IPv4 address. IPv6 addresses and hostnames are not supported. The address must must be valid for the interface being used.
timeout=seconds	TACACS+ server(s) communication timeout. This parameter defaults to 10 seconds in the /etc/tacplus_servers file, but defaults to 5 seconds in the /etc/tacplus_nss.conf file.
include=/file/name	A supplemental configuration file to avoid duplicating configuration information. You can include up to 8 more configuration files.
min_uid=value	The minimum user ID that the NSS plugin looks up. Setting it to 0 means uid 0 (root) is never looked up, which is desirable for performance reasons. The value should not be greater than the local TACACS+ user IDs (0 through 15), to ensure they can be looked up.
exclude_users=user1,user2,...	A comma-separated list of

Configuration Option	Description
	usernames that are never looked up by the NSS plugin, set in the tacplus_nss.conf file. You cannot use * (asterisk) as a wild card in the list. While it's not a legal username, bash may lookup this as a user name during pathname completion, so it is included in this list as a username string. **Note**: Do not remove the cumulus user from the exclude_users list; doing so can make it impossible to log in as the cumulus user, which is the primary administrative account in Cumulus Linux. If you do remove the cumulus user, add some other local fallback user that does not rely on TACACS but is a member of sudo and netedit groups, so that these accounts can run sudo and NCLU commands.
login=string	TACACS+ authentication service (pap, chap, or login). The default value is pap.
user_homedir=1	This is not enabled by default. When enabled, a separate home directory for each TACACS+ user is created when the TACACS+ user first logs in. By default, the home directory in the mapping

Configuration Option	Description
	accounts in /etc/passwd (/home/tacacs0 ... /home/tacacs15) is used. If the home directory does not exist, it is created with the mkhomedir_helper program, in the same way as pam_mkhomedir. This option is not honored for accounts with restricted shells when per-command authorization is enabled.
acct_all=1	Configuration option for audisp_tacplus and pam_tacplus sending accounting records to all supplied servers (1), or the first server to respond (0). The default value is 1.
timeout=seconds	Sets the timeout in seconds for connections to each TACACS+ server. The default is 10 seconds for all lookups except that NSS lookups use a 5 second timeout.
vrf=vrf-name	If the management network is in a VRF, set this variable to the VRF name. This is typically mgmt. When this variable is set, the connection to the TACACS+ accounting servers is made through the named VRF.

Configuration Option	Description
service	TACACS+ accounting and authorization service. Examples include shell, pap, raccess, ppp, and slip. The default value is shell.
protocol	TACACS+ protocol field. This option is use dependent. PAM uses the SSH protocol.

Remove the TACACS+ Client Packages

To remove all of the TACACS+ client packages, use the following commands:

```
cumulus@switch:~$ sudo -E apt-get remove tacplus-client
cumulus@switch:~$ sudo -E apt-get autoremove
```

To remove the TACACS+ client configuration files as well as the packages (recommended), use this command:

```
cumulus@switch:~$ sudo -E apt-get autoremove --purge
```

Troubleshooting

Basic Server Connectivity or NSS Issues

You can use the `getent` command to determine if TACACS+ is configured correctly and if the local password is stored in the configuration files. In the example commands below, the cumulus user represents the local user, while cumulusTAC represents the TACACS user.

To look up the username within all NSS methods:

```
cumulus@switch:~$ sudo getent passwd cumulusTAC
cumulusTAC:x:1016:1001:TACACS+ mapped user at privilege level
15,,,:/home/tacacs15:/bin/bash
```

To look up the user within the local database only:

```
cumulus@switch:~$ sudo getent -s compat passwd cumulus
cumulus:x:1000:1000:cumulus,,,:/home/cumulus:/bin/bash
```

To look up the user within the TACACS+ database only:

```
cumulus@switch:~$ sudo getent -s tacplus passwd cumulusTAC
cumulusTAC:x:1016:1001:TACACS+ mapped user at privilege level
15,,,:/home/tacacs15:/bin/bash
```

If TACACS does not appear to be working correctly, debug the following configuration files by adding the *debug=1* parameter to one or more of these files:

- `/etc/tacplus_servers`
- `/etc/tacplus_nss.conf`

> **NOTE**
>
> You can also add `debug=1` to individual pam_tacplus lines in `/etc/pam.d/common*`.

All log messages are stored in `/var/log/syslog`.

Incorrect Shared Key

The TACACS client on the switch and the TACACS server should have the same shared secret key. If this key is incorrect, the following message is printed to `syslog`:

```
2017-09-05T19:57:00.356520+00:00 leaf01 sshd[3176]:
nss_tacplus: TACACS+ server 192.168.0.254:49 read failed with
protocol error (incorrect shared secret?) user cumulus
```

Issues with Per-command Authorization

To debug TACACS user command authorization, have the TACACS+ user enter the following command at a shell prompt, then try the command again:

```
tacuser0@switch:~$ export TACACSAUTHDEBUG=1
```

When this debugging is enabled, additional information is shown for the command authorization conversation with the TACACS+ server:

```
tacuser0@switch:~$ net pending
tacplus-auth: found matching command (/usr/bin/net) request
authorization
tacplus-auth: error connecting to 10.0.3.195:49 to request
authorization for net: Transport endpoint is not connected
tacplus-auth: cmd not authorized (16)
tacplus-auth: net not authorized from 192.168.3.189:49
```

```
net not authorized by TACACS+ with given arguments, not
executing
```

```
tacuser0@switch:~$ net show version
tacplus-auth: found matching command (/usr/bin/net) request
authorization
tacplus-auth: error connecting to 10.0.3.195:49 to request
authorization for net: Transport endpoint is not connected
tacplus-auth: 192.168.3.189:49 authorized command net
tacplus-auth: net authorized, executing
DISTRIB_ID="Cumulus Linux"
DISTRIB_RELEASE=4.1.0
DISTRIB_DESCRIPTION="Cumulus Linux 4.1.0"
```

To disable debugging:

```
tacuser0@switch:~$ export -n TACACSAUTHDEBUG
```

Debug Issues with Accounting Records

If you have added or deleted TACACS+ servers from the configuration files, make sure you notify the `audisp` plugin with this command:

```
cumulus@switch:~$ sudo killall -HUP audisp-tacplus
```

If accounting records are still not being sent, add *debug=1* to the `/etc/audisp/audisp-tac_plus.conf` file, then issue the command above to notify the plugin. Ask the TACACS+ user to run a command and examine the end of `/var/log/syslog` for messages from the plugin. You can also check the auditing log file `/var/log/audit audit.log` to be sure the auditing records are being written. If they are not, restart the audit daemon with:

```
cumulus@switch:~$ sudo systemctl restart auditd.service
```

TACACS Component Software Descriptions

The following table describes the different pieces of software involved with delivering TACACS.

Package Name	Description
audisp-tacplus_1.0.0-1-cl3u3	This package uses auditing data from `auditd` to send accounting records to the TACACS+ server and is started as part of `auditd`.
libtac2_1.4.0-cl3u2	Basic TACACS+ server utility and communications routines.

Package Name	Description
libnss-tacplus_1.0.1-cl3u3	Provides an interface between `libc` username lookups, the mapping functions, and the TACACS+ server.
tacplus-auth-1.0.0-cl3u1	This package includes the `tacplus-restrict` setup utility, which enables you to perform per-command TACACS+ authorization. Per-command authorization is not done by default.
libpam-tacplus_1.4.0-1-cl3u2	A modified version of the standard Debian package.
libtacplus-map1_1.0.0-cl3u2	The mapping functionality between local and TACACS+ users on the server. Sets the immutable `sessionid` and auditing UID to ensure the original user can be tracked through multiple processes and privilege changes. Sets the auditing `loginuid` as immutable if supported. Creates and maintains a status database in `/run/tacacs_client_map` to manage and lookup mappings.
libsimple-tacacct1_1.0.0-cl3u2	Provides an interface for programs to send accounting records to the TACACS+ server. Used by `audisp-tacplus`.

Package Name	Description
libtac2-bin_1.4.0-cl3u2	Provides the `tacc` testing program and TACACS+ man page.

Considerations

TACACS+ Client Is only Supported through the Management Interface

The TACACS+ client is only supported through the management interface on the switch: eth0, eth1, or the VRF management interface. The TACACS+ client is not supported through bonds, switch virtual interfaces (SVIs), or switch port interfaces (swp).

Multiple TACACS+ Users

If two or more TACACS+ users are logged in simultaneously with the same privilege level, while the accounting records are maintained correctly, a lookup on either name will match both users, while a UID lookup will only return the user that logged in first.

This means that any processes run by either user will be attributed to both, and all files created by either user will be attributed to the first name matched. This is similar to adding two local users to the password file with the same UID and GID, and is an inherent limitation of using the UID for the base user from the password file.

> **NOTE**
>
> The current algorithm returns the first name matching the UID from the mapping file; this can be the first or the second user that logged in.

To work around this issue, you can use the switch audit log or the TACACS server accounting logs to determine which processes and files are created by each user.

- For commands that do not execute other commands (for example, changes to configurations in an editor, or actions with tools like `clagctl` and `vtysh`), no additional accounting is done.
- Per-command authorization is implemented at the most basic level (commands are permitted or denied based on the standard Linux user permissions for the local TACACS users and only privilege level 15 users can run `sudo` commands by default).

The Linux `auditd` system does not always generate audit events for processes when terminated with a signal (with the `kill` system call or internal errors such as SIGSEGV). As a result, processes that exit on a signal that is not caught and handled, might not generate a STOP accounting record.

Issues with deluser Command

TACACS+ and other non-local users that run the `deluser` command with the `--remove-home` option will see an error about not finding the user in `/etc/passwd`:

```
tacuser0@switch: deluser --remove-home USERNAME
userdel: cannot remove entry 'USERNAME' from /etc/passwd
/usr/sbin/deluser: `/usr/sbin/userdel USERNAME' returned error
code 1. Exiting
```

However, the command does remove the home directory. The user can still log in on that account, but will not have a valid home directory. This is a known upstream issue with the `deluser` command for all non-local users.

Only use the `--remove-home` option when the `user_homedir=1` configuration command is in use.

When Both TACACS+ and RADIUS AAA Clients Are Installed

When you have both the TACACS+ and the RADIUS AAA client installed, RADIUS login is not attempted. As a workaround, do not install both the TACACS+ and the RADIUS AAA client on the same switch.

RADIUS AAA

Cumulus Linux works with add-on packages that enable RADIUS you to log in to the switch in a transparent way with minimal configuration. There is no need to create accounts or directories on the switch. Authentication is handled with PAM and includes login, `ssh`, `sudo` and `su`.

Install the RADIUS Packages

You can install the RADIUS packages even if the switch is not connected to the internet, as they are contained in the `cumulus-local-apt-archive` repository that is embedded in the Cumulus Linux image.

To install the RADIUS packages:

```
cumulus@switch:~$ sudo apt-get update
cumulus@switch:~$ sudo apt-get install libnss-mapuser libpam-
radius-auth
```

After installation is complete, either reboot the switch or run the `sudo systemctl restart netd` command.

The `libpam-radius-auth` package supplied with the Cumulus Linux RADIUS client is a newer version than the one in Debian Buster. This package contains support for IPv6, the `src_ip` option described below, as well as a

number of bug fixes and minor features. The package also includes VRF support, provides man pages describing the PAM and RADIUS configuration, and sets the `SUDO_PROMPT` environment variable to the login name for RADIUS mapping support.

The `libnss-mapuser` package is specific to Cumulus Linux and supports the `getgrent`, `getgrnam` and `getgrgid` library interfaces. These interfaces add logged in RADIUS users to the group member list for groups that contain the `mapped_user` (`radius_user`) if the RADIUS account is unprivileged, and add privileged RADIUS users to the group member list for groups that contain the `mapped_priv_user` (`radius_priv_user`) during the group lookups.

During package installation:

- The PAM configuration is modified automatically using `pam-auth-update (8)`, and the NSS configuration file `/etc/nsswitch.conf` is modified to add the *mapuser* and *mapuid* plugins. If you remove or purge the packages, these files are modified to remove the configuration for these plugins.
- The `radius_shell` package is added, which installs the `/sbin/radius_shell` and `setcap cap_setuid` program used as the login shell for RADIUS accounts. The package adjusts the `UID` when needed, then runs the bash shell with the same arguments. When installed, the package changes the shell of the RADIUS accounts to `/sbin//radius_shell`, and to `/bin/shell` if the package is removed. This package is required for privileged RADIUS users to be enabled. It is not required for regular RADIUS client use.
- The `radius_user` account is added to the `netshow` group and the

`radius_priv_user` account to the `netedit` and `sudo` groups. This change enables all RADUS logins to run NCLU `net show` commands and all privileged RADIUS users to also run `net add`, `net del`, and `net commit` commands, and to use `sudo`.

Configure the RADIUS Client

To configure the RADIUS client, edit the `/etc/pam_radius_auth.conf` file:

1. Add the hostname or IP address of at least one RADIUS server (such as a *freeradius* server on Linux), and the shared secret used to authenticate and encrypt communication with each server.

> **TIP**
>
> The hostname of the switch must be resolvable to an IP address, which, in general, is fixed in DNS. If for some reason you cannot find the hostname in DNS, you can add the hostname to the `/etc/hosts` file manually. However, this can cause problems since the IP address is usually assigned by DHCP, which can change at any time.

Multiple server configuration lines are verified in the order listed. Other than memory, there is no limit to the number of RADIUS servers you can

use.

The server port number or name is optional. The system looks up the port in the `/etc/services` file. However, you can override the ports in the `/etc/pam_radius_auth.conf` file.

2. If the server is slow or latencies are high, change the `timeout` setting. The setting defaults to 3 seconds.

3. If you want to use a specific interface to reach the RADIUS server, specify the `src_ip` option. You can specify the hostname of the interface, an IPv4, or an IPv6 address. If you specify the `src_ip` option, you must also specify the `timeout` option.

4. Set the `vrf-name` field. This is typically set to *mgmt* if you are using a management VRF. You cannot specify more than one VRF.

The configuration file includes the `mapped_priv_user` field that sets the account used for privileged RADIUS users and the `priv-lvl` field that sets the minimum value for the privilege level to be considered a privileged login (the default value is 15). If you edit these fields, make sure the values match those set in the `/etc/nss_mapuser.conf` file.

The following example provides a sample `/etc/pam_radius_auth.conf` file configuration:

```
mapped_priv_user   radius_priv_user
# server[:port]    shared_secret   timeout (secs)  src_ip
192.168.0.254      secretkey
other-server       othersecret     3               192.168.1.10
# when mgmt vrf is in use
vrf-name mgmt
```

TIP

If this is the first time you are configuring the RADIUS client, uncomment the `debug` line to help with troubleshooting. The debugging messages are written to `/var/log/syslog`. When the RADIUS client is working correctly, comment out the `debug` line.

As an optional step, you can set PAM configuration keywords by editing the `/usr/share/pam-configs/radius` file. After you edit the file, you must run the `pam-auth-update --package` command. PAM configuration keywords are described in the `pam_radius_auth (8)` man page.

NOTE

The privilege level for the user on the switch is determined by the value of the VSA (Vendor Specific Attribute) `shell:priv-lvl`. If the attribute is not returned, the user is unprivileged. The following shows an example using the freeradius server for a fully-privileged user.

```
Service-Type = Administrative-User,
Cisco-AVPair = "shell:roles=network-administrator",
Cisco-AVPair += "shell:priv-lvl=15"
```

The VSA vendor name (Cisco-AVPair in the example above) can have any content. The RADIUS client only checks for the string `shell:priv-lvl`.

Enable Login without Local Accounts

Because LDAP is not commonly used with switches and adding accounts locally is cumbersome, Cumulus Linux includes a mapping capability with the `libnss-mapuser` package.

Mapping is done using two NSS (Name Service Switch) plugins, one for account name, and one for UID lookup. These accounts are configured automatically in `/etc/nsswitch.conf` during installation and are removed

when the package is removed. See the `nss_mapuser (8)` man page for the full description of this plugin.

A username is mapped at login to a fixed account specified in the configuration file, with the fields of the fixed account used as a template for the user that is logging in.

For example, if the name being looked up is *dave* and the fixed account in the configuration file is *radius_user*, and that entry in `/etc/passwd` is:

```
radius_user:x:1017:1002:radius user:/home/radius_user:/bin/bash
```

then the matching line returned by running `getent passwd dave` is:

```
cumulus@switch:~$ getent passwd dave
dave:x:1017:1002:dave mapped user:/home/dave:/bin/bash
```

The home directory `/home/dave` is created during the login process if it does not already exist and is populated with the standard skeleton files by the `mkhomedir_helper` command.

The configuration file `/etc/nss_mapuser.conf` is used to configure the plugins. The file includes the mapped account name, which is `radius_user` by default. You can change the mapped account name by editing the file. The `nss_mapuser (5)` man page describes the configuration file.

A flat file mapping is done based on the session number assigned during login, which persists across `su` and `sudo`. The mapping is removed at logout.

Local Fallback Authentication

If a site wants to allow local fallback authentication for a user when none of the RADIUS servers can be reached you can add a privileged user account as a local account on the switch. The local account must have the same unique identifier as the privileged user and the shell must be the same.

To configure local fallback authentication:

1. Add a local privileged user account. For example, if the `radius_priv_user` account in the `/etc/passwd` file is `radius_priv_user:x:1002:1001::/home/radius_priv_user:/sbin/radius_shell`, run the following command to add a local privileged user account named johnadmin:

```
cumulus@switch:~$ sudo useradd -u 1002 -g 1001 -o -s /sbin/
radius_shell johnadmin
```

2. To enable the local privileged user to run `sudo` and NCLU commands, run the following commands:

```
cumulus@switch:~$ sudo adduser johnadmin netedit
```

```
cumulus@switch:~$ sudo adduser johnadmin sudo
cumulus@switch:~$ sudo systemctl restart netd
```

3. Edit the `/etc/passwd` file to move the local user line before to the `radius_priv_user` line:

```
cumulus@switch:~$ sudo vi /etc/passwd
...
johnadmin:x:1002:1001::/home/johnadmin:/sbin/radius_shell
radius_priv_user:x:1002:1001::/home/radius_priv_user:/sbin/
radius_shell
```

4. To set the local password for the local user, run the following command:

```
cumulus@switch:~$ sudo passwd johnadmin
```

Verify RADIUS Client Configuration

To verify that the RADIUS client is configured correctly, log in as a non-privileged user and run a `net add interface` command.

In this example, the `ops` user is not a privileged RADIUS user so they cannot

add an interface.

```
ops@leaf01:~$ net add interface swp1
ERROR: User ops does not have permission to make networking
changes.
```

In this example, the `admin` user is a privileged RADIUS user (with privilege level 15) so is able to add interface swp1.

```
admin@leaf01:~$ net add interface swp1
admin@leaf01:~$ net pending
--- /etc/network/interfaces    2018-04-06 14:49:33.099331830
+0000
+++ /var/run/nclu/iface/interfaces.tmp    2018-04-06
16:01:16.057639999 +0000
@@ -3,10 +3,13 @@

  source /etc/network/interfaces.d/*.intf

  # The loopback network interface
  auto lo
  iface lo inet loopback

  # The primary network interface
```

```
  auto eth0
  iface eth0 inet dhcp
+
+auto swp1
iface swp1
...
```

Remove RADIUS Client Packages

Remove the RADIUS packages with the following command:

```
cumulus@switch:~$ sudo apt-get remove libnss-mapuser libpam-
radius-auth
```

When you remove the packages, the plugins are removed from the `/etc/nsswitch.conf` file and from the PAM files.

To remove all configuration files for these packages, run:

```
cumulus@switch:~$ sudo apt-get purge libnss-mapuser libpam-
radius-auth
```

> **NOTE**
>
> The RADIUS fixed account is not removed from the `/etc/passwd` or `/etc/group` file and the home directories are not removed. They remain in case there are modifications to the account or files in the home directories.

To remove the home directories of the RADIUS users, first get the list by running:

```
cumulus@switch:~$ sudo ls -l /home | grep radius
```

For all users listed, except the *radius_user*, run this command to remove the home directories:

```
cumulus@switch:~$ sudo deluser --remove-home USERNAME
```

where USERNAME is the account name (the home directory relative portion). This command gives the following warning because the user is not listed in the `/etc/passwd` file.

```
userdel: cannot remove entry 'USERNAME' from /etc/passwd
/usr/sbin/deluser: `/usr/sbin/userdel USERNAME' returned error
code 1. Exiting.
```

After removing all the RADIUS users, run the command to remove the fixed account. If the account has been changed in the `/etc/nss_mapuser.conf` file, use that account name instead of *radius_user*.

```
cumulus@switch:~$ sudo deluser --remove-home radius_user
cumulus@switch:~$ sudo deluser --remove-home radius_priv_user
cumulus@switch:~$ sudo delgroup radius_users
```

Considerations

- If two or more RADIUS users are logged in simultaneously, a UID lookup only returns the user that logged in first. Any processes run by either user get attributed to both, and all files created by either user get attributed to the first name matched. This is similar to adding two local users to the password file with the same UID and GID, and is an inherent limitation of using the UID for the fixed user from the password file. The current algorithm returns the first name matching the UID from the mapping file; this might be the first or second user that logged in.
- When you have both the TACACS+ and the RADIUS AAA client installed,

RADIUS login is not attempted. As a workaround, do not install both the TACACS+ and the RADIUS AAA client on the same switch.

Related Information

- TACACS+ client
- Cumulus Networks RADIUS demo on GitHub
- Cumulus Network TACACS demo on GitHub

Netfilter - ACLs

Netfilter is the packet filtering framework in Cumulus Linux as well as most other Linux distributions. There are a number of tools available for configuring ACLs in Cumulus Linux:

- `iptables`, `ip6tables`, and `ebtables` are Linux userspace tools used to administer filtering rules for IPv4 packets, IPv6 packets, and Ethernet frames (layer 2 using MAC addresses).
- NCLU is a Cumulus Linux-specific userspace tool used to configure custom ACLs.
- `cl-acltool` is a Cumulus Linux-specific userspace tool used to administer filtering rules and configure default ACLs.

NCLU and `cl-acltool` operate on various configuration files and use `iptables`, `ip6tables`, and `ebtables` to install rules into the kernel. In addition, NCLU and `cl-acltool` program rules in hardware for interfaces involving switch port interfaces, which `iptables`, `ip6tables` and `ebtables` cannot do on their own.

In many instances, you can use NCLU to configure ACLs; however, in some cases, you must use `cl-acltool`. The examples below specify when to use which tool.

If you need help to configure ACLs, run `net example acl` to see a basic configuration:

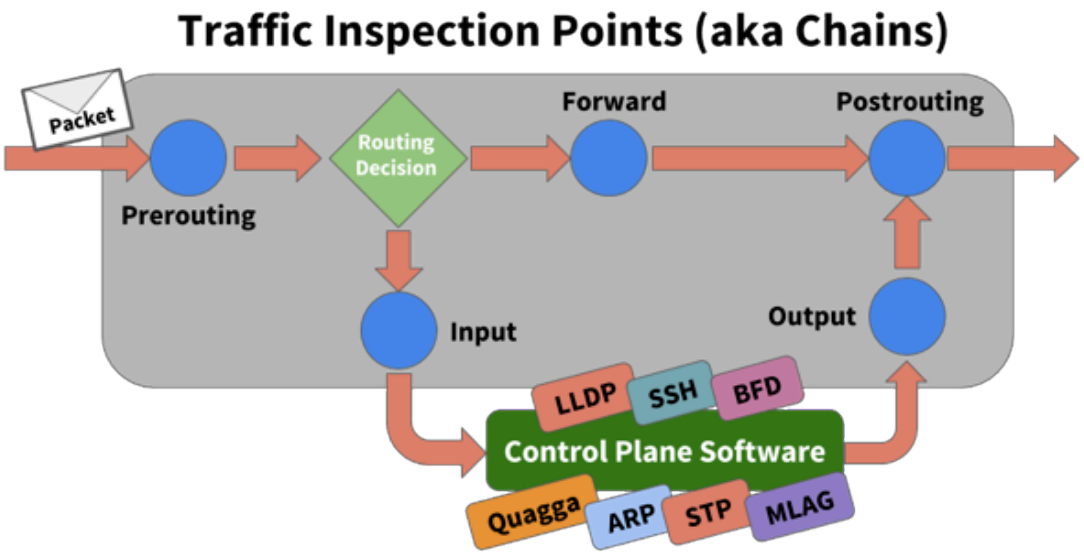
▼ Example

Traffic Rules In Cumulus Linux

Chains

Netfilter describes the mechanism for which packets are classified and controlled in the Linux kernel. Cumulus Linux uses the Netfilter framework to control the flow of traffic to, from, and across the switch. Netfilter does not require a separate software daemon to run; it is part of the Linux kernel itself. Netfilter asserts policies at layers 2, 3 and 4 of the OSI model by inspecting packet and frame headers based on a list of rules. Rules are defined using syntax provided by the `iptables`, `ip6tables` and `ebtables` userspace applications.

The rules created by these programs inspect or operate on packets at several points in the life of the packet through the system. These five points are known as *chains* and are shown here:



The chains and their uses are:

- **PREROUTING** touches packets before they are routed
- **INPUT** touches packets after they are determined to be destined for the local system but before they are received by the control plane software
- **FORWARD** touches transit traffic as it moves through the box
- **OUTPUT** touches packets that are sourced by the control plane software before they are put on the wire
- **POSTROUTING** touches packets immediately before they are put on the wire but after the routing decision has been made

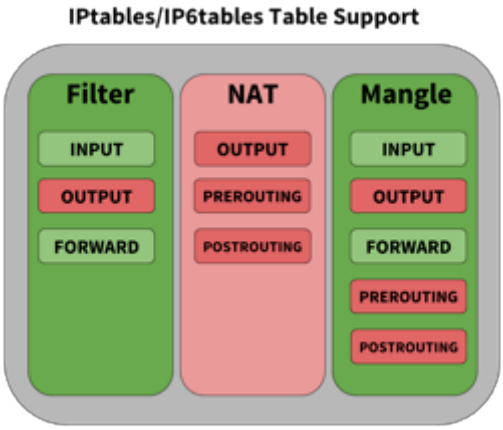
Tables

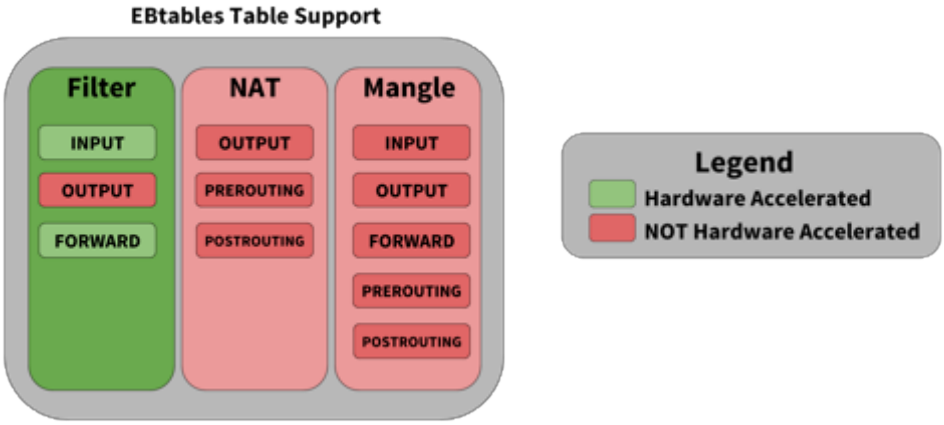
When building rules to affect the flow of traffic, the individual chains can be accessed by *tables*. Linux provides three tables by default:

- **Filter** classifies traffic or filters traffic

- **NAT** applies Network Address Translation rules
- **Mangle** alters packets as they move through the switch

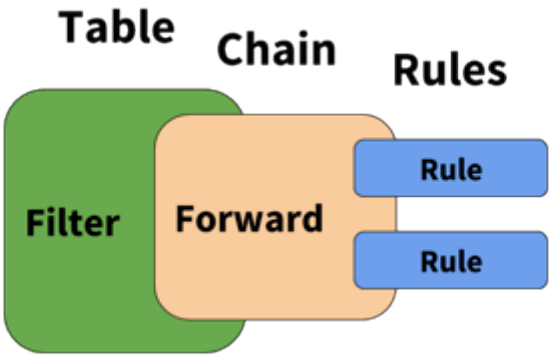
Each table has a set of default chains that can be used to modify or inspect packets at different points of the path through the switch. Chains contain the individual rules to influence traffic. Each table and the default chains they support are shown below. Tables and chains in green are supported by Cumulus Linux, those in red are not supported (that is, they are not hardware accelerated) at this time.



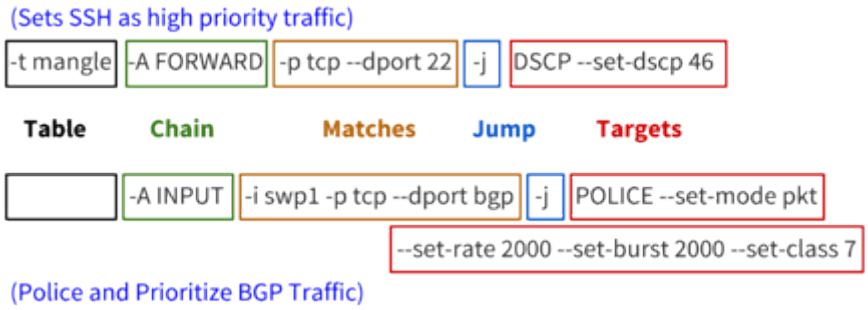


Rules

Rules are the items that actually classify traffic to be acted upon. Rules are applied to chains, which are attached to tables, similar to the graphic below.



Rules have several different components; the examples below highlight those different components.



- **Table:** The first argument is the *table*. Notice the second example does not specify a table, that is because the filter table is implied if a table is not specified.
- **Chain:** The second argument is the *chain*. Each table supports several different chains. See Understanding Tables above.
- **Matches:** The third argument(s) are called the *matches*. You can specify multiple matches in a single rule. However, the more matches you use in a rule, the more memory that rule consumes.
- **Jump:** The *jump* specifies the target of the rule; that is, what action to take if the packet matches the rule. If this option is omitted in a rule, then matching the rule will have no effect on the packet's fate, but the counters on the rule will be incremented.
- **Target(s):** The *target* can be a user-defined chain (other than the one this rule is in), one of the special built-in targets that decides the fate of the packet immediately (like DROP), or an extended target. See the Supported Rule Types section below for examples of different targets.

How Rules Are Parsed and Applied

All the rules from each chain are read from `iptables`, `ip6tables`, and `ebtables` and entered in order into either the filter table or the mangle

table. The rules are read from the kernel in the following order:

- IPv6 (`ip6tables`)
- IPv4 (`iptables`)
- `ebtables`

When rules are combined and put into one table, the order determines the relative priority of the rules; `iptables` and `ip6tables` have the highest precedence and `ebtables` has the lowest.

The Linux packet forwarding construct is an overlay for how the silicon underneath processes packets. Be aware of the following:

- The order of operations for how rules are processed is not perfectly maintained when you compare how `iptables` and the switch silicon process packets. The switch silicon reorders rules when `switchd` writes to the ASIC, whereas traditional `iptables` execute the list of rules in order.
- All rules are terminating; after a rule matches, the action is carried out and no more rules are processed. However, as an exception, when a SETCLASS rule is placed immediately before another rule, it exists multiple times in the default ACL configuration. In the example below, the SETCLASS action applied with the `--in-interface` option, creates the internal ASIC classification, and continues to process the next rule, which does the rate-limiting for the matched protocol:

```
-A $INGRESS_CHAIN --in-interface $INGRESS_INTF -p udp --dport
```

```
$BFD_ECHO_PORT -j SETCLASS --class 7
-A $INGRESS_CHAIN -p udp --dport $BFD_ECHO_PORT -j POLICE --
set-mode pkt --set-rate 2000 --set-burst 2000
```

WARNING

If multiple contiguous rules with the same match criteria are applied to `--in-interface`, **only** the first rule gets processeand then terminates processing. This is a misconfiguration; there is no reason to have duplicate rules with different actions.

- When processing traffic, rules affecting the FORWARD chain that specify an ingress interface are performed prior to rules that match on an egress interface. As a workaround, rules that only affect the egress interface can have an ingress interface wildcard (currently, only *swp+* and *bond+* are supported as wildcard names; see below) that matches any interface applied so that you can maintain order of operations with other input interface rules. For example, with the following rules:

```
-A FORWARD -i $PORTA -j ACCEPT
```

```
-A FORWARD -o $PORTA -j ACCEPT   <-- This rule is performed
LAST (because of egress interface matching)
-A FORWARD -i $PORTB -j DROP
```

If you modify the rules like this, they are performed in order:

```
-A FORWARD -i $PORTA -j ACCEPT
-A FORWARD -i swp+ -o $PORTA -j ACCEPT   <-- These rules are
performed in order (because of wildcard match on ingress
interface)
-A FORWARD -i $PORTB -j DROP
```

- When using rules that do a mangle and a filter lookup for a packet, Cumulus Linux processes them in parallel and combines the action.
- If a switch port is assigned to a bond, any egress rules must be assigned to the bond.
- When using the OUTPUT chain, rules must be assigned to the source. For example, if a rule is assigned to the switch port in the direction of traffic but the source is a bridge (VLAN), the traffic is not affected by the rule and must be applied to the bridge.
- If all transit traffic needs to have a rule applied, use the FORWARD chain,

not the OUTPUT chain.

- `ebtable` rules are put into either the IPv4 or IPv6 memory space depending on whether the rule utilizes IPv4 or IPv6 to make a decision. Layer 2-only rules that match the MAC address are put into the IPv4 memory space.

> **NOTE**
>
> On Broadcom switches, the ingress INPUT chain rules match layer 2 and layer 3 multicast packets **before** multicast packet replication has occurred; therefore, a DROP rule affects all copies.

Rule Placement in Memory

INPUT and ingress (`FORWARD -i`) rules occupy the same memory space. A rule counts as ingress if the `-i` option is set. If both input and output options (`-i` and `-o`) are set, the rule is considered as ingress and occupies that memory space. For example:

```
-A FORWARD -i swp1 -o swp2 -s 10.0.14.2 -d 10.0.15.8 -p tcp -j
ACCEPT
```

NOTE

If you set an output flag with the INPUT chain, you see an error. For example, running `cl-acltool -i` on the following rule:

```
-A FORWARD,INPUT -i swp1 -o swp2 -s 10.0.14.2 -d
10.0.15.8 -p tcp -j ACCEPT
```

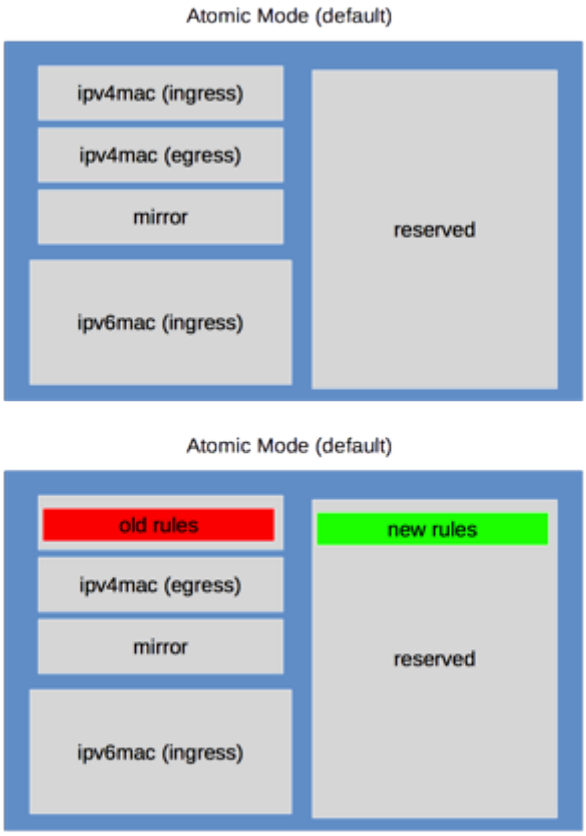
generates the following error:

```
error: line 2 : output interface specified with INPUT
chain error processing rule '-A FORWARD,INPUT -i swp1 -o
swp2 -s 10.0.14.2 -d 10.0.15.8 -p tcp -j ACCEPT'
```

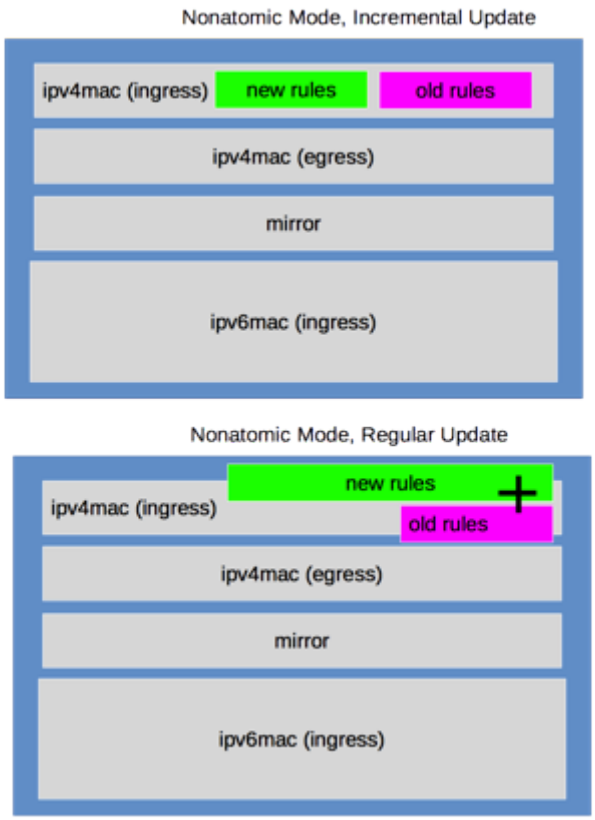
However, removing the `-o` option and interface make it a valid rule.

Nonatomic Update Mode and Atomic Update Mode

In Cumulus Linux, *atomic update mode* is enabled by default. However, this mode limits the number of ACL rules that you can configure.

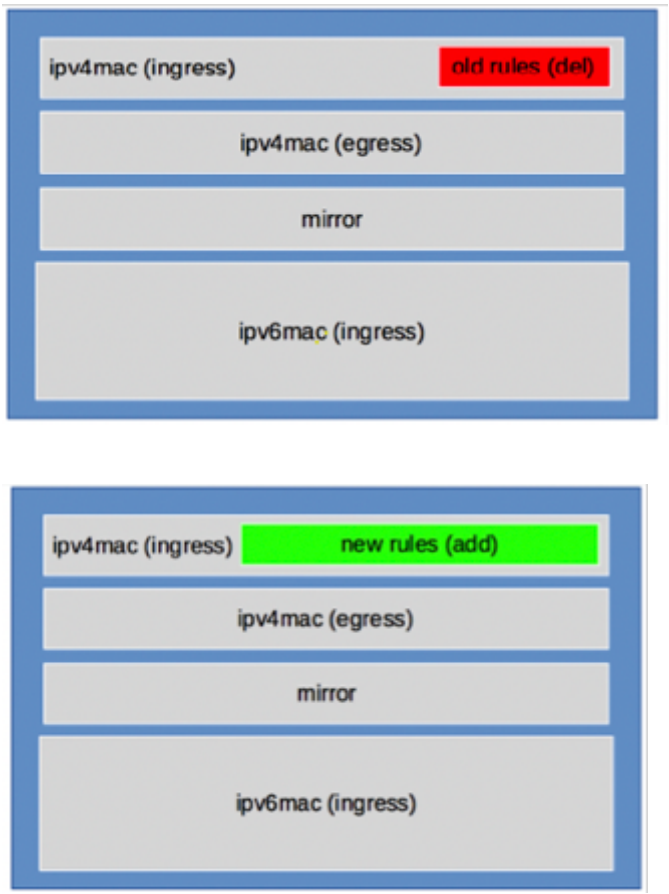


To increase the number of ACL rules that can be configured, configure the switch to operate in *nonatomic mode*.



How the Rules Get Installed

Instead of reserving 50% of your TCAM space for atomic updates, incremental update uses the available free space to write the new TCAM rules and swap over to the new rules after this is complete. Cumulus Linux then deletes the old rules and frees up the original TCAM space. If there is insufficient free space to complete this task, the original nonatomic update is performed, which interrupts traffic.



Enable Nonatomic Update Mode

You can enable nonatomic updates for `switchd`, which offer better scaling because all TCAM resources are used to actively impact traffic. With atomic updates, half of the hardware resources are on standby and do not actively impact traffic.

Incremental nonatomic updates are table based, so they do not interrupt network traffic when new rules are installed. The rules are mapped into the following tables and are updated in this order:

- mirror (ingress only)

- ipv4-mac (can be both ingress and egress)
- ipv6 (ingress only)

> **WARNING**
>
> Only switches with the Broadcom ASIC support *incremental* nonataomic updates. Mellanox switches with the Spectrum-based ASIC only support *standard* nonatomic updates; using nonatomic mode on Spectrum-based ASICs impacts traffic on ACL updates.

The incremental nonatomic update operation follows this order:

1. Updates are performed incrementally, one table at a time without stopping traffic.
2. Cumulus Linux checks if the rules in a table have changed since the last time they were installed; if a table does not have any changes, it is not reinstalled.
3. If there are changes in a table, the new rules are populated in new groups or slices in hardware, then that table is switched over to the new groups or slices.
4. Finally, old resources for that table are freed. This process is repeated for each of the tables listed above.
5. If sufficient resources do not exist to hold both the new rule set and old rule set, the regular nonatomic mode is attempted. This interrupts

network traffic.

6. If the regular nonatomic update fails, Cumulus Linux reverts back to the previous rules.

To always start `switchd` with nonatomic updates:

1. Edit `/etc/cumulus/switchd.conf`.

2. Add the following line to the file:

```
acl.non_atomic_update_mode = TRUE
```

3. Restart `switchd`:

```
cumulus@switch:~$ sudo systemctl restart switchd.service
```

WARNING

Restarting the `switchd` service causes all network ports to reset, interrupting network services, in addition to resetting the switch hardware configuration.

> **NOTE**
>
> During regular *non-incremental nonatomic updates*, traffic is stopped first, then enabled after the new configuration is written into the hardware completely.

Use iptables, ip6tables, and ebtables Directly

Using `iptables`, `ip6tables`, `ebtables` directly is not recommended because any rules installed in these cases only are applied to the Linux kernel and are not hardware accelerated using synchronization to the switch silicon. Running `cl-acltool -i` (the installation command) resets all rules and deletes anything that is not stored in `/etc/cumulus/acl/policy.conf`.

For example, performing:

```
cumulus@switch:~$ sudo iptables -A INPUT -p icmp --icmp-type
echo-request -j DROP
```

Appears to work, and the rule appears when you run `cl-acltool -L`:

```
cumulus@switch:~$ sudo cl-acltool -L ip
```

```
------------------------------
Listing rules of type iptables:
------------------------------

TABLE filter :
Chain INPUT (policy ACCEPT 72 packets, 5236 bytes)
pkts bytes target prot opt in out source destination
0 0 DROP icmp -- any any anywhere anywhere icmp echo-request
```

However, the rule is not synced to hardware when applied in this way and running `cl-acltool -i` or `reboot` removes the rule without replacing it. To ensure all rules that can be in hardware are hardware accelerated, place them in `/etc/cumulus/acl/policy.conf` and install them by running `cl-acltool -i`.

Estimate the Number of Rules

To estimate the number of rules you can create from an ACL entry, first determine if that entry is an ingress or an egress. Then, determine if it is an IPv4-mac or IPv6 type rule. This determines the slice to which the rule belongs. Use the following to determine how many entries are used up for each type.

By default, each entry occupies one double wide entry, except if the entry is one of the following:

- An entry with multiple comma-separated input interfaces is split into one rule for each input interface (listed after `--in-interface` below). For example, this entry splits into two rules:

```
-A FORWARD --in-interface swp1s0,swp1s1 -p icmp -j ACCEPT
```

- An entry with multiple comma-separated output interfaces is split into one rule for each output interface (listed after `--out-interface` below). This entry splits into two rules:

```
-A FORWARD --in-interface swp+ --out-interface swp1s0,swp1s1
-p icmp -j ACCEPT
```

- An entry with both input and output comma-separated interfaces is split into one rule for each combination of input and output interface (listed after `--in-interface` and `--out-interface` below). This entry splits into four rules:

```
-A FORWARD --in-interface swp1s0,swp1s1 --out-interface
swp1s2,swp1s3 -p icmp -j ACCEPT
```

- An entry with multiple layer 4 port ranges is split into one rule for each

range (listed after `--dports` below). For example, this entry splits into two rules:

```
-A FORWARD --in-interface swp+ -p tcp -m multiport --dports
1050:1051,1055:1056 -j ACCEPT
```

> **NOTE**
>
> Port ranges are only allowed for ingress rules.

Match SVI and Bridged Interfaces in Rules

Cumulus Linux supports matching ACL rules for both ingress and egress interfaces on both VLAN-aware and traditional mode bridges, including bridge SVIs (switch VLAN interfaces) for input and output. However, keep the following in mind:

- If a traditional mode bridge has a mix of different VLANs, or has both access and trunk members, output interface matching is not supported.
- For `iptables` rules, all IP packets in a bridge are matched, not just routed packets.
- You cannot match both input and output interfaces in a rule.
- For routed packets, Cumulus Linux cannot match the output bridge for SPAN/ERSPAN.
- Matching SVI interfaces in `ebtable` rules is supported on switches based

on Broadcom ASICs. This feature is not currently supported on switches with Mellanox Spectrum ASICs.

Example rules for a VLAN-aware bridge:

```
[ebtables]
-A FORWARD -i vlan100 -p IPv4 --ip-protocol icmp -j DROP
-A FORWARD -o vlan100 -p IPv4 --ip-protocol icmp -j ACCEPT

[iptables]
-A FORWARD -i vlan100 -p icmp -j DROP
-A FORWARD --out-interface vlan100 -p icmp -j ACCEPT
-A FORWARD --in-interface vlan100 -j POLICE --set-mode  pkt  --
set-rate  1 --set-burst 1 --set-class 0
```

Example rules for a traditional mode bridge:

```
[ebtables]
-A FORWARD -i br0 -p IPv4 --ip-protocol icmp -j DROP
-A FORWARD -o br0 -p IPv4 --ip-protocol icmp -j ACCEPT

[iptables]
-A FORWARD -i br0 -p icmp -j DROP
-A FORWARD --out-interface br0 -p icmp -j ACCEPT
```

```
-A FORWARD --in-interface br0 -j POLICE --set-mode  pkt  --set-
rate  1 --set-burst 1 --set-class 0
```

Match on VLAN IDs on Layer 2 Interfaces

On switches with Spectrum ASICs, you can match on VLAN IDs on layer 2 interfaces for ingress rules.

The following example matches on a VLAN and DSCP class, and sets the internal class of the packet. This can be combined with ingress iptable rules to get extended matching on IP fields.

```
[ebtables]
-A FORWARD -p 802_1Q --vlan-id 100 -j mark --mark-set 0x66


[iptables]
-A FORWARD -i swp31 -m mark --mark 0x66 -m dscp --dscp-class
CS1 -j SETCLASS --class 2
```

Install and Manage ACL Rules with NCLU

NCLU provides an easy way to create custom ACLs in Cumulus Linux. The rules you create live in the `/var/lib/cumulus/nclu/nclu_acl.conf` file, which gets converted to a rules file, `/etc/cumulus/acl/policy.d/`

`50_nclu_acl.rules`. This way, the rules you create with NCLU are independent of the two default files in `/etc/cumulus/acl/policy.d/00control_plane.rules` and `99control_plane_catch_all.rules`, as the content in these files might get updated after you upgrade Cumulus Linux.

Instead of crafting a rule by hand then installing it using `cl-acltool`, NCLU handles many of the options automatically. For example, consider the following `iptables` rule:

```
-A FORWARD -i swp1 -o swp2 -s 10.0.14.2 -d 10.0.15.8 -p tcp -j
ACCEPT
```

You create this rule, called *EXAMPLE1*, using NCLU like this:

```
cumulus@switch:~$ net add acl ipv4 EXAMPLE1 accept tcp source-
ip 10.0.14.2/32 source-port any dest-ip 10.0.15.8/32 dest-port
any
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

All options, such as the `-j` and `-p`, even `FORWARD` in the above rule, are added automatically when you apply the rule to the control plane; NCLU figures it all out for you.

You can also set a priority value, which specifies the order in which the rules get executed and the order in which they appear in the rules file. Lower numbers are executed first. To add a new rule in the middle, first run `net show config acl`, which displays the priority numbers. Otherwise, new rules get appended to the end of the list of rules in the `nclu_acl.conf` and `50_nclu_acl.rules` files.

> **NOTE**
>
> If you need to hand edit a rule, do not edit the `50_nclu_acl.rules` file. Instead, edit the `nclu_acl.conf` file.

After you add the rule, you need to apply it to an inbound or outbound interface using `net add int acl`. The inbound interface in our example is swp1:

```
cumulus@switch:~$ net add int swp1 acl ipv4 EXAMPLE1 inbound
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

After you commit your changes, you can verify the rule you created with NCLU by running `net show configuration acl`:

```
cumulus@switch:~$ net show configuration acl
acl ipv4 EXAMPLEv4 priority 10 accept tcp source-ip 10.0.14.2/
32 source-port any dest-ip 10.0.15.8/32 dest-port any

interface swp1
acl ipv4 EXAMPLE1 inbound
```

Or you can see all of the rules installed by running `cat` on the `50_nclu_acl.rules` file:

```
cumulus@switch:~$ cat /etc/cumulus/acl/policy.d/
50_nclu_acl.rules
[iptables]
# swp1: acl ipv4 EXAMPLE1 inbound
-A FORWARD --in-interface swp1 --out-interface swp2 -j ACCEPT
-p tcp -s 10.0.14.2/32 -d 10.0.15.8/32 --dport 110
```

For INPUT and FORWARD rules, apply the rule to a control plane interface using `net add control-plane`:

```
cumulus@switch:~$ net add control-plane acl ipv4 EXAMPLE1
```

```
inbound
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

To remove a rule, use `net del acl ipv4|ipv6|mac RULENAME`:

```
cumulus@switch:~$ net del acl ipv4 EXAMPLE1
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

This deletes all rules from the `50_nclu_acl.rules` file with that name. It also deletes the interfaces referenced in the `nclu_acl.conf` file.

Install and Manage ACL Rules with cl-acltool

You can manage Cumulus Linux ACLs with `cl-acltool`. Rules are first written to the `iptables` chains, as described above, and then synced to hardware via `switchd`.

NOTE

Use `iptables`/`ip6tables`/`ebtables` and `cl-acltool` to manage rules in the default files, `00control_plane.rules` and `99control_plane_catch_all.rules`; they are not aware of rules created using NCLU.

To examine the current state of chains and list all installed rules, run:

```
cumulus@switch:~$ sudo cl-acltool -L all
 ------------------------------
Listing rules of type iptables:
------------------------------

TABLE filter :
Chain INPUT (policy ACCEPT 90 packets, 14456 bytes)
pkts bytes target prot opt in out source destination
0 0 DROP all -- swp+ any 240.0.0.0/5 anywhere
0 0 DROP all -- swp+ any loopback/8 anywhere
0 0 DROP all -- swp+ any base-address.mcast.net/8 anywhere
0 0 DROP all -- swp+ any 255.255.255.255 anywhere ...
```

To list installed rules using native `iptables`, `ip6tables` and `ebtables`, use the `-L` option with the respective commands:

```
cumulus@switch:~$ sudo iptables -L
cumulus@switch:~$ sudo ip6tables -L
cumulus@switch:~$ sudo ebtables -L
```

To flush all installed rules, run:

```
cumulus@switch:~$ sudo cl-acltool -F all
```

To flush only the IPv4 `iptables` rules, run:

```
cumulus@switch:~$ sudo cl-acltool -F ip
```

If the install fails, ACL rules in the kernel and hardware are rolled back to the previous state. Errors from programming rules in the kernel or ASIC are reported appropriately.

Install Packet Filtering (ACL) Rules

`cl-acltool` takes access control list (ACL) rules input in files. Each ACL policy file contains `iptables`, `ip6tables` and `ebtables` categories under the tags `[iptables]`, `[ip6tables]` and `[ebtables]`.

Each rule in an ACL policy must be assigned to one of the rule categories above.

See `man cl-acltool(5)` for ACL rule details. For `iptables` rule syntax, see `man iptables(8)`. For `ip6tables` rule syntax, see `man ip6tables(8)`. For `ebtables` rule syntax, see `man ebtables(8)`.

See `man cl-acltool(5)` and `man cl-acltool(8)` for further details on using `cl-acltool`. Some examples are listed here and more are listed later in this chapter.

> **NOTE**
>
> By default:
>
> - ACL policy files are located in `/etc/cumulus/acl/policy.d/`.
> - All `*.rules` files in this directory are included in `/etc/cumulus/acl/policy.conf`.
> - All files included in this `policy.conf` file are installed when the switch boots up.
> - The `policy.conf` file expects rules files to have a `.rules` suffix as part of the file name.

Here is an example ACL policy file:

```
[iptables]
-A INPUT --in-interface swp1 -p tcp --dport 80 -j ACCEPT
-A FORWARD --in-interface swp1 -p tcp --dport 80 -j ACCEPT

[ip6tables]
-A INPUT --in-interface swp1 -p tcp --dport 80 -j ACCEPT
-A FORWARD --in-interface swp1 -p tcp --dport 80 -j ACCEPT

[ebtables]
-A INPUT -p IPv4 -j ACCEPT
-A FORWARD -p IPv4 -j ACCEPT
```

You can use wildcards or variables to specify chain and interface lists to ease administration of rules.

NOTE

Currently only *swp+* and *bond+* are supported as wildcard names. There might be kernel restrictions in supporting more complex wildcards like *swp1+ etc*.

swp+ rules are applied as an aggregate, *not* per port. If you want to apply per port policing, specify a specific port instead of the wildcard.

```
INGRESS = swp+
INPUT_PORT_CHAIN = INPUT,FORWARD

[iptables]
-A $INPUT_PORT_CHAIN --in-interface $INGRESS -p tcp --dport 80
-j ACCEPT

[ip6tables]
-A $INPUT_PORT_CHAIN --in-interface $INGRESS -p tcp --dport 80
-j ACCEPT

[ebtables]
-A INPUT -p IPv4 -j ACCEPT
```

You can write ACL rules for the system into multiple files under the default `/etc/cumulus/acl/policy.d/` directory. The ordering of rules during installation follows the sort order of the files based on their file names.

Use multiple files to stack rules. The example below shows two rules files separating rules for management and datapath traffic:

```
cumulus@switch:~$ ls /etc/cumulus/acl/policy.d/
00sample_mgmt.rules 01sample_datapath.rules
```

```
cumulus@switch:~$ cat /etc/cumulus/acl/policy.d/
00sample_mgmt.rules

INGRESS_INTF = swp+
INGRESS_CHAIN = INPUT

[iptables]
# protect the switch management
-A $INGRESS_CHAIN --in-interface $INGRESS_INTF -s 10.0.14.2 -d
10.0.15.8 -p tcp -j ACCEPT
-A $INGRESS_CHAIN --in-interface $INGRESS_INTF -s 10.0.11.2 -d
10.0.12.8 -p tcp -j ACCEPT
-A $INGRESS_CHAIN --in-interface $INGRESS_INTF -d 10.0.16.8 -p
udp -j DROP

cumulus@switch:~$ cat /etc/cumulus/acl/policy.d/
01sample_datapath.rules
INGRESS_INTF = swp+
INGRESS_CHAIN = INPUT, FORWARD

[iptables]
-A $INGRESS_CHAIN --in-interface $INGRESS_INTF -s 192.0.2.5 -p
icmp -j ACCEPT
-A $INGRESS_CHAIN --in-interface $INGRESS_INTF -s 192.0.2.6 -d
```

```
192.0.2.4 -j DROP
-A $INGRESS_CHAIN --in-interface $INGRESS_INTF -s 192.0.2.2 -d
192.0.2.8 -j DROP
```

Install all ACL policies under a directory:

```
cumulus@switch:~$ sudo cl-acltool -i -P ./rules
Reading files under rules
Reading rule file ./rules/01_http_rules.txt ...
Processing rules in file ./rules/01_http_rules.txt ...
Installing acl policy ...
Done.
```

Apply all rules and policies included in `/etc/cumulus/acl/policy.conf`:

```
cumulus@switch:~$ sudo cl-acltool -i
```

In addition to ensuring that the rules and policies referenced by `/etc/cumulus/acl/policy.conf` are installed, this will remove any currently active rules and policies that are not contained in the files referenced by `/etc/cumulus/acl/policy.conf`.

Specify the Policy Files to Install

By default, Cumulus Linux installs any `.rules` file you configure in `/etc/cumulus/acl/policy.d/`. To add other policy files to an ACL, you need to include them in `/etc/cumulus/acl/policy.conf`. For example, for Cumulus Linux to install a rule in a policy file called `01_new.datapathacl`, add `include /etc/cumulus/acl/policy.d/01_new.rules` to `policy.conf`, as in this example:

```
cumulus@switch:~$ sudo nano /etc/cumulus/acl/policy.conf


#
# This file is a master file for acl policy file inclusion
#
# Note: This is not a file where you list acl rules.
#
# This file can contain:
# - include lines with acl policy files
#   example:
#     include <filepath>
#
# see manpage cl-acltool(5) and cl-acltool(8) for how to write
policy files
#
```

```
include /etc/cumulus/acl/policy.d/01_new.datapathacl
```

Hardware Limitations on Number of Rules

The maximum number of rules that can be handled in hardware is a function of the following factors:

- The platform type (switch silicon, like Tomahawk or Spectrum - see the HCL to determine which platform type applies to a particular switch).
- The mix of IPv4 and IPv6 rules; Cumulus Linux does not support the maximum number of rules for both IPv4 and IPv6 simultaneously.
- The number of default rules provided by Cumulus Linux.
- Whether the rules are applied on ingress or egress.
- Whether the rules are in atomic or nonatomic mode; nonatomic mode rules are used when nonatomic updates are enabled (see above).

If the maximum number of rules for a particular table is exceeded, `cl-acltool -i` generates the following error:

```
error: hw sync failed (sync_acl hardware installation failed)
Rolling back .. failed.
```

In the tables below, the default rules count toward the limits listed. The raw

limits below assume only one ingress and one egress table are present.

Broadcom Tomahawk Limits

Direction	Atomic Mode IPv4 Rules	Atomic Mode IPv6 Rules	Nonatomic Mode IPv4 Rules	Nonatomic Mode IPv6 Rules
Ingress raw limit	512	512	1024	1024
Ingress limit with default rules	256 (36 default)	256 (29 default)	768 (36 default)	768 (29 default)
Egress raw limit	256	0	512	0
Egress limit with default rules	256 (29 default)	0	512 (29 default)	0

Broadcom Trident3 Limits

The Trident3 ASIC is divided into 12 slices, organized into 4 groups for ACLs. Each group contains 3 slices. Each group can support a maximum of 768 rules. You cannot mix IPv4 and IPv6 rules within the same group. IPv4 and MAC rules can be programmed into the same group.

Direction	Atomic Mode IPv4 Rules	Atomic Mode IPv6 Rules	Nonatomic Mode IPv4 Rules	Nonatomic Mode IPv6 Rules
Ingress raw limit	768	768	2304	2304
Ingress limit with default rules	768 (44 default)	768 (41 default)	2304 (44 default)	2304 (41 default)
Egress raw limit	512	0	512	0
Egress limit with default rules	512 (28 default)	0	512 (28 default)	0

Due to a hardware limitation on Trident3 switches, certain broadcast packets that are VXLAN decapsulated and sent to the CPU do not hit the normal INPUT chain ACL rules installed with `cl-acltool`. See default ACL considerations.

Broadcom Trident II+ Limits

Direction	Atomic Mode IPv4 Rules	Atomic Mode IPv6 Rules	Nonatomic Mode IPv4 Rules	Nonatomic Mode IPv6 Rules
Ingress	4096	4096	8192	8192

Direction	Atomic Mode IPv4 Rules	Atomic Mode IPv6 Rules	Nonatomic Mode IPv4 Rules	Nonatomic Mode IPv6 Rules
raw limit				
Ingress limit with default rules	2048 (36 default)	3072 (29 default)	6144 (36 default)	6144 (29 default)
Egress raw limit	256	0	512	0
Egress limit with default rules	256 (29 default)	0	512 (29 default)	0

Broadcom Trident II Limits

Direction	Atomic Mode IPv4 Rules	Atomic Mode IPv6 Rules	Nonatomic Mode IPv4 Rules	Nonatomic Mode IPv6 Rules
Ingress raw limit	1024	1024	2048	2048
Ingress limit with default rules	512 (36 default)	768 (29 default)	1536 (36 default)	1536 (29 default)

Direction	Atomic Mode IPv4 Rules	Atomic Mode IPv6 Rules	Nonatomic Mode IPv4 Rules	Nonatomic Mode IPv6 Rules
Egress raw limit	256	0	512	0
Egress limit with default rules	256 (29 default)	0	512 (29 default)	0

Broadcom Helix4 Limits

Direction	Atomic Mode IPv4 Rules	Atomic Mode IPv6 Rules	Nonatomic Mode IPv4 Rules	Nonatomic Mode IPv6 Rules
Ingress raw limit	1024	512	2048	1024
Ingress limit with default rules	768 (36 default)	384 (29 default)	1792 (36 default)	896 (29 default)
Egress raw limit	256	0	512	0
Egress limit with default rules	256 (29 default)	0	512 (29 default)	0

Mellanox Spectrum Limits

The Mellanox Spectrum ASIC has one common TCAM for both ingress and egress, which can be used for other non-ACL-related resources. However, the number of supported rules varies with the TCAM profile specified for the switch.

Profile	Atomic Mode IPv4 Rules	Atomic Mode IPv6 Rules	Nonatomic Mode IPv4 Rules	Nonatomic Mode IPv6 Rules
default	500	250	1000	500
ipmc-heavy	750	500	1500	1000
acl-heavy	1750	1000	3500	2000
ipmc-max	1000	500	2000	1000
ip-acl-heavy	7500	0	15000	0

> **NOTE**
>
> Even though the table above specifies that zero IPv6 rules are supported with the ip-acl-heavy profile, nothing prevents you from configuring IPv6 rules. However, there is no guarantee that IPv6

rules work under the ip-acl-heavy profile.

Supported Rule Types

The `iptables`/`ip6tables`/`ebtables` construct tries to layer the Linux implementation on top of the underlying hardware but they are not always directly compatible. Here are the supported rules for chains in `iptables`, `ip6tables` and `ebtables`.

NOTE

To learn more about any of the options shown in the tables below, run `iptables -h [name of option]`. The same help syntax works for options for `ip6tables` and `ebtables`.

```
root@leaf1# ebtables -h tricolorpolice
<...snip...>
tricolorpolice option:
--set-color-mode STRING setting the mode in blind or
aware
--set-cir INT setting committed information rate in
```

```
kbits per second
--set-cbs INT setting committed burst size in kbyte
--set-pir INT setting peak information rate in kbits per
second
--set-ebs INT setting excess burst size in kbyte
--set-conform-action-dscp INT setting dscp value if the
action is accept for conforming packets
--set-exceed-action-dscp INT setting dscp value if the
action is accept for exceeding packets
--set-violate-action STRING setting the action (accept/
drop) for violating packets
--set-violate-action-dscp INT setting dscp value if the
action is accept for violating packets
Supported chains for the filter table:
INPUT FORWARD OUTPUT
```

iptables and ip6tables Rule Support

Rule Element	Supported	Unsupported
Matches	Src/Dst, IP protocol In/out interface	Rules with input/ output Ethernet

Rule Element	Supported	Unsupported
	IPv4: icmp, ttl, IPv6: icmp6, frag, hl, IP common: tcp (with flags), udp, multiport, DSCP, addrtype	interfaces are ignored Inverse matches
Standard Targets	ACCEPT, DROP	RETURN, QUEUE, STOP, Fall Thru, Jump
Extended Targets	LOG (IPv4/IPv6); UID is not supported for LOG TCP SEQ, TCP options or IP options ULOG SETQOS DSCP Unique to Cumulus Linux: SPAN ERSPAN (IPv4/IPv6) POLICE TRICOLORPOLICE SETCLASS	

ebtables Rule Support

Rule Element	Supported	Unsupported
Matches	ether type input interface/ wildcard output interface/ wildcard Src/Dst MAC IP: src, dest, tos, proto, sport, dport IPv6: tclass, icmp6: type, icmp6: code range, src/dst addr, sport, dport 802.1p (CoS) VLAN	Inverse matches Proto length
Standard Targets	ACCEPT, DROP	RETURN, CONTINUE, Jump, Fall Thru
Extended Targets	ULOG LOG Unique to Cumulus Linux: SPAN ERSPAN POLICE TRICOLORPOLICE SETCLASS	

Other Unsupported Rules

- Rules that have no matches and accept all packets in a chain are currently ignored.
- Chain default rules (that are ACCEPT) are also ignored.

IPv6 Egress Rules on Broadcom Switches

Cumulus Linux supports IPv6 egress rules in `ip6tables` on Broadcom switches. Because there are no slices to allocate in the egress TCAM for IPv6, the matches are implemented using a combination of the ingress IPv6 slice and the existing egress IPv4 MAC slice:

- Cumulus Linux compares all the match fields in the IPv6 ingress slice, except the `--out-interface` field, and marks the packet with a `classid.`
- The egress IPv4 MAC slice matches on the `classid` and the `out-interface`, and performs the actions.

For example, the `-A FORWARD --out-interface vlan100 -p icmp6 -j ACCEPT` rule is split into the following:

- IPv6 ingress: `-A FORWARD -p icmp6` → action mark (for example, `classid 4`)
- IPv4 MAC egress: `<match mark 4>` and `--out-interface vlan100 -j ACCEPT`

NOTE

- IPv6 egress rules in `ip6tables` are not supported on Hurricane2 switches.
- You cannot match both input and output interfaces in the same rule.
- The egress TCAM IPv4 MAC slice is shared with other rules, which constrains the scale to a much lower limit.

Considerations

Splitting rules across the ingress TCAM and the egress TCAM causes the ingress IPv6 part of the rule to match packets going to all destinations, which can interfere with the regular expected linear rule match in a sequence. For example:

A higher rule can prevent a lower rule from being matched unexpectedly:

Rule 1: `-A FORWARD --out-interface vlan100 -p icmp6 -j ACCEPT`

Rule 2: `-A FORWARD --out-interface vlan101 -p icmp6 -s 01::02 -j ACCEPT`

Rule 1 matches all icmp6 packets from to all out interfaces in the ingress TCAM.`

This prevents rule 2 from getting matched, which is more specific but with a different out interface. Make sure to put more specific matches above more general matches even if the output interfaces are different.

When you have two rules with the same output interface, the lower rule might match unexpectedly depending on the presence of the previous rules.

Rule 1: `-A FORWARD --out-interface vlan100 -p icmp6 -j ACCEPT`

Rule 2: `-A FORWARD --out-interface vlan101 -s 00::01 -j DROP`

Rule 3: `-A FORWARD --out -interface vlan101 -p icmp6 -j ACCEPT`

Rule 3 still matches for an icmp6 packet with sip 00:01 going out of vlan101. Rule 1 interferes with the normal function of rule 2 and/or rule 3.

When you have two adjacent rules with the same match and different output interfaces, such as:

Rule 1: `-A FORWARD --out-interface vlan100 -p icmp6 -j ACCEPT`

Rule 2: `-A FORWARD --out-interface vlan101 -p icmp6 -j DROP`

Rule 2 will never be match on ingress. Both rules share the same mark.

Matching Untagged Packets (Trident3 Switches)

Untagged packets do not have an associated VLAN to match on egress; therefore, the match must be on the underlying layer 2 port. For example, for a bridge configured with pvid 100, member port swp1s0 and swp1s1, and

SVI vlan100, the output interface match on vlan100 has to be expanded into each member port. The `-A FORWARD -o vlan100 -p icmp6 -j ACCEPT` rule must be specified as two rules:

Rule 1: `-A FORWARD -o swp1s0 -p icmp6 -J ACCEPT`

Rule 2: `-A FORWARD -o swp1s1 -p icmp6 -j ACCEPT`

Matching on an egress port matches all packets egressing the port, tagged as well as untagged. Therefore, to match only untagged traffic on the port, you must specify additional rules above this rule to prevent tagged packets matching the rule. This is true for bridge member ports as well as regular layer 2 ports. In the example rule above, if vlan101 is also present on the bridge, add a rule above rule 1 and rule 2 to protect vlan101 tagged traffic:

Rule 0: `-A FORWARD -o vlan101 -p icmp6 -j ACCEPT`

Rule 1: `-A FORWARD -o swp1s0 -p icmp6 -j ACCEPT`

Rule 2: `-A FORWARD -o swp1s1 -p icmp6 -j ACCEPT`

For a standalone port or subinterface on swp1s2:

Rule 0: `-A FORWARD -o swp1s2.101 -p icmp6 -j ACCEPT`

Rule 1: `-A FORWARD -o swp1s2 -p icmp6 -j ACCEPT`

Common Examples

Control Plane and Data Plane Traffic

You can configure quality of service for traffic on both the control plane and the data plane. By using QoS policers, you can rate limit traffic so incoming packets get dropped if they exceed specified thresholds.

> **NOTE**
>
> Counters on POLICE ACL rules in `iptables` do not currently show the packets that are dropped due to those rules.

Use the `POLICE` target with `iptables`. `POLICE` takes these arguments:

- `--set-class value` sets the system internal class of service queue configuration to *value*.
- `--set-rate value` specifies the maximum rate in kilobytes (KB) or packets.
- `--set-burst value` specifies the number of packets or kilobytes (KB) allowed to arrive sequentially.
- `--set-mode string` sets the mode in *KB* (kilobytes) or *pkt* (packets) for rate and burst size.

For example, to rate limit the incoming traffic on swp1 to 400 packets per

second with a burst of 100 packets per second and set the class of the queue for the policed traffic as 0, set this rule in your appropriate `.rules` file:

```
-A INPUT --in-interface swp1 -j POLICE --set-mode pkt --set-
rate 400 --set-burst 100 --set-class 0
```

Here is another example of control plane ACL rules to lock down the switch. You specify them in `/etc/cumulus/acl/policy.d/00control_plane.rules`:

▼ View the contents of the file ...

Set DSCP on Transit Traffic

The examples here use the *mangle* table to modify the packet as it transits the switch. DSCP is expressed in decimal notation in the examples below.

```
[iptables]


#Set SSH as high priority traffic.
-t mangle -A FORWARD -p tcp --dport 22  -j DSCP --set-dscp 46


#Set everything coming in SWP1 as AF13
-t mangle -A FORWARD --in-interface swp1 -j DSCP --set-dscp 14
```

```
#Set Packets destined for 10.0.100.27 as best effort
-t mangle -A FORWARD -d 10.0.100.27/32 -j DSCP --set-dscp 0

#Example using a range of ports for TCP traffic
-t mangle -A FORWARD -p tcp -s 10.0.0.17/32 --sport 10000:20000
-d 10.0.100.27/32 --dport 10000:20000 -j DSCP --set-dscp 34
```

Verify DSCP Values on Transit Traffic

The examples here use the DSCP match criteria in combination with other IP, TCP, and interface matches to identify traffic and count the number of packets.

```
[iptables]

#Match and count the packets that match SSH traffic with DSCP EF
-A FORWARD -p tcp --dport 22 -m dscp --dscp 46 -j ACCEPT

#Match and count the packets coming in SWP1 as AF13
-A FORWARD --in-interface swp1 -m dscp --dscp 14 -j ACCEPT
#Match and count the packets with a destination 10.0.0.17
marked best effort
-A FORWARD -d 10.0.100.27/32 -m dscp --dscp 0 -j ACCEPT
```

```
#Match and count the packets in a port range with DSCP AF41
-A FORWARD -p tcp -s 10.0.0.17/32 --sport 10000:20000 -d
10.0.100.27/32 --dport 10000:20000 -m dscp --dscp 34 -j ACCEPT
```

Check the Packet and Byte Counters for ACL Rules

To verify the counters using the above example rules, first send test traffic matching the patterns through the network. The following example generates traffic with `mz` (or `mausezahn`), which can be installed on host servers or even on Cumulus Linux switches. After traffic is sent to validate the counters, they are matched on switch1 using `cl-acltool`.

> **NOTE**
>
> Policing counters do not increment on switches with the Spectrum ASIC.

```
# Send 100 TCP packets on host1 with a DSCP value of EF with a
destination of host2 TCP port 22:

cumulus@host1$ mz eth1 -A 10.0.0.17 -B 10.0.100.27 -c 100 -v -t
```

```
tcp "dp=22,dscp=46"
  IP:  ver=4, len=40, tos=184, id=0, frag=0, ttl=255, proto=6,
sum=0, SA=10.0.0.17, DA=10.0.100.27,
      payload=[see next layer]
  TCP: sp=0, dp=22, S=42, A=42, flags=0, win=10000, len=20,
sum=0,
      payload=

# Verify the 100 packets are matched on switch1

cumulus@switch1$ sudo cl-acltool -L ip
--------------------------------
Listing rules of type iptables:
--------------------------------
TABLE filter :
Chain INPUT (policy ACCEPT 9314 packets, 753K bytes)
  pkts bytes target     prot opt in     out
source               destination
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out
source               destination
  100  6400 ACCEPT     tcp  --  any    any
anywhere             anywhere             tcp dpt:ssh DSCP
match 0x2e
```

```
    0     0 ACCEPT     all  --  swp1   any
anywhere             anywhere             DSCP match 0x0e
    0     0 ACCEPT     all  --  any    any
10.0.0.17            anywhere             DSCP match 0x00
    0     0 ACCEPT     tcp  --  any    any
10.0.0.17            10.0.100.27          tcp spts:webmin:20000
    dpts:webmin:2002

# Send 100 packets with a small payload on host1 with a DSCP
value of AF13 with a destination of host2:

cumulus@host1$ mz eth1 -A 10.0.0.17 -B 10.0.100.27 -c 100 -v -t
ip
  IP:  ver=4, len=20, tos=0, id=0, frag=0, ttl=255, proto=0,
sum=0, SA=10.0.0.17, DA=10.0.100.27,
      payload=

# Verify the 100 packets are matched on switch1

cumulus@switch1$ sudo cl-acltool -L ip
--------------------------------
Listing rules of type iptables:
--------------------------------
TABLE filter :
```

```
Chain INPUT (policy ACCEPT 9314 packets, 753K bytes)
  pkts bytes target     prot opt in     out
source               destination
  chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out
source               destination
  100  6400 ACCEPT     tcp  --  any    any
anywhere             anywhere             tcp dpt:ssh DSCP
match 0x2e
  100  7000 ACCEPT     all  --  swp3   any
anywhere             anywhere             DSCP match 0x0e
  100  6400 ACCEPT     all  --  any    any
10.0.0.17            anywhere             DSCP match 0x00
    0     0 ACCEPT     tcp  --  any    any
10.0.0.17            10.0.100.27          tcp spts:webmin:20000
dpts:webmin:2002

# Send 100 packets on host1 with a destination of host2:

cumulus@host1$ mz eth1 -A 10.0.0.17 -B 10.0.100.27 -c 100 -v -t
ip
 IP:  ver=4, len=20, tos=56, id=0, frag=0, ttl=255, proto=0,
sum=0, SA=10.0.0.17, DA=10.0.100.27,
     payload=
```

```
# Verify the 100 packets are matched on switch1

cumulus@switch1$ sudo cl-acltool -L ip
--------------------------------
Listing rules of type iptables:
--------------------------------
TABLE filter :
Chain INPUT (policy ACCEPT 9314 packets, 753K bytes)
  pkts bytes target     prot opt in     out
source               destination
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out
source               destination
  100  6400 ACCEPT     tcp  --  any    any
anywhere             anywhere             tcp dpt:ssh DSCP
match 0x2e
  100  7000 ACCEPT     all  --  swp3   any
anywhere             anywhere             DSCP match 0x0e
    0     0 ACCEPT     all  --  any    any
10.0.0.17            anywhere             DSCP match 0x00
    0     0 ACCEPT     tcp  --  any    any
10.0.0.17            10.0.100.27          tcp spts:webmin:20000
dpts:webmin:2002Still working
```

Filter Specific TCP Flags

The example solution below creates rules on the INPUT and FORWARD chains to drop ingress IPv4 and IPv6 TCP packets when the SYN bit is set and the RST, ACK, and FIN bits are reset. The default for the INPUT and FORWARD chains allows all other packets. The ACL is applied to ports swp20 and swp21. After configuring this ACL, new TCP sessions that originate from ingress ports swp20 and swp21 are not allowed. TCP sessions that originate from any other port are allowed.

```
INGRESS_INTF = swp20,swp21


[iptables]
-A INPUT,FORWARD --in-interface $INGRESS_INTF -p tcp --syn -j
DROP
[ip6tables]
-A INPUT,FORWARD --in-interface $INGRESS_INTF -p tcp --syn -j
DROP
```

The `--syn` flag in the above rule matches packets with the SYN bit set and the ACK, RST, and FIN bits are cleared. It is equivalent to using `-tcp-flags SYN,RST,ACK,FIN SYN`. For example, you can write the above rule as:

```
-A INPUT,FORWARD --in-interface $INGRESS_INTF -p tcp --tcp-
flags SYN,RST,ACK,FIN SYN -j DROP
```

Control Who Can SSH into the Switch

Run the following NCLU commands to control who can SSH into the switch. In the following example, 10.0.0.11/32 is the interface IP address (or loopback IP address) of the switch and 10.255.4.0/24 can SSH into the switch.

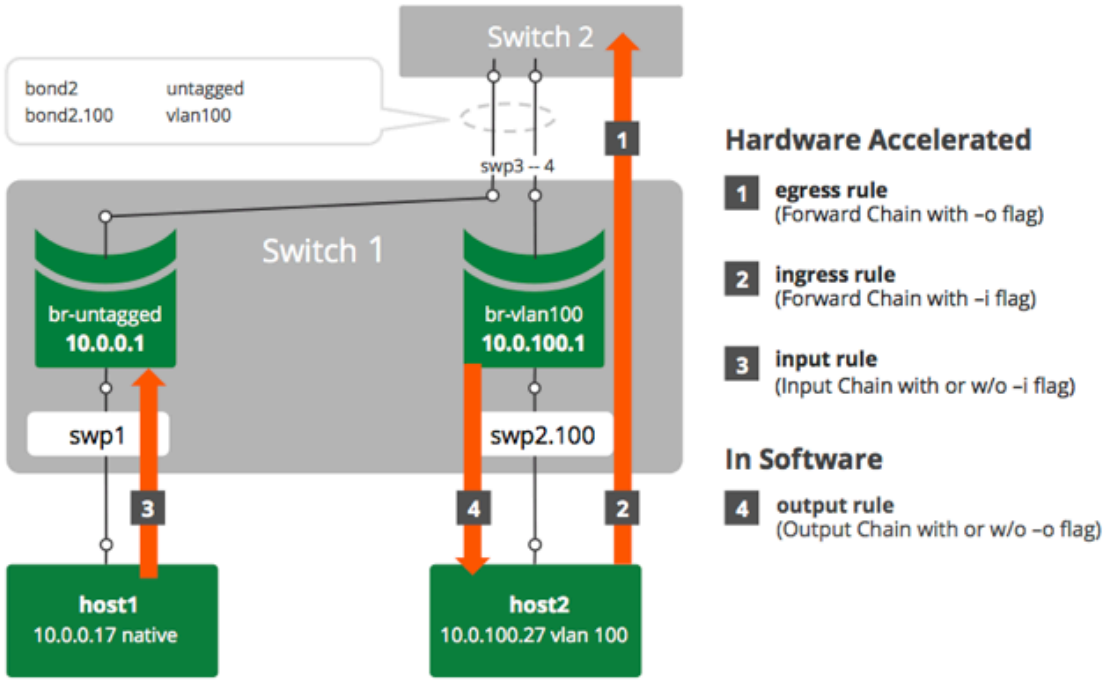
```
cumulus@switch:~$ net add acl ipv4 test priority 10 accept
source-ip 10.255.4.0/24 dest-ip 10.0.0.11/32
cumulus@switch:~$ net add acl ipv4 test priority 20 drop source-
ip any dest-ip 10.0.0.11/32
cumulus@switch:~$ net add control-plane acl ipv4 test inbound
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

> **NOTE**
>
> Cumulus Linux does not support the keyword `iprouter` (typically used for traffic sent to the CPU, where the destination MAC

address is that of the router but the destination IP address is not the router).

Example Configuration

The following example demonstrates how several different rules are applied.



Following are the configurations for the two switches used in these examples. The configuration for each switch appears in `/etc/network/interfaces` on that switch.

Switch 1 Configuration

```
cumulus@switch1:~$ net show configuration files
...
/etc/network/interfaces
=======================

auto swp1
iface swp1

auto swp2
iface swp2

auto swp3
iface swp3

auto swp4
iface swp4

auto bond2
iface bond2
  bond-slaves swp3 swp4

auto br-untagged
iface br-untagged
```

```
  address 10.0.0.1/24
  bridge_ports swp1 bond2
  bridge_stp on

auto br-tag100
iface br-tag100
  address 10.0.100.1/24
  bridge_ports swp2.100 bond2.100
  bridge_stp on
...
```

Switch 2 Configuration

```
cumulus@switch2:~$ net show configuration files
...
/etc/network/interfaces
=======================

auto swp3
iface swp3

auto swp4
iface swp4
```

```
auto br-untagged
iface br-untagged
  address 10.0.0.2/24
  bridge_ports bond2
  bridge_stp on

auto br-tag100
iface br-tag100
  address 10.0.100.2/24
  bridge_ports bond2.100
  bridge_stp on

auto bond2
iface bond2
  bond-slaves swp3 swp4
...
```

Egress Rule

The following rule blocks any TCP traffic with destination port 200 going from host1 or host2 through the switch (corresponding to rule 1 in the diagram above).

```
[iptables] -A FORWARD -o bond2 -p tcp --dport 200 -j DROP
```

Ingress Rule

The following rule blocks any UDP traffic with source port 200 going from host1 through the switch (corresponding to rule 2 in the diagram above).

```
[iptables] -A FORWARD -i swp2 -p udp --sport 200 -j DROP
```

Input Rule

The following rule blocks any UDP traffic with source port 200 and destination port 50 going from host1 to the switch (corresponding to rule 3 in the diagram above).

```
[iptables] -A INPUT -i swp1 -p udp --sport 200 --dport 50 -j
DROP
```

Output Rule

The following rule blocks any TCP traffic with source port 123 and destination port 123 going from Switch 1 to host2 (corresponding to rule 4 in the diagram above).

```
[iptables] -A OUTPUT -o br-tag100 -p tcp --sport 123 --dport
123 -j DROP
```

Combined Rules

The following rule blocks any TCP traffic with source port 123 and destination port 123 going from any switch port egress or generated from Switch 1 to host1 or host2 (corresponding to rules 1 and 4 in the diagram above).

```
[iptables] -A OUTPUT,FORWARD -o swp+ -p tcp --sport 123 --dport
123 -j DROP
```

This also becomes two ACLs and is the same as:

```
[iptables]
-A FORWARD -o swp+ -p tcp --sport 123 --dport 123 -j DROP
-A OUTPUT -o swp+ -p tcp --sport 123 --dport 123 -j DROP
```

Layer 2-only Rules/ebtables

The following rule blocks any traffic with source MAC address 00:00:00:00:00:12 and destination MAC address 08:9e:01:ce:e2:04 going

from any switch port egress/ingress.

```
[ebtables] -A FORWARD -s 00:00:00:00:00:12 -d 08:9e:01:ce:e2:04
-j DROP
```

Considerations

Not All Rules Supported

Not all `iptables`, `ip6tables`, or `ebtables` rules are supported. Refer to the Supported Rules section above for specific rule support.

ACL Log Policer Limits Traffic

To protect the CPU from overloading, traffic copied to the CPU is limited to 1 pkt/s by an ACL Log Policer.

Bridge Traffic Limitations

Bridge traffic that matches LOG ACTION rules are not logged in syslog; the kernel and hardware identify packets using different information.

Log Actions Cannot Be Forwarded

Logged packets cannot be forwarded. The hardware cannot both forward a packet and send the packet to the control plane (or kernel) for logging. To emphasize this, a log action must also have a drop action.

Broadcom Range Checker Limitations

Broadcom platforms have only 24 range checkers. This is a separate resource from the total number of ACLs allowed. If you are creating a large ACL configuration, use port ranges for large ranges of more than 5 ports.

Inbound LOG Actions Only for Broadcom Switches

On Broadcom-based switches, LOG actions can only be done on inbound interfaces (the ingress direction), not on outbound interfaces (the egress direction).

SPAN Sessions that Reference an Outgoing Interface

SPAN sessions that reference an outgoing interface create mirrored packets based on the ingress interface before the routing/switching decision. See SPAN Sessions that Reference an Outgoing Interface and Use the CPU Port as the SPAN Destination in the Network Troubleshooting section.

Tomahawk Hardware Limitations

Rate Limiting per Pipeline, Not Global

On Tomahawk switches, the field processor (FP) polices on a per-pipeline basis instead of globally, as with a Trident II switch. If packets come in to different switch ports that are on different pipelines on the ASIC, they might be rate limited differently.

For example, your switch is set so BFD is rate limited to 2000 packets per second. When the BFD packets are received on port1/pipe1 and port2/

pipe2, they are each rate limited at 2000 pps; the switch is rate limiting at 4000 pps overall. Because there are four pipelines on a Tomahawk switch, you might see a fourfold increase of your configured rate limits.

Atomic Update Mode Enabled by Default

In Cumulus Linux, atomic update mode is enabled by default. If you have Tomahawk switches and plan to use SPAN and/or mangle rules, you must disable atomic update mode.

To do so, enable nonatomic update mode by setting the value for `acl.non_atomic_update_mode` to TRUE in `/etc/cumulus/switchd.conf`, then restart `switchd`.

```
acl.non_atomic_update_mode = TRUE
```

Packets Undercounted during ACL Updates

On Tomahawk switches, when updating egress FP rules, some packets do no get counted. This results in an underreporting of counts during ping-pong or incremental switchover.

Trident II+ Hardware Limitations

On a Trident II+ switch, the TCAM allocation for ACLs is limited to 2048 rules in atomic mode for a default setup instead of 4096, as advertised for ingress rules.

Trident3 Hardware Limitations

TCAM Allocation

On a Trident3 switch, the TCAM allocation for ACLs is limited to 2048 rules in atomic mode for a default setup instead of 4096, as advertised for ingress rules.

Enable Nonatomic Mode

On a Trident3 switch, you must enable nonatomic update mode before you can configure ERSPAN. To do so, set the value for `acl.non_atomic_update_mode` to TRUE in `/etc/cumulus/switchd.conf`, then restart `switchd`.

```
acl.non_atomic_update_mode = TRUE
```

Egress ACL Rules

On Trident3 switches, egress ACL rules matching on the output SVI interface match layer 3 routed packets only, not bridged packets. To match layer 2 traffic, use egress bridge member port-based rules.

iptables Interactions with cl-acltool

Because Cumulus Linux is a Linux operating system, the `iptables` commands can be used directly. However, consider using `cl-acltool` instead because:

- Without using `cl-acltool`, rules are not installed into hardware.
- Running `cl-acltool -i` (the installation command) resets all rules and deletes anything that is not stored in `/etc/cumulus/acl/policy.conf`.

For example, running the following command works:

```
cumulus@switch:~$ sudo iptables -A INPUT -p icmp --icmp-type
echo-request -j DROP
```

And the rules appear when you run `cl-acltool -L`:

```
cumulus@switch:~$ sudo cl-acltool -L ip
-------------------------------
Listing rules of type iptables:
-------------------------------
TABLE filter :
Chain INPUT (policy ACCEPT 72 packets, 5236 bytes)
pkts bytes target  prot opt in   out   source    destination

0     0 DROP    icmp --  any  any   anywhere  anywhere
icmp echo-request
```

However, running `cl-acltool -i` or `reboot` removes them. To ensure all rules that can be in hardware are hardware accelerated, place them in the

`/etc/cumulus/acl/policy.conf` file, then run `cl-acltool -i`.

Mellanox Spectrum Hardware Limitations

Due to hardware limitations in the Spectrum ASIC, BFD policers are shared between all BFD-related control plane rules. Specifically the following default rules share the same policer in the `00control_plan.rules` file:

```
[iptables]
-A $INGRESS_CHAIN -p udp --dport $BFD_ECHO_PORT -j POLICE --set-
mode pkt --set-rate 2000 --set-burst 2000
-A $INGRESS_CHAIN -p udp --dport $BFD_PORT -j POLICE --set-mode
pkt --set-rate 2000 --set-burst 2000
-A $INGRESS_CHAIN -p udp --dport $BFD_MH_PORT -j POLICE --set-
mode pkt --set-rate 2000 --set-burst 2000

[ip6tables]
-A $INGRESS_CHAIN --in-interface $INGRESS_INTF -p udp --dport
$BFD_ECHO_PORT -j POLICE --set-mode pkt --set-rate 2000 --set-
burst 2000 --set-class 7
-A $INGRESS_CHAIN --in-interface $INGRESS_INTF -p udp --dport
$BFD_PORT -j POLICE --set-mode pkt --set-rate 2000 --set-burst
2000 --set-class 7
-A $INGRESS_CHAIN --in-interface $INGRESS_INTF -p udp --dport
$BFD_MH_PORT -j POLICE --set-mode pkt --set-rate 2000 --set-
burst 2000 --set-class 7
```

To work around this limitation, set the rate and burst of all 6 of these rules to the same values, using the `--set-rate` and `--set-burst` options.

Where to Assign Rules

- If a switch port is assigned to a bond, any egress rules must be assigned to the bond.
- When using the OUTPUT chain, rules must be assigned to the source. For example, if a rule is assigned to the switch port in the direction of traffic but the source is a bridge (VLAN), the traffic is not affected by the rule and must be applied to the bridge.
- If all transit traffic needs to have a rule applied, use the FORWARD chain, not the OUTPUT chain.

Generic Error Message Displayed after ACL Rule Installation Failure

After an ACL rule installation failure, a generic error message like the following is displayed:

```
cumulus@switch:$ sudo cl-acltool -i -p 00control_plane.rules
Using user provided rule file 00control_plane.rules
Reading rule file 00control_plane.rules ...
Processing rules in file 00control_plane.rules ...
error: hw sync failed (sync_acl hardware installation failed)
Installing acl policy... Rolling back ..
failed.
```

Dell S3048-ON Supports only 24K MAC Addresses

The Dell S3048-ON has a limit of 24576 MAC address entries instead of 32K for other 1G switches.

Mellanox Spectrum ASICs and INPUT Chain Rules

On switches with Mellanox Spectrum ASICs, INPUT chain rules are implemented using a trap mechanism. Packets headed to the CPU are assigned trap IDs. The default INPUT chain rules are mapped to these trap IDs. However, if a packet matches multiple traps, they are resolved by an internal priority mechanism that might be different from the rule priorities. Packets might not get policed by the default expected rule, but by another rule instead. For example, ICMP packets headed to the CPU are policed by the LOCAL rule instead of the ICMP rule. Also, multiple rules might share the same trap. In this case the policer that is applied is the largest of the policer values.

To work around this issue, create rules on the INPUT and FORWARD chains (INPUT,FORWARD).

Hardware Policing of Packets in the Input Chain

On certain platforms, there are limitations on hardware policing of packets in the INPUT chain. To work around these limitations, Cumulus Linux supports kernel based policing of these packets in software using limit/hashlimit matches. Rules with these matches are not hardware offloaded, but are ignored during hardware install.

ACLs Do not Match when the Output Port on the ACL is a

Subinterface

Packets don't get matched when a subinterface is configured as the output port. The ACL matches on packets only if the primary port is configured as an output port. If a subinterface is set as an output or egress port, the packets match correctly.

For example:

```
-A FORWARD --out-interface swp49s1.100 -j ACCEPT
```

Mellanox Switches and Egress ACL Matching on Bonds

On the Mellanox switch, ACL rules that match on an outbound *bond* interface are not supported. For example, the following rule is not supported:

```
[iptables]
-A FORWARD --out-interface <bond_intf> -j DROP
```

To work around this issue, duplicate the ACL rule on each physical port of the bond. For example:

```
[iptables]
-A FORWARD --out-interface <bond-member-port-1> -j DROP
-A FORWARD --out-interface <bond-member-port-2> -j DROP
```

Related Information

- Netfilter website
- Netfilter.org packet filtering how-to

Default Cumulus Linux ACL Configuration

The Cumulus Linux default ACL configuration is split into three parts: `iptables`, `ip6tables`, and `ebtables`. The sections below describe the default configurations for each part. You can see the default file by clicking the Default ACL Configuration link:

▼ Default ACL Configuration

iptables

Action/Value	Protocol/IP Address
Drop Destination IP: Any	Source IPv4: 240.0.0.0/5 loopback/8 224.0.0.0/4 255.255.255.255
Set class: 7 Police: Packet rate 2000 burst 2000 Source IP: Any Destination IP: Any	Protocol: UDP/BFD Echo UDP/BFD Control UDP BFD Multihop Control OSPF TCP/BGP (spt dpt 179) TCP/MLAG (spt dpt 5342)
Set Class: 6 Police: Rate 300 burst 100 Source IP: Any	Protocol: IGMP

Action/Value	Protocol/IP Address
Destination IP: Any	
Set class: 2 Police: Rate 100 burst 40 Source IP : Any Destination IP: Any	Protocol: ICMP
Set class: 2 Police: Rate 100 burst 100 Source IP: Any Destination IP: Any	Protocol: UDP/bootpc, bootps
Set class: 0 Police: Rate 1000 burst 1000 Source IP: Any Destination IP: Any	ADDRTYPE match dst-type LOCAL **Note**: LOCAL is any local address -> Receiving a packet with a destination matching a local IP address on the switch will go to the CPU.
Set class: 0 Police: Rate 400 burst 100 Source IP: Any Destination IP: Any	ADDRTYPE match dst-type IPROUTER **Note**: IPROUTER is any unresolved address -> On a l2/l3 boundary receiving a packet from L3 and needs to go to CPU in order to ARP for the destination.
Set class 0	All

> **NOTE**
>
> Set class is internal to the switch - it does not set any precedence bits.

ip6tables

Action/Value	Protocol/IP Address
Drop	Source IPv6: ff00::/8 :: ::ffff:0.0.0.0/96 localhost
Set class: 7 Police: Packet rate 2000 burst 2000 Source IPv6: Any Destination IPv6: Any	Protocol: UDP/BFD Echo UDP/BFD Control UDP BFD Multihop Control OSPF TCP/BGP (spt dpt 179)
Set class: 6 Police: Packet Rte: 200 burst 100 Source IPv6: Any Destination IPv6: Any	Protocol: Multicast Listener Query (MLD) Multicast Listener Report (MLD) Multicast Listener Done (MLD Multicast Listener Report V2
Set class: 2 Police: Packet rate: 100 burst 100	Protocol: ipv6-icmp router-solicitation

Action/Value	Protocol/IP Address
Source IPv6: Any Destination IPv6: Any	
Set class: 2 Police: Packet rate: 500 burst 500 Source IPv6: Any Destination IPv6: Any	Protocol: ipv6-icmp router-advertisement POLICE
Set class: 2 Police: Packet rate: 400 burst 400 Source IPv6: Any Destination IPv6: Any	Protocol: ipv6-icmp neighbour-solicitation ipv6-icmp neighbour-advertisement
Set class: 2 Police: Packet rate: 64 burst: 40 Source IPv6: Any Destination IPv6: Any	Protocol: Ipv6 icmp
Set class: 2 Police: Packet rate: 100 burst: 100 Source IPv6: Any Destination IPv6: Any	Protocol: UDP/ dhcpv6-client:dhcpv6-server (Spts & dpts)
Police: Packet rate: 1000 burst 1000 Source IPv6: Any Destination IPv6: Any	ADDRTYPE match dst-type LOCAL **Note**: LOCAL is any local address -> Receiving a packet with a destination matching a local IPv6 address on the switch will go to the CPU.

Action/Value	Protocol/IP Address
Set class: 0 Police: Packet rate: 400 burst 100	ADDRTYPE match dst-type IPROUTER **Note:** IPROUTER is an unresolved address -> On a l2/l3 boundary receiving a packet from L3 and needs to go to CPU in order to ARP for the destination.
Set class 0	All

> **NOTE**
>
> Set class is internal to the switch - it does not set any precedence bits.

ebtables

Action/Value	Protocol/MAC Address
Set Class: 7 Police: packet rate: 2000 burst rate:2000 Any switchport input interface	BDPU LACP= Cisco PVST
Set Class: 6 Police: packet rate: 200 burst rate: 200	LLDP CDP

Action/Value	Protocol/MAC Address
Any switchport input inteface	
Set Class: 2 Police: packet rate: 400 burst rate: 100 Any switchport input interface	ARP
Catch All: Allow all traffic Any switchport input interface	IPv4 IPv6
Catch All (applied at end): Set class: 0 Police: packet rate 100 burst rate 100 Any switchport	ALL OTHER

> **NOTE**
>
> Set class is internal to the switch. It does not set any precedence bits.

Considerations

Due to a hardware limitation on **Trident3 switches**, certain broadcast packets that are VXLAN decapsulated and sent to the CPU do not hit the normal INPUT chain ACL rules installed with `cl-acltool`.

You can configure policers for broadcast packets in the `/etc/cumulus/switchd.conf` file. The policers configuration format and default value is shown below:

```
cumulus@switch:~$ sudo cat /etc/cumulus/switchd.conf
...
#hal.bcm.vxlan_policers =
tunnel_arp=400,tunnel_dhcp_v4=100,tunnel_dhcp_v6=100,tunnel_ttl1=100,tunnel_rs=300
```

Filtering Learned MAC Addresses

On Broadcom switches, a MAC address is learned on a bridge regardless of whether or not a received packet is dropped by an ACL. This is due to how the hardware learns MAC addresses and occurs before the ACL lookup. This can be a security or resource problem as the MAC address table has the potential to get filled with bogus MAC addresses; a malfunctioning host, network error, loop, or malicious attack on a shared layer 2 platform can create an outage for other hosts if the same MAC address is learned on another port.

To prevent this from happening, Cumulus Linux filters frames before MAC learning occurs. Because MAC addresses and their port/VLAN associations are known at configuration time, you can create static MAC addresses, then create ingress ACLs to whitelist traffic from these MAC addresses and drop traffic otherwise.

> **NOTE**
>
> This feature is specific to switches on the Broadcom platform only; on switches with Mellanox Spectrum ASICs, the input port ACL does not have these issues when learning MAC addresses.

Create a configuration similar to the following, where you associate a port

and VLAN with a given MAC address, adding each one to the bridge:

```
cumulus@switch:~$ net add bridge bridge vids 100,200,300
cumulus@switch:~$ net add bridge bridge pvid 1
cumulus@switch:~$ net add bridge bridge ports swp1-3
cumulus@switch:~$ net add bridge pre-up bridge fdb add
00:00:00:00:00:11 dev swp1 master static vlan 100
cumulus@switch:~$ net add bridge pre-up bridge fdb add
00:00:00:00:00:22 dev swp2 master static vlan 200
cumulus@switch:~$ net add bridge pre-up bridge fdb add
00:00:00:00:00:33 dev swp3 master static vlan 300
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following configuration in the `/etc/network/interfaces` file:

```
auto swp1
iface swp1

auto swp2
iface swp2

auto swp3
```

```
iface swp3


auto bridge
iface bridge
    bridge-ports swp1 swp2 swp3
    bridge-pvid 1
    bridge-vids 100 200 300
    bridge-vlan-aware yes
    pre-up bridge fdb add 00:00:00:00:00:11 dev swp1 master
static vlan 100
    pre-up bridge fdb add 00:00:00:00:00:22 dev swp2 master
static vlan 200
    pre-up bridge fdb add 00:00:00:00:00:33 dev swp3 master
static vlan 300
```

If you need to list many MAC addresses, you can run a script to create the same configuration. For example, create a script called `macs.txt` and put in the `bridge fdb add` commands for each MAC address you need to configure:

```
cumulus@switch:~$ cat /etc/networks/macs.txt
#!/bin/bash
bridge fdb add 00:00:00:00:00:11 dev swp1 master static vlan 100
```

```
bridge fdb add 00:00:00:00:00:22 dev swp2 master static vlan 200
bridge fdb add 00:00:00:00:00:33 dev swp3 master static vlan 300
bridge fdb add 00:00:00:00:00:44 dev swp4 master static vlan 400
bridge fdb add 00:00:00:00:00:55 dev swp5 master static vlan 500
bridge fdb add 00:00:00:00:00:66 dev swp6 master static vlan 600
```

Then create the configuration using NCLU:

```
cumulus@switch:~$ net add bridge bridge vids 100,200,300
cumulus@switch:~$ net add bridge bridge pvid 1
cumulus@switch:~$ net add bridge bridge ports swp1-3
cumulus@switch:~$ net add bridge pre-up /etc/networks/macs.txt
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following configuration in the `/etc/network/interfaces` file:

```
auto swp1
iface swp1
```

```
auto swp2
iface swp2

auto swp3
iface swp3

auto swp4
iface swp4

auto swp5
iface swp5

auto swp6
iface swp6

auto bridge
iface bridge
    bridge-ports swp1 swp2 swp3 swp4 swp5 swp6
    bridge-pvid 1
    bridge-vids 100 200 300
    bridge-vlan-aware yes
    pre-up bridge fdb add 00:00:00:00:00:11 dev swp1 master
static vlan 100
    pre-up bridge fdb add 00:00:00:00:00:22 dev swp2 master
```

```
static vlan 200
    pre-up bridge fdb add 00:00:00:00:00:33 dev swp3 master
static vlan 300
    pre-up bridge fdb add 00:00:00:00:00:44 dev swp4 master
static vlan 400
    pre-up bridge fdb add 00:00:00:00:00:55 dev swp5 master
static vlan 500
    pre-up bridge fdb add 00:00:00:00:00:66 dev swp6 master
static vlan 600
```

Interactions with EVPN

If you are using EVPN, local static MAC addresses added to the local FDB are exported as static MAC addresses to remote switches. Remote MAC addresses are added as MAC addresses to the remote FDB.

Services and Daemons in Cumulus Linux

Services (also known as *daemons*) and *processes* are at the heart of how a Linux system functions. Most of the time, a service takes care of itself; you just enable and start it, then let it run. However, because a Cumulus Linux switch is a Linux system, you can dig deeper if you like. Services can start multiple processes as they run. Services are important to monitor on a Cumulus Linux switch.

You manage services in Cumulus Linux in the following ways:

- Identify currently active or stopped services
- Identify boot time state of a specific service
- Disable or enable a specific service
- Identify active listener ports

systemd and the systemctl Command

In general, you manage services using `systemd` via the `systemctl` command. You use it with any service on the switch to start, stop, restart, reload, enable, disable, reenable, or get the status of the service.

```
cumulus@switch:~$ sudo systemctl start | stop | restart |
status | reload | enable | disable | reenable
```

```
SERVICENAME.service
```

For example to restart networking, run the command:

```
cumulus@switch:~$ sudo systemctl restart networking.service
```

> **NOTE**
>
> The service name is written **after** the `systemctl` subcommand, not before it.

To show all the services currently running, run the `systemctl status` command. For example:

```
cumulus@switch:~$ sudo systemctl status
● switch
    State: running
      Jobs: 0 queued
    Failed: 0 units
    Since: Thu 2019-01-10 00:19:34 UTC; 23h ago
```

```
    CGroup: /
            ├─init.scope
            │ └─1 /sbin/init
            └─system.slice
              ├─haveged.service
              │ └─234 /usr/sbin/haveged --Foreground --
verbose=1 -w 1024
              ├─sysmonitor.service
              │ ├─  658 /bin/bash /usr/lib/cumulus/sysmonitor
              │ └─26543 sleep 60
              ├─systemd-udevd.service
              │ └─218 /lib/systemd/systemd-udevd
              ├─system-ntp.slice
              │ └─ntp@mgmt.service
              │   └─vrf
              │     └─mgmt
              │       └─12108 /usr/sbin/ntpd -n -u ntp:ntp -g
              ├─cron.service
              │ └─274 /usr/sbin/cron -f -L 38
              ├─system-serial\x2dgetty.slice
              │ └─serial-getty@ttyS0.service
              │   └─745 /sbin/agetty -o -p -- \u --keep-baud
115200,38400,9600 ttyS0 vt220
              ├─nginx.service
```

```
              | ├─332 nginx: master process /usr/sbin/nginx -g
daemon on; master_process on;
              | └─333 nginx: worker process
              ├─auditd.service
              | └─235 /sbin/auditd
              ├─rasdaemon.service
              | └─275 /usr/sbin/rasdaemon -f -r
              ├─clagd.service
              | └─11443 /usr/bin/python /usr/sbin/clagd --
daemon 169.254.1.2 peerlink.4094 44:39:39:ff:40:9
             --priority 100 --vxlanAnycas
              ├─switchd.service
              | └─430 /usr/sbin/switchd -vx
              ...
```

systemctl Subcommands

`systemctl` has a number of subcommands that perform a specific operation on a given service.

- **status** returns the status of the specified service.
- **start** starts the service.
- **stop** stops the service.
- **restart** stops, then starts the service, all the while maintaining state. If there are dependent services or services that mark the restarted service

as *Required*, the other services also restart. For example, running `systemctl restart frr.service` restarts any of the routing protocol services that are enabled and running, such as `bgpd` or `ospfd`.

- **reload** reloads the configuration for the service.
- **enable** enables the service to start when the system boots, but does not start it unless you use the `systemctl start SERVICENAME.service` command or reboot the switch.
- **disable** disables the service, but does not stop it unless you use the `systemctl stop SERVICENAME.service` command or reboot the switch. You can start or stop a disabled service.
- **reenable** disables, then enables a service. You might need to do this so that any new *Wants* or *WantedBy* lines create the symlinks necessary for ordering. This has no side effects on other services.

There is often little reason to interact with the services directly using these commands. If a critical service crashes or encounters an error, it is automatically respawned by systemd. systemd is effectively the caretaker of services in modern Linux systems and is responsible for starting all the necessary services at boot time.

Ensure a Service Starts after Multiple Restarts

By default, `systemd` is configured to try to restart a particular service only a certain number of times within a given interval before the service fails to start at all. The settings, *StartLimitInterval* (which defaults to 10 seconds) and *StartBurstLimit* (which defaults to 5 attempts) are stored in the service script; however, many services override these defaults, sometimes with

much longer times. For example, `switchd.service` sets *StartLimitInterval=10m* and *StartBurstLimit=3;* therefore, if you restart `switchd` more than 3 times in 10 minutes, it does not start.

When the restart fails for this reason, you see a message similar to the following:

```
Job for switchd.service failed. See 'systemctl status
switchd.service' and 'journalctl -xn' for details.
```

`systemctl status switchd.service` shows output similar to:

```
Active: failed (Result: start-limit) since Thu 2016-04-07
21:55:14 UTC; 15s ago
```

To clear this error, run `systemctl reset-failed switchd.service`. If you know you are going to restart frequently (multiple times within the StartLimitInterval), you can run the same command before you issue the restart request. This also applies to stop followed by start.

Keep systemd Services from Hanging after Starting

If you start, restart, or reload any `systemd` service that can be started from another `systemd` service, you must use the `--no-block` option with `systemctl`. Otherwise, that service or even the switch itself might hang after

starting or restarting.

Identify Active Listener Ports for IPv4 and IPv6

You can identify the active listener ports under both IPv4 and IPv6 using the `netstat` command:

```
cumulus@switch:~$ netstat -nlp --inet --inet6
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign
Address         State       PID/Program name
tcp        0      0 0.0.0.0:53
0.0.0.0:*               LISTEN      444/dnsmasq
tcp        0      0 0.0.0.0:22
0.0.0.0:*               LISTEN      874/sshd
tcp6       0      0 :::53
:::*                    LISTEN      444/dnsmasq
tcp6       0      0 :::22
:::*                    LISTEN      874/sshd
udp        0      0 0.0.0.0:28450
0.0.0.0:*                           839/dhclient
udp        0      0 0.0.0.0:53
0.0.0.0:*                           444/dnsmasq
udp        0      0 0.0.0.0:68
0.0.0.0:*                           839/dhclient
```

```
udp        0      0 192.168.0.42:123
0.0.0.0:*                           907/ntpd
udp        0      0 127.0.0.1:123
0.0.0.0:*                           907/ntpd
udp        0      0 0.0.0.0:123
0.0.0.0:*                           907/ntpd
udp        0      0 0.0.0.0:4784
0.0.0.0:*                           909/ptmd
udp        0      0 0.0.0.0:3784
0.0.0.0:*                           909/ptmd
udp        0      0 0.0.0.0:3785
0.0.0.0:*                           909/ptmd
udp6       0      0 :::58352
:::*                                839/dhclient
udp6       0      0 :::53
:::*                                444/dnsmasq
udp6       0      0 fe80::a200:ff:fe00::123
:::*                                907/ntpd
udp6       0      0 ::1:123
:::*                                907/ntpd
udp6       0      0 :::123
:::*                                907/ntpd
udp6       0      0 :::4784
:::*                                909/ptmd
```

```
udp6       0      0 :::3784
:::*                                    909/ptmd
```

Identify Services Currently Active or Stopped

To determine which services are currently active or stopped, run the `cl-service-summary` command:

```
cumulus@switch:~$ cl-service-summary
Service cron                enabled    active
Service ssh                 enabled    active
Service syslog              enabled    active
Service asic-monitor        enabled    inactive
Service clagd               enabled    inactive
Service cumulus-poe                    inactive
Service lldpd               enabled    active
Service mstpd               enabled    active
Service neighmgrd           enabled    active
Service netd                enabled    active
Service netq-agent          enabled    active
Service ntp                 enabled    active
Service portwd              enabled    active
Service ptmd                enabled    active
```

```
Service pwmd                enabled    active
Service smond               enabled    active
Service switchd             enabled    active
Service sysmonitor          enabled    active
Service rdnbrd              disabled   inactive
Service frr                 enabled    inactive
...
```

You can also run the `systemctl list-unit-files --type service` command to list all services on the switch and see which ones are enabled:

```
cumulus@switch:~$ systemctl list-unit-files --type service
UNIT FILE                                STATE
aclinit.service                          enabled
acltool.service                          enabled
acpid.service                            disabled
asic-monitor.service                     enabled
auditd.service                           enabled
autovt@.service                          disabled
bmcd.service                             disabled
bootlog.service                          enabled
bootlogd.service                         masked
bootlogs.service                         masked
```

```
bootmisc.service                        masked
checkfs.service                         masked
checkroot-bootclean.service             masked
checkroot.service                       masked
clagd.service                           enabled
console-getty.service                   disabled
console-shell.service                   disabled
container-getty@.service                static
cron.service                            enabled
cryptdisks-early.service                masked
cryptdisks.service                      masked
cumulus-aclcheck.service                static
cumulus-core.service                    static
cumulus-fastfailover.service            enabled
cumulus-firstboot.service               disabled
cumulus-hyperconverged.service          disabled
cumulus-platform.service                enabled
...
```

Identify Essential Services

If you need to know which services are required to run when the switch boots, run:

```
cumulus@switch:~$ systemctl list-dependencies --before

basic.target
```

To see which services are needed for networking, run:

```
cumulus@switch:~$ systemctl list-dependencies --after
network.target
● ├─switchd.service
● ├─wd_keepalive.service
● └─network-pre.target
```

To identify the services needed for a multi-user environment, run:

```
cumulus@switch:~$ systemctl list-dependencies --before multi-
user.target

● ├─bootlog.service
● ├─systemd-readahead-done.service
● ├─systemd-readahead-done.timer
● ├─systemd-update-utmp-runlevel.service
● └─graphical.target
● └─systemd-update-utmp-runlevel.service
```

Important Services

The following table lists the most important services in Cumulus Linux.

Service Name	Description	Affects Forwarding?
switchd	Hardware abstraction daemon. Synchronizes the kernel with the ASIC.	YES
sx_sdk	Interfaces with the Spectrum ASIC. Only on Spectrum switches.	YES
portwd	Port watch daemon. Broadcom switches only. Reads pluggable information over the I2C bus. Identifies and classifies the modules that are inserted into the system. Manages setting related to the module types that are inserted.	YES, eventually, if modules are added or removed
frr	FRRouting. Handles routing protocols. There are separate processes for each routing protocol, such as `bgpd` and `ospfd`.	YES if routing

Service Name	Description	Affects Forwarding?
clag	Cumulus link aggregation daemon. Handles MLAG.	YES if using MLAG
neighmgrd	Synchronizes MAC address information if using MLAG.	YES if using MLAG
mstpd	Spanning tree protocol daemon.	YES if using layer 2
ptmd	Prescriptive Topology Manager. Verifies cabling based on LLDP output. Also sets up BFD sessions.	YES if using BFD
netd	NCLU back end.	NO
rsyslog	Handles logging of syslog messages.	NO
ntp	Network time protocol.	NO
ledmgrd	LED manager. Reads the state of system LEDs.	NO
sysmonitor	Watches and logs critical system load (free memory, disk,	NO

Service Name	Description	Affects Forwarding?
	CPU).	
lldpd	Handles Tx/Rx of LLDP information.	NO
smond	Reads platform sensors and fan information from pwmd.	NO
pwmd	Reads and sets fan speeds.	NO

Configuring switchd

`switchd` is the daemon at the heart of Cumulus Linux. It communicates between the switch and Cumulus Linux, and all the applications running on Cumulus Linux.

The `switchd` configuration is stored in `/etc/cumulus/switchd.conf`.

The switchd File System

`switchd` also exports a file system, mounted on `/cumulus/switchd`, that presents all the `switchd` configuration options as a series of files arranged in a tree structure. To show the contents, run the `tree /cumulus/switchd` command. The following example shows output for a switch with one switch port configured:

```
cumulus@switch:~$ sudo tree /cumulus/switchd/
/cumulus/switchd/
├── clear
│   └── stats
│       ├── vlan
│       └── vxlan
├── config
│   ├── acl
│   │   ├── flow_based_mirroring
```

```
|   |   ├── non_atomic_update_mode
|   |   ├── optimize_hw
|   |   └── vxlan_tnl_arp_punt_disable
|   ├── arp
|   |   ├── drop_during_failed_state
|   |   └── next_hops
|   ├── bridge
|   |   ├── broadcast_frame_to_cpu
|   |   └── optimized_mcast_flood
|   ├── buf_util
|   |   ├── measure_interval
|   |   └── poll_interval
|   ├── coalesce
|   |   ├── offset
|   |   ├── reducer
|   |   └── timeout
|   ├── disable_internal_hw_err_restart
|   ├── disable_internal_parity_restart
|   ├── hal
|   |   └── bcm
|   |       ├── l3
|   |       |   └── per_vlan_router_mac_lookup_for_vrrp
|   |       ├── linkscan_interval
|   |       ├── logging
```

```
|   |       |   └── l3mc
|   |       ├── per_vlan_router_mac_lookup
|   |       └── vxlan_support
|   ├── ignore_non_swps
|   ├── interface
|   |   ├── swp1
|   |   |   ├── ethtool_mode
|   |   |   ├── interface_mode
|   |   |   ├── port_security
|   |   |   |   ├── enable
|   |   |   |   ├── mac_limit
|   |   |   |   ├── static_mac
|   |   |   |   ├── sticky_aging
|   |   |   |   ├── sticky_mac
|   |   |   |   ├── sticky_timeout
|   |   |   |   ├── violation_mode
|   |   |   |   └── violation_timeout
|   |   |   └── storm_control
|   |   |       ├── broadcast
|   |   |       ├── multicast
|   |   |       └── unknown_unicast
...
```

Configure switchd Parameters

To configure the `switchd` parameters, edit the `/etc/cumulus/switchd.conf` file. An example is provided below.

```
cumulus@switch:~$ sudo nano /etc/cumulus/switchd.conf
#
# /etc/cumulus/switchd.conf - switchd configuration file
#

# Statistic poll interval (in msec)
#stats.poll_interval = 2000

# Buffer utilization poll interval (in msec), 0 means disable
#buf_util.poll_interval = 0

# Buffer utilization measurement interval (in mins)
#buf_util.measure_interval = 0

# Optimize ACL HW resources for better utilization
#acl.optimize_hw = FALSE

# Enable Flow based mirroring.
#acl.flow_based_mirroring = TRUE
```

```
# Enable non atomic acl update
acl.non_atomic_update_mode = FALSE

# Send ARPs for next hops
#arp.next_hops = TRUE

# Kernel routing table ID, range 1 - 2^31, default 254
#route.table = 254
...
```

When you update the `/etc/cumulus/switchd.conf` file, you must restart `switchd` for the changes to take effect. See Restart switchd, below.

Restart switchd

Whenever you modify a `switchd` hardware configuration file (for example, you update any `*.conf` file that requires making a change to the switching hardware, like `/etc/cumulus/datapath/traffic.conf`), you must restart the `switchd` service for the change to take effect:

```
cumulus@switch:~$ sudo systemctl restart switchd.service
```

You do not have to restart the `switchd` service when you update a network

interface configuration (for example, when you edit the `/etc/network/interfaces` file).

WARNING

Restarting the `switchd` service causes all network ports to reset in addition to resetting the switch hardware configuration.

Power over Ethernet - PoE

Cumulus Linux supports Power over Ethernet (PoE) and PoE+, so certain Cumulus Linux switches can supply power from Ethernet switch ports to enabled devices over the Ethernet cables that connect them. PoE is capable of powering devices up to 15W, while PoE+ can power devices up to 30W. Configuration for power negotiation is done over LLDP.

The currently supported platforms include:

- Cumulus Express CX-1048-P
- Dell N3048EP-ON
- Delta AG6248C PoE
- EdgeCore AS4610-54P

PoE Basics

PoE functionality is provided by the `cumulus-poe` package. When a powered device is connected to the switch via an Ethernet cable:

- If the available power is greater than the power required by the connected device, power is supplied to the switch port, and the device powers on
- If available power is less than the power required by the connected device and the switch port's priority is less than the port priority set on all powered ports, power is **not** supplied to the port
- If available power is less than the power required by the connected device and the switch port's priority is greater than the priority of a

currently powered port, power is removed from lower priority port(s) and power is supplied to the port

- If the total consumed power exceeds the configured power limit of the power source, low priority ports are turned off. In the case of a tie, the port with the lower port number gets priority

Power is available as follows:

PSU 1	PSU 2	PoE Power Budget
920W	x	750W
x	920W	750W
920W	920W	1650W

The AS4610-54P has an LED on the front panel to indicate PoE status:

- Green: The `poed` daemon is running and no errors are detected
- Yellow: One or more errors are detected or the `poed` daemon is not running

> **NOTE**
>
> Link state and PoE state are completely independent of each other. When a link is brought down on a particular port using `ip link <port> down`, power on that port is not turned off; however, LLDP negotiation is not possible.

Configure PoE

You use the `poectl` command utility to configure PoE on a switch that supports the feature. You can:

- Enable or disable PoE for a given switch port
- Set a switch port's PoE priority to one of three values: *low*, *high* or *critical*

The PoE configuration resides in `/etc/cumulus/poe.conf`. The file lists all the switch ports, whether PoE is enabled for those ports and the priority for each port.

▼ Sample poe.conf file ...

By default, PoE and PoE+ are enabled on all Ethernet/1G switch ports, and these ports are set with a low priority. Switch ports can have low, high or critical priority.

There is no additional configuration for PoE+.

To change the priority for one or more switch ports, run `poectl -p swp# [low|high|critical]`. For example:

```
cumulus@switch:~$ sudo poectl -p swp1-swp5,swp7 high
```

To disable PoE for one or more ports, run `poectl -d [port_numbers]`:

```
cumulus@switch:~$ sudo poectl -d swp1-swp5,swp7
```

To display PoE information for a set of switch ports, run `poectl -i [port_numbers]`:

```
cumulus@switch:~$ sudo poectl -i swp10-swp13
Port          Status            Allocated    Priority  PD
type      PD class   Voltage   Current    Power
-----   ---------------------   -----------   --------
-----------   --------   -------   -------   ---------
swp10   connected             negotiating   low
IEEE802.3at   4          53.5 V     25 mA    3.9 W
swp11   searching             n/a           low
IEEE802.3at   none        0.0 V      0 mA    0.0 W
swp12   connected             n/a           low
IEEE802.3at   2          53.5 V     25 mA    1.4 W
swp13   connected             51.0 W        low
IEEE802.3at   4          53.6 V     72 mA    3.8 W
```

The **Status** can be one of the following:

- **searching:** PoE is enabled but no device has been detected.
- **disabled:** The PoE port has been configured as disabled.

- **connected:** A powered device is connected and receiving power.
- **power-denied:** There is insufficient PoE power available to enable the connected device.

The **Allocated** column displays how much PoE power has been allocated to the port, which can be one of the following:

- **n/a:** No device is connected or the connected device does not support LLDP negotiation.
- **negotiating:** An LLDP-capable device is connected and is negotiating for PoE power.
- **XX.X W:** An LLDP-capable device has negotiated for XX.X watts of power (for example, 51.0 watts for swp13 above).

To see all the PoE information for a switch, run `poectl -s`:

```
cumulus@switch:~$ poectl -s
System power:
  Total:      730.0 W
  Used:        11.0 W
  Available:  719.0 W
Connected ports:
  swp11, swp24, swp27, swp48
```

The set commands (priority, enable, disable) either succeed silently or display an error message if the command fails.

The `poectl` command takes the following arguments:

Argument	Description
`-h`, `--help`	Show this help message and exit.
`-i`, `--port-info <port-list>`	Returns detailed information for the specified ports. You can specify a range of ports. For example: `-i swp1-swp5,swp10`. **Note**: On an Edge-Core AS4610-54P switch, the voltage reported by the poectl -i command and measured through a power meter connected to the device varies by 5V. The current and power readings are correct and no difference is seen for them.
`-a`, `--all`	Returns PoE status and detailed information for all ports.
`-p`, `--priority <port-list> <priority>`	Sets priority for the specified ports: low, high, critical.
`-d`, `--disable-ports <port-list>`	Disables PoE operation on the specified ports.
`-e`, `--enable-ports <port-list>`	Enables PoE operation on the specified ports.
`-s`, `--system`	Returns PoE status for the entire switch.

Argument	Description
`-r`, `--reset <port-list>`	Performs a hardware reset on the specified ports. Use this if one or more ports are stuck in an error state. This does not reset any configuration settings for the specified ports.
`-v`, `--version`	Displays version information.
`-j`, `--json`	Displays output in JSON format.
`--save`	Saves the current configuration. The saved configuration is automatically loaded on system boot.
`--load`	Loads and applies the saved configuration.

Troubleshooting

You can troubleshoot PoE and PoE+ using the following utilities and files:

- `poectl -s`, as described above.
- The Cumulus Linux `cl-support` script, which includes PoE-related output from `poed.conf`, `syslog`, `poectl --diag-info` and `lldpctl`.
- `lldpcli show neighbors ports <swp> protocol lldp hidden details`
- `tcpdump -v -v -i <swp> ether proto 0x88cc`
- The contents of the PoE/PoE+ `/etc/lldpd.d/poed.conf` configuration file, as described above.

Verify the Link Is Up

LLDP requires network connectivity, so verify that the link is up.

```
cumulus@switch:~$ net show interface swp20
    Name    MAC                Speed      MTU  Mode
--  ------  -----------------  -------  -----  ---------
UP  swp20   44:38:39:00:00:04  1G        9216  Access/L2
```

View LLDP Information Using lldpcli

You can run `lldpcli` to view the LLDP information that has been received on a switch port. For example:

```
cumulus@switch:~$ sudo lldpcli show neighbors ports swp20
protocol lldp hidden details
-------------------------------------------------------------------------------
LLDP neighbors:
-------------------------------------------------------------------------------
Interface:    swp20, via: LLDP, RID: 2, Time: 0 day, 00:03:34
  Chassis:
    ChassisID:    mac 68:c9:0b:25:54:7c
    SysName:      ihm-ubuntu
    SysDescr:     Ubuntu 14.04.2 LTS Linux 3.14.4+ #1 SMP Thu
```

```
Jun 26 00:54:44 UTC 2014 armv7l
    MgmtIP:       fe80::6ac9:bff:fe25:547c
    Capability:   Bridge, off
    Capability:   Router, off
    Capability:   Wlan, off
    Capability:   Station, on
  Port:
    PortID:       mac 68:c9:0b:25:54:7c
    PortDescr:    eth0
    PMD autoneg:  supported: yes, enabled: yes
      Adv:          10Base-T, HD: yes, FD: yes
      Adv:          100Base-TX, HD: yes, FD: yes
      MAU oper type: 100BaseTXFD - 2 pair category 5 UTP, full
duplex mode
    MDI Power:    supported: yes, enabled: yes, pair control: no
      Device type:  PD
      Power pairs:  spare
      Class:        class 4
      Power type:   2
      Power Source: Primary power source
      Power Priority: low
      PD requested power Value: 51000
      PSE allocated power Value: 51000
  UnknownTLVs:
```

```
    TLV:          OUI: 00,01,42, SubType: 1, Len: 1 05
    TLV:          OUI: 00,01,42, SubType: 1, Len: 1 0D
--------------------------------------------------------------------------------
```

View LLDP Information Using tcpdump

You can use `tcpdump` to view the LLDP frames being transmitted and received. For example:

```
cumulus@switch:~$ sudo tcpdump -v -v -i swp20 ether proto 0x88cc
tcpdump: listening on swp20, link-type EN10MB (Ethernet),
capture size 262144 bytes
18:41:47.559022 LLDP, length 211
    Chassis ID TLV (1), length 7
      Subtype MAC address (4): 00:30:ab:f2:d7:a5 (oui Unknown)
      0x0000:  0400 30ab f2d7 a5
    Port ID TLV (2), length 6
      Subtype Interface Name (5): swp20
      0x0000:  0573 7770 3230
    Time to Live TLV (3), length 2: TTL 120s
      0x0000:  0078
    System Name TLV (5), length 13: dni-3048up-09
      0x0000:  646e 692d 3330 3438 7570 2d30 39
```

```
    System Description TLV (6), length 68
      Cumulus Linux version 3.0.1~1466303042.2265c10 running on
dni 3048up
      0x0000:  4375 6d75 6c75 7320 4c69 6e75 7820 7665
      0x0010:  7273 696f 6e20 332e 302e 317e 3134 3636
      0x0020:  3330 3330 3432 2e32 3236 3563 3130 2072
      0x0030:  756e 6e69 6e67 206f 6e20 646e 6920 3330
      0x0040:  3438 7570
    System Capabilities TLV (7), length 4
      System  Capabilities [Bridge, Router] (0x0014)
      Enabled Capabilities [Router] (0x0010)
      0x0000:  0014 0010
    Management Address TLV (8), length 12
      Management Address length 5, AFI IPv4 (1): 10.0.3.190
      Interface Index Interface Numbering (2): 2
      0x0000:  0501 0a00 03be 0200 0000 0200
    Management Address TLV (8), length 24
      Management Address length 17, AFI IPv6 (2):
fe80::230:abff:fef2:d7a5
      Interface Index Interface Numbering (2): 2
      0x0000:  1102 fe80 0000 0000 0000 0230 abff fef2
      0x0010:  d7a5 0200 0000 0200
    Port Description TLV (4), length 5: swp20
      0x0000:  7377 7032 30
```

```
    Organization specific TLV (127), length 9: OUI IEEE 802.3
Private (0x00120f)
      Link aggregation Subtype (3)
        aggregation status [supported], aggregation port ID 0
      0x0000:  0012 0f03 0100 0000 00
    Organization specific TLV (127), length 9: OUI IEEE 802.3
Private (0x00120f)
      MAC/PHY configuration/status Subtype (1)
        autonegotiation [supported, enabled] (0x03)
        PMD autoneg capability [10BASE-T fdx, 100BASE-TX fdx,
1000BASE-T fdx] (0x2401)
        MAU type 100BASEFX fdx (0x0012)
      0x0000:  0012 0f01 0324 0100 12
    Organization specific TLV (127), length 12: OUI IEEE 802.3
Private (0x00120f)
      Power via MDI Subtype (2)
        MDI power support [PSE, supported, enabled], power pair
spare, power class class4
      0x0000:  0012 0f02 0702 0513 01fe 01fe
    Organization specific TLV (127), length 5: OUI Unknown
(0x000142)
      0x0000:  0001 4201 0d
    Organization specific TLV (127), length 5: OUI Unknown
(0x000142)
```

```
      0x0000:  0001 4201 01
    End TLV (0), length 0
```

Log poed Events in syslog

The `poed` service logs the following events to `syslog` when:

- A switch provides power to a powered device.
- A device that was receiving power is removed.
- The power available to the switch changes.
- Errors are detected.

Configuring a Global Proxy

You configure global HTTP and HTTPS proxies in the `/etc/profile.d/` directory of Cumulus Linux. To do so, set the `http_proxy` and `https_proxy` variables, which tells the switch the address of the proxy server to use to fetch URLs on the command line. This is useful for programs such as `apt`/`apt-get`, `curl` and `wget`, which can all use this proxy.

1. In a terminal, create a new file in the `/etc/profile.d/` directory. In the code example below, the file is called `proxy.sh`, and is created using the text editor `nano`.

```
cumulus@switch:~$ sudo nano /etc/profile.d/proxy.sh
```

2. Add a line to the file to configure either an HTTP or an HTTPS proxy, or both:

   - HTTP proxy:

```
http_proxy=http://myproxy.domain.com:8080
export http_proxy
```

   - HTTPS proxy:

```
https_proxy=https://myproxy.domain.com:8080
export https_proxy
```

3. Create a file in the `/etc/apt/apt.conf.d` directory and add the following lines to the file for acquiring the HTTP and HTTPS proxies; the example below uses `http_proxy` as the file name:

```
cumulus@switch:~$ sudo nano /etc/apt/apt.conf.d/http_proxy
Acquire::http::Proxy "http://myproxy.domain.com:8080";
Acquire::https::Proxy "https://myproxy.domain.com:8080";
```

4. Add the proxy addresses to `/etc/wgetrc`; you may have to uncomment the `http_proxy` and `https_proxy` lines:

```
cumulus@switch:~$ sudo nano /etc/wgetrc
...
https_proxy = https://myproxy.domain.com:8080
http_proxy = http://myproxy.domain.com:8080
...
```

5. Run the `source` command, to execute the file in the current environment:

```
cumulus@switch:~$ source /etc/profile.d/proxy.sh
```

The proxy is now configured. The `echo` command can be used to confirm aproxy is set up correctly:

- HTTP proxy:

```
cumulus@switch:~$ echo $http_proxy
http://myproxy.domain.com:8080
```

- HTTPS proxy:

```
cumulus@switch:~$ echo $https_proxy
https://myproxy.domain.com:8080
```

Related Information

Set up an apt package cache

HTTP API

Cumulus Linux implements an HTTP application programming interface to the OpenStack ML2 driver and NCLU. Instead of accessing Cumulus Linux using SSH, you can interact with the switch using an HTTP client, such as cURL, HTTPie or a web browser.

HTTP API Basics

The supporting software for the API is installed with Cumulus Linux.

To use the REST API, you must enable `nginx` on the switch:

```
cumulus@switch:~$ sudo systemctl enable nginx; systemctl
restart nginx
```

To enable the HTTP API service, run the following `systemd` command:

```
cumulus@switch:~$ sudo systemctl enable restserver
```

Use the `systemctl start` and `systemctl stop` commands to start or stop the service:

```
cumulus@switch:~$ sudo systemctl start restserver
cumulus@switch:~$ sudo systemctl stop restserver
```

Each service runs as a background daemon.

Configure API Services

To configure the HTTP API services, edit the `/etc/nginx/sites-available/nginx-restapi.conf` configuration file, then run the `sudo systemctl restart nginx` command.

IP and Port Settings

You can modify the IP:port combinations to which services listen by changing the parameters of the `listen` directive(s). By default, `nginx-restapi.conf` has only one `listen` parameter.

> **NOTE**
>
> All URLs must use HTTPS instead of HTTP.

For more information on the listen directive, refer to the NGINX documentation.

Configure Security

Authentication

The default configuration requires all HTTP requests from external sources (not internal switch traffic) to set the HTTP Basic Authentication header.

The user and password must correspond to a user on the host switch.

Transport Layer Security

All traffic must be secured in transport using TLSv1.2 by default. Cumulus Linux contains a self-signed certificate and private key used server-side in this application so that it works out of the box, but using your own certificates and keys is highly recommended. Certificates must be in the PEM format.

For step by step documentation for generating self-signed certificates and keys, and installing them to the switch, refer to the Ubuntu Certificates and Security documentation.

WARNING

Do not copy the `cumulus.pem` or `cumulus.key` files. After installation, edit the `ssl_certificate` and `ssl_certificate_key` values in the

configuration file for your hardware.

cURL Examples

This section includes several example cURL commands you can use to send HTTP requests to a host. The following settings are used for these examples:

- Username: `user`
- Password: `pw`
- IP: `192.168.0.32`
- Port: `8080`

NOTE

Requests for NCLU require setting the Content-Type request header to be set to `application/json`.

The cURL `-k` flag is necessary when the server uses a self-signed certificate. This is the default configuration (see the Security section). To display the response headers, include the `-D` flag in the command.

To retrieve a list of all available HTTP endpoints:

```
cumulus@switch:~$ curl -X GET -k -u user:pw
https://192.168.0.32:8080
```

To run `net show counters` on the host as a remote procedure call:

```
cumulus@switch:~$ curl -X POST -k -u user:pw -H "Content-Type:
application/json" -d '{"cmd": "show counters"}'
https://192.168.0.32:8080/nclu/v1/rpc
```

To add a bridge using ML2:

```
cumulus@switch:~$ curl -X PUT -k -u user:pw
https://192.168.0.32:8080/ml2/v1/bridge/"br1"/200
```

Considerations

The `/etc/restapi.conf` file is *not* listed in the `net show configuration files` command output.

Layer 1 and Switch Ports

This section discusses how to configure network interfaces and DHCP delays and servers. The Prescriptive Topology Manager (PTM) cabling verification tool is also discussed.

Interface Configuration and Management

`ifupdown` is the network interface manager for Cumulus Linux. Cumulus Linux uses an updated version of this tool, `ifupdown2`.

For more information on network interfaces, see Switch Port Attributes.

> ⚠ **IMPORTANT**
>
> By default, `ifupdown` is quiet. Use the verbose option (`-v`) to show commands as they are executed when bringing an interface down or up.

Basic Commands

To bring up the physical connection to an interface or apply changes to an existing interface, run the `sudo ifup <interface>` command. The following example command brings up the physical connection to swp1:

```
cumulus@switch:~$ sudo ifup swp1
```

To bring down the physical connection to a single interface, run the `sudo ifdown <interface>` command. The following example command brings down the physical connection to swp1:

```
cumulus@switch:~$ sudo ifdown swp1
```

The `ifdown` command always deletes logical interfaces after bringing them down. When you bring down the physical connection to an interface, it is brought back up automatically after any future reboots or configuration changes with `ifreload -a`.

To administratively bring the interface up or down; for example, to bring down a port, bridge, or bond but not the physical connection for a port, bridge, or bond, you can use the `--admin-state` option. Alternatively, you can use NCLU commands.

When you put an interface into an admin down state, the interface *remains down* after any future reboots or configuration changes with `ifreload -a`.

NCLU Commands | Linux Commands

To put an interface into an admin *down* state, run the `net add interface <interface> link down` command.

```
cumulus@switch:~$ net add interface swp1 link down
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following configuration in the `/etc/network/interfaces` file:

```
auto swp1
iface swp1
    link-down yes
```

To bring the interface back *up*, run the `net del interface <interface> link down` command.

```
cumulus@switch:~$ net del interface swp1 link down
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

To see the link and administrative state, use the `ip link show` command. In the following example, swp1 is administratively UP and the physical link is UP (LOWER_UP flag).

```
cumulus@switch:~$ ip link show dev swp1
3: swp1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
pfifo_fast state UP mode DEFAULT qlen 500
    link/ether 44:38:39:00:03:c1 brd ff:ff:ff:ff:ff:ff
```

For additional information on interface administrative state and physical state, refer to this knowledge base article.

ifupdown2 Interface Classes

`ifupdown2` enables you to group interfaces into separate classes, where a class is a user-defined label that groups interfaces that share a common function (such as uplink, downlink or compute). You specify classes in the `/etc/network/interfaces` file.

The most common class is *auto*, which you configure like this:

```
auto swp1
iface swp1
```

You can add other classes using the *allow* prefix. For example, if you have multiple interfaces used for uplinks, you can define a class called *uplinks:*

```
auto swp1
allow-uplink swp1
iface swp1 inet static
    address 10.1.1.1/31

auto swp2
allow-uplink swp2
iface swp2 inet static
    address 10.1.1.3/31
```

This allows you to perform operations on only these interfaces using the `--allow=uplinks` option. You can still use the `-a` options because these interfaces are also in the *auto* class:

```
cumulus@switch:~$ sudo ifup --allow=uplinks
cumulus@switch:~$ sudo ifreload -a
```

If you are using Management VRF, you can use the special interface class called *mgmt* and put the management interface into that class. The management VRF must have an IPv6 address in addition to an IPv4 address to work correctly.

WARNING

The *mgmt* interface class is not supported with NCLU commands.

```
allow-mgmt eth0
iface eth0 inet dhcp
    vrf mgmt

allow-mgmt mgmt
iface mgmt
    address 127.0.0.1/8
    address ::1/128
    vrf-table auto
```

All `ifupdown2` commands (`ifup`, `ifdown`, `ifquery`, `ifreload`) can take a class. Include the `--allow=<class>` option when you run the command. For example, to reload the configuration for the management interface described above, run:

```
cumulus@switch:~$ sudo ifreload --allow=mgmt
```

Use the `-a` option to bring up or down all interfaces that are marked with the common `auto` class in the `/etc/network/interfaces` file.

To administratively bring up all interfaces marked `auto`, run:

```
cumulus@switch:~$ sudo ifup -a
```

To administratively bring down all interfaces marked `auto`, run:

```
cumulus@switch:~$ sudo ifdown -a
```

To reload all network interfaces marked `auto`, use the `ifreload` command. This command is equivalent to running `ifdown` then `ifup`; however, `ifreload` skips unchanged configurations:

```
cumulus@switch:~$ sudo ifreload -a
```

TIP

Certain syntax checks are done by default. As a precaution, apply

configurations only if the syntax check passes. Use the following compound command:

```
cumulus@switch:~$ sudo bash -c "ifreload -s -a &&
ifreload -a"
```

For more information, see the individual man pages for `ifup(8)`, `ifdown(8)`, `ifreload(8)`.

Configure a Loopback Interface

Cumulus Linux has a loopback interface preconfigured in the `/etc/network/interfaces` file. When the switch boots up, it has a loopback interface called *lo*, which is up and assigned an IP address of 127.0.0.1.

TIP

The loopback interface *lo* must always be specified in the `/etc/`

`network/interfaces` file and must always be up.

To see the status of the loopback interface (lo):

NCLU Commands | Linux Commands

Use the `net show interface lo` command.

```
cumulus@switch:~$ net show interface lo
    Name    MAC                Speed    MTU    Mode
--  ------  -----------------  -------  -----  --------
UP  lo      00:00:00:00:00:00  N/A      65536  Loopback


Alias
-----
loopback interface
IP Details
-------------------------  --------------------
IP:                        127.0.0.1/8, ::1/128
IP Neighbor(ARP) Entries:  0
```

The loopback is up and is assigned an IP address of 127.0.0.1.

To add an IP address to a loopback interface, configure the *lo* interface:

```
cumulus@switch:~$ net add loopback lo ip address 10.1.1.1/32
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Configure Multiple Loopbacks

You can configure multiple loopback addresses by assigning additional IP addresses to the lo interface.

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add loopback lo ip address 172.16.2.1/
24
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following configuration in the `/etc/network/interfaces` file:

```
cumulus@leaf01:~$ cat /etc/network/interfaces
...

# The loopback network interface
auto lo
iface lo inet loopback
    # The primary network interface
    address 172.16.2.1/24
```

ifupdown2 Behavior with Child Interfaces

By default, `ifupdown2` recognizes and uses any interface present on the system that is listed as a dependent of an interface (for example, a VLAN, bond, or physical interface). You are not required to list interfaces in the `interfaces` file unless they need a specific configuration for MTU, link speed, and so on. If you need to delete a child interface, delete all references to that interface from the `interfaces` file.

In the following example, swp1 and swp2 do not need an entry in the `interfaces` file. The following stanzas defined in `/etc/network/interfaces` provide the exact same configuration:

With Child Interfaces Defined:

```
auto swp1
iface swp1

auto swp2
iface swp2

auto bridge
iface bridge
    bridge-vlan-aware yes
    bridge-ports swp1 swp2
```

```
    bridge-vids 1-100
    bridge-pvid 1
    bridge-stp on
```

Without Child Interfaces Defined

```
auto bridge
iface bridge
    bridge-vlan-aware yes
    bridge-ports swp1 swp2
    bridge-vids 1-100
    bridge-pvid 1
    bridge-stp on</code></pre></td>
```

In the following example, swp1.100 and swp2.100 do not need an entry in the `interfaces` file. The following stanzas defined in `/etc/network/interfaces` provide the exact same configuration:

With Child Interfaces Defined

```
auto swp1.100
```

```
iface swp1.100


auto swp2.100
iface swp2.100


auto br-100
iface br-100
    address 10.0.12.2/24
    address 2001:dad:beef::3/64
    bridge-ports swp1.100 swp2.100
    bridge-stp on
```

Without Child Interfaces Defined

```
auto br-100
iface br-100
    address 10.0.12.2/24
    address 2001:dad:beef::3/64
    bridge-ports swp1.100 swp2.100
    bridge-stp on
```

For more information about bridges in traditional mode and bridges in VLAN-aware mode, read this knowledge base article.

ifupdown2 Interface Dependencies

`ifupdown2` understands interface dependency relationships. When you run `ifup` and `ifdown` with all interfaces, the commands always run with all interfaces in dependency order. When you run `ifup` and `ifdown` with the interface list on the command line, the default behavior is to *not* run with dependents; however, if there are any built-in dependents, they will be brought up or down.

To run with dependents when you specify the interface list, use the `--with-depends` option. The `--with-depends` option walks through all dependents in the dependency tree rooted at the interface you specify. Consider the following example configuration:

```
auto bond1
iface bond1
    address 100.0.0.2/16
    bond-slaves swp29 swp30

auto bond2
iface bond2
    address 100.0.0.5/16
    bond-slaves swp31 swp32
```

```
auto br2001
iface br2001
    address 12.0.1.3/24
    bridge-ports bond1.2001 bond2.2001
    bridge-stp on
```

The `ifup --with-depends br2001` command brings up all dependents of br2001: bond1.2001, bond2.2001, bond1, bond2, bond1.2001, bond2.2001, swp29, swp30, swp31, swp32.

```
cumulus@switch:~$ sudo ifup --with-depends br2001
```

The `ifdown --with-depends br2001` command brings down all dependents of br2001: bond1.2001, bond2.2001, bond1, bond2, bond1.2001, bond2.2001, swp29, swp30, swp31, swp32.

```
cumulus@switch:~$ sudo ifdown --with-depends br2001
```

WARNING

`ifdown2` always deletes logical interfaces after bringing them down. Use the `--admin-state` option if you only want to administratively bring the interface up or down. In the above example, `ifdown br2001` deletes `br2001`.

To guide you through which interfaces will be brought down and up, use the `--print-dependency` option.

For example, run `ifquery --print-dependency=list -a` to show the dependency list for all interfaces:

```
cumulus@switch:~$ sudo ifquery --print-dependency=list -a
lo : None
eth0 : None
bond0 : ['swp25', 'swp26']
bond1 : ['swp29', 'swp30']
bond2 : ['swp31', 'swp32']
br0 : ['bond1', 'bond2']
bond1.2000 : ['bond1']
bond2.2000 : ['bond2']
```

```
br2000 : ['bond1.2000', 'bond2.2000']
bond1.2001 : ['bond1']
bond2.2001 : ['bond2']
br2001 : ['bond1.2001', 'bond2.2001']
swp40 : None
swp25 : None
swp26 : None
swp29 : None
swp30 : None
swp31 : None
swp32 : None
```

To print the dependency list of a single interface, run the `ifquery --print-dependency=list <interface>` command. The following example command shows the dependency list for br2001:

```
cumulus@switch:~$ sudo ifquery --print-dependency=list br2001
br2001 : ['bond1.2001', 'bond2.2001']
bond1.2001 : ['bond1']
bond2.2001 : ['bond2']
bond1 : ['swp29', 'swp30']
bond2 : ['swp31', 'swp32']
swp29 : None
```

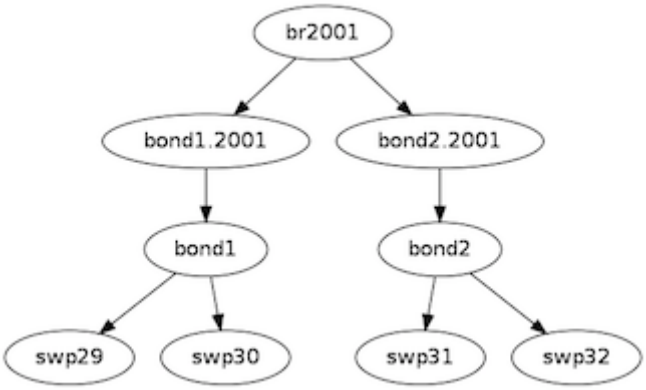
```
swp30 : None
swp31 : None
swp32 : None
```

To show the dependency information for an interface in `dot` format, run the `ifquery --print-dependency=dot <interface>` command. The following example command shows the dependency information for interface br2001 in `dot` format:

```
cumulus@switch:~$ sudo ifquery --print-dependency=dot br2001
/* Generated by GvGen v.0.9 (http://software.inl.fr/trac/wiki/
GvGen) */
digraph G {
    compound=true;
    node1 [label="br2001"];
    node2 [label="bond1.2001"];
    node3 [label="bond2.2001"];
    node4 [label="bond1"];
    node5 [label="bond2"];
    node6 [label="swp29"];
    node7 [label="swp30"];
    node8 [label="swp31"];
```

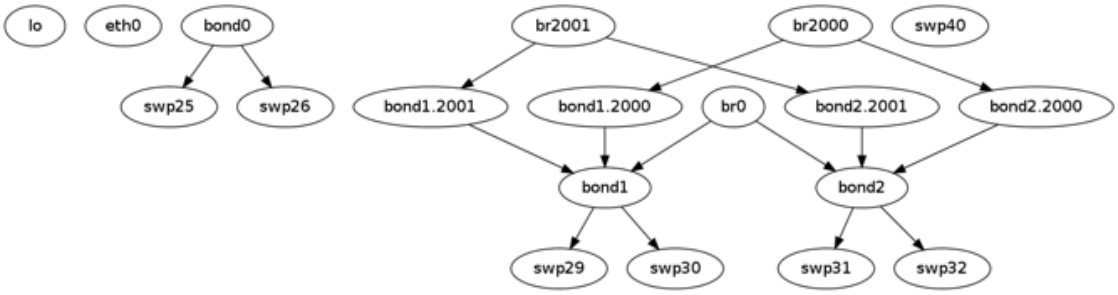
```
    node9 [label="swp32"];
    node1->node2;
    node1->node3;
    node2->node4;
    node3->node5;
    node4->node6;
    node4->node7;
    node5->node8;
    node5->node9;
}
```

You can use `dot` to render the graph on an external system where `dot` is installed.



To print the dependency information of the entire `interfaces` file, run the following command:

```
cumulus@switch:~$ sudo ifquery --print-dependency=dot -a
>interfaces_all.dot
```



Subinterfaces

On Linux, an *interface* is a network device that can be either physical, like a switch port (for example, swp1) or virtual, like a VLAN (for example, vlan100). A *VLAN subinterface* is a VLAN device on an interface, and the VLAN ID is appended to the parent interface using dot (.) VLAN notation. For example, a VLAN with ID 100 that is a subinterface of swp1 is named swp1.100. The dot VLAN notation for a VLAN device name is a standard way to specify a VLAN device on Linux. Many Linux configuration tools, such as `ifupdown2` and its predecessor `ifupdown`, recognize such a name as a VLAN interface name.

A VLAN subinterface only receives traffic tagged for that VLAN; therefore, swp1.100 only receives packets tagged with VLAN 100 on switch port swp1. Similarly, any packets transmitted from swp1.100 are tagged with VLAN 100.

In an MLAG configuration, the peer link interface that connects the two switches in the MLAG pair has a VLAN subinterface named 4094 by default if you configured the subinterface with NCLU. The peerlink.4094 subinterface only receives traffic tagged for VLAN 4094.

ifup and Upper (Parent) Interfaces

When you run `ifup` on a logical interface (like a bridge, bond or VLAN interface), if the `ifup` results in the creation of the logical interface, it implicitly tries to execute on the interface's upper (or parent) interfaces as well.

Consider this example configuration:

```
auto br100
iface br100
    bridge-ports bond1.100 bond2.100

auto bond1
iface bond1
    bond-slaves swp1 swp2
```

If you run `ifdown bond1`, `ifdown` deletes bond1 and the VLAN interface on bond1 (bond1.100); it also removes bond1 from the bridge br100. Next, when you run `ifup bond1`, it creates bond1 and the VLAN interface on bond1

(bond1.100); it also executes `ifup br100` to add the bond VLAN interface (bond1.100) to the bridge br100.

There can be cases where an upper interface (like br100) is not in the right state, which can result in warnings. The warnings are mostly harmless.

If you want to disable these warnings, you can disable the implicit upper interface handling by setting `skip_upperifaces=1` in the `/etc/network/ifupdown2/ifupdown2.conf` file.

With `skip_upperifaces=1`, you have to explicitly execute `ifup` on the upper interfaces. In this case, you will have to run `ifup br100` after an `ifup bond1` to add bond1 back to bridge br100.

> **NOTE**
>
> Although specifying a subinterface like swp1.100 and then running `ifup swp1.100` results in the automatic creation of the swp1 interface in the kernel, consider also specifying the parent interface swp1. A parent interface is one where any physical layer configuration can reside, such as `link-speed 1000` or `link-duplex full`. If you only create swp1.100 and not swp1, then you cannot run `ifup swp1` because you did not specify it.

Configure IP Addresses

To configure IP addresses, run the following commands.

NCLU Commands | Linux Commands

The following commands configure three IP addresses for swp1: two IPv4 addresses, and one IPv6 address.

```
cumulus@switch:~$ net add interface swp1 ip address
12.0.0.1/30
cumulus@switch:~$ net add interface swp1 ip address
12.0.0.2/30
cumulus@switch:~$ net add interface swp1 ipv6 address
2001:DB8::1/126
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following code snippet in the `/etc/network/interfaces` file:

```
auto swp1
iface swp1
    address 12.0.0.1/30
    address 12.0.0.2/30
    address 2001:DB8::1/126
```

NOTE

You can specify both IPv4 and IPv6 addresses for the same interface.

For IPv6 addresses, you can create or modify the IP address

To show the assigned IP address on an interface, run the `ip addr show` command. The following example command shows the assigned IP address on swp1.

```
cumulus@switch:~$ ip addr show dev swp1
3: swp1: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc
pfifo_fast state UP qlen 500
    link/ether 44:38:39:00:03:c1 brd ff:ff:ff:ff:ff:ff
    inet 192.0.2.1/30 scope global swp1
    inet 192.0.2.2/30 scope global swp1
    inet6 2001:DB8::1/126 scope global tentative
        valid_lft forever preferred_lft forever
```

Specify IP Address Scope

`ifupdown2` does not honor the configured IP address scope setting in the `/etc/network/interfaces` file, treating all addresses as global. It does not report an error. Consider this example configuration:

```
auto swp2
iface swp2
    address 35.21.30.5/30
    address 3101:21:20::31/80
    scope link
```

When you run `ifreload -a` on this configuration, `ifupdown2` considers all IP addresses as global.

```
cumulus@switch:~$ ip addr show swp2
5: swp2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
pfifo_fast state UP group default qlen 1000
link/ether 74:e6:e2:f5:62:82 brd ff:ff:ff:ff:ff:ff
inet 35.21.30.5/30 scope global swp2
valid_lft forever preferred_lft forever
inet6 3101:21:20::31/80 scope global
valid_lft forever preferred_lft forever
inet6 fe80::76e6:e2ff:fef5:6282/64 scope link
valid_lft forever preferred_lft forever
```

To work around this issue, configure the IP address scope:

NCLU Commands | Linux Commands

Run the following commands:

```
cumulus@switch:~$ net add interface swp6 post-up ip address
add 71.21.21.20/32 dev swp6 scope site
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following code snippet in the `/etc/network/interfaces` file:

```
auto swp6
iface swp6
    post-up ip address add 71.21.21.20/32 dev swp6 scope
site
```

The following configuration shows the correct scope:

```
cumulus@switch:~$ ip addr show swp6
9: swp6: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
```

```
pfifo_fast state UP group default qlen 1000
link/ether 74:e6:e2:f5:62:86 brd ff:ff:ff:ff:ff:ff
inet 71.21.21.20/32 scope site swp6
valid_lft forever preferred_lft forever
inet6 fe80::76e6:e2ff:fef5:6286/64 scope link
valid_lft forever preferred_lft forever
```

Purge Existing IP Addresses on an Interface

By default, `ifupdown2` purges existing IP addresses on an interface. If you have other processes that manage IP addresses for an interface, you can disable this feature.

NCLU Commands | Linux Commands

To disable IP address purge on an interface, run the following commands:

```
cumulus@switch:~$ net add interface swp1 address-purge no
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following configuration snippet in the `/etc/network/interfaces` file:

```
auto swp1
iface swp1
    address-purge no
```

NOTE

Purging existing addresses on interfaces with multiple `iface` stanzas is not supported. Doing so can result in the configuration of multiple addresses for an interface after you change an interface

address and reload the configuration with `ifreload -a`. If this happens, you must shut down and restart the interface with `ifup` and `ifdown`, or manually delete superfluous addresses with `ip address delete specify.ip.address.here/mask dev DEVICE`. See also the Considerations section below for cautions about using multiple `iface` stanzas for the same interface.

Specify User Commands

You can specify additional user commands in the `/etc/network/interfaces` file. The interface stanzas in `/etc/network/interfaces` can have a command that runs at pre-up, up, post-up, pre-down, down, and post-down:

NCLU Commands | Linux Commands

To add a command to an interface stanza, run the following commands:

```
cumulus@switch:~$ net add interface swp1 post-up /sbin/foo
bar
cumulus@switch:~$ net add interface ip address 12.0.0.1/30
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following configuration in the `/etc/network/interfaces` file:

```
auto swp1
iface swp1
    address 12.0.0.1/30
    post-up /sbin/foo bar
```

WARNING

If your `post-up` command also starts, restarts, or reloads any `systemd` service, you must use the `--no-block` option with `systemctl`. Otherwise, that service or even the switch itself might hang after starting or restarting. For example, to restart the `dhcrelay` service after bringing up VLAN 100, first run:

You can add any valid command in the sequence to bring an interface up or down; however, limit the scope to network-related commands associated with the particular interface. For example, it does not make sense to install a Debian package on `ifup` of swp1, even though it is technically possible. See `man interfaces` for more details.

Source Interface File Snippets

Sourcing interface files helps organize and manage the `interfaces` file. For example:

```
cumulus@switch:~$ sudo cat /etc/network/interfaces
# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet dhcp

source /etc/network/interfaces.d/bond0
```

The contents of the sourced file used above are:

```
cumulus@switch:~$ sudo cat /etc/network/interfaces.d/bond0
auto bond0
iface bond0
    address 14.0.0.9/30
    address 2001:ded:beef:2::1/64
    bond-slaves swp25 swp26
```

Use Globs for Port Lists

Globs define a range of ports.

NCLU Commands | Linux Commands

NCLU supports globs to define port lists (a range of ports). You must use commas to separate different ranges of ports in the NCLU command; for example:

```
cumulus@switch:~$ net add bridge bridge ports swp1-4,6,10-12
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands produce the following snippet in the `/etc/network/interfaces` file. The file renders the list of ports individually.

```
...

auto bridge
iface bridge
    bridge-ports swp1 swp2 swp3 swp4 swp6 swp10 swp11 swp12
    bridge-vlan-aware yes
auto swp1
iface swp1

auto swp2
iface swp2

auto swp3
iface swp3

auto swp4
iface swp4
```

Mako Templates

`ifupdown2` supports Mako-style templates. The Mako template engine is run over the `interfaces` file before parsing.

> **WARNING**
>
> While `ifupdown2` supports Mako templates, NCLU does not understand them. As a result, NCLU cannot read or write to the `/etc/network/interfaces` file.

Use the template to declare cookie-cutter bridges in the `interfaces` file:

And use it to declare addresses in the `interfaces` file:

```
%for i in [1,12]:
auto swp${i}
iface swp${i}
    address 10.20.${i}.3/24
```

> **NOTE**
>
> In Mako syntax, use square brackets (`[1,12]`) to specify a list of individual numbers (in this case, 1 and 12). Use `range(1,12)` to specify a range of interfaces.

> **TIP**
>
> You can test your template and confirm it evaluates correctly by running `mako-render /etc/network/interfaces`.

To comment out content in Mako templates, use double hash marks (##). For example:

```
## % for i in range(1, 4):
## auto swp${i}
## iface swp${i}
## % endfor
##
```

For more examples of configuring Mako templates, read this knowledge base article.

Run ifupdown Scripts under /etc/network/ with ifupdown2

Unlike the traditional `ifupdown` system, `ifupdown2` does not run scripts installed in `/etc/network/*/` automatically to configure network interfaces.

To enable or disable `ifupdown2` scripting, edit the `addon_scripts_support` line in the `/etc/network/ifupdown2/ifupdown2.conf` file. `1` enables scripting and `2` disables scripting. The following example enables scripting.

```
cumulus@switch:~$ sudo nano /etc/network/ifupdown2/
ifupdown2.conf
# Support executing of ifupdown style scripts.
# Note that by default python addon modules override scripts
with the same name
addon_scripts_support=1
```

`ifupdown2` sets the following environment variables when executing commands:

- `$IFACE` represents the physical name of the interface being processed; for example, `br0` or vxlan42. The name is obtained from the `/etc/network/interfaces` file.

- `$LOGICAL` represents the logical name (configuration name) of the interface being processed.
- `$METHOD` represents the address method; for example, loopback, DHCP, DHCP6, manual, static, and so on.
- `$ADDRFAM` represents the address families associated with the interface, formatted in a comma-separated list for example, `"inet,inet6"`.

Add Descriptions to Interfaces

You can add descriptions to interfaces configured in the `/etc/network/interfaces` file by using the *alias* keyword.

NCLU Commands | Linux Commands

The following commands create an alias for swp1:

```
cumulus@switch:~$ net add interface swp1 alias
hypervisor_port_1
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following code snippet:

```
auto swp1
iface swp1
    alias hypervisor_port_1
```

You can query the interface description.

NCLU Commands | Linux Commands

To show the description (alias) for an interface, run the `net show interface <interface>` command. The following example command shows the description for swp1:

```
cumulus@switch$ net show interface swp1
    Name    MAC                Speed     MTU   Mode
--  ----    -----------------  -------   -----  ---------
UP  swp1    44:38:39:00:00:04  1G        1500   Access/L2
Alias
-----
hypervisor_port_1
```

To show the interface description (alias) for all interfaces on the switch, run the `net show interface alias` command. For example:

```
cumulus@switch:~$ net show interface alias
State    Name            Mode              Alias
-----    -------------   -------------
------------------
UP       bond01          LACP
UP       bond02          LACP
UP       bridge          Bridge/L2
UP       eth0            Mgmt
UP       lo              Loopback          loopback
interface
UP       mgmt            Interface/L3
UP       peerlink        LACP
```

Interface descriptions also appear in the SNMP OID IF-MIB::ifAlias.

NOTE

- Aliases are limited to 256 characters.
- Avoid using apostrophes or non-ASCII characters in the alias string. Cumulus Linux does not parse these characters.

Considerations

Even though `ifupdown2` supports the inclusion of multiple `iface` stanzas for the same interface, consider using a single `iface` stanza for each interface. If you must specify more than one `iface` stanza; for example, if the configuration for a single interface comes from many places, like a template or a sourced file, make sure the stanzas do not specify the same interface attributes. Otherwise, unexpected behavior can result.

In the following example, swp1 is configured in two places: the `/etc/network/interfaces` file and the `/etc/network/interfaces.d/speed_settings` file. `ifupdown2` correctly parses this configuration because the same attributes are not specified in multiple `iface` stanzas.

```
cumulus@switch:~$ sudo cat /etc/network/interfaces
```

```
source /etc/network/interfaces.d/speed_settings

auto swp1
iface swp1
  address 10.0.14.2/24

cumulus@switch:~$ cat /etc/network/interfaces.d/speed_settings

auto swp1
iface swp1
  link-speed 1000
  link-duplex full
```

> **NOTE**
>
> You cannot purge existing addresses on interfaces with multiple `iface` stanzas.

ifupdown2 and sysctl

For `sysctl` commands in the `pre-up`, `up`, `post-up`, `pre-down`, `down`, and `post-down` lines that use the `$IFACE` variable, if the interface name contains a dot (.), `ifupdown2` does not change the name to work with `sysctl`. For example,

the interface name `bridge.1` is not converted to `bridge/1`.

ifupdown2 and the gateway Parameter

The default route created by the `gateway` parameter in ifupdown2 is not installed in FRRouting, therefore cannot be redistributed into other routing protocols. Define a static default route instead, which is installed in FRR and redistributed, if needed.

The following shows an example of the `/etc/network/interfaces` file when you use a static route instead of a gateway parameter:

```
auto swp2
iface swp2
address 172.16.3.3/24
up ip route add default via 172.16.3.2
```

Interface Name Limitations

Interface names are limited to 15 characters in length, the first character cannot be a number and the name cannot include a dash (-). In addition, any name that matches with the regular expression `.{0,13}\-v.*` is not supported.

If you encounter issues, remove the interface name from the `/etc/network/interfaces` file, then restart the `networking.service`.

```
cumulus@switch:~$ sudo nano /etc/network/interfaces
cumulus@switch:~$ sudo systemctl restart networking.service
```

Related Information

- Debian - Network Configuration
- Linux Foundation - Bonds
- Linux Foundation - VLANs
- man ifdown(8)
- man ifquery(8)
- man ifreload
- man ifup(8)
- man ifupdown-addons-interfaces(5)
- man interfaces(5)

Switch Port Attributes

Cumulus Linux exposes network interfaces for several types of physical and logical devices:

- `lo` is the network loopback device
- `ethN` are switch management ports (for out of band management only)
- `swpN` are switch front panel ports
- (optional) `brN` are bridges (IEEE 802.1Q VLANs)
- (optional) `bondN` are bonds (IEEE 802.3ad link aggregation trunks, or port channels)

Each physical network interface (port) has a number of configurable settings:

- Auto-negotiation
- Duplex Mode
- Link speed
- MTU (maximum transmission unit)
- FEC (forward error correction)

Most of these settings are configured automatically for you, depending upon your switch ASIC; however, you must always set MTU manually.

For **Spectrum ASICs**, MTU is the only port attribute you can directly configure. The Spectrum firmware configures FEC, link speed, duplex mode and auto-negotiation automatically, following a predefined list of parameter settings until the link comes up. However, you can disable FEC if necessary,

which forces the firmware to not try any FEC options.

For **Broadcom-based switches**, consider enabling auto-negotiation on each port. When enabled, Cumulus Linux automatically configures the best link parameter settings based on the module type (speed, duplex, auto-negotiation, and FEC, where supported).

This topic describes the auto-negotiation, link speed, duplex mode, MTU, and FEC settings and provides a table showing the default configuration for various port and cable types. Breakout port configuration, logical switch port limitations, and troubleshooting is also provided.

Auto-negotiation

By default on a Broadcom-based switch, auto-negotiation is disabled - except on 10G and 1000BASE-T fixed copper switch ports, where it is required for links to work. For RJ-45 SFP adapters, you need to manually configure the desired link speed and auto-negotiation as described in the default settings table below.

If you disable auto-negotiation later or never enable it, then you have to configure any settings that deviate from the port default - such as duplex mode, FEC, and link speed settings.

Some module types support auto-negotiation while others do not. To enable a simpler configuration, Cumulus Linux allows you to configure auto-negotiation on all port types on Broadcom switches; the port configuration software then configures the underlying hardware according to its capabilities.

If you do decide to disable auto-negotiation, be aware of the following:

- You must manually set any non-default link speed, duplex, pause, and FEC.
- Disabling auto-negotiation on a 1G optical cable prevents detection of single fiber breaks.
- You cannot disable auto-negotiation on 1GT or 10GT fixed copper switch ports.

For 1000BASE-T RJ-45 SFP adapters, auto-negotiation is automatically done on the SFP PHY, so enabling auto-negotiation on the port settings is not required. You must manually configure these ports using the settings below.

Depending upon the connector used for a port, enabling auto-negotiation

also enables forward error correction (FEC), if the cable requires it (see the table below). The correct FEC mode is set based on the speed of the cable when auto-negotiation is enabled.

To configure auto-negotiation for a switch:

NCLU Commands | Linux Commands

Run the `net add interface <interface> link autoneg` command. The following example commands enable auto-negotiation for the swp1 interface:

```
cumulus@switch:~$ net add interface swp1 link autoneg on
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

NOTE

Any time you enable auto-negotiation, Cumulus Linux restores the default configuration settings specified in the table below.

Port Speed and Duplex Mode

Cumulus Linux supports both half- and full-duplex configurations. Half-

duplex is supported only with speeds of less than 1G.

Supported port speeds include 100M, 1G, 10G, 25G, 40G, 50G and 100G. In Cumulus Linux, you set the speed on a Broadcom switch in Mbps, where the setting for 1G is *1000*, 40G is *40000*, and 100G is *100000*.

You can configure ports to the following speeds (unless there are restrictions in the `/etc/cumulus/ports.conf` file of a particular platform).

Switch Port Type	Other Configurable Speeds
1G	100 Mb
10G	1 Gigabit (1000 Mb)
40G	4x10G (10G lanes) creates four 1-lane ports each running at 10G
100G	50G or 2x50G (25G lanes) - 50G creates one 2-lane port running at 25G and 2x50G creates two 2-lane ports each running at 25G 40G (10G lanes) creates one 4-lane port running at 40G 4x25G (25G lanes) creates four 1-lane ports each running at 25G 4x10G (10G lanes) creates four 1-lane ports each running at 10G

NOTE

Platform Limitations

- On Lenovo NE2572O switches, swp1 through swp8 only support 25G speed.
- For 10G and 1G SFPs inserted in a 25G port on a Broadcom switch, you must edit the `/etc/cumulus/ports.conf` file and configure the four ports in the same core to be 10G. See Considerations below.
- A switch with the Maverick ASIC limits multicast traffic by the lowest speed port that has joined a particular group. For example, if you are sending 100G multicast through and subscribe with one 100G and one 25G port, traffic on both egress ports is limited to 25Gbps. If you remove the 25G port from the group, traffic correctly forwards at 100Gbps.

To configure the port speed and duplex mode:

NCLU Commands | Linux Commands

Run the `net add interface <interface> link speed` command. The following commands configure the port speed for the swp1 interface. The duplex mode setting defaults to *full*. You only need to specify `link duplex` if you want to set half-duplex mode.

```
cumulus@switch:~$ net add interface swp1 link speed 10000
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The above commands create the following `/etc/network/interfaces` file code snippet:

```
auto swp1
iface swp1
    link-speed 10000
```

The following commands configure the port speed and set half-duplex mode for the swp31 interface.

```
cumulus@switch:~$ net add interface swp31 link speed 100
cumulus@switch:~$ net add interface swp31 link duplex half
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```


The above commands create the following `/etc/network/interfaces`

MTU

Interface MTU applies to traffic traversing the management port, front panel or switch ports, bridge, VLAN subinterfaces, and bonds (both physical and logical interfaces). MTU is the only interface setting that you must set manually.

In Cumulus Linux, `ifupdown2` assigns 9216 as the default MTU setting. On a Mellanox switch, the initial MTU value set by the driver is 9238. After you configure the interface, the default MTU setting is 9216.

To change the MTU setting, run the following commands:

NCLU Commands | Linux Commands

Run the `net add interface <interface> mtu` command. The following example command sets the MTU to 1500 for the swp1 interface.

```
cumulus@switch:~$ net add interface swp1 mtu 1500
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following code snippet:

```
auto swp1
iface swp1
    mtu 1500
```

NOTE

Some switches might not support the same maximum MTU setting in hardware for both the management interface (eth0) and the data plane ports.

Set a Policy for Global System MTU

For a global policy to set MTU, create a policy document (called `mtu.json`). For example:

```
cumulus@switch:~$ sudo cat /etc/network/ifupdown2/policy.d/
mtu.json
{
  "address": {"defaults": { "mtu": "9216" }
             }
}
```

WARNING

The policies and attributes in any file in `/etc/network/ifupdown2/policy.d/` override the default policies and attributes in `/var/lib/ifupdown2/policy.d/`.

MTU for a Bridge

The MTU setting is the lowest MTU of any interface that is a member of the bridge (every interface specified in `bridge-ports` in the bridge configuration of the `/etc/network/interfaces` file). There is **no** need to specify an MTU on

the bridge. Consider this bridge configuration:

```
auto bridge
iface bridge
    bridge-ports bond1 bond2 bond3 bond4 peer5
    bridge-vids 100-110
    bridge-vlan-aware yes
```

For *bridge* to have an MTU of 9000, set the MTU for each of the member interfaces (bond1 to bond 4, and peer5), to 9000 at minimum.

TIP

Use MTU 9216 for a bridge

Two common MTUs for jumbo frames are 9216 (the default value) and 9000 bytes. The corresponding MTUs for the VNIs are 9166 and 8950.

When configuring MTU for a bond, configure the MTU value directly under the bond interface; the configured value is inherited by member links/slave interfaces. If you need a different MTU on the bond, set it on the bond interface, as this ensures the slave interfaces pick it up. There is no need to

specify MTU on the slave interfaces.

VLAN interfaces inherit their MTU settings from their physical devices or their lower interface; for example, swp1.100 inherits its MTU setting from swp1. Therefore, specifying an MTU on swp1 ensures that swp1.100 inherits the MTU setting for swp1.

If you are working with VXLANs, the MTU for a virtual network interface (VNI must be 50 bytes smaller than the MTU of the physical interfaces on the switch, as those 50 bytes are required for various headers and other data. Also, consider setting the MTU much higher than 1500.

> **NOTE**
>
> The MTU for an SVI interface, such as vlan100, is derived from the bridge. When you use NCLU to change the MTU for an SVI and the MTU setting is higher than it is for the other bridge member interfaces, the MTU for all bridge member interfaces changes to the new setting. If you need to use a mixed MTU configuration for SVIs, (if some SVIs have a higher MTU and some lower), set the MTU for all member interfaces to the maximum value, then set the MTU on the specific SVIs that need to run at a lower MTU.

To show the MTU setting for an interface:

NCLU Commands | Linux Commands

Run the `net show interface <interface>` command:

```
cumulus@switch:~$ net show interface swp1
    Name    MAC                Speed      MTU  Mode
--  ------  -----------------  -------  -----  ----------
UP  swp1    44:38:39:00:00:04  1G         9216  Access/L2
```

FEC

Forward Error Correction (FEC) is an encoding and decoding layer that enables the switch to detect and correct bit errors introduced over the cable between two interfaces. The target IEEE bit error rate (BER) on high speed ethernet link is 10^{-12}. Because 25G transmission speeds can introduce a higher than acceptable BER on a link, FEC is often required to correct errors to achieve the target BER at 25G, 4x25G, 100G, and higher link speeds. The type and grade of a cable or module and the medium of transmission will determine which FEC setting is needed.

For the link to come up, the two interfaces on each end must use the same FEC setting.

> **NOTE**
>
> There is a very small latency overhead required for FEC. For most applications, this small amount of latency is preferable to error packet retransmission latency.

There are two FEC types:

- Reed Solomon (**RS**), IEEE 802.3 Clause 108 (CL108) on individual 25G channels and Clause 91 on 100G (4channels). This is the highest FEC algorithm, providing the best bit-error correction.
- Base-R (**BaseR**), Fire Code (FC), IEEE 802.3 Clause 74 (CL74). Base-R provides less protection from bit errors than RS FEC but adds less latency.

Cumulus Linux includes additional FEC options:

- *Auto* FEC instructs the hardware to select the best FEC. For copper DAC, FEC can be negotiated with the remote end. However, optical modules do not have auto-negotiation capability; if the device chooses a preferred mode, it might not match the remote end. This is the current default on a Spectrum switch.
- *No* FEC (no error correction is done). This is the current default on a Broadcom switch.

> **NOTE**
>
> The Trident II switch does not support FEC.
>
> The Tomahawk switch does not support RS FEC or auto-negotiation of FEC on 25G lanes that are broken out (Tomahawk pre-dates 802.3by). If you are using a 4x25G breakout DAC or AOC on a Tomahawk switch, you can configure either Base-R FEC or no FEC, and choose cables appropriate for that limitation (CA-25G-S, CA-25G-N or fiber). Tomahawk+, Tomahawk2, Trident3 and Maverick switches do not have this limitation.

For **25G DAC, 4x25G Breakouts DAC and 100G DAC cables**, the IEEE 802.3by specification creates 3 classes:

- CA-25G-L (Long cable) - Requires RS FEC - Achievable cable length of at least 5m. dB loss less or equal to 22.48. Expected BER of 10^{-5} or better without RS FEC enabled.
- CA-25G-S (Short cable) - Requires Base-R FEC - Achievable cable length of at least 3m. dB loss less or equal to 16.48. Expected BER of 10^{-8} or better without Base-R FEC enabled.
- CA-25G-N (No FEC) - Does not require FEC - Achievable cable length of at least 3m. dB loss less or equal to 12.98. Expected BER 10^{-12} or better with no FEC enabled.

The IEEE classification is based on various dB loss measurements and minimum achievable cable length. You can build longer and shorter cables

if they comply to the dB loss and BER requirements.

If a cable is manufactured to CA-25G-S classification and FEC is not enabled, the BER might be unacceptable in a production network. It is important to set the FEC according to the cable class (or better) to have acceptable bit error rates. See Determining Cable Class below.

You can check bit errors using `cl-netstat` (`RX_ERR` column) or `ethtool -S` (`HwIfInErrors` counter) after a large amount of traffic has passed through the link. A non-zero value indicates bit errors. Expect error packets to be zero or extremely low compared to good packets. If a cable has an unacceptable rate of errors with FEC enabled, replace the cable.

For **25G, 4x25G Breakout, and 100G Fiber modules and AOCs**, there is no classification of 25G cable types for dB loss, BER or length. FEC is recommended but might not be required if the BER is low enough.

Determine Cable Class of 100G and 25G DACs

You can determine the cable class for 100G and 25G DACs from the Extended Specification Compliance Code field (SFP28: 0Ah, byte 35, QSFP28: Page 0, byte 192) in the cable EEPROM programming.

For 100G DACs, most manufacturers use the 0x0Bh *100GBASE-CR4 or 25GBASE-CR CA-L* value (the 100G DAC specification predates the IEEE 802.3by 25G DAC specification). RS FEC is the expected setting for 100G DAC but might not be required with shorter or better cables.

> **NOTE**
>
> A manufacturer's EEPROM setting might not match the dB loss on a cable or the actual bit error rates that a particular cable introduces. Use the designation as a guide, but set FEC according to the bit error rate tolerance in the design criteria for the network. For most applications, the highest mutual FEC ability of both end devices is the best choice.

You can determine for which grade the manufacturer has designated the cable as follows.

For the **SFP28 DAC**, run the following command:

```
cumulus@switch:~$ sudo ethtool -m swp35 hex on | grep 0020 |
awk '{ print $6}'
0c
```

The values at location 0x0024 are:

- 0x0b : CA-L (long cable - RS FEC required)
- 0x0c : CA-S (short cable - Base-R or better FEC required)
- 0x0d : CA-N (no FEC required)

For the **QSFP28 DAC**, run the following command:

```
cumulus@switch:~$ sudo ethtool -m swp51s0 hex on | grep 00c0 |
awk '{print $2}'
0b
```

The values at 0x00c0 are:

- 0x0b : CA-L (long cable - RS FEC required) or 100G CR4
- 0x0c : CA-S (short cable - Base-R or better FEC required)
- 0x0d : CA-N (no FEC required)

In each example below, the *Compliance* field is derived using the method described above and is not visible in the `ethool -m` output.

```
3meter cable that does not require FEC
(CA-N)
Cost: More expensive
Cable size: 26AWG (Note that AWG does not necessarily
correspond to overall dB loss or BER performance)
Compliance Code: 25GBASE-CR CA-N

3meter cable that requires Base-R FEC
(CA-S)
```

```
Cost: Less expensive
Cable size: 26AWG
Compliance Code: 25GBASE-CR CA-S
```

When in doubt, consult the manufacturer directly to determine the cable classification.

Spectrum ASIC FEC Behavior

The firmware in a Spectrum ASIC applies an FEC configuration to 25G and 100G cables based on the cable type and whether the peer switch also has a Spectrum ASIC.

When the link is between two switches with Spectrum ASICs:

- For 25G optical modules, the Spectrum ASIC firmware chooses Base-R/FC-FEC.
- For 25G DAC cables with attenuation less or equal to 16db, the firmware chooses Base-R/FC-FEC.
- For 25G DAC cables with attenuation higher than 16db, the firmware chooses RS-FEC.
- For 100G cables/modules, the firmware chooses RS-FEC.

Cable Type	FEC Mode
25G optical cables	Base-R/FC-FEC

Cable Type	FEC Mode
25G 1,2 meters: CA-N, loss <13db	Base-R/FC-FEC
25G 2.5,3 meters: CA-S, loss <16db	Base-R/FC-FEC
25G 2.5,3,4,5 meters: CA-L, loss > 16db	RS-FEC
100G DAC or optical	RS-FEC

When linking to a non-Spectrum peer, the firmware lets the peer decide.

The Spectrum ASIC supports RS-FEC (for both 100G and 25G), Base-R/FC-FEC (25G only), or no-FEC (for both 100G and 25G).

Cable Type	FEC Mode
25G pptical cables	Let peer decide
25G 1,2 meters: CA-N, loss <13db	Let peer decide
25G 2.5,3 meters: CA-S, loss <16db	Let peer decide
25G 2.5,3,4,5 meters: CA-L, loss > 16db	Let peer decide
100G	Let peer decide: RS-FEC or No FEC

How Does Cumulus Linux use FEC?

This depends upon the make of the switch you are using.

A Spectrum switch enables FEC automatically when it powers up; that is, the setting is `fec auto`. The port firmware tests and determines the correct FEC mode to bring the link up with the neighbor. It is possible to get a link up to a Spectrum switch without enabling FEC on the remote device as the switch eventually finds a working combination to the neighbor without FEC.

On a Broadcom switch, Cumulus Linux does not enable FEC by default; that is, the setting is `fec off`. Consider configuring FEC explicitly to match the configured FEC on the link neighbor. On 100G DACs, you can configure `link-autoneg` so that the port attempts to negotiate FEC settings with the remote peer.

The following sections describe how to show the current FEC mode, and to enable and disable FEC.

Show the Current FEC Mode

Cumulus Linux returns different output for the `ethtool --show-fec` command, depending upon whether you are using a Broadcom or Mellanox Spectrum switch.

On a Broadcom switch, the `--show-fec` output tells you exactly what you configured, even if the link is down due to a FEC mismatch with the neighbor.

On a Spectrum switch, the `--show-fec` output tells you the current active state of FEC **only if the link is up**; that is, if the FEC modes matches that of the neighbor. If the link is not up, the value displays *None*, which is not valid.

To show the FEC mode currently enabled on a given switch port, run the `ethtool --show-fec <interface>` command.

```
cumulus@switch:~$ sudo ethtool --show-fec swp23
FEC parameters for swp23:
Configured FEC encodings: Auto
Active FEC encoding: Off
```

Enable or Disable FEC

To enable **Reed Solomon (RS) FEC** on a link:

NCLU Commands | Linux Commands

Run the `net add interface <interface> link fec rs` command. For example:

```
cumulus@switch:~$ sudo net add interface swp23 link fec rs
cumulus@switch:~$ sudo net pending
cumulus@switch:~$ sudo net commit
```

To enable **Base-R/FireCode FEC** on a link:

NCLU Commands | Linux Commands

Run the `net add interface <interface> link fec baser` command.

For example:

```
cumulus@switch:~$ sudo net add interface swp23 link fec
baser
cumulus@switch:~$ sudo net pending
cumulus@switch:~$ sudo net commit
```

To enable FEC with Auto-negotiation:

NOTE

FEC with auto-negotiation is supported on DACs only.

NCLU Commands | Linux Commands

Run the `net add interface <interface> link autoneg on` command. The following example command enables FEC with auto-negotiation on the swp12 interface:

```
cumulus@switch:~$ sudo net add interface swp12 link autoneg
on
cumulus@switch:~$ sudo net pending
cumulus@switch:~$ sudo net commit
```

To show the FEC and auto-negotiation settings for an interface, run the following command:

```
cumulus@switch:~$ sudo ethtool swp12 | egrep 'FEC|auto'
Supports auto-negotiation: Yes
Supported FEC modes: RS
Advertised auto-negotiation: Yes
Advertised FEC modes: RS
Link partner advertised auto-negotiation: Yes
Link partner advertised FEC modes: Not reported
```

To disable FEC on a link:

NCLU Commands | Linux Commands

Run the `net add interface <interface> link fec off` command. For example:

```
cumulus@switch:~$ sudo net add interface swp23 link fec off
cumulus@switch:~$ sudo net pending
cumulus@switch:~$ sudo net commit
```

Interface Configuration Recommendations for Broadcom Platforms

The recommended configuration for each type of interface is described in the following table. These are the link settings that are applied to the port hardware when auto-negotiation is enabled on a Broadcom-based switch. If further troubleshooting is required to bring a link up, use the table below as a guide to set the link parameters.

Except as noted below, the settings for both sides of the link are expected to be the same.

ⓘ NOTE

Spectrum switches automatically configure these settings following a predefined list of parameter settings until the link comes up.

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

100BASE-T (RJ-45 SFP adapter)	Off	N/A	**NCLU commands**
			`$ net add interface swp1 link speed 100`
			`$ net add interface swp1 link autoneg`
				The module has two sets of electronics: the port side, which communicates with the switch ASIC and the RJ-45 adapter side.
				
				Auto-negotiation is always used on the RJ-45

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

			`off`
			
			Configuration in /etc/network/interfaces
			
			`auto swp1`
			`iface swp1`
			`  link-autoneg off`
			`  link-speed 100`
				adapter side of the link by the PHY built into the module. This is independent of the switch setting. Set auto-negotiation to off.
				
				Auto-negotiation must be enabled on the server side in this scenario.

100BASE-T on a 1G fixed copper	On	N/A	**NCLU commands**	10M or 100M speeds are

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

port			`$ net add interface swp1 link speed 100`
			`$net add interface swp1 link autoneg on`
			
			Configuration in /etc/network/interfaces
			
			`auto swp1`
			`iface`
				possible with auto-negotiation off on both sides.
				
				Testing on an Edgecore AS4610-54P showed the ASIC reporting auto-negotiation as on.
				
				Power over Ethernet might require auto-negotiation to be on.

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes
			`swp1 ink-autoneg on link-speed 100`	
1000BASE-T (RJ-45 SFP adapter)	Off	N/A	**NCLU commands** `$ net add interface swp1 link speed 1000 $ net`	The module has two sets of electronics: the port side, which communicates with the switch ASIC and the RJ-45 side. Auto-negotiation

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

			`add interface swp1 link autoneg off`
			
			Configuration in /etc/network/interfaces
			
			`auto swp1`
			`iface swp1`
			`  link-autoneg off`
			`  link-speed 1000`
				is always used on the RJ-45 side of the link by the PHY built into the module. This is independent of the switch setting. Set auto-negotiation to off.
				
				Auto-negotiation must be enabled on the server side.

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

1000BASE-T on a 1G fixed copper port	On	N/A	**NCLU commands**
			`$ net add interface swp1 link speed 1000`
			`$ net add interface swp1 link autoneg on`
			Configuration in /etc/network/interfaces	

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes
			`auto swp1 iface swp1  link-autoneg on  link-speed 1000`	
1000BASE-T on a 10G fixed copper port	On	N/A	**NCLU commands** `$ net add interface swp1 link speed`	

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes
			1000 $ net add interface swp1 link autoneg on **Configuration in /etc/network/interfaces** auto swp1 iface swp1 link- autoneg on	

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes
			`link-speed 1000`	

1000BASE-SX 1000BASE-LX (1G Fiber)	Recommended On	N/A	**NCLU commands**
			`$ net add interface swp1 link speed 1000`
			`$ net add interface swp1 link autoneg on`
				Without auto-negotiation, the link stays up when there is a single fiber break.
				
				See the limitation discussed in 10G and 1G SFPs Inserted in a 25G Port below.

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

			Configuration in /etc/network/interfaces
			`auto swp1`
			`iface swp1`
			`  link-autoneg on`
			`  link-speed 1000`	

10GBASE-T (RJ-45 SFP Module)	Off	N/A	**NCLU commands** `$ net add`	The module has two sets of electronics - the port side, which communicates to the

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

			`interface swp1 link speed 10000`
			`$ net add interface swp1 link autoneg off`
			
			Configuration in /etc/network/interfaces
			
			`auto swp1`
			`iface swp1`
				switch ASIC and the RJ-45 side.
				
				Auto-negotiation is always used on the RJ-45 side of the link by the PHY built into the module. This is independent of the switch setting. Set link-autoneg to off.
				
				Auto-negotiation needs to be enabled on the

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes
			`link-autoneg off` `link-speed 10000`	server side.

10GBASE-T fixed copper port	On	N/A	**NCLU commands**
			`$ net add interface swp1 link speed 10000`
			`$ net add interface`	

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

			`swp1 link autoneg on`
			
			Configuration in /etc/network/interfaces
			
			`auto swp1 iface swp1  link-autoneg on  link-speed 10000`	

10GBASE-	Off	N/A	**NCLU**	

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

CR
10GBASE-LR
10GBASE-SR
10G AOC
			commands
			`$ net add interface swp1 link speed 10000`
			`$ net add interface swp1 link autoneg off`
			Configuration in /etc/network/interfaces
			`auto`	

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes
			`swp1 iface swp1 link-autoneg off link-speed 10000`	
40GBASE-CR4	Recommended On	Disable	**NCLU commands** `$ net add interface swp1 link speed 40000`	40G standards mandate auto-negotiation be enabled for DAC connections.

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

			`$ net add interface swp1 link autoneg on`
			
			Configuration in /etc/network/interfaces
			
			`auto swp1 iface swp1 link-autoneg on link-`	

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes
			`speed 40000`	

40GBASE-SR4
40GBASE-LR4
40G AOC
	Off	Disable	**NCLU commands**
			`$ net add interface swp1 link speed 40000`
			`$ net add interface swp1 link autoneg off`
			Configuration	

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

			in /etc/network/interfaces
			`auto swp1`
			`iface swp1`
			`  link-autoneg off`
			`  link-speed 40000`	

100GBASE-CR4	On	auto-negotiated	**NCLU commands** `$ net add interface`	

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

			`swp1 link speed 100000 $ net add interface swp1 link autoneg on`
			
			Configuration in /etc/network/interfaces
			
			`auto swp1 iface swp1 link-`	

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes
			`autoneg on` `link-speed 100000`	

100GBASE-SR4
100G AOC
	Off	RS	**NCLU commands**
			`$ net add interface swp1 link speed 100000`
			`$ net add interface swp1`	

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

			`link autoneg off`
			`$ net add interface swp1 link fec rs`
			
			Configuration in /etc/network/interfaces
			
			`auto swp1`
			`iface swp1`
			`  link-autoneg`	

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

			`off`
			
			`link-speed 100000`
			
			`link-fec rs`	

100GBASE-LR4	Off	None	**NCLU commands**
			
			`$ net add interface swp1 link speed 100000`
			`$ net`	

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

			`add interface swp1 link autoneg off`
			`$ net add interface swp1 link fec off`
			Configuration in /etc/network/interfaces
			`auto swp1`
			`iface swp1`	

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

			`link-autoneg off`
			`link-speed 100000`
			`link-fec off`	

25GBASE-CR	On	auto-negotiated	**NCLU commands** `$ net add interface swp1 link speed`	Tomahawk predates 802.3by. It does not support RS FEC or auto-negotiation of RS FEC on a 25G port or subport. It does

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes
			25000 $ net add interface swp1 link autoneg on **Configuration in /etc/network/interfaces** auto swp1 iface swp1 link- autoneg on	support Base-R FEC.

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes
			`link-speed 25000`	

25GBASE-SR	Off	RS	**NCLU commands**
			`$ net add interface swp1 link speed 25000`
			`$ net add interface swp1 link autoneg off`
				Tomahawk predates 802.3by and does not support RS FEC on a 25G port or subport; however it does support Base-R FEC. The configuration for Base-R FEC is as follows:
				NCLU commands

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

			`$ net add interface swp1 link fec rs`
			
			Configuration in /etc/network/interfaces
			
			`auto swp1 iface swp1 link-autoneg off link-`
				`$ net add interface swp1 link speed 25000 $ net add interface swp1 link autoneg off $ net add interface swp1 link fec baser`
				
				Configuration in /etc/

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

			NCLU
			`speed 25000`
			`link-fec rs`
			network/interfaces
			`auto swp1`
			`iface swp1`
			`link-autoneg off`
			`link-speed 25000`
			`link-fec baser`
			Cumulus Linux 3.x
			`speed 25000`
			`link-fec rs`
			Cumulus Linux 2.5.x
			`speed 25000`
			`link-fec rs`
				Configure FEC to the setting that the cable requires.

25GBASE-	Off	None	**NCLU**	

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

LR			**commands**
			`$ net add interface swp1 link speed 25000`
			`$ net add interface swp1 link autoneg off`
			`$ net add interface swp1 link fec off`
			Configuration	

Speed	Auto-negotiation	FEC Setting	Manual Configuration Examples	Notes

			in /etc/network/interfaces
			
			`auto swp1`
			`iface swp1`
			
			`link-autoneg off`
			
			`link-speed 25000`
			
			`link-fec off`	

Default Policies for Interface Settings

Instead of configuring settings for each individual interface, you can specify a policy for all interfaces on a switch or tailor custom settings for each

interface. Create a file in `/etc/network/ifupdown2/policy.d/` and populate the settings accordingly. The following example shows a file called `address.json`.

```
cumulus@switch:~$ cat /etc/network/ifupdown2/policy.d/
address.json
{
    "ethtool": {
        "defaults": {
            "link-duplex": "full"
        },
        "iface_defaults": {
            "swp1": {
                "link-autoneg": "on",
                "link-speed": "1000"
          },
            "swp16": {
                "link-autoneg": "off",
                "link-speed": "10000"
            },
            "swp50": {
                "link-autoneg": "off",
                "link-speed": "100000",
                "link-fec": "rs"
            }
```

```
        }
    },
    "address": {
        "defaults": { "mtu": "9000" },
        "iface_defaults": {
            "eth0": {"mtu": "1500"}
        }
    }
}
```

> **NOTE**
>
> Setting the default MTU also applies to the management interface. Be sure to add the *iface_defaults* to override the MTU for eth0, to remain at 9216.

Breakout Ports

Cumulus Linux lets you:

- Break out 100G switch ports into 2x50G, 4x25G, or 4x10G with breakout cables.
- Break out 40G switch ports into four separate 10G ports (4x10G) for use with breakout cables.

- Combine (*aggregate* or *gang*) four 10G switch ports into one 40G port for use with a breakout cable (not to be confused with a bond).

NOTE

- For Broadcom switches with ports that support 100G speeds, you *cannot* have more than 128 logical ports.
- On Mellanox switches with the Spectrum ASIC running in *nonatomic* ACL mode, if you break out a port, then reload the `switchd` service, temporary disruption to traffic occurs while the ACLs are reinstalled.
- Port ganging is not supported on Mellanox switches with the Spectrum ASIC.
- Mellanox switches with the Spectrum 1 ASIC have a limit of 64 logical ports. 64-port Broadcom switches with the Tomahawk2 ASIC have a limit of 128 total logical ports. If you want to break ports out to 4x25G or 4x10G, you must configure the logical ports as follows:
  - You can only break out odd-numbered ports into four logical ports.

    - You must disable the next even-numbered port. For example, if you break out port 11 into four logical ports, you must disable port 12.

  These restrictions do *not* apply to a 2x50G breakout configuration or to the Mellanox SN2100 and SN2010 switches.

- Mellanox switches with the Spectrum 2 and Spectrum 3 ASIC have a limit of 128 logical ports. To ensure that the number of total logical interfaces does not exceed the limit, if you split ports into four interfaces on Spectrum 2 and Spectrum 3 switches with 64 interfaces, you must disable the adjacent port. For example, when splitting port 1 into four 25G interfaces, you must disable port 2 in the `/etc/cumulus/ports.conf` file:

  ```
  1=4x25G
  2=disabled
  ```

  When you split a port into two interfaces, such as 2x50G, you do **not** have to disable the adjacent port.

Valid port configuration and breakout guidance for each platform

is provided in the `/etc/cumulus/ports.conf` file.

Spectrum 2 and Spectrum 3

Mellanox switches with the Spectrum 2 and Spectrum 3 ASICs have a limit of 128 logical ports. To ensure that the number of total logical interfaces does not exceed the limit, Spectrum 2 and Spectrum 3 platforms with 64 interfaces have the following breakout limitation:

When you split ports into four interfaces, you must configure the adjacent port as “disabled” in this file. When splitting a port into two interfaces, such as 2x50G, you do not have to disable the adjacent port. Adjacent ports only need to be disabled when a port is split into four interfaces. For example, when splitting port 1 into four 25G interfaces, port 2 must be configured as “disabled” like this:

1=4x25G 2=disabled

Configure a Breakout Port

To configure a breakout port:

NCLU Commands | Linux Commands

This example command breaks out the 100G port on swp3 into four 25G ports:

```
cumulus@switch:~$ net add interface swp3 breakout 4x
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

To break out swp3 into four 10G ports, run the `net add interface swp3 breakout 4x10G` command.

On Mellanox switches with the Spectrum ASIC and 64-port Broadcom switches, you need to disable the next port. The following example command disables swp4.

```
cumulus@switch:~$ net add interface swp4 breakout disabled
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands break out swp3 into four 25G interfaces in the `/etc/cumulus/ports.conf` file and create four interfaces in the `/etc/network/interfaces` file:

```
cumulus@switch:~$ cat /etc/network/interfaces
...
auto swp3s0
iface swp3s0
```

Remove a Breakout Port

To remove a breakout port:

NCLU Commands | Linux Commands

1. Run the `net del interface <interface>` command. For example:

```
cumulus@switch:~$ net del interface swp3s0
cumulus@switch:~$ net del interface swp3s1
cumulus@switch:~$ net del interface swp3s2
cumulus@switch:~$ net del interface swp3s3
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

2. Manually edit the `/etc/cumulus/ports.conf` file to configure the interface for the original speed. For example:

```
cumulus@switch:~$ sudo nano /etc/cumulus/ports.conf
...

1=100G
2=100G
3=100G
4=100G
...
```

3. On a Broadcom switch, restart `switchd` with the `sudo systemctl restart switchd.service` command. The restart interrupts network services.

```
cumulus@switch:~$ sudo systemctl restart switchd.service
```

Combine Four 10G Ports into One 40G Port

You can *gang* (combine) four 10G ports into one 40G port for use with a breakout cable, provided you follow these requirements:

- You must gang four 10G ports in sequential order. For example, you cannot gang swp1, swp10, swp20 and swp40 together.
- The ports must be in increments of four, with the starting port being swp1 (or swp5, swp9, or so forth); so you cannot gang swp2, swp3, swp4 and swp5 together.

> **NOTE**
>
> - Port ganging is not supported on Mellanox switches with the Spectrum ASIC.
> - The `/etc/cumulus/ports.conf` file varies across different hardware platforms. Check the current list of supported platforms on the hardware compatibility list.

NCLU Commands | Linux Commands

To gang swp1 through swp4 into a 40G port, run the following commands:

```
cumulus@switch:~$ net add int swp1-4 breakout /4
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following configuration snippet in the `/etc/cumulus/ports.conf` file:

```
# SFP+ ports#
# <port label 1-48> = [10G|40G/4]
1=40G/4
2=40G/4
3=40G/4
4=40G/4
5=10G
```

Logical Switch Port Limitations

100G and 40G switches can support a certain number of logical ports,

depending on the manufacturer; these include:

- Mellanox SN2700, SN2700B, SN2410, and SN2410B switches
- Switches with Broadcom Tomahawk, Trident II, Trident II+, and Trident3 chipsets (check the HCL)

Before you configure any logical/unganged ports on a switch, check the limitations listed in `/etc/cumulus/ports.conf`; this file is specific to each manufacturer.

The following example shows the logical port limitation provided in the Dell Z9254F-ON `ports.conf` file. The maximum number of ports for this switch is 128.

```
# ports.conf --
#
#   configure port speed, aggregation, and subdivision.
#
# The Dell Z9264F has:
#      64 QSFP28 ports numbered 1-64
#         These ports are configurable as 100G, 50G, 40G, or
split into
#         2x50G, 4x25G, or 4x10G ports.
#
# NOTE:  You must restart switchd for any changes to take
effect.
```

```
# Only "odd-numbered " port can be split into 4 interfaces and
if an odd-numbered
# port is split in a 4X configuration, the port adjacent to it
(even-numbered port)
# has to be set to "disabled " in this file. When splitting a
port into two
# interfaces, like 2x50G, it is NOT required that the adjacent
port be
# disabled. For example, when splitting port 11 into 4 10G
interfaces, port
# 12 must be configured as "disabled" like this:
#
#   11=4x10G
#   12=disabled

# QSFP28 ports
#
# <port label> = [100G|50G|40G|2x50G|4x25G|4x10G|disabled]
```

Mellanox SN2700 and SN2700B switches have a limit of 64 logical ports in total. However, the logical ports must be configured in a specific way. See the note above.

Verification and Troubleshooting Commands

Statistics

To show high-level interface statistics, run the `net show interface` command:

```
cumulus@switch:~$ net show interface swp1

    Name    MAC                Speed      MTU  Mode
--  ------  -----------------  -------  -----  ---------
UP  swp1    44:38:39:00:00:04  1G        1500  Access/L2

Vlans in disabled State
-------------------------
br0

Counters      TX    RX
----------  ----  ----
errors         0     0
unicast        0     0
broadcast      0     0
multicast      0     0
```

```
LLDP
------  ----  --------------------------
swp1    ====  44:38:39:00:00:03(server01)
```

To show low-level interface statistics, run the following `ethtool` command:

```
cumulus@switch:~$ sudo ethtool -S swp1
NIC statistics:
      HwIfInOctets: 21870
      HwIfInUcastPkts: 0
      HwIfInBcastPkts: 0
      HwIfInMcastPkts: 243
      HwIfOutOctets: 1148217
      HwIfOutUcastPkts: 0
      HwIfOutMcastPkts: 11353
      HwIfOutBcastPkts: 0
      HwIfInDiscards: 0
      HwIfInL3Drops: 0
      HwIfInBufferDrops: 0
      HwIfInAclDrops: 0
      HwIfInBlackholeDrops: 0
      HwIfInDot3LengthErrors: 0
      HwIfInErrors: 0
```

```
      SoftInErrors: 0
      HwIfOutErrors: 0
      HwIfOutQDrops: 0
      HwIfOutNonQDrops: 0
      SoftOutErrors: 0
      SoftOutDrops: 0
      SoftOutTxFifoFull: 0
      HwIfOutQLen: 0
```

Query SFP Port Information

To verify SFP settings, run the `ethtool -m` command. The following example shows the vendor, type and power output for the swp4 interface.

```
cumulus@switch:~$ sudo ethtool -m swp4 | egrep
'Vendor|type|power\s+:'
        Transceiver type                        : 10G
Ethernet: 10G Base-LR
        Vendor name                             : FINISAR
CORP.
        Vendor OUI                              : 00:90:65
        Vendor PN                               : FTLX2071D327
        Vendor rev                              : A
```

```
        Vendor SN                                 : UY30DTX
        Laser output power                        : 0.5230 mW /
-2.81 dBm
        Receiver signal average optical power     : 0.7285 mW /
-1.38 dBm
```

Considerations

Port Speed and the ifreload -a Command

When configuring port speed or break outs in the `/etc/cumulus/ports.conf` file, you need to run the `ifreload -a` command to reload the configuration after restarting `switchd` in the following cases:

- If you configure, or configure then remove, the port speed in the `/etc/cumulus/ports.conf` file and you also set or remove the speed on the same physical port or breakouts of that port in the `/etc/network/interfaces` file since the last time you restarted `switchd`.
- If you break out a switch port or remove a break out port and the port speed is set in both the `/etc/cumulus/ports.conf` file and the `/etc/network/interfaces` file.

Port Speed Configuration

If you change the port speed in the `/etc/cumulus/ports.conf` file but the

speed is also configured for that port in the `/etc/network/interfaces` file, after you edit the `/etc/cumulus/ports.conf` file and restart `switchd`, you must also run the `ifreload -a` command so that the `/etc/network/interfaces` file is also updated with your change.

10G and 1G SFPs Inserted in a 25G Port

For 10G and 1G SFPs inserted in a 25G port on a Broadcom switch, you must configure the four ports in the same core to be 10G. Each set of four 25G ports are controlled by a single core; therefore, each core must run at the same clock speed. The four ports must be in sequential order; for example, swp1, swp2, swp3, and swp4, unless a particular core grouping is specified in the `/etc/cumulus/ports.conf` file.

1. Edit the `/etc/cumulus/ports.conf` file and configure the four ports to be 10G. 1G SFPs are clocked at 10G speeds; therefore, for 1G SFPs, the `/etc/cumulus/ports.conf` file entry must also specify 10G. Currently you cannot use NCLU commands for this step.

```
...
# SFP28 ports
#
# <port label 1-48> = [25G|10G|100G/4|40G/4]
1=25G
2=25G
3=25G
```

```
4=25G
5=10G
6=10G
7=10G
8=10G
9=25G
...
```

NOTE

You cannot use `ethtool -s speed XX` (or `ifreload -a` after setting the speed in the `/etc/network/interfaces` file) to change the port speed unless the four ports in a core group are already configured to 10G and `switchd` has been restarted. If the ports are still in 25G mode, using `ethtool` or `ifreload` to change the speed to 10G or 1G returns an error (and a return code of 255). If you change the speed with `ethtool` to a setting already in use in the `/etc/cumulus/ports.conf` file, `ethtool` (and `ifreload -a`) do not return an error and no changes are made.

2. Restart `switchd`.

```
cumulus@switch:~$ sudo systemctl restart switchd.service
```

WARNING

Restarting the `switchd` service causes all network ports to reset, interrupting network services, in addition to resetting the switch hardware configuration.

3. If you want to set the speed of any SFPs to 1G, set the port speed to 1000 Mbps using NCLU commands; this is *not* necessary for 10G SFPs. You don't need to set the port speed to 1G for all four ports. For example, if you intend only for swp5 and swp6 to use 1G SFPs, do the following:

```
cumulus@switch:~$ net add interface swp5-swp6 link speed 1000
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

NOTE

100G switch ASICs do not support 1000Base-X auto-negotiation (Clause 37), which is recommended for 1G fiber optical modules. As a result, single fiber breaks cannot be detected when using 1G optical modules on these switches.
The auto-negotiation setting must be the same on both sides of the connection. If using 1G fiber modules in 25G SFP28 ports, ensure auto-negotiation is disabled on the link partner interface as well.

Delta AGV848v1 Switch and Breakout Ports

Breaking out the 100G ports to 4x10G and 4x25G is not supported on the Delta AGV848v1 switch.

Timeout Error on Quanta LY8 and LY9 Switches

On Quanta T5048-LY8 and T3048-LY9 switches, an *Operation timed out* error occurs when you remove and reinsert a QSFP module.

You cannot remove the QSFPx2 module while the switch is powered on; it is *not* hot-swappable. However, if an *Operation timed out* error occurs, restart switchd to bring the link up. Be aware that this disrupts your network.

On the T3048-LY9, run the following commands:

```
cumulus@switch:~$ sudo echo 0 > qsfpd_power_enable/value
cumulus@switch:~$ sudo rmmod quanta_ly9_rangeley_platform
cumulus@switch:~$ sudo modprobe quanta_ly9_rangeley_platform
cumulus@switch:~$ sudo systemctl restart switchd.service
```

On the T5048-LY8, run the following commands:

```
cumulus@switch:~$ sudo echo 0 > qsfpd_power_enable/value
cumulus@switch:~$ sudo systemctl restart switchd.service
```

swp33 and swp34 Disabled on Some Switches

The front SFP+ ports (swp33 and swp34) are disabled in Cumulus Linux on the following switches:

- Dell Z9100-ON
- Penguin Arctica 3200-series switches (the 3200C, 3200XL and 3200XLP)
- Supermicro SSE-C3632S

These ports appear as disabled in the `/etc/cumulus/ports.conf` file.

200G Interfaces on the Dell S5248F Switch

On the Dell S5248F switch, the 2x200G QSFP-DD interfaces labeled 49/50

and 51/52 are not supported natively at 200G speeds. The interfaces are supported with 100G cables; however, you can only use one 100G from each QSFP-DD port. The upper QSFP-DD port is named swp49 and the lower QSFP-DD port is named swp52.

QSFP+ Ports on the Dell S5232F Switch

Cumulus Linux does not support the 2x10G QSFP+ ports on the Dell S5232F switch.

QSFP+ Ports on the Dell S4148T Switch

On the Dell S4148T switch, the two QSFP+ ports are set to `disabled` by default and the four QSFP28 ports are configured for 100G. The following example shows the default settings in the `/etc/cumulus/ports.conf` file for this switch:

```
cumulus@switch:~$ sudo cat /etc/cumulus/ports.conf
...
# QSFP+ ports
#
# <port label 27-28> = [4x10G|40G]
27=disabled
28=disabled
# QSFP28 ports
#
# <port label 25-26, 29-30> = [4x10G|4x25G|2x50G|40G|50G|100G]
```

```
25=100G
26=100G
29=100G
30=100G
```

To enable the two QSFP+ ports, you *must* configure all four QSFP28 ports for either 40G or 4x10G. You cannot use either of the QSFP+ ports if any of the QSFP28 ports are configured for 100G.

The following example shows the `/etc/cumulus/ports.conf` file with all four QSFP28 ports configured for 40G and both QSFP+ ports enabled:

```
cumulus@switch:~$ sudo cat /etc/cumulus/ports.conf
...
# QSFP+ ports
#
# <port label 27-28> = [4x10G|40G]
27=40G
28=40G
# QSFP28 ports
#
# <port label 25-26, 29-30> = [4x10G|4x25G|2x50G|40G|50G|100G]
25=40G
```

```
26=40G
29=40G
30=40G
```

> **NOTE**
>
> To disable the QSFP+ ports, you must set the ports to `disabled`. Do not comment out the lines as this prevents `switchd` from restarting.

1000BASE-T SFP Modules Supported Only on Certain 25G Platforms

1000BASE-T SFP modules are supported on only the following 25G platforms:

- Cumulus Express CX-5148-S and the Edgecore AS7326-56X, provided the switch has board revision R01D (to determine the revision of the board, look for the output in the `label revision` field when you run `decode-syseeprom`)
- Dell S5248F-ON
- Mellanox SN2410
- Mellanox SN2010

1000BASE-T SFP modules are not supported on any 100G or faster platforms.

Mellanox SN2100 Switch and eth0 Link Speed

After rebooting the Melllanox SN2100 switch, eth0 always has a speed of 100Mb/s. If you bring the interface down and then back up again, the interface negotiates 1000Mb. This only occurs the first time the interface comes up.

To work around this issue, add the following commands to the `/etc/rc.local` file to flap the interface automatically when the switch boots:

```
modprobe -r igb
sleep 20
modprobe igb
```

Link Speed on the EdgeCore AS7326-56X Switch

On the EdgeCore AS7326-56X switch, all four switch ports in each port group must be set to the same link speed; otherwise, the links do not come up. These ports are set to 25G by default, but can also be set to 10G. The port groups on this switch are as follows, where each row is a port group:

- 1 2 3 6*
- 4 5 7* 9
- 8 10 11* 12
- 13 14 15 18*
- 16 17 19* 21
- 20 22 23* 24

- 25 26 27 30*
- 28 29 31* 33
- 32 34 35* 36
- 37 38 39 42*
- 40* 41 43 45
- 44* 46 47 48

For example, if you configure port 19 for 10G, you must also configure ports 16, 17 and 21 for 10G.

Additionally, you can gang each port group together as a 100G or 40G port. When ganged together, one port (based on the arrangement of the ports) is designated as the gang leader. This port's number is used to configure the ganged ports and is marked with an asterisk (*) above.

> **NOTE**
>
> The EdgeCore AS7326-56X is a 48x25G + 8x100G + 2x10G switch. The dedicated 10G ports are not currently supported in Cumulus Linux. However, you can configure all other ports to run at 10G speeds.

Link Speed on the Lenovo NE2572O Switch

The Lenovo NE2572O switch has external retimers on swp1 through swp8. Currently, these ports only support a speed of 25G.

Link Speed and Auto-negotiation on Switches with SOL

The following switches that use Serial over LAN technology (SOL) do not support eth0 speed or auto-negotiation changes:

- EdgeCore AS7816-64X
- Penguin Arctica 4804ip
- Penguin Arctica NX3200c
- Penguin Arctica NX4808xxv

Delay in Reporting Interface as Operational Down

When you remove two transceivers simultaneously from a switch, both interfaces show the `carrier down` status immediately. However, it takes one second for the second interface to show the `operational down` status. In addition, the services on this interface also take an extra second to come down.

Mellanox Spectrum-2 and Tomahawk-based Switches Support Different FEC Modes

The Mellanox Spectrum-2 (25G) switch only supports RS FEC. The Tomahawk-based switch only supports BASE-R FEC. These two switches do not share compatible FEC modes and do not interoperate reliably.

Maverick Switches with Modules that Don’t Support Auto-negotiation

On a Maverick switch, if auto-negotiation is configured on a 10G interface and the installed module does not support auto-negotiation (for example,

10G DAC, 10G Optical, 1G RJ45 SFP), the link breaks. To work around this issue, disable auto-negotiation on interfaces where it is not supported.

Related Information

- Debian - Network Configuration
- Linux Foundation - VLANs
- Linux Foundation - Bonds

ifplugd

`ifplugd` is an Ethernet link-state monitoring daemon that executes user-specified scripts to configure an Ethernet device when a cable is plugged in, or automatically unconfigure an Ethernet device when a cable is removed. Follow the steps below to install and configure the `ifplugd` daemon.

Install ifplugd

You can install this package even if the switch is not connected to the internet, as it is contained in the `cumulus-local-apt-archive` repository that is embedded in the Cumulus Linux image.

To install `ifplugd`:

1. Update the switch before installing the daemon:

```
cumulus@switch:~$ sudo -E apt-get update
```

2. Install the `ifplugd` package:

```
cumulus@switch:~$ sudo -E apt-get install ifplugd
```

Configure ifplugd

After you install `ifplugd`, you must edit two configuration files:

- `/etc/default/ifplugd`
- `/etc/ifplugd/action.d/ifupdown`

The example configuration below configures `ifplugd` to bring down all uplinks when the peer bond goes down in an MLAG environment.

1. Open `/etc/default/ifplugd` in a text editor and configure the file as appropriate. Add the `peerbond` name before you save the file.

```
INTERFACES="peerbond"
HOTPLUG_INTERFACES=""
ARGS="-q -f -u0 -d1 -w -I"
SUSPEND_ACTION="stop"
```

2. Open the `/etc/ifplugd/action.d/ifupdown` file in a text editor. Configure the script, then save the file.

```
#!/bin/sh
set -e
case "$2" in
```

```
up)
        clagrole=$(clagctl | grep "Our Priority" | awk
'{print $8}')
        if [ "$clagrole" = "secondary" ]
        then
            #List all the interfaces below to bring up when
clag peerbond comes up.
            for interface in swp1 bond1 bond3 bond4
            do
                echo "bringing up : $interface"
                ip link set $interface up
            done
        fi
    ;;
down)
        clagrole=$(clagctl | grep "Our Priority" | awk
'{print $8}')
        if [ "$clagrole" = "secondary" ]
        then
            #List all the interfaces below to bring down when
clag peerbond goes down.
            for interface in swp1 bond1 bond3 bond4
            do
                echo "bringing down : $interface"
```

```
                ip link set $interface down
            done
        fi
    ;;
esac
```

3. Restart the `ifplugd` daemon to implement the changes:

```
cumulus@switch:~$ sudo systemctl restart ifplugd.service
```

Considerations

The default shell for `ifplugd` is `dash` (`/bin/sh`) instead of `bash`, as it provides a faster and more nimble shell. However, `dash` contains fewer features than `bash` (for example, `dash` is unable to handle multiple uplinks).

Buffer and Queue Management

Hardware datapath configuration manages packet buffering, queueing and scheduling in hardware. To configure priority groups, and assign the scheduling alogorithm and weights, you edit the `/etc/cumulus/datapath/traffic.conf`.

> **NOTE**
>
> The `/usr/lib/python2.7/dist-packages/cumulus/__chip_config/[bcm|mlx]/datapath.conf` assigns buffer space and egress queues. Working with a support engineer to change buffer limits in the `datapath.conf` file is recommended.

Each packet is assigned to an ASIC Class of Service (CoS) value based on the priority value of the packet stored in the 802.1p (Class of Service) or DSCP (Differentiated Services Code Point) header field. The choice to schedule packets based on COS or DSCP is a configurable option in the `/etc/cumulus/datapath/traffic.conf` file.

Priority groups include:

- *Control*: Highest priority traffic
- *Service*: Second-highest priority traffic

- *Bulk*: All remaining traffic

The scheduler is configured to use a hybrid scheduling algorithm. It applies strict priority to control traffic queues and a weighted round robin selection from the remaining queues. Unicast packets and multicast packets with the same priority value are assigned to separate queues, which are assigned equal scheduling weights.

> **NOTE**
>
> You can configure Quality of Service (QoS) for switches on the following platforms only:
>
> - Broadcom Tomahawk, Trident II, Trident II+, and Trident3
> - Mellanox Spectrum, Spectrum-2, and Spectrum-3

Syntax Checker

Cumulus Linux provides a syntax checker for the `/etc/cumulus/datapath/traffic.conf` file to check for errors, such missing parameters, or invalid parameter labels and values.

On Broadcom switches, the syntax checker runs automatically during `switchd` initialization and reports syntax errors to the `/var/log/switchd.log` file.

On both Broadcom and Mellanox switches, you can run the syntax checker manually from the command line by issuing the `cl-consistency-check --datapath-syntax-check` command. If errors exist, they are written to `stderr` by default. If you run the command with `-q`, errors are written to the `/var/log/switchd.log` file.

The `cl-consistency-check --datapath-syntax-check` command takes the following options:

Option	Description
-h	Displays this list of command options.
-q	Runs the command in quiet mode. Errors are written to the `/var/log/switchd.log` file instead of `stderr`.
-t `<file-name>`	Runs the syntax check on a non-default `traffic.conf` file; for example, `/mypath/test-traffic.conf`.

You can run the syntax checker when `switchd` is either running or stopped.

Example Commands

The following example command runs the syntax checker on the default `/etc/cumulus/datapath/traffic.conf` file and shows that no errors are detected:

```
cumulus@switch:~$ cl-consistency-check --datapath-syntax-check
No errors detected in traffic config file /etc/cumulus/datapath/
traffic.conf
```

The following example command runs the syntax checker on the default `/etc/cumulus/datapath/traffic.conf` file in quiet mode. If errors exist, they are written to the `/var/log/switchd.log` file.

```
cumulus@switch:~$ cl-consistency-check --datapath-syntax-check
-q
```

The following example command runs the syntax checker on the `/mypath/test-traffic.conf` file and shows that errors are detected:

```
cumulus@switch:~$ cl-consistency-check --datapath-syntax-check
-t /path/test-traffic.conf
Traffic source 8021p: missing mapping for priority value '7'
Errors detected while checking traffic config file /mypath/test-
traffic.conf
```

The following example command runs the syntax checker on the `/mypath/`

`test-traffic.conf` file in quiet mode. If errors exist, they are written to the `/var/log/switchd.log` file.

```
cumulus@switch:~$ cl-consistency-check --datapath-syntax-check
-t /path/test-traffic.conf -q
```

Configure Traffic Marking through ACL Rules

You can mark traffic for egress packets through `iptables` or `ip6tables` rule classifications. To enable these rules, you do one of the following:

- Mark DSCP values in egress packets.
- Mark 802.1p CoS values in egress packets.

To enable traffic marking, use `cl-acltool`. Add the `-p` option to specify the location of the policy file. By default, if you do not include the `-p` option, `cl-acltool` looks for the policy file in `/etc/cumulus/acl/policy.d/`.

The `iptables`-/`ip6tables`-based marking is supported with the following action extension:

```
-j SETQOS --set-dscp 10 --set-cos 5
```

For `ebtables`, the setqos keyword must be in lowercase, as in:

```
[ebtables]
-A FORWARD -o swp5 -j setqos --set-cos 5
```

You can specify one of the following targets for SETQOS/setqos:

Option	Description
`--set-cos INT`	Sets the datapath resource/ queuing class value. Values are defined in IEEE P802.1p.
`--set-dscp value`	Sets the DSCP field in packet header to a value, which can be either a decimal or hex value.
`--set-dscp-class class`	Sets the DSCP field in the packet header to the value represented by the DiffServ class value. This class can be EF, BE or any of the CSxx or AFxx classes.

NOTE

You can specify either `--set-dscp` or `--set-dscp-class`, but not both.

Here are two example rules:

```
[iptables]
-t mangle -A FORWARD --in-interface swp+ -p tcp --dport bgp -j
SETQOS --set-dscp 10 --set-cos 5

[ip6tables]
-t mangle -A FORWARD --in-interface swp+ -j SETQOS --set-dscp 10
```

You can put the rule in either the *mangle* table or the default *filter* table; the mangle table and filter table are put into separate TCAM slices in the hardware.

To put the rule in the mangle table, include `-t mangle`; to put the rule in the filter table, omit `-t mangle`.

Priority Flow Control

Priority flow control, as defined in the IEEE 802.1Qbb standard, provides a link-level flow control mechanism that can be controlled independently for each Class of Service (CoS) with the intention to ensure no data frames are lost when congestion occurs in a bridged network.

> **NOTE**
>
> PFC is not supported on switches with the Helix4 ASIC.

PFC is a layer 2 mechanism that prevents congestion by throttling packet transmission. When PFC is enabled for received packets on a set of switch ports, the switch detects congestion in the ingress buffer of the receiving port and signals the upstream switch to stop sending traffic. If the upstream switch has PFC enabled for packet transmission on the designated priorities, it responds to the downstream switch and stops sending those packets for a period of time.

PFC operates between two adjacent neighbor switches; it does not provide end-to-end flow control. However, when an upstream neighbor throttles packet transmission, it could build up packet congestion and propagate PFC frames further upstream: eventually the sending server could receive PFC frames and stop sending traffic for a time.

The PFC mechanism can be enabled for individual switch priorities on specific switch ports for RX and/or TX traffic. The switch port's ingress buffer occupancy is used to measure congestion. If congestion is present, the switch transmits flow control frames to the upstream switch. Packets with priority values that do not have PFC configured are not counted during congestion detection; neither do they get throttled by the upstream switch when it receives flow control frames.

PFC congestion detection is implemented on the switch using xoff and xon threshold values for the specific ingress buffer which is used by the targeted switch priorities. When a packet enters the buffer and the buffer occupancy is above the xoff threshold, the switch transmits an Ethernet PFC frame to the upstream switch to signal packet transmission should stop. When the buffer occupancy drops below the xon threshold, the switch sends another PFC frame upstream to signal that packet transmission can resume. (PFC frames contain a quanta value to indicate a timeout value for the upstream switch: packet transmission can resume after the timer has expired, or when a PFC frame with quanta == 0 is received from the downstream switch.)

After the downstream switch has sent a PFC frame upstream, it continues to receive packets until the upstream switch receives and responds to the PFC frame. The downstream ingress buffer must be large enough to store those additional packets after the xoff threshold has been reached.

Priority flow control is fully supported on both Broadcom (including the Edgecore Minipack-AS8000/Trident3) and Mellanox switches.

PFC is disabled by default in Cumulus Linux. To enable priority flow control (PFC), you must configure the following settings in the `/etc/cumulus/datapath/traffic.conf` file on the switch:

- Specify the name of the port group in `pfc.port_group_list` in brackets; for example, *pfc.port_group_list = [pfc_port_group]*.
- Assign a CoS value to the port group in `pfc.pfc_port_group.cos_list` setting. *pfc_port_group* is the name of a port group you specified above

and is used throughout the following settings.

- Populate the port group with its member ports in `pfc.pfc_port_group.port_set`.
- Set a PFC buffer size in `pfc.pfc_port_group.port_buffer_bytes`. This is the maximum number of bytes allocated for storing bursts of packets, guaranteed at the ingress port. The default is *25000* bytes.
- Set the xoff byte limit in `pfc.pfc_port_group.xoff_size`. This is a threshold for the PFC buffer; when this limit is reached, an xoff transition is initiated, signaling the upstream port to stop sending traffic, during which time packets continue to arrive due to the latency of the communication. The default is *10000* bytes.
- Set the xon delta limit in `pfc.pfc_port_group.xon_delta`. This is the number of bytes to subtract from the xoff limit, which results in a second threshold at which the egress port resumes sending traffic. After the xoff limit is reached and the upstream port stops sending traffic, the buffer begins to drain. When the buffer reaches 8000 bytes (assuming default xoff and xon settings), the egress port signals that it can start receiving traffic again. The default is *2000* bytes.
- Enable the egress port to signal the upstream port to stop sending traffic (`pfc.pfc_port_group.tx_enable`). The default is *true*.
- Enable the egress port to receive notifications and act on them (`pfc.pfc_port_group.rx_enable`). The default is *true*.
- The switch priority value(s) are mapped to the specific ingress buffer for each targeted switch port. Cumulus Linux looks at either the 802.1p bits or the IP layer DSCP bits depending on which is configured in the `traffic.conf` file to map packets to internal switch priority values.

The following configuration example shows PFC configured for ports swp1 through swp4 and swp6:

```
# to configure priority flow control on a group of ports:
# -- assign cos value(s) to the cos list
# -- add or replace a port group names in the port group list
# -- for each port group in the list
#    -- populate the port set, e.g.
#       swp1-swp4,swp8,swp50s0-swp50s3
#    -- set a PFC buffer size in bytes for each port in the
group
#    -- set the xoff byte limit (buffer limit that triggers PFC
frame transmit to start)
#    -- set the xon byte delta (buffer limit that triggers PFC
frame transmit to stop)
#    -- enable PFC frame transmit and/or PFC frame receive
# priority flow control
pfc.port_group_list = [pfc_port_group]
pfc.pfc_port_group.cos_list = []
pfc.pfc_port_group.port_set = swp1-swp4,swp6
pfc.pfc_port_group.port_buffer_bytes = 25000
pfc.pfc_port_group.xoff_size = 10000
pfc.pfc_port_group.xon_delta = 2000
pfc.pfc_port_group.tx_enable = true
pfc.pfc_port_group.rx_enable = true
```

Port Groups

A *port group* refers to one or more sequences of contiguous ports. You can define multiple port groups by adding:

- A comma-separated list of port group names to the port_group_list.
- The port_set, rx_enable, and tx_enable configuration lines for each port group.

You can specify the set of ports in a port group in comma-separate sequences of contiguous ports; you can see which ports are contiguous in the `/var/lib/cumulus/porttab` file. The syntax supports:

- A single port (swp1s0 or swp5).
- A sequence of regular swp ports (swp2-swp5).
- A sequence within a breakout swp port (swp6s0-swp6s3).
- A sequence of regular and breakout ports, provided they are all in a contiguous range. For example:

```
...
swp2
swp3
swp4
swp5
swp6s0
swp6s1
```

```
swp6s2
swp6s3
swp7
...
```

On a Broadcom switch, restart `switchd` with the `sudo systemctl restart switchd.service` command to allow the PFC configuration changes to take effect. On a Mellanox switch with the Spectrum ASIC, restarting `switchd` is not necessary.

```
cumulus@switch:~$ sudo systemctl restart switchd.service
```

WARNING

Restarting the `switchd` service causes all network ports to reset, interrupting network services, in addition to resetting the switch hardware configuration.

Link Pause

The PAUSE frame is a flow control mechanism that halts the transmission of the transmitter for a specified period of time. A server or other network node within the data center may be receiving traffic faster than it can handle it, thus the PAUSE frame. In Cumulus Linux, you can configure individual ports to execute link pause by:

- Transmitting pause frames when its ingress buffers become congested (TX pause enable).
- Responding to received pause frames (RX pause enable).

Link pause is disabled by default. To enabling link pause, you must configure settings in the `/etc/cumulus/datapath traffic.conf` file.

TIP

What's the difference between link pause and priority flow control?

- Priority flow control is applied to an individual priority group for a specific ingress port.
- Link pause (also known as port pause or global pause) is applied to all the traffic for a specific ingress port.

Here is an example configuration that enables both types of link pause for swp1 through swp4 and swp6:

```
# to configure pause on a group of ports:
# -- add or replace port group names in the port group list
# -- for each port group in the list
#    -- populate the port set, e.g.
#       swp1-swp4,swp8,swp50s0-swp50s3
#    -- set a pause buffer size in bytes for each port in the
group
#    -- set the xoff byte limit (buffer limit that triggers
pause frames transmit to start)
#    -- set the xon byte delta (buffer limit that triggers
pause frames transmit to stop)

# link pause
link_pause.port_group_list = [pause_port_group]
link_pause.pause_port_group.port_set = swp1-swp4,swp6
link_pause.pause_port_group.port_buffer_bytes = 25000
link_pause.pause_port_group.xoff_size = 10000
link_pause.pause_port_group.xon_delta = 2000
link_pause.pause_port_group.rx_enable = true
link_pause.pause_port_group.tx_enable = true
```

On a Broadcom switch, restart `switchd` with the `sudo systemctl restart`

`switchd.service` command to allow the PFC configuration changes to take effect. On a Mellanox switch with the Spectrum ASIC, restarting `switchd` is not necessary.

```
cumulus@switch:~$ sudo systemctl restart switchd.service
```

WARNING

Restarting the `switchd` service causes all network ports to reset, interrupting network services, in addition to resetting the switch hardware configuration.

Cut-through Mode and Store and Forward Switching

Cut-through mode is disabled in Cumulus Linux by default on switches with Broadcom ASICs. With cut-though mode enabled and link pause is asserted, Cumulus Linux generates a TOVR and TUFL ERROR; certain error counters increment on a given physical port.

```
cumulus@switch:~$ sudo ethtool -S swp49 | grep Error
```

```
HwIfInDot3LengthErrors: 0
HwIfInErrors: 0
HwIfInDot3FrameErrors: 0
SoftInErrors: 0
SoftInFrameErrors: 0
HwIfOutErrors: 35495749
SoftOutErrors: 0

cumulus@switch:~$ sudo ethtool -S swp50 | grep Error
HwIfInDot3LengthErrors: 3038098
HwIfInErrors: 297595762
HwIfInDot3FrameErrors: 293710518
```

To work around this issue, disable link pause or disable cut-through mode in the `/etc/cumulus/datapath/traffic.conf` file.

To disable link pause, comment out the `link_pause*` section in the `/etc/cumulus/datapath/traffic.conf` file:

```
cumulus@switch:~$ sudo nano /etc/cumulus/datapath/traffic.conf
#link_pause.port_group_list = [port_group_0]
#link_pause.port_group_0.port_set = swp45-swp54
#link_pause.port_group_0.rx_enable = true
```

```
#link_pause.port_group_0.tx_enable = true
```

To enable store and forward switching, set `cut_through_enable` to *false* in the `/etc/cumulus/datapath/traffic.conf` file:

```
cumulus@switch:~$ sudo nano /etc/cumulus/datapath/traffic.conf
cut_through_enable = false
```

> **NOTE**
>
> On switches using Broadcom Tomahawk, Trident II, Trident II+, and Trident3 ASICs, Cumulus Linux supports store and forward switching but does **not** support cut-through mode.
> On switches with the Mellanox Spectrum ASIC, Cumulus Linux supports cut-through mode but does **not** support store and forward switching.

Congestion Notification

Explicit Congestion Notification (ECN) is defined by RFC 3168. ECN enables the Cumulus Linux switch to mark a packet to signal impending congestion instead of dropping the packet, which is how TCP typically behaves when

ECN is not enabled.

ECN is a layer 3 end-to-end congestion notification mechanism only. Packets can be marked as *ECN-capable transport* (ECT) by the sending server. If congestion is observed by any switch while the packet is getting forwarded, the ECT-enabled packet can be marked by the switch to indicate the congestion. The end receiver can respond to the ECN-marked packets by signaling the sending server to slow down transmission. The sending server marks a packet *ECT* by setting the least 2 significant bits in an IP header `DiffServ` (ToS) field to *01* or *10*. A packet that has the least 2 significant bits set to *00* indicates a non-ECT-enabled packet.

The ECN mechanism on a switch only marks packets to notify the end receiver. It does not take any other action or change packet handling in any way, nor does it respond to packets that have already been marked ECN by an upstream switch.

NOTE

On Trident II switches only, if ECN is enabled on a specific queue, the ASIC also enables RED on the same queue. If the packet is ECT marked (the ECN bits are 01 or 10), the ECN mechanism executes as described above. However, if it is entering an ECN-enabled queue but is not ECT marked (the ECN bits are 00), then the RED mechanism uses the same threshold and probability values to decide whether to drop the packet. Packets entering a non-ECN-

enabled queue do not get marked or dropped due to ECN or RED in any case.

ECN is implemented on the switch using minimum and maximum threshold values for the egress queue length. When a packet enters the queue and the average queue length is between the minimum and maximum threshold values, a configurable probability value will determine whether the packet will be marked. If the average queue length is above the maximum threshold value, the packet is always marked.

The downstream switches with ECN enabled perform the same actions as the traffic is received. If the ECN bits are set, they remain set. The only way to overwrite ECN bits is to set the ECN bits to *11*.

ECN is supported on Broadcom Tomahawk, Tomahawk2, Trident II, Trident II+ and Trident3, and Mellanox Spectrum ASICs.

▼ Click to learn how to configure ECN

Check Interface Buffer Status

- On switches with ASICs, you can collect a fine-grained history of queue lengths using histograms maintained by the ASIC; see the ASIC Monitoring for details.
- On Broadcom switches, the buffer status is not visible currently.

Example Configuration File

The following example `/etc/cumulus/datapath/traffic.conf` datapath configuration file applies to 10G, 40G, and 100G switches on Broadcom Tomahawk, Trident II, Trident II+, or Trident3 and Mellanox Spectrum platforms only.

- For the default source packet fields and mapping, each selected packet field must have a block of mapped values. Any packet field value that is not specified in the configuration is assigned to a default internal switch priority. The configuration applies to every forwarding port unless a custom remark configuration is defined for that port (see below).
- For the default remark packet fields and mapping, each selected packet field should have a block of mapped values. Any internal switch priority value that is not specified in the configuration is assigned to a default packet field value. The configuration applies to every forwarding port unless a custom remark configuration is defined for that port (see below).
- Per-port source packet fields and mapping apply to the designated set of ports.
- Per-port remark packet fields and mapping apply to the designated set of ports.

▼ Click to see the traffic.conf file

NOTE

On switches with Spectrum ASICs, you must enable packet priority remark on the **ingress** port. A packet received on a remark-enabled port is remarked according to the priority mapping configured on the **egress** port. If you configure packet priority remark the same way on every port, the default configuration example above is correct. However, per-port customized configurations require two port groups, one for the ingress ports and one for the egress ports, as below:

```
remark.port_group_list = [ingress_remark_group,
egress_remark_group]
remark.ingress_remark_group.packet_priority_remark_set =
[dscp]
remark.remark_port_group.port_set = swp1-swp4,swp6
remark.egress_remark_group.port_set = swp10-swp20
remark.egress_remark_group.cos_0.priority_remark.dscp =
[2]
remark.egress_remark_group.cos_1.priority_remark.dscp =
[10]
remark.egress_remark_group.cos_2.priority_remark.dscp =
[18]
```

```
remark.egress_remark_group.cos_3.priority_remark.dscp =
[26]
remark.egress_remark_group.cos_4.priority_remark.dscp =
[34]
remark.egress_remark_group.cos_5.priority_remark.dscp =
[42]
remark.egress_remark_group.cos_6.priority_remark.dscp =
[50]
remark.egress_remark_group.cos_7.priority_remark.dscp =
[58]
```

On Broadcom switches, if you modify the configuration in the `/etc/cumulus/datapath/traffic.conf` file, you *must* restart `switchd` for the changes to take effect; run the `cumulus@switch:~$ sudo systemctl restart switchd.service` command.

```
cumulus@switch:~$ sudo systemctl restart switchd.service
```

WARNING

Restarting the `switchd` service causes all network ports to reset, interrupting network services, in addition to resetting the switch hardware configuration.

On Mellanox switches with the Spectrum ASIC, the following options in the `/etc/cumulus/datapath/traffic.conf` file do *not* require you to restart `switchd`. However, you must run the `echo 1 > /cumulus/switchd/config/traffic/reload` command after you change the options.

```
cumulus@switch:~$ echo 1 > /cumulus/switchd/config/traffic/
reload
```

- DSCP/802.1p to COS remark assignments (`traffic.*`)
- Explicit congestion notification (ECN) settings (`ecn_red.*`)
- Priority flow control (PFC) settings (`pfc.*`)
- Link Pause settings (`link_pause.*`)
- Queue weight settings (`priority_group.*.weight`)
- ECMP hash and LAG hash settings

Related Information

- iptables-extensions man page

Hardware-enabled DDOS Protection

It is crucial to protect the control plane on the switch to ensure that the proper control plane applications have access to the CPU. Failure to do so increases vulnerabilities to a Denial of Service (DOS attack. Cumulus Linux provides control plane protection by default. In addition, you can configure DDOS protection to protect data plane, control plane, and management plane traffic on the switch. You can configure Cumulus Linux to drop packets that match one or more of the following criteria while incurring no performance impact:

- Source IP address matches the destination address for IPv4 and IPv6 packets
- Source MAC address matches the destination MAC address
- Unfragmented or first fragment SYN packets with a source port of 0-1023
- TCP packets with control flags =0 and seq number == 0
- TCP packets with FIN, URG and PSH bits set and seq number == 0
- TCP packets with both SYN and FIN bits set
- TCP source PORT matches the destination port
- UDP source PORT matches the destination port
- First TCP fragment with partial TCP header
- TCP header has fragment offset value of 1
- ICMPv6 ping packets payload larger than programmed value of ICMP max size

- ICMPv4 ping packets payload larger than programmed value of ICMP max size
- Fragmented ICMP packet
- IPv6 fragment lower than programmed minimum IPv6 packet size

> **NOTE**
>
> DDOS protection is not supported on Broadcom Hurricane2 and Mellanox Spectrum ASICs.

Configure DDOS Protection

1. Open the `/etc/cumulus/datapath/traffic.conf` file in a text editor.
2. Enable DOS prevention checks by setting the `dos_enable` value to `true`:

```
# To turn on/off Denial of Service (DOS) prevention checks
dos_enable = true
```

3. Open the `/usr/lib/python2.7/dist-packages/cumulus/__chip_config/bcm/datapath.conf` file in a text editor. Set any of the DOS checks to *true*. For example:

```
cumulus@switch:~$ sudo nano /usr/lib/python2.7/dist-packages/
cumulus/__chip_config/bcm/datapath.conf
# Enabling/disabling Denial of service (DOS) prevetion checks
# To change the default configuration:
# enable/disable the individual DOS checks.
dos.sip_eq_dip = true
dos.smac_eq_dmac = true
dos.tcp_hdr_partial = true
dos.tcp_syn_frag = true
dos.tcp_ports_eq = true
dos.tcp_flags_syn_fin = true
dos.tcp_flags_fup_seq0 = true
dos.tcp_offset1 = true
dos.tcp_ctrl0_seq0 = true
dos.udp_ports_eq = true
dos.icmp_frag = true
dos.icmpv4_length = true
dos.icmpv6_length = true
dos.ipv6_min_frag = true
```

NOTE

Configuring any of the following settings affects the BFD echo

function. For example, if you enable `dos.udp_ports_eq`, all the BFD packets are dropped because the BFD protocol uses the same source and destination UDP ports.

```
dos.sip_eq_dip
dos.smac_eq_dmac
dos.tcp_ctrl0_seq0
dos.tcp_flags_fup_seq0
dos.tcp_flags_syn_fin
dos.tcp_ports_eq
dos.tcp_syn_frag
dos.udp_ports_eq
```

4. Restart `switchd`:

```
cumulus@switch:~$ sudo systemctl restart switchd.service
```

WARNING

Restarting the `switchd` service causes all network ports to reset, interrupting network services, in addition to resetting the switch hardware configuration.

DHCP

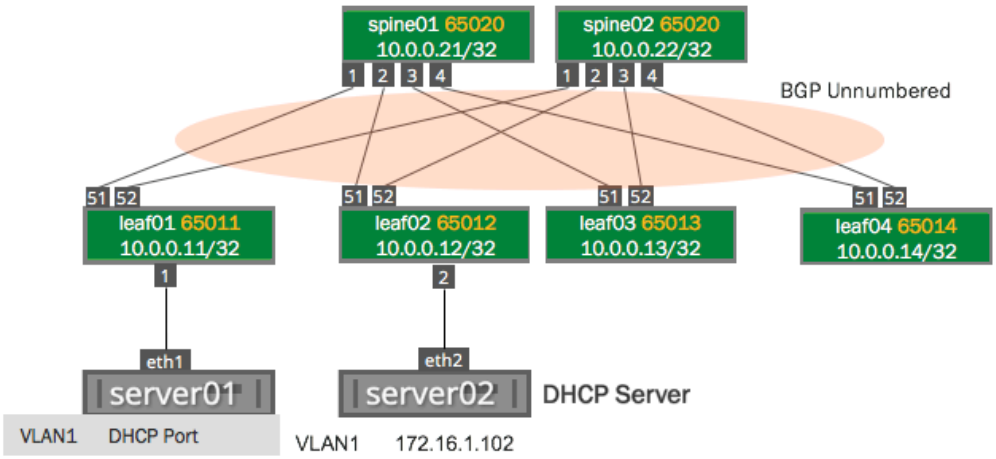
This section descibes how to configure:

- DHCP relays for IPv4 and IPv6
- DHCP servers for IPv4 and IPv6
- DCHP snooping for IPv4 and IPv6

DHCP Relays

DHCP is a client/server protocol that automatically provides IP hosts with IP addresses and other related configuration information. A DHCP relay (agent) is a host that forwards DHCP packets between clients and servers. DHCP relays forward requests and replies between clients and servers that are not on the same physical subnet.

This topic describes how to configure DHCP relays for IPv4 and IPv6. Configurations on the server hosts, DHCP relays, and DHCP server are provided using the following topology:



> **NOTE**
>
> The `dhcpd` and `dhcrelay` services are disabled by default. After you

finish configuring the DHCP relays and servers, you need to start those services. If you intend to run these services within a VRF, including the management VRF, follow these steps.

Configure IPv4 DHCP Relays

To configure IPv4 DHCP relays, run the following commands.

NCLU Commands | Linux Commands

WARNING

You configure a DHCP relay on a per-VLAN basis, specifying the SVI, not the parent bridge. In the example below, you specify v*lan1* as the SVI for VLAN 1 but you do not specify the bridge named *bridge* in this case.

Specify the IP address of each DHCP server and the interfaces that are used as the uplinks. In the example commands below, the DHCP server IP address is 172.16.1.102, VLAN 1 (the SVI is vlan1) and the uplinks are swp51 and swp52. As per RFC 3046, you can specify as many server IP addresses that can fit in 255 octets. You can specify each address only once.

```
cumulus@switch:~$ net add dhcp relay interface swp51
cumulus@switch:~$ net add dhcp relay interface swp52
cumulus@switch:~$ net add dhcp relay interface vlan1
cumulus@switch:~$ net add dhcp relay server 172.16.1.102
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following configuration in the `/etc/default/isc-dhcp-relay` file:

To see the DHCP relay status, use the `systemctl status dhcrelay.service` command:

```
cumulus@switch:~$ sudo systemctl status dhcrelay.service
● dhcrelay.service - DHCPv4 Relay Agent Daemon
    Loaded: loaded (/lib/systemd/system/dhcrelay.service;
enabled)
    Active: active (running) since Fri 2016-12-02 17:09:10 UTC;
2min 16s ago
      Docs: man:dhcrelay(8)
Main PID: 1997 (dhcrelay)
    CGroup: /system.slice/dhcrelay.service
            └─1997 /usr/sbin/dhcrelay --nl -d -q -i vlan1 -i
swp51 -i swp52 172.16.1.102
```

DHCP Agent Information Option (Option 82)

Cumulus Linux supports DHCP Agent Information Option 82, which allows a DHCP relay to insert circuit or relay specific information into a request that is being forwarded to a DHCP server. Two sub-options are provided:

- The Circuit ID sub-option includes information about the circuit on which the request comes in, such as the SVI or physical port.
- The Remote ID sub-option includes information that identifies the relay agent, such as the MAC address.

To enable the DHCP Agent Information Option, you configure the `-a` option.

By default, when you enable this option, the Circuit ID is the printable name of the interface on which the client request is received, typically an SVI. The Remote ID is the System MAC of the device on which DHCP relay is running.

> **NOTE**
>
> NCLU commands are not currently available for this feature. Use the following Linux commands.

- To configure the DHCP relay to inject the ingress *SVI interface* against which the relayed DHCP discover packet is processed, edit `/etc/default/isc-dhcp-relay` file and add `-a` to the `OPTIONS` line. For example:

```
cumulus@switch:~$ sudo nano /etc/default/isc-dhcp-relay
...
# Additional options that are passed to the DHCP relay daemon?
OPTIONS="-a"
```

- To configure the DHCP relay to inject the *physical switch port* on which the relayed DHCP discover packet arrives instead of the SVI, edit the `/etc/default/isc-dhcp-relay` file and add `-a --use-pif-circuit-id` to the `OPTIONS` line. For example:

```
cumulus@switch:~$ sudo nano /etc/default/isc-dhcp-relay
...
# Additional options that are passed to the DHCP relay daemon?
OPTIONS="-a --use-pif-circuit-id"
```

- To customize the Remote ID sub-option, edit `/etc/default/isc-dhcp-relay` file and add `-a -r` to the `OPTIONS` line followed by a custom string (up to 255 characters that is used for the Remote ID. For example:

```
cumulus@switch:~$ sudo nano /etc/default/isc-dhcp-relay
...
# Additional options that are passed to the DHCP relay daemon?
OPTIONS="-a -r CUSTOMVALUE"
```

Make sure to restart the `dhcrelay` service to apply the new configuration:

```
cumulus@switch:~$ sudo systemctl restart dhcrelay.service
```

Control the Gateway IP Address with RFC 3527

When DHCP relay is required in an environment that relies on an anycast gateway (such as EVPN), a unique IP address is necessary on each device

for return traffic. By default, in a BGP unnumbered environment with DHCP relay, the source IP address is set to the loopback IP address and the gateway IP address (giaddr) is set as the SVI IP address. However with anycast traffic, the SVI IP address is not unique to each rack; it is typically shared amongst all racks. Most EVPN ToR deployments only possess a single unique IP address, which is the loopback IP address.

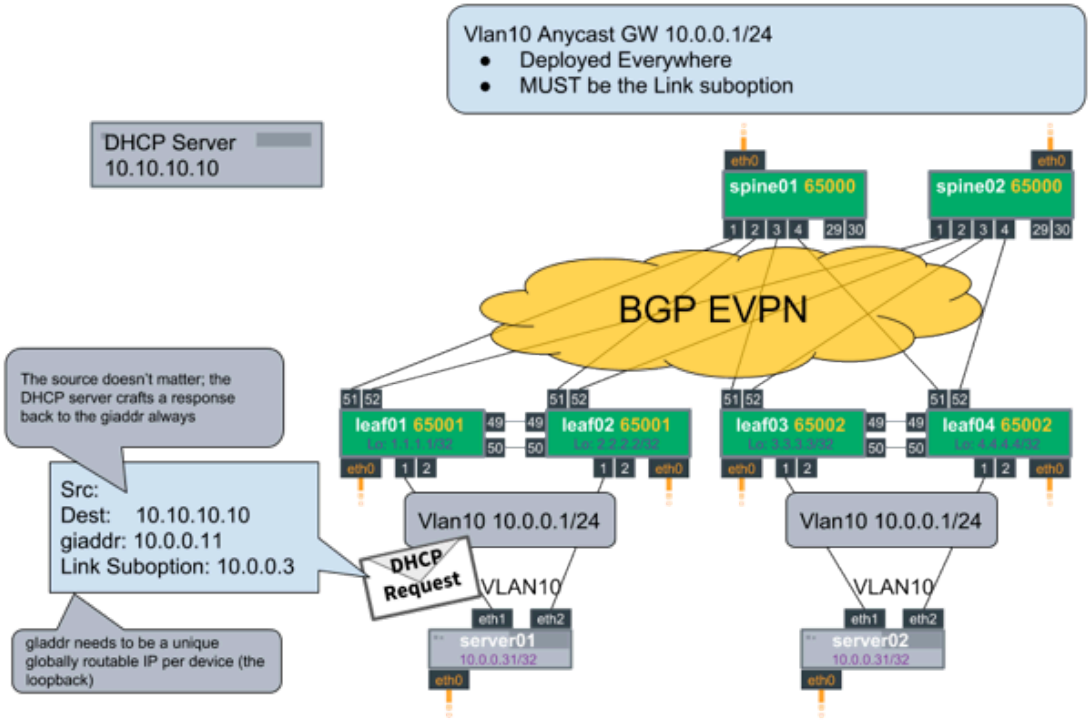
RFC 3527 enables the DHCP server to react to these environments by introducing a new parameter to the DHCP header called the link selection sub-option, which is built by the DHCP relay agent. The link selection sub-option takes on the normal role of the giaddr in relaying to the DHCP server which subnet is correlated to the DHCP request. When using this sub-option, the giaddr continues to be present but only relays the return IP address that is to be used by the DHCP server; the giaddr becomes the unique loopback IP address.

When enabling RFC 3527 support, you can specify an interface, such as the loopback interface or a switch port interface to be used as the giaddr. The relay picks the first IP address on that interface. If the interface has multiple IP addresses, you can specify a specific IP address for the interface.

> **NOTE**
>
> RFC 3527 is supported for IPv4 DHCP relays only.

The following illustration demonstrates how you can control the giaddr with

RFC 3527.



To enable RFC 3527 support and control the giaddr, run the following commands.

NCLU Commands | Linux Commands

1. Run the `net add dhcp relay giaddr-interface` command with the interface/IP address you want to use. The following example uses the first IP address on the loopback interface as the giaddr:

```
cumulus@switch:~$ net add dhcp relay giaddr-interface lo
```

The above command creates the following configuration in the `/etc/default/isc-dhcp-relay` file:

```
# Additional options that are passed to the DHCP relay
daemon?
OPTIONS="-U lo"
```

The first IP address on the loopback interface is typically the 127.0.0.1 address; Consider using more specific syntax, as shown in the next example.

The following example uses IP address 10.0.0.1 on the loopback interface as the giaddr:

```
cumulus@switch:~$ net add dhcp relay giaddr-interface lo
10.0.0.1
```

The above command creates the following configuration in the `/etc/default/isc-dhcp-relay` file:

Gateway IP Address as Source IP for Relayed DHCP Packets (Advanced)

You can configure the `dhcrelay` service to forward IPv4 (only) DHCP packets to a DHCP server and ensure that the source IP address of the relayed packet is the same as the gateway IP address.

> **NOTE**
>
> This option impacts all relayed IPv4 packets globally.

To use the gateway IP address as the source IP address:

NCLU Commands | Linux Commands

Run these commands:

```
cumulus@leaf:~$ net add dhcp relay use-giaddr-as-src
cumulus@leaf:~$ net pending
cumulus@leaf:~$ net commit
```

Configure IPv6 DHCP Relays

> **NOTE**
>
> NCLU commands are not currently available to configure IPv6 relays.

1. Edit the `/etc/default/isc-dhcp-relay6` file to add the upstream and downstream interfaces. In the example below, the SVI is vlan1, and the interfaces are swp51 and swp52.

```
cumulus@switch:$ sudo nano /etc/default/isc-dhcp-relay6
SERVERS=" -u 2001:db8:100::2%swp51 -u 2001:db8:100::2%swp52"
INTF_CMD="-l vlan1"
```

2. Enable, then restart the `dhcrelay6` service so that the configuration persists between reboots:

```
cumulus@switch:~$ sudo systemctl enable dhcrelay6.service
cumulus@switch:~$ sudo systemctl restart dhcrelay6.service
```

To see the status of the IPv6 DHCP relay, use the `systemctl status`

`dhcrelay6.service` command:

```
cumulus@switch:~$ sudo systemctl status dhcrelay6.service
● dhcrelay6.service - DHCPv6 Relay Agent Daemon
   Loaded: loaded (/lib/systemd/system/dhcrelay6.service;
disabled)
   Active: active (running) since Fri 2016-12-02 21:00:26
UTC; 1s ago
      Docs: man:dhcrelay(8)
Main PID: 6152 (dhcrelay)
   CGroup: /system.slice/dhcrelay6.service
           └─6152 /usr/sbin/dhcrelay -6 --nl -d -q -l vlan1
-u 2001:db8:100::2 swp51 -u 2001:db8:100::2 swp52
```

Configure Multiple DHCP Relays

Cumulus Linux supports multiple DHCP relay daemons on a switch to enable relaying of packets from different bridges to different upstream interfaces.

To configure multiple DHCP relay daemons on a switch:

1. Create a configuration file in the `/etc/default` directory for each DHCP relay daemon. Use the naming scheme `isc-dhcp-relay-<dhcp-name>` for IPv4 or `isc-dhcp-relay6-<dhcp-name>` for IPv6. An example configuration file for IPv4 is shown below:

```
# Defaults for isc-dhcp-relay initscript
# sourced by /etc/init.d/isc-dhcp-relay
# installed at /etc/default/isc-dhcp-relay by the maintainer
scripts

#
# This is a POSIX shell fragment
#

# What servers should the DHCP relay forward requests to?
SERVERS="102.0.0.2"
# On what interfaces should the DHCP relay (dhrelay) serve
DHCP requests?
# Always include the interface towards the DHCP server.
# This variable requires a -i for each interface configured
above.
# This will be used in the actual dhcrelay command
# For example, "-i eth0 -i eth1"
INTF_CMD="-i swp2s2 -i swp2s3"

# Additional options that are passed to the DHCP relay daemon?
OPTIONS=""
```

An example configuration file for IPv6 is shown below:

```
# Defaults for isc-dhcp-relay6 initscript
# sourced by /etc/init.d/isc-dhcp-relay6
# installed at /etc/default/isc-dhcp-relay6 by the maintainer
scripts

#
# This is a POSIX shell fragment
#

# Specify upstream and downstream interfaces
# For example, "-u eth0 -l swp1"
INTF_CMD=""

# Additional options that are passed to the DHCP relay daemon?
OPTIONS=""
```

2. Run the following command to start a `dhcrelay` instance, where `<dhcp-name>` is the instance name or number.

```
cumulus@switch:~$ sudo systemctl start dhcrelay@<dhcp-name>
```

Configure a DHCP Relay with VRR

The configuration procedure for DHCP relay with VRR is the same as documented above.

> **NOTE**
>
> The DHCP relay must run on the SVI and not on the -v0 interface.

Troubleshooting

If you are experiencing issues with DHCP relay, you can check if there is a problem with `systemd:`

- For IPv4, run this command:

```
cumulus@switch:~$  /usr/sbin/dhcrelay -4 -i <interface-facing-
host> <ip-address-dhcp-server> -i <interface-facing-dhcp-
server>
```

- For IPv6, run this command:

```
cumulus@switch:~$  /usr/sbin/dhcrelay -6 -l <interface-facing-
host> -u <ip-address-hcp-server>%<interface-facing-dhcp-
server>
```

For example:

```
cumulus@switch:~$ /usr/sbin/dhcrelay -4 -i vlan1 172.16.1.102
-i swp51
cumulus@switch:~$ /usr/sbin/dhcrelay -6 -l vlan1 -u
2001:db8:100::2%swp51
```

The above commands manually activate the DHCP relay process and they do not persist when you reboot the switch.

To see how DHCP relay is working on your switch, run the `journalctl` command:

```
cumulus@switch:~$ sudo journalctl -l -n 20 | grep dhcrelay
Dec 05 20:58:55 leaf01 dhcrelay[6152]: sending upstream swp52
Dec 05 20:58:55 leaf01 dhcrelay[6152]: sending upstream swp51
Dec 05 20:58:55 leaf01 dhcrelay[6152]: Relaying Reply to
fe80::4638:39ff:fe00:3 port 546 down.
```

```
Dec 05 20:58:55 leaf01 dhcrelay[6152]: Relaying Reply to
fe80::4638:39ff:fe00:3 port 546 down.
Dec 05 21:03:55 leaf01 dhcrelay[6152]: Relaying Renew from
fe80::4638:39ff:fe00:3 port 546 going up.
Dec 05 21:03:55 leaf01 dhcrelay[6152]: sending upstream swp52
Dec 05 21:03:55 leaf01 dhcrelay[6152]: sending upstream swp51
Dec 05 21:03:55 leaf01 dhcrelay[6152]: Relaying Reply to
fe80::4638:39ff:fe00:3 port 546 down.
Dec 05 21:03:55 leaf01 dhcrelay[6152]: Relaying Reply to
fe80::4638:39ff:fe00:3 port 546 down.
```

To specify a time period with the `journalctl` command, use the `--since` flag:

```
cumulus@switch:~$ sudo journalctl -l --since "2 minutes ago" |
grep dhcrelay
Dec 05 21:08:55 leaf01 dhcrelay[6152]: Relaying Renew from
fe80::4638:39ff:fe00:3 port 546 going up.
Dec 05 21:08:55 leaf01 dhcrelay[6152]: sending upstream swp52
Dec 05 21:08:55 leaf01 dhcrelay[6152]: sending upstream swp51
```

Configuration Errors

If you configure DHCP relays by editing the `/etc/default/isc-dhcp-relay` file manually instead of running NCLU commands, you might introduce configuration errors that can cause the switch to crash.

For example, if you see an error similar to the following, there might be a space between the DHCP server address and the interface used as the uplink.

```
Core was generated by /usr/sbin/dhcrelay --nl -d -i vx-40 -i
vlan100 10.0.0.4 -U 10.0.1.2  %vlan120.
Program terminated with signal SIGSEGV, Segmentation fault.
```

To resolve the issue, manually edit the `/etc/default/isc-dhcp-relay` file to remove the space, then run the `systemctl restart dhcrelay.service` command to restart the `dhcrelay` service and apply the configuration change.

Considerations

Interface Names Cannot Be Longer than 14 Characters

The `dhcrelay` command does not bind to an interface if the interface's name is longer than 14 characters. To work around this issue, change the interface

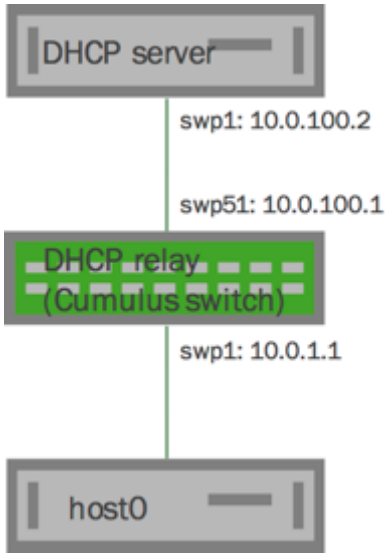
name to be 14 or fewer characters if `dhcrelay` is required to bind to it.

This is a known limitation in `dhcrelay`.

DHCP Servers

A DHCP Server automatically provides and assigns IP addresses and other network parameters to client devices. It relies on the Dynamic Host Configuration Protocol to respond to broadcast requests from clients.

This topic describes how to configure a DHCP server for IPv4 and IPv6. Configurations on the hosts, DHCP relay and DHCP server are provided using the following topology. The DHCP server is a switch running Cumulus Linux; however, the DHCP server can also be located on a dedicated server in your environment.



> **NOTE**
>
> The `dhcpd` and `dhcrelay` services are disabled by default. After you

finish configuring the DHCP relays and servers, you need to start those services. If you intend to run these services within a VRF, including the management VRF, follow these steps.

For information about DHCP relays, refer to DHCP Relays.

Configure the DHCP Server on Cumulus Linux Switches

To configure the DHCP server on a Cumulus Linux switch for IPv4 and IPv6, you need to edit the `/etc/dhcp/dhcp.conf` and `/etc/dhcp/dhcpd6.conf` configuration files. Sample configurations are provided as a starting point.

You must include two pools in the DHCP configuration files:

- Pool 1 is the subnet that includes the IP addresses of the interfaces on the DHCP server
- Pool 2 is the subnet that includes the IP addresses being assigned

Configure the IPv4 DHCP Server

In a text editor, edit the `/etc/dhcp/dhcpd.conf` file. Use following configuration as an example:

```
cumulus@switch:~$ cat /etc/dhcp/dhcpd.conf
ddns-update-style none;

default-lease-time 600;
max-lease-time 7200;

subnet 10.0.100.0 netmask 255.255.255.0 {
}
subnet 10.0.1.0 netmask 255.255.255.0 {
      range 10.0.1.50 10.0.1.60;
}
```

Edit the `/etc/default/isc-dhcp-server` configuration file so that the DHCP server launches when the system boots. Here is an example configuration:

```
cumulus@switch:~$ cat /etc/default/isc-dhcp-server
DHCPD_CONF="-cf /etc/dhcp/dhcpd.conf"

INTERFACES="swp1"
```

Enable and start the `dhcpd` service:

```
cumulus@switch:~$ sudo systemctl enable dhcpd.service
cumulus@switch:~$ sudo systemctl start dhcpd.service
```

Configure the IPv6 DHCP Server

In a text editor, edit the `/etc/dhcp/dhcpd6.conf` file. Use following configuration as an example:

```
cumulus@switch:~$ cat /etc/dhcp/dhcpd6.conf
ddns-update-style none;

default-lease-time 600;
max-lease-time 7200;

subnet6 2001:db8:100::/64 {
}
subnet6 2001:db8:1::/64 {
    range6 2001:db8:1::100 2001:db8:1::200;
}
```

Edit the `/etc/default/isc-dhcp-server6` file so that the DHCP server launches when the system boots. Here is an example configuration:

```
cumulus@switch:~$ cat /etc/default/isc-dhcp-server6
DHCPD_CONF="-cf /etc/dhcp/dhcpd6.conf"


INTERFACES="swp1"
```

Enable and start the `dhcpd6` service:

```
cumulus@switch:~$ sudo systemctl enable dhcpd6.service
cumulus@switch:~$ sudo systemctl start dhcpd6.service
```

Assign Port-based IP Addresses

You can assign an IP address and other DHCP options based on physical location or port regardless of MAC address to clients that are attached directly to the Cumulus Linux switch through a switch port. This is helpful when swapping out switches and servers; you can avoid the inconvenience of collecting the MAC address and sending it to the network administrator to modify the DHCP server configuration.

Edit the `/etc/dhcp/dhcpd.conf` file and add the interface name `ifname` to assign an IP address through DHCP. The following provides an example:

```
host myhost {
    ifname "swp1" ;
    fixed-address 10.10.10.10 ;
}
```

Troubleshooting

The DHCP server determines if a DHCP request is a relay or a non-relay DHCP request. You can run the following command to see the DHCP request:

```
cumulus@server02:~$ sudo tail /var/log/syslog | grep dhcpd
2016-12-05T19:03:35.379633+00:00 server02 dhcpd: Relay-forward
message from 2001:db8:101::1 port 547, link address
2001:db8:101::1, peer address fe80::4638:39ff:fe00:3
2016-12-05T19:03:35.380081+00:00 server02 dhcpd: Advertise NA:
address 2001:db8:1::110 to client with duid
00:01:00:01:1f:d8:75:3a:44:38:39:00:00:03 iaid = 956301315
valid for 600 seconds
2016-12-05T19:03:35.380470+00:00 server02 dhcpd: Sending Relay-
reply to 2001:db8:101::1 port 547
```

DHCP Snooping

DHCP snooping enables Cumulus Linux to act as a middle layer between the DHCP infrastructure and DHCP clients by scanning DHCP control packets and building an IP-MAC database. Cumulus Linux accepts DHCP offers from only trusted interfaces and can rate limit packets.

NOTE

- DHCP option 82 processing is not supported.
- DHCPv6 is supported on Broadcom switches only.

Configure DHCP Snooping

To configure DHCP snooping, you need to:

- Enable DHCP snooping on a VLAN.
- Add a trusted interface. Cumulus Linux allows DHCP offers from only trusted interfaces to prevent malicious DHCP servers from assigning IP addresses inside the network. The interface must be a member of the bridge specified.
- Set the rate limit for DHCP requests to avoid DoS attacks. The default value is 100 packets per second.

The following example commands show you how to configure DHCP

snooping for IPv4 and IPv6.

IPv4 IPv6

```
cumulus@leaf01:~$ net add bridge br0 dhcp-snoop vlan 100
cumulus@leaf01:~$ net add bridge br0 dhcp-snoop vlan 100
trust swp6
cumulus@leaf01:~$ net add bridge br0 dhcp-snoop vlan 100
rate-limit 50
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The NCLU commands save the configuration in the `/etc/dhcpsnoop/dhcp_snoop.json` file. For example:

```
{
  "bridge": [
    {
      "bridge_id": "br0",  <<< Bridge name
      "vlan": [
        {
          "vlan_id": 100,
          "snooping": 1,
          "rate_limit": 50,
```

```
            "ip_version": 6,
            "trusted_interface": [
              "swp6"
            ],
          }
        ]
      }
    ]
}
```

To remove all DHCP snooping configuration, run the `net del dhcp-snoop all` command. For example:

```
cumulus@leaf01:~$ net del dhcp-snoop all
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

When DHCP snooping detects a violation, the packet is dropped and a message is logged to the `/var/log/dhcpsnoop.log` file.

Show the DHCP Binding Table

To show the DHCP binding table, run the `net show dhcp-snoop table`

command for IPv4 or the `net show dhcp-snoop6 table` command for IPv6. The following example command shows the DHCP binding table for IPv4:

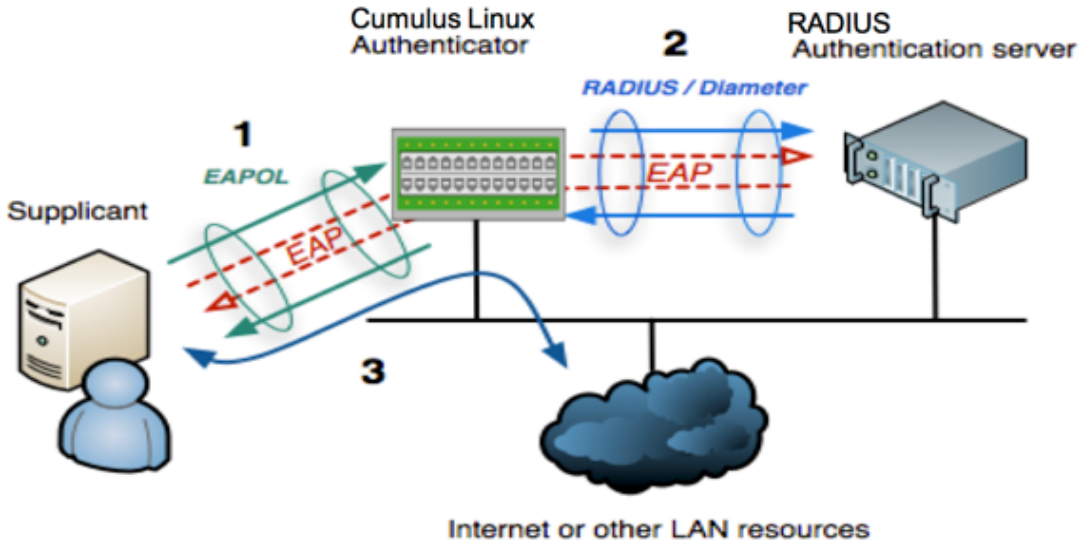
```
cumulus@leaf01:~$ net show dhcp-snoop table
Port VLAN IP        MAC               Lease State Bridge
---- ---- --------- ----------------- ----- ----- ------

swp5 1002 10.0.0.3  00:02:00:00:00:04 7200  ACK   br0

swp5 1000 10.0.1.3  00:02:00:00:00:04 7200  ACK   br0
```

802.1X Interfaces

The IEEE 802.1X protocol provides a method of authenticating a client (called a *supplicant*) over wired media. It also provides access for individual MAC addresses on a switch (called the *authenticator*) after those MAC addresses have been authenticated by an authentication server, typically a RADIUS (Remote Authentication Dial In User Service, defined by RFC 2865) server.

A Cumulus Linux switch acts as an intermediary between the clients connected to the wired ports and the authentication server, which is reachable over the existing network. EAPOL (Extensible Authentication Protocol (EAP) over LAN - EtherType value of 0x888E, defined by RFC 3748) operates on top of the data link layer; the switch uses EAPOL to communicate with supplicants connected to the switch ports.

Cumulus Linux implements 802.1X through the Debian `hostapd` package, which has been modified to provide the PAE (port access entity).



Supported Features

- 802.1X is supported on Broadcom-based switches (except the Hurricane2 switch). The Tomahawk, Tomahawk2, and Trident3 switch *must* be running in nonatomic mode.
- 802.1X is supported on physical interfaces only, such as swp1 or swp2s0 (bridged/access only and routed interfaces).
  - The interfaces cannot be part of a bond.
  - 802.1X is **not** supported on eth0.
- You can configure 802.1X interfaces for bridges in both VLAN-aware mode and traditional mode using the following features:
  - Parking VLAN.
  - Dynamic VLAN.
  - MAB (MAC-based authentication bypass).
- MAB, parking VLAN, and dynamic VLAN all require a bridge access port.
- In traditional bridge mode, parking VLANs and dynamic VLANs both

require the destination bridge to have a parking VLAN ID or dynamic VLAN ID tagged subinterface.

- When you enable or disable 802.1X on ports, `hostapd` reloads; however, existing authorized sessions do not reset.
- Changing the 802.1X interface, MAB, or parking VLAN settings do *not* reset existing authorized user ports. However, removing all 802.1X interfaces or changing any of the following RADIUS parameters restarts `hostapd`, which forces existing, authorized users to re-authenticate:
  - RADIUS server IP address, shared secret, authentication port or accounting port.
  - Parking VLAN ID.
  - MAB activation delay.
  - EAP reauthentication period.
- You can configure up to three RADIUS servers (in case of failover). However, do not use a Cumulus Linux switch as the RADIUS server.
- You can configure 802.1X interfaces with dynamic ACLs in VLAN-aware bridge mode only.
- 802.1X on Cumulus Linux has been tested with only a few `wpa_supplicant` (Debian), Windows 10 and Windows 7 supplicants.
- RADIUS authentication is supported with FreeRADIUS and Cisco ACS.
- 802.1X supports simple login and password, PEAP/MSCHAPv2 (Win7) and EAP-TLS (Debian).
- 802.1X supports RFC 5281 for EAP-TTLS, which provides more secure transport layer security.
- Mako template-based configurations are not supported.
- Cumulus Linux supports Multi Domain Authentication (MDA), where

802.1X is extended to allow authorization of multiple devices (a data and a voice device) on a single port and assign different VLANs to the devices based on authorization.

- MDA is enabled by default; however, you need to assign a tagged VLAN for voice devices (see Configure 802.1X Interfaces for a VLAN-aware Bridge).
- A maximum of four authorized devices (MAB + EAPOL) per port are supported.
- The 802.1X-enabled port must be a trunk port to allow tagged voice traffic from a phone; you cannot enable 802.1X on an access port.
- Only one untagged VLAN and one tagged VLAN is supported on the 802.1X enabled ports.
- Multiple MAB (non voice) devices on a port are supported for VLAN-aware bridges only. Authorization of multiple MAB devices for different VLANs is not supported.

Configure the RADIUS Server

Before you can authenticate with 802.1x on your switch, you must configure a RADIUS server somewhere in your network. Popular examples of commercial software with RADIUS capability include Cisco ISE and Aruba ClearPass (you can read our six-part blog series on campus design, which discusses configuring ClearPass and ISE in your network).

There are also open source versions of software supporting RADIUS such as PacketFence and FreeRADIUS. This section discusses how to add FreeRADIUS to a Debian server on your network.

> **NOTE**
>
> Do not use a Cumulus Linux switch as the RADIUS server.

To add FreeRADIUS on a Debian server, do the following:

```
root@radius:~# apt-get update
root@radius:~# apt-get install freeradius
```

When installed and configured, the FreeRADIUS server can serve Cumulus Linux running `hostapd` as a RADIUS client.

For more information, see the FreeRADIUS documentation.

Configure 802.1X Interfaces

All the 802.1X interfaces share the same RADIUS server settings. Make sure you configure the RADIUS server before you configure the 802.1X interfaces. See Configure the RADIUS Server above.

To configure an 802.1X interface, you need to set the following parameters, then enable 802.1X on the interface:

- The RADIUS accounting port, which defaults to *1813*.

- The RADIUS Server IPv4 or IPv6 address, which has no default, but is required. You can also specify a VRF.
- The RADIUS shared secret, which has no default, but is required.

Configure 802.1X Interfaces for a VLAN-aware Bridge

NCLU Commands | Linux Commands

NCLU handles all the 802.1X interface configuration, updating `hostapd` and other components so you do not have to manually modify configuration files.

1. Create a simple interface bridge configuration on the switch and add the switch ports that are members of the bridge. You can use glob syntax to add a range of interfaces. The MAB and parking VLAN configurations require interfaces to be bridge access ports. The VLAN-aware bridge must be named *bridge* and there can be only one VLAN-aware bridge on a switch.

```
cumulus@switch:~$ net add bridge bridge ports swp1-4
```

2. Add the 802.1X RADIUS server IP address and shared secret:

```
cumulus@switch:~$ net add dot1x radius server-ip 127.0.0.1
cumulus@switch:~$ net add dot1x radius shared-secret
mysecret
```

You can specify a VRF for outgoing RADIUS accounting and authorization packets. The following example specifies a VRF called turtle:

```
cumulus@switch:~$ net add dot1x radius server-ip
127.0.0.1 vrf turtle
cumulus@switch:~$ net add dot1x radius shared-secret
```

Configure 802.1X Interfaces for a Traditional Mode Bridge

NCLU Commands | Linux Commands

NCLU and `hostapd` might change traditional mode configurations on the `bridge-ports` line in the `/etc/network/interface` file by adding or deleting special 802.1X traditional mode `bridge-ports` configuration stanzas in `/etc/network/interfaces.d/`. The `source` configuration command in `/etc/network/interfaces` must include these special configuration filenames. It must include at least `source /etc/network/interfaces.d/*.intf` so that these files are sourced during an `ifreload`.

1. Create uplink ports. The following example uses bonds:

```
cumulus@switch:~$ net add bond bond1 bond slaves swp5-6
cumulus@switch:~$ net add bond bond2 bond slaves swp7-8
```

2. Create a traditional mode bridge configuration on the switch and add the switch ports that are members of the bridge. A traditional bridge cannot be named **** *bridge* as that name is reserved for the single VLAN-aware bridge on the switch. You can use glob syntax to add a range of interfaces.

```
cumulus@switch:~$ net add bridge bridge1 ports swp1-4
```

3. Create bridge associations with the parking VLAN ID and the dynamic VLAN IDs. In this example, *600* is used for the parking VLAN ID and 700 is used for the dynamic VLAN ID:

Configure the Linux Supplicants

A sample FreeRADIUS server configuration needs to contain the entries for users *host1* and *host2* on swp1 and swp2 for them to be placed in a VLAN.

```
host1 Cleartext-Password := "host1password"
host2 Cleartext-Password := "host2password"
```

After being configured, each supplicant needs the proper credentials:

```
user@host1:~# cat /etc/wpa_supplicant.conf


ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
eapol_version=2
ap_scan=0
network={
        key_mgmt=IEEE8021X
        eap=TTLS MD5
        identity="host1"
        anonymous_identity="host1"
        password="host1password"
        phase1="auth=MD5"
        eapol_flags=0
```

```
}
```

```
user@host2:~# cat /etc/wpa_supplicant.conf


ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
eapol_version=2
ap_scan=0
network={
        key_mgmt=IEEE8021X
        eap=TTLS MD5
        identity="host2"
        anonymous_identity="host2"
        password="host2password"
        phase1="auth=MD5"
        eapol_flags=0
  }
```

To test that a supplicant (client) can communicate with the Cumulus Linux Authenticator switch, install the wpasupplicant package:

```
root@radius:~# apt-get update
root@radius:~# apt-get install wpasupplicant
```

And run the following command from the supplicant:

```
root@host1:/home/cumulus# wpa_supplicant -c /etc/
wpa_supplicant.conf -D wired -i swp1
Successfully initialized wpa_supplicant
swp1: Associated with 01:80:c2:00:00:03
swp1: CTRL-EVENT-EAP-STARTED EAP authentication started
swp1: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=4
swp1: CTRL-EVENT-EAP-METHOD EAP vendor 0 method 4 (MD5) selected
swp1: CTRL-EVENT-EAP-SUCCESS EAP authentication completed
successfully
swp1: CTRL-EVENT-CONNECTED - Connection to 01:80:c2:00:00:03
compl
```

Or from another supplicant:

```
root@host2:/home/cumulus# wpa_supplicant -c /etc/
wpa_supplicant.conf -D wired -i swp1
```

```
Successfully initialized wpa_supplicant
swp1: Associated with 01:80:c2:00:00:03
swp1: CTRL-EVENT-EAP-STARTED EAP authentication started
swp1: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=4
swp1: CTRL-EVENT-EAP-METHOD EAP vendor 0 method 4 (MD5) selected
swp1: CTRL-EVENT-EAP-SUCCESS EAP authentication completed
successfully
swp1: CTRL-EVENT-CONNECTED - Connection to 01:80:c2:00:00:03
comp
```

Configure Accounting and Authentication Ports

You can configure the accounting and authentication ports in Cumulus Linux. The default values are *1813* for the accounting port and *1812* for the authentication port. You can also change the reauthentication period for Extensible Authentication Protocol (EAP). The period defaults to *0* (no re-authentication is performed by the switch).

To use different ports:

NCLU Commands | Linux Commands

The following example commands change:

- The authentication port to 2812
- The accounting port to 2813
- The reauthentication period for EAP to 86400

```
cumulus@switch:~$ net add dot1x radius authentication-port
2812
cumulus@switch:~$ net add dot1x radius accounting-port 2813
cumulus@switch:~$ net add dot1x eap-reauth-period 86400
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Configure MAC Authentication Bypass

MAC authentication bypass (MAB) enables bridge ports to allow devices to bypass authentication based on their MAC address. This is useful for devices that do not support PAE, such as printers or phones.

MAB must be configured on both the RADIUS server and the RADIUS client (the Cumulus Linux switch).

When using a VLAN-aware bridge, the switch port must be part of bridge named *bridge*.

To configure MAB:

NCLU Commands | Linux Commands

Enable a bridge port for MAB. The following example commands enable bridge port swp1 for MAB:

```
cumulus@switch:~$ net add interface swp1 dot1x mab
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Configure a Parking VLAN

If a non-authorized supplicant tries to communicate with the switch, you can route traffic from that device to a different VLAN and associate that VLAN with one of the switch ports to which the supplicant is attached.

For VLAN-aware bridges, the parking VLAN is assigned by manipulating the PVID of the switch port. For traditional mode bridges, Cumulus Linux identifies the bridge associated with the parking VLAN ID and moves the switch port into that bridge. If an appropriate bridge is not found for the move, the port remains in an unauthenticated state where no packets can be received or transmitted.

When using a VLAN-aware bridge, the switch port must be part of bridge named *bridge*.

NCLU Commands | Linux Commands

Run the following commands:

```
cumulus@switch:~$ net add dot1x parking-vlan-id 777
cumulus@switch:~$ net add interface swp1 dot1x parking-vlan
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

If the authentication for swp1 fails, the port is moved to the parking VLAN:

```
cumulus@switch:~$ net show dot1x interface swp1 details

Interface  MAC Address        Attribute
Value
---------  -----------------  ----------------------------
-----------------
swp1       00:02:00:00:00:08  Status Flags
[PARKED_VLAN]
                              Username
vlan60
                              Authentication Type
MD5
                              VLAN
777
                              Session Time (seconds)
24772
                              EAPOL Frames RX
9
```

Configure Dynamic VLAN Assignments

A common requirement for campus networks is to assign dynamic VLANs to specific users in combination with IEEE 802.1x. After authenticating a supplicant, the user is assigned a VLAN based on the RADIUS configuration.

For VLAN-aware bridges, the dynamic VLAN is assigned by manipulating the PVID of the switch port. For traditional mode bridges, Cumulus Linux identifies the bridge associated with the dynamic VLAN ID and moves the switch port into that bridge. If an appropriate bridge is not found for the move, the port remains in an unauthenticated state where no packets can be received or transmitted.

To enable dynamic VLAN assignment globally, where VLAN attributes sent from the RADIUS server are applied to the bridge:

NCLU Commands | Linux Commands

Run the following commands:

```
cumulus@switch:~$ net add dot1x dynamic-vlan
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

You can specify the `require` option in the command so that VLAN attributes are required. If VLAN attributes do not exist in the access response packet returned from the RADIUS server, the user is not authorized and has no connectivity. If the RADIUS server returns VLAN attributes but the user has an incorrect password, the user is placed in the parking VLAN (if you have configured parking VLAN).

```
cumulus@switch:~$ net add dot1x dynamic-vlan require
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The following example shows a typical RADIUS configuration (shown for FreeRADIUS, not typically configured or run on the Cumulus Linux device) for a user with dynamic VLAN assignment:

```
# # VLAN 100 Client Configuration for Freeradius RADIUS
Server.
# # This is not part of the CL configuration.
vlan100client Cleartext-Password := "client1password"
      Service-Type = Framed-User,
```

To disable dynamic VLAN assignment, where VLAN attributes sent from the RADIUS server are ignored and users are authenticated based on existing credentials:

NCLU Commands | Linux Commands

Run the `net del dot1x dynamic-vlan` command:

```
cumulus@switch:~$ net del dot1x dynamic-vlan
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

NOTE

Enabling or disabling dynamic VLAN assignment restarts `hostapd`, which forces existing, authorized users to re-authenticate.

Dynamic ACLs

In high-security campus environments where 802.1X interfaces are in use, you can implement network access control at the user (supplicant) level using *dynamic access control lists*, or DACLs. A *pre-auth ACL* permits some traffic to traverse the network before 802.1X authorization takes place, then a dynamic ACL can be applied for that supplicant that is specific to an

interface and the MAC address that was authorized (sometimes called a *station*).

Since DACLs restrict access to network resources at the user level, multiple users on the same VLAN can access different resources based on the policy provided by the RADIUS server. DACLs utilize NAS-Filter-Rule (RADIUS attribute 92), so you can configure them in your RADIUS server configuration and not on each switch.

The DACLs are also dynamically modified to fit the specific authenticating supplicant. For example, specific MAC addresses may be restricted to talk only to certain L3/L4 destinations.

DACLs work with Voice VLAN for phones (MDA).

NOTE

- You can configure DACLs for VLAN-aware bridges only.
- Port security (MAC address restrictions) cannot be used at the same time as DACLs.
- Cumulus Linux does not support configuring both Dynamic VLAN and DACLs on a given switch port at the same time.
- The source MAC address of the user gaining authorization in the `ebtables` filter replaces the `from any` source IPv4 address.

- Only a single destination port integer is supported; port ranges are not supported.
- Any IPv4 protocol is supported either by name or number as supported in the Cumulus Linux `ebtables` implementation.

How It Works

1. A supplicant sends packets over a network port. A pre-802.1X authorization ACL executes. You can manually create your own pre-auth ACL filter or just use the Cumulus Linux default (see below). There are no NCLU commands for creating the filter itself.
2. When `dot1x dynamic-acl` is enabled on an interface, Cumulus Linux installs the pre-auth ACL defaults for the port (once you execute `net commit`).
3. When a supplicant on the port tries to get 802.1X authorized, the RADIUS server may (or may not) send along some NAS-Filter-Rule attributes in the Access-Accept message.
4. If any filters are sent from the RADIUS server, Cumulus Linux applies them before the default pre-auth ACL.
5. If no filters are sent, Cumulus Linux leaves the defaults in place, and no special access is granted to the user.

The NAS-Filter-Rule Attribute

The NAS-Filter-Rule attribute is a string of one or more octets that contains filter rules in the IPFilterRule syntax defined by RFC 6733. The IPFilterRule filters **must** follow this format:

```
action dir proto from src to dst [options]
```

Keyword	Definition
`action`	*permit*: Allow packets that match the rule. *deny*: Drop packets that match the rule.
`dir`	Direction: *in* is from the terminal, *out* is to the terminal. Only the *in* direction is supported.
`proto`	An IP protocol specified by number. The `ip` keyword means any protocol will match. Only IPv4 ACLs are supported.
`src` / `dst`	Source and destination IP address/subnet mask, and optional ports.

The syntax for NAS-Filter-Rule attributes configured in the RADIUS server varies widely by RADIUS vendor. But the resulting format for these rules

contained in the Access-Accept must conform to the IPFilterRule syntax defined in by RFC 6733, Section 4.3, as mentioned above. When the Cumulus Linux switch gets these rules for a particular user, they are converted to `ebtables` rules using the actual user MAC address, and are then combined with the default pre-auth ACL rules.

The rules for the appropriate direction are evaluated in order, with the first matched rule terminating the evaluation. Each packet is evaluated once. If no rule matches, the packet is dropped if the last rule was a deny.

If these rules are invalid — for example, they contain contain port ranges or IPv6 addresses — the port does not get authorized and a log message is written to `/var/log/syslog`.

Get Started

To start applying a DACL to a port, configure the RADIUS server and client, then configure the port with the following:

- An untagged data VLAN
- The DACL and pre-auth ACL
- Optionally, a voice VLAN

Configure a Dynamic ACL

You configure DACLs on the RADIUS server on your network using the methods provided by the RADIUS software, then you enable it for one or more switch ports on a given switch. This section shows the configuration methods for the FreeRADIUS server.

Configure the RADIUS Server

On the RADIUS server, set the password for the RADIUS client (that is, the Cumulus Linux switch) in the `/etc/freeradius/3.0/clients.conf` file as follows, using the src IP address of the switch:

```
client leaf01 {
        ipaddr = 10.0.0.1
        secret = CumulusLinux!
}
```

Add the DACL configuration to the `/etc/freeradius/3.0/users` file. For example:

```
leaf01  Cleartext-Password := "CumulusLinux!"
        Service-Type = Framed-User,
        Tunnel-Type = VLAN,
        Tunnel-Medium-Type = "IEEE-802",
        Tunnel-Private-Group-ID = 222,
        NAS-Filter-Rule = "permit in udp from any to any 67",
        NAS-Filter-Rule = "permit in udp from any to 10.0.0.0/9
53",
        NAS-Filter-Rule = "permit in udp from any to 10.0.0.0/9
```

```
123",
        NAS-Filter-Rule = "permit in icmp from any to any",
        NAS-Filter-Rule = "permit in ip from any to
172.16.0.99",
        NAS-Filter-Rule = "permit in ip from any to
172.16.0.33",
        NAS-Filter-Rule = "permit in ip from any to
172.16.0.105",
        NAS-Filter-Rule = "permit in ip from any to
172.16.0.224",
        NAS-Filter-Rule = "permit in ip from any to
172.16.224.142",
        NAS-Filter-Rule = "permit in tcp from any to
172.16.224.0/9 8883",
        NAS-Filter-Rule = "deny in ip from any to any"
```

`ebtables` converts this to a temporary file on the switch called something like `/etc/cumulus/acl/policy.d/150_dot1x_dacl_swp2_000200000002.rules` (the filename is always prefaced with *150_*; default rules filenames are prefaced with *200_*). It looks like the following:

```
cumulus@switch:~$ cat /etc/cumulus/acl/policy.d/
150_dot1x_dacl_swp2_000200000002.rules
```

```
######## hostapd generated Dynamic ACL EBTABLES rule file
########
[ebtables]
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-protocol
UDP --ip-dport 67 -j mark --set-mark 2
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-protocol
UDP --ip-dport 67 -j ACCEPT
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
10.0.0.0/9 --ip-protocol UDP --ip-dport 53 -j mark --set-mark 2
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
10.0.0.0/9 --ip-protocol UDP --ip-dport 53 -j ACCEPT
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
10.0.0.0/9 --ip-protocol UDP --ip-dport 123 -j mark --set-mark 2
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
10.0.0.0/9 --ip-protocol UDP --ip-dport 123 -j ACCEPT
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
10.0.0.3 --ip-protocol ICMP -j mark --set-mark 2
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
10.0.0.3 --ip-protocol ICMP -j DROP
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
172.16.0.99 -j mark --set-mark 2
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
172.16.0.99 -j ACCEPT
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
```

```
172.16.131.99 -j mark --set-mark 2
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
172.16.131.99 -j ACCEPT
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
172.16.0.33 -j mark --set-mark 2
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
172.16.0.33 -j ACCEPT
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
172.16.131.105 -j mark --set-mark 2
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
172.16.131.105 -j ACCEPT
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
10.72.169.224 -j mark --set-mark 2
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
10.72.169.224 -j ACCEPT
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
10.72.168.142 -j mark --set-mark 2
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
10.72.168.142 -j ACCEPT
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
10.0.0.0/9 --ip-protocol TCP --ip-dport 8883 -j mark --set-mark
2
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
10.0.0.0/9 --ip-protocol TCP --ip-dport 8883 -j ACCEPT
```

```
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
10.0.0.0/9 --ip-protocol TCP --ip-dport 32768 -j mark --set-
mark 2
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  --ip-dst
10.0.0.0/9 --ip-protocol TCP --ip-dport 32768 -j ACCEPT
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  -j mark --set-
mark 2
-A FORWARD -i swp2 -s 00:02:00:00:00:02 -p IPV4  -j DROP
```

TIP

In the above rules file, the `--set-mark 2` option ensures that the nearly identical next rule gets installed in the dedicated TCAM slice for 802.1X.

Configure the RADIUS Client

The Cumulus Linux switch is the RADIUS client.

NCLU Commands | Linux Commands

Configure the Cumulus Linux switch as a RADIUS client using the `net add dot1x radius` command, and include your RADIUS server's IP address and secret:

```
cumulus@leaf01:~$ net add dot1x radius server-ip 10.0.0.1
cumulus@leaf01:~$ net add dot1x radius shared-secret
mysecret
```

Enable one or more switch ports for DACLs by running the `net add dot1x interface <INTERFACE> dot1x dynamic-acl` command. You can also enable MAC authentication bypass by including the `mab` option at the end of the command.

```
cumulus@leaf01:~$ net add interface swp1 dot1x dynamic-acl
[mab]
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

Pre-auth ACLs

A *pre-auth ACL* is a static ACL that is applied to **all** 802.1X dynamic ACL-

enabled ports by default. It provides some basic services that are available before 802.1X authorization occurs. The default pre-auth ACL in Cumulus Linux allows for DHCP and DNS to operate without authorizing the supplicant.

The default pre-auth ACL file is `/etc/cumulus/acl/policy.d/dot1x_preauth_dacl/default_preauth_dacl.rules`, which you can modify, or you can create your own. The default pre-auth ACL permits DHCP (using source port 68 and destination port 67) and DNS (using destination port 53) before 802.1X authorization. You configure pre-auth ACLs only with `ebtables` syntax.

```
cumulus@switch:~$ cat /etc/cumulus/acl/policy.d/
dot1x_preauth_dacl/default_preauth_dacl.rules
[ebtables]
-A FORWARD -p IPV4 --ip-protocol UDP --ip-dport 53 -j ACCEPT
-A INPUT -p IPV4 --ip-protocol UDP --ip-dport 67 --ip-sport 68
-j ACCEPT
```

The pre-auth ACL is always applied to dynamic ACL-enabled 802.1X ports, even after authentication has already completed for any clients on a given switch port.

WARNING

If you don't use the default pre-auth ACL and don't create your own, all traffic gets denied.

To create your own pre-auth ACL file, complete the following steps.

NCLU Commands | Linux Commands

Create the pre-auth ACL file as shown in **Linux Commands** below, then run the `net add dot1x default-dacl-preauth-filename <FILE>` command.

```
cumulus@switch:~$ net add dot1x default-dacl-preauth-
filename my_preauth_dacl.rules
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Troubleshooting

To see which interfaces are enabled for 802.1X, run the `net show dot1x status` command. The Interfaces line shows all 802.1X-enabled interfaces

while the Dynamic ACL Interfaces line shows only those 802.1X interfaces that are enabled for DACLs:

```
cumulus@switch:~$ net show dot1x status

Hostapd IEEE 802.11 AP and IEEE 802.1X/WPA/WPA2/EAP
Authenticator Daemon
Attribute                Value
-----------------------  ----------------
Current Status           active (running)
Reload Status            enabled
Interfaces               swp1 swp2
MAB Interfaces
Voice Interfaces
Parking VLAN Interfaces
Dynamic ACL Interfaces   swp2
Dynamic VLAN Status      Disabled
8021x ACL Rules          10 used/256 max
```

To see which interfaces have attempted authorization for DACLs, run `net show dot1x interface summary`:

```
cumulus@switch:~$ net show dot1x interface summary
```

```
Interface  MAC Address        Username  State
Authentication Type  MAB  VLAN  DACL Active
---------  -----------------  --------  ----------
------------------  ---  ----  -----------
swp1       00:02:00:00:00:01  host1     AUTHORIZED
MD5                 NO         NO
swp2       00:02:00:00:00:02  host2     AUTHORIZED
MD5                 NO         YES
```

To determine the name of the DACL rules file for an interface after it has been authorized and has received DACL rules, run `net show dot1x interface swp1 detail`. Look for the DACL Filename line:

```
cumulus@switch:~$ net show dot1x interface swp2 detail


Interface  MAC Address        Attribute
Value
---------  -----------------  ----------------------------
-----------------
swp2       00:02:00:00:00:01  Status Flags
[AUTHORIZED]
                              Username
host1
```

```
                            Authentication Type           MD5
                            VLAN
                            DACL Filename
150_dot1x_dacl_swp2_000200000002.rules
                            Session Time (seconds)        65
                            EAPOL Frames RX               3
                            EAPOL Frames TX               3
                            EAPOL Start Frames RX         1
                            EAPOL Logoff Frames RX        0
                            EAPOL Response ID Frames RX   1
```

To see which ACLs are applied to a given interface, run `net show dot1x interface <INTERFACE> applied-acls`, which is similar to the output of `cl-acltool -L eb | grep swp1`.

```
cumulus@switch:~$ net show dot1x interface swp2 applied-acls


swp2 EBTABLES ACLs
=======================
-p ! 802_1Q -s 0:2:0:0:0:2 -i swp2 -j mark --mark-set 0x2 --
mark-target ACCEPT, pcnt = 1 -- bcnt = 421
-p ! 802_1Q -s 0:2:0:0:0:2 -i swp2 -j ACCEPT , pcnt = 1 -- bcnt
```

```
= 421
-p IPv4 -i swp2 --ip-proto udp --ip-sport 68 --ip-dport 67 -j
mark --mark-set 0x2 --mark-target ACCEPT, pcnt = 0 -- bcnt = 0
-p IPv4 -i swp2 --ip-proto udp --ip-sport 68 --ip-dport 67 -j
ACCEPT , pcnt = 0 -- bcnt = 0
-p 0x888e -i swp2 -j mark --mark-set 0x2 --mark-target ACCEPT,
pcnt = 3 -- bcnt = 192
-p 0x888e -i swp2 -j police --set-mode pkt --set-rate 100 --set-
burst 100 , pcnt = 3 -- bcnt = 192
-i swp2 -j mark --mark-set 0x2 --mark-target ACCEPT, pcnt = 4
-- bcnt = 1684
-i swp2 -j DROP , pcnt = 4 -- bcnt = 1684
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-proto udp --ip-dport 67 -j
mark --mark-set 0x2 --mark-target ACCEPT, pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-proto udp --ip-dport 67 -j
ACCEPT , pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 10.0.0.0/9 --ip-proto
udp --ip-dport 53 -j mark --mark-set 0x2 --mark-target ACCEPT,
pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 10.0.0.0/9 --ip-proto
udp --ip-dport 53 -j ACCEPT , pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 10.0.0.0/9 --ip-proto
udp --ip-dport 123 -j mark --mark-set 0x2 --mark-target ACCEPT,
pcnt = 0 -- bcnt = 0
```

```
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 10.0.0.0/9 --ip-proto
udp --ip-dport 123 -j ACCEPT , pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-proto icmp -j mark --mark-
set 0x2 --mark-target ACCEPT, pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-proto icmp -j ACCEPT , pcnt
= 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 172.16.0.99 -j mark --
mark-set 0x2 --mark-target ACCEPT, pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 172.16.0.99 -j ACCEPT ,
pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 172.16.131.99 -j mark --
mark-set 0x2 --mark-target ACCEPT, pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 172.16.131.99 -j ACCEPT
, pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 172.16.0.33 -j mark --
mark-set 0x2 --mark-target ACCEPT, pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 172.16.0.33 -j ACCEPT ,
pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 172.16.131.105 -j mark
--mark-set 0x2 --mark-target ACCEPT, pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 172.16.131.105 -j
ACCEPT , pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 10.72.169.224 -j mark --
mark-set 0x2 --mark-target ACCEPT, pcnt = 0 -- bcnt = 0
```

```
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 10.72.169.224 -j ACCEPT
, pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 10.72.168.142 -j mark --
mark-set 0x2 --mark-target ACCEPT, pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 10.72.168.142 -j ACCEPT
, pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 10.0.0.0/9 --ip-proto
tcp --ip-dport 8883 -j mark --mark-set 0x2 --mark-target
ACCEPT, pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 10.0.0.0/9 --ip-proto
tcp --ip-dport 8883 -j ACCEPT , pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 10.0.0.0/9 --ip-proto
tcp --ip-dport 32768 -j mark --mark-set 0x2 --mark-target
ACCEPT, pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 --ip-dst 10.0.0.0/9 --ip-proto
tcp --ip-dport 32768 -j ACCEPT , pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 -j mark --mark-set 0x2 --mark-
target ACCEPT, pcnt = 0 -- bcnt = 0
-p IPv4 -s 0:2:0:0:0:2 -i swp2 -j DROP , pcnt = 0 -- bcnt = 0
-p IPv4 -i swp2 --ip-proto udp --ip-dport 53 -j mark --mark-set
0x2 --mark-target ACCEPT, pcnt = 0 -- bcnt = 0
-p IPv4 -i swp2 --ip-proto udp --ip-dport 53 -j ACCEPT , pcnt =
0 -- bcnt = 0
-i swp2 -j mark --mark-set 0x2 --mark-target ACCEPT, pcnt = 0
```

```
-- bcnt = 0
-i swp2 -j DROP , pcnt = 0 -- bcnt = 0
```

Configure MAC Addresses per Port

You can specify the maximum number of authenticated MAC addresses allowed on a port.

NCLU Commands | Linux Commands

Run the `net add dot1x max-number-stations <value>` command. You can specify any number between 0 and 255. The default value is 4.

```
cumulus@switch:~$ net add dot1x max-number-stations 10
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Configure EAP Requests from the Switch

Cumulus Linux provides the `send-eap-request-id` option, which you can use to trigger EAP packets to be sent from the host side of a connection. For example, this option is required in a configuration where a PC connected to a phone attempts to send EAP packets to the switch via the phone but the

PC does not receive a response from the switch (the phone might not be ready to forward packets to the switch after a reboot). Because the switch does not receive EAP packets, it attempts to authorize the PC with MAB instead of waiting for the packets. In this case, the PC might be placed into a parking VLAN to isolate it. To remove the PC from the parking VLAN, the switch needs to send an EAP request to the PC to trigger EAP.

To configure the switch send an EAP request, run these commands:

```
cumulus@switch:~$ net add dot1x send-eap-request-id
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

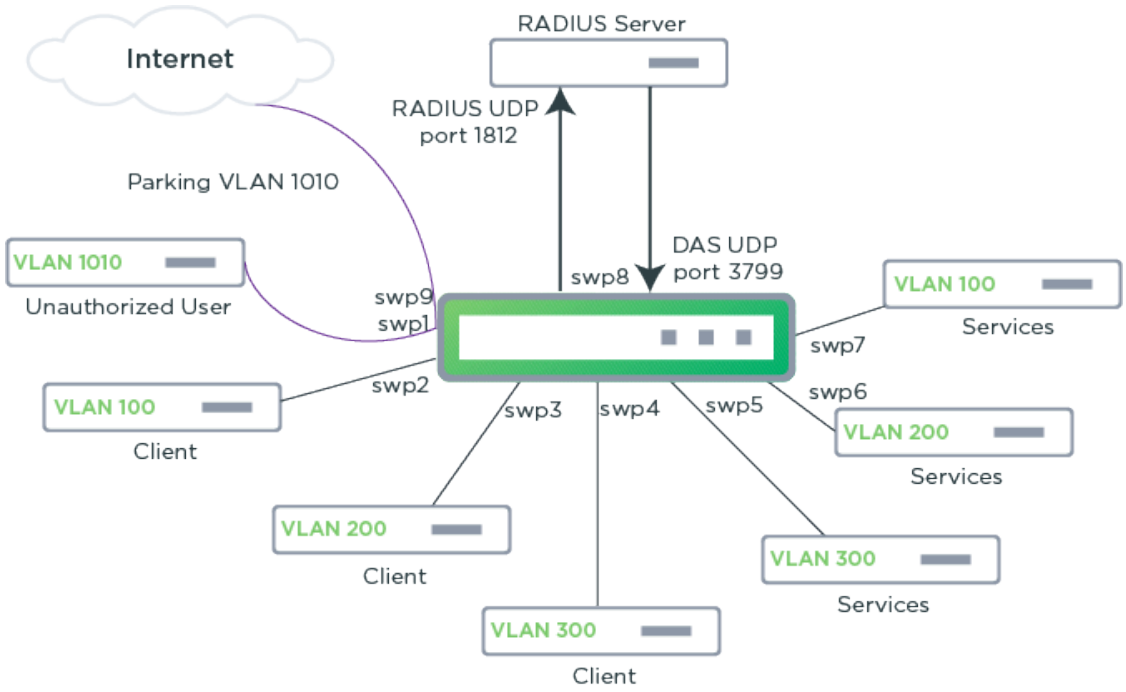
> **NOTE**
>
> Only run this command if MAB is configured on an interface.

> **NOTE**
>
> The PC might attempt 802.1X authorization through the bridged connection in the back of the phone before the phone completes MAB authorization. In this case, 802.1X authorization fails.

The `net del dot1x send-eap-request-id` command disables this feature.

RADIUS Change of Authorization and Disconnect Requests

Extensions to the RADIUS protocol (RFC 5176) enable the Cumulus Linux switch to act as a Dynamic Authorization Server (DAS) by listening for Change of Authorization (CoA) requests from the RADIUS server (Dynamic Authorization Client (DAC)) and taking action when needed, such as bouncing a port or terminating a user session. The IEEE 802.1x server (`hostapd`) running on Cumulus Linux has been adapted to handle these additional, unsolicited RADIUS requests.



Configure DAS

To configure DAS, provide the UDP port (3799 is the default port), the IP

address, and the secret key for the DAS client.

NCLU Commands | Linux Commands

The following example commands set the UDP port to the default port, the IP address of the DAS client to 10.0.2.228, and the secret key to myclientsecret:

```
cumulus@switch:~$ net add dot1x radius das-port default
cumulus@switch:~$ net add dot1x radius das-client-ip
10.0.2.228 das-client-secret mysecret123
cumulus@switch:~$ net commit
```

You can specify a VRF so that incoming RADIUS disconnect and CoA commands are received and acknowledged on the correct interface when VRF is configured. The following example specifies VRF turtle:

```
cumulus@switch:~$ net add dot1x radius das-port default
cumulus@switch:~$ net add dot1x radius das-client-ip
10.0.2.228 vrf turtle das-client-secret mysecret123
cumulus@switch:~$ net commit
```

You can configure up to four DAS clients to be authorized to send CoA commands. For example:

```
cumulus@switch:~$ net add dot1x radius das-port default
cumulus@switch:~$ net add dot1x radius das-client-ip
10.20.250.53 das-client-secret mysecret1
cumulus@switch:~$ net add dot1x radius das-client-ip
10.0.1.7 das-client-secret mysecret2
```

You can disable DAS in Cumulus Linux at any time by running the following commands:

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net del dot1x radius das-port
cumulus@switch:~$ net del dot1x radius das-client-ip
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Terminate a User Session

From the DAC, users can create a disconnect message using the `radclient` utility (included in the Debian `freeradius-utils` package) on the RADIUS server or other authorized client. A disconnect message is sent as an unsolicited RADIUS Disconnect-Request packet to the switch to terminate a user session and discard all associated session context. The Disconnect-Request packet is used when the RADIUS server wants to disconnect the user after the session has been accepted by the RADIUS Access-Accept packet.

This is an example of a disconnect message created using the radclient utility:

```
$ echo "Acct-Session-Id=D91FE8E51802097" > disconnect-packet.txt
```

```
$ ## OPTIONAL ## echo "User-Name=somebody" >> disconnect-
packet.txt
$ echo "Message-Authenticator=1" >> disconnect-packet.txt
$ echo "Event-Timestamp=1532974019" >> disconnect-packet.txt
# now send the packet with the radclient utility (from
freeradius-utils deb package)
$ cat disconnect-packet.txt | radclient -x 10.0.0.1:3799
disconnect myclientsecret
```

To prevent unauthorized servers from disconnecting users, the Disconnect-Request packet must include certain identification attributes (described below). For a session to be disconnected, all parameters must match their expected values at the switch. If the parameters do not match, the switch discards the Disconnect-Request packet and sends a Disconnect-NAK (negative acknowledgment message).

- The `Message-Authenticator` attribute is required.
- If the packet comes from a different source IP address than the one defined by `das-client-ip`, the session is not disconnected and the `hostapd` logs the debug message: `DAS: Drop message from unknown client.`
- The `Event-Timestamp` attribute is required. If `Event-Timestamp` in the packet is outside the time window, a debug message is shown in the `hostapd` logs: `DAS: Unacceptable Event-Timestamp (1532978602; local time 1532979367) in packet from 10.10.0.21:45263 - drop`

- If the `Acct-Session-Id` attribute is omitted, the `User-Name`attribute is used to find the session. If the `User-Name` attribute is omitted, the `Acct-Session-Id` attribute is used. If both the `User-Name` and the `Acct-Session-Id` attributes are supplied, they must match the username provided by the supplicant with the `Acct-Session-Id` provided. If neither are given or there is no match, a Disconnect-NAK message is returned to the RADIUS server with `Error-Cause "Session-Context-Not-Found"` and the following debug message is shown in the log:

```
RADIUS DAS: Acct-Session-Id match
RADIUS DAS: No matches remaining after User-Name check
hostapd_das_find_global_sta: checking ifname=swp2
RADIUS DAS: No matches remaining after Acct-Session-Id check
RADIUS DAS: No matching session found
DAS: Session not found for request from 10.10.0.1:58385
DAS: Reply to 10.10.0.1:58385
```

The following is an example of the Disconnect-Request packet received by the switch:

```
RADIUS Protocol
Code: Disconnect-Request (40)
Packet identifier: 0x4f (79)
```

```
Length: 53
Authenticator: c0e1fa75fdf594a1cfaf35151a43c6a7
Attribute Value Pairs
AVP: t=Acct-Session-Id(44) l=17 val=D91FE8E51802097
AVP: t=User-Name(1) l=10 val=somebody
AVP: t=Message-Authenticator(80) l=18
val=38cb3b6896623b4b7d32f116fa976cdc
AVP: t=Event-Timestamp(55) l=6 val=1532974019
AVP: t=NAS-IP-Address(4) l=6 val=10.0.0.1
```

Bounce a Port

You can create a CoA bounce-host-port message from the RADIUS server using the `radclient` utility (included in the Debian `freeradius-utils` package). The bounce port can cause a link flap on an authentication port, which triggers DHCP renegotiation from one or more hosts connected to the port.

The following is an example of a Cisco AVPair CoA bounce-host-port message sent from the radclient utility:

```
$ echo "Acct-Session-Id=D91FE8E51802097" > bounce-packet.txt
$ ## OPTIONAL ## echo "User-Name=somebody" >> bounce-packet.txt
```

```
$ echo "Message-Authenticator=1" >> bounce-packet.txt
$ echo "Event-Timestamp=1532974019" >> bounce-packet.txt
$ echo "cisco-avpair='subscriber:command=bounce-host-port' " >>
bounce-packet.txt
$ cat bounce-packet.txt | radclient -x 10.0.0.1:3799 coa
myclientsecret
```

The message received by the switch is:

```
RADIUS Protocol
Code: CoA-Request (43)
Packet identifier: 0x3a (58)
Length: 96
Authenticator: 6480d710802329269d5cae6a59bcfb59
Attribute Value Pairs
AVP: t=Acct-Session-Id(44) l=17 val=D91FE8E51802097
Type: 44
Length: 17
Acct-Session-Id: D91FE8E51802097
AVP: t=User-Name(1) l=10 val=somebody
Type: 1
Length: 10
User-Name: somebody
```

```
AVP: t=NAS-IP-Address(4) l=6 val=10.0.0.1
Type: 4
Length: 6
NAS-IP-Address: 10.0.0.1
AVP: t=Vendor-Specific(26) l=43 vnd=ciscoSystems(9)
Type: 26
Length: 43
Vendor ID: ciscoSystems (9)
VSA: t=Cisco-AVPair(1) l=37 val=subscriber:command=bounce-host-
port
Type: 1
Length: 37
Cisco-AVPair: subscriber:command=bounce-host-port
```

Configure the NAS IP Address

You can send the NAS IPv4 or IPv6 address in access request and accounting packets. You can only configure one NAS IP address on the switch, which is used for all interface authorizations.

To configure the NAS IP address, run the following commands:

NCLU Commands | Linux Commands

The following command example sets the NAS IP address to 10.0.0.1:

```
cumulus@switch:~$ net add dot1x radius nas-ip-address
10.0.0.1
```

To delete the NAS IP address, either run the NCLU `net del dot1x radius nas-ip-address` command or edit the `/etc/hostapd.conf` file.

Troubleshooting

To check connectivity between two supplicants, ping one host from the other:

```
root@host1:/home/cumulus# ping 198.51.100.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.604 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.552 ms
^C
--- 10.0.0.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.552/0.578/0
```

You can run `net show dot1x` with the following options for more data:

- `json` prints the command output in JSON format.
- `macs` displays MAC address information.
- `port-details` shows counters from the IEEE8021-PAE-MIB for ports.
- `radius-details` shows counters from the RADIUS-CLIENT MIB (RFC 2618) for ports.
- `status` displays the status of the daemon.

To check to see which MAC addresses have been authorized by RADIUS:

```
cumulus@switch:~$ net show dot1x macs
Interface     Attribute   Value
-----------   -------------  -----------------
swp1          MAC Addresses  00:02:00:00:00:01
swp2          No Data
swp3          No Data
swp4          No Data
```

To check the port detail counters:

```
cumulus@switch:~$ net show dot1x port-details

Interface  Attribute                                  Value
```

```
-----------  ----------------------------------------
---------
swp1         Mac Addresses
00:02:00:00:00:01
             authMultiSessionId
96703ADC82D77DF2
             connected_time                          182
             dot1xAuthEapolFramesRx                  3
             dot1xAuthEapolFramesTx                  3
             dot1xAuthEapolLogoffFramesRx            0
             dot1xAuthEapolReqFramesTx               2
             dot1xAuthEapolReqIdFramesTx             1
             dot1xAuthEapolRespFramesRx              2
             dot1xAuthEapolRespIdFramesRx            1
             dot1xAuthEapolStartFramesRx             1
             dot1xAuthInvalidEapolFramesRx           0
             dot1xAuthLastEapolFrameSource
00:02:00:00:00:01
             dot1xAuthLastEapolFrameVersion          2
             dot1xAuthPaeState                       5
             dot1xAuthQuietPeriod                    60
             dot1xAuthReAuthEnabled                  FALSE
             dot1xAuthReAuthPeriod                   0
             dot1xAuthServerTimeout                  30
```

```
            dot1xAuthSessionAuthenticMethod           1
            dot1xAuthSessionId
1B50FE8939FD9F5E
            dot1xAuthSessionTerminateCause            999
            dot1xAuthSessionTime                      182
            dot1xAuthSessionUserName                  testing
            dot1xPaePortProtocolVersion               2
            last_eap_type_as                          4 (MD5)
            last_eap_type_sta                         4 (MD5)
```

To check RADIUS counters:

```
cumulus@switch:~$ net show dot1x radius-details swp1

Interface  Attribute                                 Value
-----------  ----------------------------------------
---------
swp1       radiusAccClientRequests                    1
           radiusAccClientResponses                   1
           radiusAccClientServerPortNumber            1813
           radiusAccServerAddress
127.0.0.1
           radiusAuthClientAccessAccepts              1
```

```
            radiusAuthClientAccessChallenges            1
            radiusAuthClientAccessRejects               0
            radiusAuthClientAccessRequests              0
            radiusAuthClientServerPortNumber            1812
            radiusAuthServerAddress
127.0.0.1
            radiusAuthServerIndex                       1


...
```

You can also check logging with the `journalctl` command:

```
cumulus@switch:~$ sudo journalctl -f -u hostapd
Apr 19 22:17:11 switch hostapd[12462]: swp1: interface state
UNINITIALIZED->ENABLED
Apr 19 22:17:11 switch hostapd[12462]: swp1: AP-ENABLED
Apr 19 22:17:11 switch hostapd[12462]: Reading rule file /etc/
cumulus/acl/policy.d/00control_ps ...
Apr 19 22:17:11 switch hostapd[12462]: Processing rules in file
/etc/cumulus/acl/policy.d/00...
Apr 19 22:17:12 switch hostapd[12462]: Reading rule file /etc/
cumulus/acl/policy.d/100_dot1x...
Apr 19 22:17:12 switch hostapd[12462]: Processing rules in file
```

```
/etc/cumulus/acl/policy.d/ ..
Apr 19 22:17:12 switch hostapd[12462]: Reading rule file /etc/
cumulus/acl/policy.d/99control
Apr 19 22:17:12 switch hostapd[12462]: Processing rules in file
/etc/cumulus/acl/policy.d/99
Apr 19 22:17:12 switch hostapd[12462]: Installing acl policy
Apr 19 22:17:12 switch hostapd[12462]: done.
```

You can perform more advanced troubleshooting with the following commands.

To increase the debug level in `hostapd`, copy over the `hostapd` service file, then add *-d*, *-dd* or *-ddd* to the `ExecStart` line in the `hostapd.service` file:

```
cumulus@switch:~$ cp /lib/systemd/system/hostapd.service /etc/
systemd/system/hostapd.service
cumulus@switch:~$ sudo nano /etc/systemd/system/hostapd.service
...
ExecStart=/usr/sbin/hostapd -ddd -c /etc/hostapd.conf
...
```

To watch debugs with `journalctl` as supplicants attempt to connect:

```
cumulus@switch:~$ sudo journalctl -n 1000  -u hostapd      #
see the last 1000 lines of hostapd debug logging
cumulus@switch:~$ sudo journalctl -f -u hostapd            #
continuous tail of the hostapd daemon debug logging
```

To check ACL rules in `/etc/cumulus/acl/policy.d/100_dot1x_swpX.rules` before and after a supplicant attempts to authenticate:

```
cumulus@switch:~$ sudo cl-acltool -L eb | grep swpXX
cumulus@switch:~$ sudo cl-netstat | grep swpXX           # look
at interface counters
```

To check `tc` rules in `/var/lib/hostapd/acl/tc_swpX.rules` with:

```
cumulus@switch:~$ sudo tc -s filter show dev swpXX parent 1:
cumulus@switch:~$ sudo tc -s filter show dev swpXX parent ffff:
```

Related Information

Campus design feature setup (6-part blog series)

Prescriptive Topology Manager - PTM

In data center topologies, right cabling is a time-consuming endeavor and is error prone. Prescriptive Topology Manager (PTM) is a dynamic cabling verification tool to help detect and eliminate such errors. It takes a Graphviz-DOT specified network cabling plan (something many operators already generate), stored in a `topology.dot` file, and couples it with runtime information derived from LLDP to verify that the cabling matches the specification. The check is performed on every link transition on each node in the network.

You can customize the `topology.dot` file to control `ptmd` at both the global/network level and the node/port level.

PTM runs as a daemon, named `ptmd`.

For more information, see `man ptmd(8)`.

Supported Features

- Topology verification using LLDP. `ptmd` creates a client connection to the LLDP daemon, `lldpd`, and retrieves the neighbor relationship between the nodes/ports in the network and compares them against the prescribed topology specified in the `topology.dot` file.
- Only physical interfaces, such as swp1 or eth0, are currently supported. Cumulus Linux does not support specifying virtual interfaces, such as

bonds or subinterfaces, such as eth0.200 in the topology file.

- Forwarding path failure detection using Bidirectional Forwarding Detection (BFD); however, demand mode is not supported. For more information on how BFD operates in Cumulus Linux, read the Bidirectional Forwarding Detection - BFD chapter and read `man ptmd(8)`.
- Integration with FRRouting (PTM to FRRouting notification).
- Client management: `ptmd` creates an abstract named socket `/var/run/ptmd.socket` on startup. Other applications can connect to this socket to receive notifications and send commands.
- Event notifications: see Scripts below.
- User configuration via a `topology.dot` file; see below.

Configure PTM

`ptmd` verifies the physical network topology against a DOT-specified network graph file, `/etc/ptm.d/topology.dot`.

PTM supports undirected graphs.

At startup, `ptmd` connects to `lldpd`, the LLDP daemon, over a Unix socket and retrieves the neighbor name and port information. It then compares the retrieved port information with the configuration information that it read from the topology file. If there is a match, it is a PASS, else it is a FAIL.

> **NOTE**

PTM performs its LLDP neighbor check using the PortID ifname TLV information.

ptmd Scripts

`ptmd` executes scripts at `/etc/ptm.d/if-topo-pass` and `/etc/ptm.d/if-topo-fail`for each interface that goes through a change and runs `if-topo-pass` when an LLDP or BFD check passes or `if-topo-fails` when the check fails. The scripts receive an argument string that is the result of the `ptmctl` command, described in the `ptmd` commands below.

Modify these default scripts as needed.

Configuration Parameters

You can configure `ptmd` parameters in the topology file. The parameters are classified as host-only, global, per-port/node and templates.

Host-only Parameters

Host-only parameters apply to the entire host on which PTM is running. You can include the `hostnametype` host-only parameter, which specifies if PTM uses only the host name (`hostname`) or the fully-qualified domain name (`fqdn`) while looking for the `self-node` in the graph file. For example, in the

graph file below PTM ignores the FQDN and only looks for *switch04* because that is the host name of the switch on which it is running:

> **TIP**
>
> Wrap the hostname in double quotes; for example, `"www.example.com"` to prevent `ptmd` from failing.
>
> To avoid errors when starting the `ptmd` process, make sure that `/etc/hosts` and `/etc/hostname` both reflect the hostname you are using in the `topology.dot` file.

```
graph G {
          hostnametype="hostname"
          BFD="upMinTx=150,requiredMinRx=250"
          "cumulus":"swp44" --
"switch04.cumulusnetworks.com":"swp20"
          "cumulus":"swp46" --
"switch04.cumulusnetworks.com":"swp22"
}
```

In this next example, PTM compares using the FQDN and looks for

switch05.cumulusnetworks.com, which is the FQDN of the switch ion which it is running:

```
graph G {
          hostnametype="fqdn"
          "cumulus":"swp44" --
"switch05.cumulusnetworks.com":"swp20"
          "cumulus":"swp46" --
"switch05.cumulusnetworks.com":"swp22"
}
```

Global Parameters

Global parameters apply to every port listed in the topology file. There are two global parameters: LLDP and BFD. LLDP is enabled by default; if no keyword is present, default values are used for all ports. However, BFD is disabled if no keyword is present, unless there is a per-port override configured. For example:

```
graph G {
          LLDP=""
          BFD="upMinTx=150,requiredMinRx=250,afi=both"
          "cumulus":"swp44" -- "qct-ly2-04":"swp20"
          "cumulus":"swp46" -- "qct-ly2-04":"swp22"
```

```
}
```

Per-port Parameters

Per-port parameters provide finer-grained control at the port level. These parameters override any global or compiled defaults. For example:

```
graph G {
          LLDP=""
          BFD="upMinTx=300,requiredMinRx=100"
          "cumulus":"swp44" -- "qct-ly2-04":"swp20"
[BFD="upMinTx=150,requiredMinRx=250,afi=both"]
          "cumulus":"swp46" -- "qct-ly2-04":"swp22"
}
```

Templates

Templates provide flexibility in choosing different parameter combinations and applying them to a given port. A template instructs `ptmd` to reference a named parameter string instead of a default one. There are two parameter strings `ptmd` supports:

- `bfdtmpl` specifies a custom parameter tuple for BFD.
- `lldptmpl` specifies a custom parameter tuple for LLDP.

For example:

```
graph G {
          LLDP=""
          BFD="upMinTx=300,requiredMinRx=100"
          BFD1="upMinTx=200,requiredMinRx=200"
          BFD2="upMinTx=100,requiredMinRx=300"
          LLDP1="match_type=ifname"
          LLDP2="match_type=portdescr"
          "cumulus":"swp44" -- "qct-ly2-04":"swp20"
[BFD="bfdtmpl=BFD1", LLDP="lldptmpl=LLDP1"]
          "cumulus":"swp46" -- "qct-ly2-04":"swp22"
[BFD="bfdtmpl=BFD2", LLDP="lldptmpl=LLDP2"]
          "cumulus":"swp46" -- "qct-ly2-04":"swp22"
}
```

In this template, LLDP1 and LLDP2 are templates for LLDP parameters. BFD1 and BFD2 are templates for BFD parameters.

Supported BFD and LLDP Parameters

`ptmd` supports the following BFD parameters:

- `upMinTx` is the minimum transmit interval, which defaults to *300ms*, specified in milliseconds.
- `requiredMinRx` is the minimum interval between received BFD packets, which defaults to *300ms*, specified in milliseconds.

- `detectMult` is the detect multiplier, which defaults to *3*, and can be any non-zero value.
- `afi` is the address family to be supported for the edge. The address family must be one of the following:
  - *v4*: BFD sessions will be built for only IPv4 connected peer. This is the default value.
  - *v6*: BFD sessions will be built for only IPv6 connected peer.
  - *both*: BFD sessions will be built for both IPv4 and IPv6 connected peers.

The following is an example of a topology with BFD applied at the port level:

```
graph G {
          "cumulus-1":"swp44" -- "cumulus-2":"swp20"
[BFD="upMinTx=300,requiredMinRx=100,afi=v6"]
          "cumulus-1":"swp46" -- "cumulus-2":"swp22"
[BFD="detectMult=4"]
}
```

`ptmd` supports the following LLDP parameters:

- `match_type`, which defaults to the interface name (`ifname`), but can accept a port description (`portdescr`) instead if you want `lldpd` to compare the topology against the port description instead of the interface name. You can set this parameter globally or at the per-port level.

- `match_hostname`, which defaults to the host name (`hostname`), but enables PTM to match the topology using the fully-qualified domain name (`fqdn`) supplied by LLDP.

The following is an example of a topology with LLDP applied at the port level:

```
graph G {
          "cumulus-1":"swp44" -- "cumulus-2":"swp20"
[LLDP="match_hostname=fqdn"]
          "cumulus-1":"swp46" -- "cumulus-2":"swp22"
[LLDP="match_type=portdescr"]
}
```

NOTE

When you specify `match_hostname=fqdn`, `ptmd` will match the entire FQDN, (*cumulus-2.domain.com* in the example below). If you do not specify anything for `match_hostname`, `ptmd` matches based on hostname only, (*cumulus-3* below), and ignores the rest of the URL:

```
graph G {
```

```
          "cumulus-1":"swp44" --
"cumulus-2.domain.com":"swp20"
[LLDP="match_hostname=fqdn"]
          "cumulus-1":"swp46" -- "cumulus-3":"swp22"
[LLDP="match_type=portdescr"]
}
```

Bidirectional Forwarding Detection (BFD)

BFD provides low overhead and rapid detection of failures in the paths between two network devices. It provides a unified mechanism for link detection over all media and protocol layers. Use BFD to detect failures for IPv4 and IPv6 single or multihop paths between any two network devices, including unidirectional path failure detection. For information about configuring BFD using PTM, see BFD.

Check Link State with FRRouting

The FRRouting routing suite enables additional checks to ensure that routing adjacencies are formed only on links that have connectivity conformant to the specification, as determined by `ptmd`.

> **NOTE**
>
> You only need to do this to check link state; you do not need to enable PTM to determine BFD status.

When the global `ptm-enable` option is enabled, every interface has an implied `ptm-enable` line in the configuration stanza in the interfaces file.

To enable the global `ptm-enable` option, run the following FRRouting command:

```
cumulus@switch:~$ sudo vtysh


switch# configure terminal
switch(config)# ptm-enable
switch(config)# end
switch# write memory
switch# exit
cumulus@switch:~$
```

To disable the checks, delete the `ptm-enable` parameter from the interface:

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net del interface swp51 ptm-enable
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

If you need to re-enable PTM for that interface:

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add interface swp51 ptm-enable
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

With PTM enabled on an interface, the `zebra` daemon connects to `ptmd` over a Unix socket. Any time there is a change of status for an interface, `ptmd` sends notifications to `zebra`. Zebra maintains a `ptm-status` flag per interface and evaluates routing adjacency based on this flag. To check the per-interface `ptm-status`:

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net show interface swp1


Interface swp1 is up, line protocol is up
  Link ups:       0    last: (never)
  Link downs:     0    last: (never)
  PTM status: disabled
  vrf: Default-IP-Routing-Table
  index 3 metric 0 mtu 1550
  flags: <UP,BROADCAST,RUNNING,MULTICAST>
  HWaddr: c4:54:44:bd:01:41
```

ptmd Service Commands

PTM sends client notifications in CSV format.

To start or restart the `ptmd` service, run the following command. The `topology.dot` file must be present for the service to start.

```
cumulus@switch:~$ sudo systemctl start|restart|force-reload
ptmd.service
```

To instruct `ptmd` to read the `topology.dot` file again to apply the new

configuration to the running state without restarting:

```
cumulus@switch:~$ sudo systemctl reload ptmd.service
```

To stop the `ptmd` service:

```
cumulus@switch:~$ sudo systemctl stop ptmd.service
```

To retrieve the current running state of `ptmd`:

```
cumulus@switch:~$ sudo systemctl status ptmd.service
```

ptmctl Commands

`ptmctl` is a client of `ptmd` that retrieves the operational state of the ports configured on the switch and information about BFD sessions from `ptmd`. `ptmctl` parses the CSV notifications sent by `ptmd`. See `man ptmctl` for more information.

ptmctl Examples

The examples below contain the following keywords in the output of the `cbl status` column:

cbl status Keyword	Definition
pass	The interface is defined in the topology file, LLDP information is received on the interface, and the LLDP information for the interface matches the information in the topology file.
fail	The interface is defined in the topology file, LLDP information is received on the interface, and the LLDP information for the interface does not match the information in the topology file.
N/A	The interface is defined in the topology file, but no LLDP information is received on the interface. The interface might be down or disconnected, or the neighbor is not sending LLDP packets. The `N/A` and `fail` status might indicate a wiring problem to investigate. The `N/A` status is not shown when you use the `-l` option with `ptmctl`; only interfaces that are receiving LLDP information are shown.

For basic output, use `ptmctl` without any options:

```
cumulus@switch:~$ sudo ptmctl

------------------------------------------------------------
port  cbl     BFD     BFD                  BFD    BFD
      status  status  peer                 local  type
------------------------------------------------------------
swp1  pass    pass    11.0.0.2             N/A    singlehop
swp2  pass    N/A     N/A                  N/A    N/A
swp3  pass    N/A     N/A                  N/A    N/A
```

For more detailed output, use the `-d` option:

```
cumulus@switch:~$ sudo ptmctl -d

------------------------------------------------------------------------------
port  cbl    exp     act      sysname  portID  portDescr
match  last    BFD   BFD
      status nbr     nbr
on     upd     Type  state
------------------------------------------------------------------------------
swp45 pass   h1:swp1 h1:swp1  h1       swp1    swp1
IfName 5m: 5s  N/A   N/A
swp46 fail   h2:swp1 h2:swp1  h2       swp1    swp1
```

```
IfName 5m: 5s  N/A   N/A


#continuation of the output
------------------------------------------------------------------------------
BFD   BFD       det_mult  tx_timeout  rx_timeout
echo_tx_timeout  echo_rx_timeout  max_hop_cnt
peer  DownDiag
------------------------------------------------------------------------------
N/A   N/A       N/A       N/A         N/A         N/
A              N/A              N/A
N/A   N/A       N/A       N/A         N/A         N/
A              N/A              N/A
```

To return information on active BFD sessions `ptmd` is tracking, use the `-b` option:

```
cumulus@switch:~$ sudo ptmctl -b


------------------------------------------------------------
port  peer        state  local         type       diag


------------------------------------------------------------
swp1  11.0.0.2    Up     N/A           singlehop  N/A
```

```
N/A   12.12.12.1  Up     12.12.12.4    multihop   N/A
```

To return LLDP information, use the `-l` option. It returns only the active neighbors currently being tracked by `ptmd`.

```
cumulus@switch:~$ sudo ptmctl -l

------------------------------------------
port  sysname  portID  port   match  last
                       descr  on     upd
------------------------------------------
swp45 h1       swp1    swp1   IfName 5m:59s
swp46 h2       swp1    swp1   IfName 5m:59s
```

To return detailed information on active BFD sessions `ptmd` is tracking, use the `-b` and `-d` option (results are for an IPv6-connected peer):

```
cumulus@switch:~$ sudo ptmctl -b -d

------------------------------------------------------------------------------
port  peer                state  local  type       diag  det
```

```
tx_timeout  rx_timeout
                                                                mult
-------------------------------------------------------------------------------
swp1  fe80::202:ff:fe00:1  Up     N/A    singlehop  N/A   3
300         900
swp1  3101:abc:bcad::2     Up     N/A    singlehop  N/A   3
300         900


#continuation of output
-------------------------------------------------------------------
echo        echo        max      rx_ctrl  tx_ctrl  rx_echo
tx_echo
tx_timeout  rx_timeout  hop_cnt
-------------------------------------------------------------------
0           0           N/A      187172   185986   0        0
0           0           N/A      501      533      0        0
```

ptmctl Error Outputs

If there are errors in the topology file or there is no session, PTM returns appropriate outputs. Typical error strings are:

```
Topology file error [/etc/ptm.d/topology.dot] [cannot find node
```

```
cumulus] -
please check /var/log/ptmd.log for more info

Topology file error [/etc/ptm.d/topology.dot] [cannot open file
(errno 2)] -
please check /var/log/ptmd.log for more info

No Hostname/MgmtIP found [Check LLDPD daemon status] -
please check /var/log/ptmd.log for more info

No BFD sessions . Check connections

No LLDP ports detected. Check connections

Unsupported command
```

For example:

```
cumulus@switch:~$ sudo ptmctl
------------------------------------------------------------------------
cmd         error
------------------------------------------------------------------------
get-status  Topology file error [/etc/ptm.d/topology.dot]
```

```
            [cannot open file (errno 2)] - please check /var/
log/ptmd.log
            for more info
```

TIP

If you encounter errors with the `topology.dot` file, you can use `dot` (included in the Graphviz package) to validate the syntax of the topology file.

Open the topology file with Graphviz to ensure that it is readable and that the file format is correct.

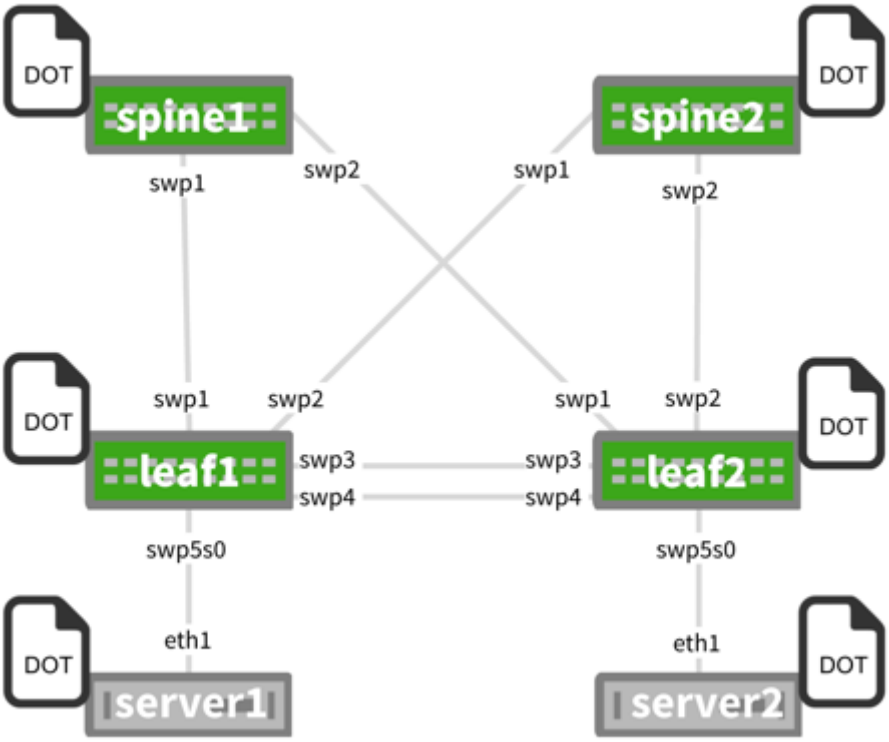
If you edit `topology.dot` file from a Windows system, be sure to double check the file formatting; there might be extra characters that keep the graph from working correctly.

Basic Topology Example

This is a basic example DOT file and its corresponding topology diagram. Use the same `topology.dot` file on all switches and do not split the file per device; this allows for easy automation by pushing/pulling the same exact

file on each device.

```
graph G {
    "spine1":"swp1" -- "leaf1":"swp1";
    "spine1":"swp2" -- "leaf2":"swp1";
    "spine2":"swp1" -- "leaf1":"swp2";
    "spine2":"swp2" -- "leaf2":"swp2";
    "leaf1":"swp3" -- "leaf2":"swp3";
    "leaf1":"swp4" -- "leaf2":"swp4";
    "leaf1":"swp5s0" -- "server1":"eth1";
    "leaf2":"swp5s0" -- "server2":"eth1";
}
```



Considerations

ptmd in Incorrect Failure State while Zebra Interface Is Enabled

When `ptmd` is incorrectly in a failure state and the Zebra interface is enabled, PIF BGP sessions do not establish the route, but the subinterface on top of it does establish routes.

If the subinterface is configured on the physical interface and the physical interface is incorrectly marked as being in a PTM FAIL state, routes on the physical interface are not processed in FRR, but the subinterface is working.

Cannot Use Commas in PortDescr

If an LLDP neighbor advertises a `PortDescr` that contains commas, `ptmctl -d` splits the string on the commas and misplaces its components in other columns. Do not use commas in your port descriptions.

Related Information

- Bidirectional Forwarding Detection (BFD)
- Graphviz
- LLDP on Wikipedia
- PTMd GitHub repo

Port Security

Port security is a layer 2 traffic control feature that enables you to manage network access from end-users. Use port security to:

- Limit port access to specific MAC addresses so that the port does not forward ingress traffic from source addresses that are not defined.
- Limit port access to only the first learned MAC address on the port (sticky MAC) so that the device with that MAC address has full bandwidth. You can provide a timeout so that the MAC address on that port no longer has access after a specified time.
- Limit port access to a specific number of MAC addresses.

You can specify what action to take when there is a port security violation (drop packets or put the port into ADMIN down state) and add a timeout for the action to take effect.

> **NOTE**
>
> Layer 2 interfaces in trunk or access mode are currently supported. However, interfaces in a bond are *not* supported.

Configure MAC Address Options

To limit port access to a specific MAC address, run the following commands.

The example commands configure swp1 to allow access to MAC address 00:02:00:00:00:05:

```
cumulus@switch:~$ net add interface swp1 port-security allowed-
mac 00:02:00:00:00:05
```

You can specify only one MAC address with the NCLU command. To specify multiple MAC addresses, set the `interface.<port>.port_security.static_mac` parameter in the `/etc/cumulus/switchd.d/port_security.conf` file. See Configure Port Security Manually below.

To enable sticky MAC on a port, where the first learned MAC address on the port is the only MAC address allowed, run the following commands.

You can add a timeout value so that after the time specified, the MAC address ages out and no longer has access to the port. The default aging timeout value is 1800 seconds. You can specify a value between 0 and 3600 seconds.

The example commands enable sticky MAC on interface swp1, set the timeout value to 2000 seconds, and enable aging.

```
cumulus@switch:~$ net add interface swp1 port-security sticky-
mac
```

```
cumulus@switch:~$ net add interface swp1 port-security sticky-
mac timeout 2000
cumulus@switch:~$ net add interface swp1 port-security sticky-
mac aging
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

To limit the number of MAC addresses that are allowed to access a port, run the following commands. You can specify a number between 0 and 512. The default is 32.

The example commands configure swp1 to limit access to 40 MAC addresses:

```
cumulus@switch:~$ net add interface swp1 port-security mac-
limit 40
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Configure Security Violation Actions

You can configure the action you want to take when there is a security violation on a port:

- `shutdown` puts a port into ADMIN down state.
- `restrict` drops packets. When packets are dropped, Cumulus Linux sends a log message.

You can also set a timeout value between 0 and 3600 seconds for the action to take effect. The default is 1800 seconds.

The following example commands put swp1 into ADMIN down state when there is a security violation and set the timeout value to 3600 seconds:

```
cumulus@switch:~$ net add interface swp1 port-security
violation shutdown
cumulus@switch:~$ net add interface swp1 port-security
violation timeout 3600
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Enable Port Security Settings

After you configure the port security settings to suit your needs, you can enable security on a port with the following commands:

```
cumulus@switch:~$ net add interface swp1 port-security
cumulus@switch:~$ net pending
```

```
cumulus@switch:~$ net commit
```

To disable port security on a port, run the `net del interface <interface> port-security` command.

Configure Port Security Manually

You can edit the `/etc/cumulus/switchd.d/port_security.conf` file manually to configure port security instead of running the NCLU commands shown above. This procedure is useful if you use configuration scripts.

Add the configuration settings you want to use to the `/etc/cumulus/switchd.d/port_security.conf` file, then restart `switchd` to apply the changes.

Setting	Description
`interface.<port>.port_security.enable`	1 enables security on the port. 0 disables security on the port.
`interface.<port>.port_security.mac_maximum`	The maximum number of MAC addresses allowed to access the port. You can specify a number between 0 and 512. The default is 32.
`interface.<port>.port_security.static_mac`	The specific MAC addresses allowed to access the port. You

Setting	Description
	can specify multiple MAC addresses. Separate each MAC address with a space.
`interface.<port>.port_security.sticky_mac`	1 enables sticky MAC, where the first learned MAC address on the port is the only MAC address allowed. 0 disables sticky MAC.
`interface.<port>.port_security.sticky_timeout`	The time period after which the first learned MAC address ages out and no longer has access to the port. The default aging timeout value is 1800 seconds. You can specify a value between 0 and 3600 seconds.
`interface.<port>.port_security.sticky_aging`	1 enables sticky MAC aging. 0 disables sticky MAC aging.
`interface.<port>.port_security.violation_mode`	The violation mode: 0 (shutdown) puts a port into ADMIN down state. 1 (restrict) drops packets.
`interface.<port>.port_security.violation_timeout`	The number of seconds after which the violation mode times out. You can specify a value between 0 and 3600 seconds. The default value is 1800 seconds.

An example `/etc/cumulus/switchd.d/port_security.conf` configuration file is shown here:

```
cumulus@switch:~$ sudo nano /etc/cumulus/switchd.d/
port_security.conf
interface.swp1.port_security.enable = 1
interface.swp1.port_security.mac_limit = 32
interface.swp1.port_security.static_mac = 00:02:00:00:00:05
00:02:00:00:00:06
interface.swp1.port_security.sticky_mac = 1
interface.swp1.port_security.sticky_timeout = 2000
interface.swp1.port_security.sticky_aging = 1
interface.swp1.port_security.violation_mode = 0
interface.swp1.port_security.violation_timeout = 3600
...
```

Show Port Security Configuration

To show port security settings for all ports:

```
cumulus@switch:~$ net show port-security
Interface  Port security  MAC limit  Sticky MAC  Sticky MAC
aging  Sticky MAC timeout  Violation mode  Timeout
---------  -------------  ---------  ----------
----------------  ------------------  --------------  -------
swp1       ENABLED        40         ENABLED
```

```
ENABLED           2000                Shutdown        3600
swp2       Disabled       NA          NA
NA                NA                  Restrict        1800
swp3       Disabled       NA          NA
NA                NA                  Restrict        1800
swp4       Disabled       NA          NA
NA                NA                  Restrict        1800
swp5       Disabled       NA          NA
NA                NA                  Restrict        1800
swp6       Disabled       NA          NA
NA                NA                  Restrict        1800
...
```

To show port security settings for a specific port:

```
cumulus@switch:~$ net show port-security swp1
Interface           swp1
Port security       Enabled
Mac limit           40
Sticky mac          ENABLED
Sticky MAC aging    Enabled
Sticky MAC timeout  1440
Violation mode      Shutdown
```

```
Violation timeout   3600
Mac addresses
00:02:00:00:00:05
00:02:00:00:00:06
```

Layer 2

This section describes layer 2 configuration, such as Ethernet bridging, bonding, spanning tree protocol, multi-chassis link aggregation (MLAG), link layer discovery protocol (LLDP), LACP bypass, virtual router redundancy (VRR) and IGMP and MLD snooping.

Link Layer Discovery Protocol

The `lldpd` daemon implements the IEEE802.1AB (Link Layer Discovery Protocol, or LLDP) standard. LLDP shows you which ports are neighbors of a given port. By default, `lldpd` runs as a daemon and starts at system boot. `lldpd` command line arguments are placed in `/etc/default/lldpd`. All `lldpd` configuration options are saved in `/etc/lldpd.conf` or under `/etc/lldpd.d/`.

For more details on the command line arguments and configuration options, see `man lldpd(8)`.

`lldpd` supports CDP (Cisco Discovery Protocol, v1 and v2) and logs by default into `/var/log/daemon.log` with an `lldpd` prefix.

You can use the `lldpcli` CLI tool to query the `lldpd` daemon for neighbors, statistics, and other running configuration information. See `man lldpcli(8)` for details.

Configure LLDP

You configure `lldpd` settings in `/etc/lldpd.conf` or `/etc/lldpd.d/`.

Here is an example persistent configuration:

```
cumulus@switch:~$ sudo cat /etc/lldpd.conf
configure lldp tx-interval 40
```

```
configure lldp tx-hold 3
configure system interface pattern *,!eth0,swp*
```

The last line in the example above shows that LLDP is disabled on eth0. To disable LLDP on a single port, edit the `/etc/default/lldpd` file. This file specifies the default options to present to the `lldpd` service when it starts. The following example uses the `-I` option to disable LLDP on swp43:

```
cumulus@switch:~$ sudo nano /etc/default/lldpd

# Add "-x" to DAEMON_ARGS to start SNMP subagent
# Enable CDP by default
DAEMON_ARGS="-c -I *, !swp43"
```

`lldpd` has two timers defined by the `tx-interval` setting that affect each switch port:

- The first timer catches any port-related changes.
- The second is a system-based refresh timer on each port that looks for other changes like hostname. This timer uses the `tx-interval` value multiplied by 20.

`lldpd` logs to `/var/log/daemon.log` with the *lldpd* prefix:

```
cumulus@switch:~$ sudo tail -f /var/log/daemon.log  | grep lldp
Aug  7 17:26:17 switch lldpd[1712]: unable to get system name
Aug  7 17:26:17 switch lldpd[1712]: unable to get system name
Aug  7 17:26:17 switch lldpcli[1711]: lldpd should resume
operations
Aug  7 17:26:32 switch lldpd[1805]: NET-SNMP version 5.4.3
AgentX subagent connected
```

Example lldpcli Commands

To show all neighbors on all ports and interfaces:

```
cumulus@switch:~$ sudo lldpcli show neighbors
-------------------------------------------------------------------------------
LLDP neighbors:
-------------------------------------------------------------------------------
Interface:    eth0, via: LLDP, RID: 1, Time: 0 day, 17:38:08
  Chassis:
    ChassisID:    mac 08:9e:01:e9:66:5a
    SysName:      PIONEERMS22
    SysDescr:     Cumulus Linux version 4.1.0 running on quanta
lb9
    MgmtIP:       192.168.0.22
```

```
    Capability:   Bridge, on
    Capability:   Router, on
  Port:
    PortID:       ifname swp47
    PortDescr:    swp47
-------------------------------------------------------------------------------
Interface:    swp1, via: LLDP, RID: 10, Time: 0 day, 17:08:27
  Chassis:
    ChassisID:    mac 00:01:00:00:09:00
    SysName:      MSP-1
    SysDescr:     Cumulus Linux version 4.1.0 running on QEMU
Standard PC (i440FX + PIIX, 1996)
    MgmtIP:       192.0.2.9
    MgmtIP:       fe80::201:ff:fe00:900
    Capability:   Bridge, off
    Capability:   Router, on
  Port:
    PortID:       ifname swp1
    PortDescr:    swp1
-------------------------------------------------------------------------------
Interface:    swp2, via: LLDP, RID: 10, Time: 0 day, 17:08:27
  Chassis:
    ChassisID:    mac 00:01:00:00:09:00
    SysName:      MSP-1
```

```
    SysDescr:     Cumulus Linux version 4.1.0 running on QEMU
Standard PC (i440FX + PIIX, 1996)
    MgmtIP:       192.0.2.9
    MgmtIP:       fe80::201:ff:fe00:900
    Capability:   Bridge, off
    Capability:   Router, on
  Port:
    PortID:       ifname swp2
    PortDescr:    swp2
-------------------------------------------------------------------------------
Interface:    swp3, via: LLDP, RID: 11, Time: 0 day, 17:08:27
  Chassis:
    ChassisID:    mac 00:01:00:00:0a:00
    SysName:      MSP-2
    SysDescr:     Cumulus Linux version 4.1.0 running on QEMU
Standard PC (i440FX + PIIX, 1996)
    MgmtIP:       192.0.2.10
    MgmtIP:       fe80::201:ff:fe00:a00
    Capability:   Bridge, off
    Capability:   Router, on
  Port:
    PortID:       ifname swp1
    PortDescr:    swp1
-------------------------------------------------------------------------------
```

```
Interface:    swp4, via: LLDP, RID: 11, Time: 0 day, 17:08:27
  Chassis:
    ChassisID:    mac 00:01:00:00:0a:00
    SysName:      MSP-2
    SysDescr:     Cumulus Linux version 4.1.0 running on QEMU
Standard PC (i440FX + PIIX, 1996)
    MgmtIP:       192.0.2.10
    MgmtIP:       fe80::201:ff:fe00:a00
    Capability:   Bridge, off
    Capability:   Router, on
  Port:
    PortID:       ifname swp2
    PortDescr:    swp2
------------------------------------------------------------------------------
Interface:    swp49s1, via: LLDP, RID: 9, Time: 0 day, 16:55:00
  Chassis:
    ChassisID:    mac 00:01:00:00:0c:00
    SysName:      TORC-1-2
    SysDescr:     Cumulus Linux version 4.1.0 running on QEMU
Standard PC (i440FX + PIIX, 1996)
    MgmtIP:       192.0.2.12
    MgmtIP:       fe80::201:ff:fe00:c00
    Capability:   Bridge, on
    Capability:   Router, on
```

```
  Port:
    PortID:       ifname swp6
    PortDescr:    swp6
-------------------------------------------------------------------------------
Interface:    swp49s0, via: LLDP, RID: 9, Time: 0 day, 16:55:00
  Chassis:
    ChassisID:    mac 00:01:00:00:0c:00
    SysName:      TORC-1-2
    SysDescr:     Cumulus Linux version 4.1.0 running on QEMU
Standard PC (i440FX + PIIX, 1996)
    MgmtIP:       192.0.2.12
    MgmtIP:       fe80::201:ff:fe00:c00
    Capability:   Bridge, on
    Capability:   Router, on
  Port:
    PortID:       ifname swp5
    PortDescr:    swp5
-------------------------------------------------------------------------------
```

To show `lldpd` statistics for all ports:

```
cumulus@switch:~$ sudo lldpcli show statistics
-------------------------------------------------------------------------
```

```
LLDP statistics:
------------------------------------------------------------------------
Interface:    eth0
  Transmitted:  9423
  Received:     17634
  Discarded:    0
  Unrecognized: 0
  Ageout:       10
  Inserted:     20
  Deleted:      10
------------------------------------------------------------------------
Interface:    swp1
  Transmitted:  9423
  Received:     6264
  Discarded:    0
  Unrecognized: 0
  Ageout:       0
  Inserted:     2
  Deleted:      0
------------------------------------------------------------------------
Interface:    swp2
  Transmitted:  9423
  Received:     6264
  Discarded:    0
```

```
  Unrecognized: 0
  Ageout:       0
  Inserted:     2
  Deleted:      0
------------------------------------------------------------------------
Interface:    swp3
  Transmitted:  9423
  Received:     6265
  Discarded:    0
  Unrecognized: 0
  Ageout:       0
  Inserted:     2
  Deleted:      0
------------------------------------------------------------------------
...
```

To show `lldpd` statistics summary for all ports:

```
cumulus@switch:~$ sudo lldpcli show statistics summary
------------------------------------------------------------------------
LLDP Global statistics:
------------------------------------------------------------------------
Summary of stats:
```

```
  Transmitted:  648186
  Received:     437557
  Discarded:    0
  Unrecognized: 0
  Ageout:       10
  Inserted:     38
  Deleted:      10
```

To show the `lldpd` running configuration:

```
cumulus@switch:~$ sudo lldpcli show running-configuration
-------------------------------------------------------------------------
Global configuration:
-------------------------------------------------------------------------
Configuration:
  Transmit delay: 30
  Transmit hold: 4
  Receive mode: no
  Pattern for management addresses: (none)
  Interface pattern: (none)
  Interface pattern blacklist: (none)
  Interface pattern for chassis ID: (none)
  Override description with: (none)
```

```
  Override platform with: Linux
  Override system name with: (none)
  Advertise version: yes
  Update interface descriptions: no
  Promiscuous mode on managed interfaces: no
  Disable LLDP-MED inventory: yes
  LLDP-MED fast start mechanism: yes
  LLDP-MED fast start interval: 1
  Source MAC for LLDP frames on bond slaves: local
  Portid TLV Subtype for lldp frames: ifname
-----------------------------------------------------------------------
```

▼ Runtime Configuration (Advanced)

Enable the SNMP Subagent in LLDP

LLDP does not enable the SNMP subagent by default. You need to edit `/etc/default/lldpd` and enable the `-x` option.

```
cumulus@switch:~$ sudo nano /etc/default/lldpd


# Add "-x" to DAEMON_ARGS to start SNMP subagent
```

```
# Enable CDP by default

DAEMON_ARGS="-c"
```

Considerations

Annex E (and hence Annex D) of IEEE802.1AB (lldp) is not supported.

Related Information

- GitHub - lldpd project
- Wikipedia - Link Layer Discovery Protocol

Voice VLAN

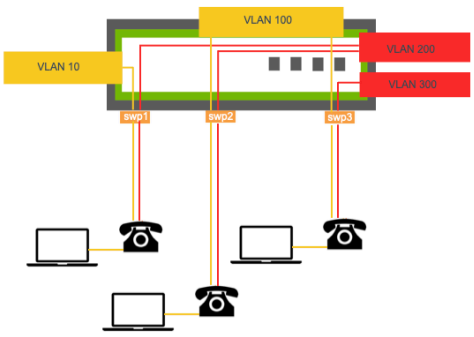
In Cumulus Linux, a *voice VLAN* is a VLAN dedicated to voice traffic on a switch port. Voice VLAN is part of a trunk port with two VLANs that comprises either of the following:

- Native VLAN, which carries both data and voice traffic.
- Voice VLAN, which carries the voice traffic, and a data or native VLAN, which carries the data traffic in a trunk port.

The voice traffic is an 802.1q-tagged packet with a VLAN ID (that might or might not be 0) and an 802.1p (3-bit layer 2 COS) with a specific value (typically 5 is assigned for voice traffic).

Data traffic is always untagged.

Example Configuration

	In this example configuration: • swp1 data traffic traverses the native VLAN of the bridge and the voice traffic traverses VLAN 200 • swp2 data traffic traverses

	VLAN 100 and the voice traffic traverses VLAN 200 • swp3 data traffic traverses VLAN 100 and voice traffic traverses VLAN 300

To configure the topology shown above:

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add bridge bridge ports swp1-3
cumulus@switch:~$ net add bridge bridge vids 10,100,200,300
cumulus@switch:~$ net add bridge bridge pvid 10
cumulus@switch:~$ net add interface swp1 bridge voice-vlan
200
cumulus@switch:~$ net add interface swp2 bridge voice-vlan
200 data-vlan 100
cumulus@switch:~$ net add interface swp3 bridge voice-vlan
300 data-vlan 100
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Troubleshooting

To show the bridge VIDs, run the `net show bridge vlan` command:

```
cumulus@switch:~$ net show bridge vlan

Interface      VLAN  Flags
-----------  ------  --------------------
swp1             10  PVID, Egress Untagged
                200
swp2            100  PVID, Egress Untagged
                200
swp3            100  PVID, Egress Untagged
                300
```

To obtain MAC address information, run the NCLU `net show bridge macs` command or the Linux `sudo brctl showmacs <bridge>` command. For example:

```
cumulus@switch:~$ net show bridge macs

VLAN      Master    Interface    MAC
TunnelDest  State      Flags    LastSeen
--------  --------  -----------  -----------------
-------------  ---------  -------  ----------
untagged  bridge    bridge
08:00:27:d5:00:93                permanent          00:13:54
```

```
untagged  bridge    swp1
08:00:27:6a:ad:da                permanent           00:13:54
untagged  bridge    swp2
08:00:27:e3:0c:a7                permanent           00:13:54
untagged  bridge    swp3
08:00:27:9e:98:86                permanent           00:13:54
```

To capture LLDP information, check `syslog` or use `tcpdump` on an interface.

Considerations

- A static voice VLAN configuration overwrites the existing configuration for the switch port.
- Removing the `bridge-vids` or `bridge-pvid` configuration from a voice VLAN does not remove the VLAN from the bridge.
- Configuring voice VLAN with NCLU does not configure `lldpd` in Cumulus Linux; LLDP-MED does not provide data and voice VLAN information. You can configure LLDP-MED for each interface in a new file in `/etc/lldp.d`. In the following example, the file is called `/etc/lldpd.d/voice_vlan.conf`:

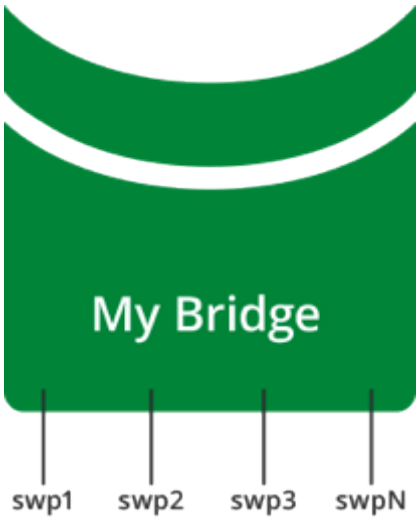
```
cumulus@switch:~$ sudo nano /etc/lldpd.d/voice_vlan.conf
configure ports swp1 med policy application voice tagged vlan
200 priority voice dscp 46
configure ports swp2 med policy application voice tagged vlan
100 priority voice dscp 46
configure ports swp3 med policy application voice tagged vlan
300 priority voice dscp 46
```

You can also use the `lldpcli` command to configure an LLDP-MED network policy. However, `lldpcli` commands do not persist across switch reboots.

Ethernet Bridging - VLANs

Ethernet bridges enable hosts to communicate through layer 2 by connecting all of the physical and logical interfaces in the system into a single layer 2 domain. The bridge is a logical interface with a MAC address and an MTU (maximum transmission unit). The bridge MTU is the minimum MTU among all its members. By default, the bridge's MAC address is the MAC address of the first port in the `bridge-ports` list. The bridge can also be assigned an IP address, as discussed below.

> **NOTE**
>
> Bridge members can be individual physical interfaces, bonds, or logical interfaces that traverse an 802.1Q VLAN trunk.



TIP

Consider using *VLAN-aware mode* bridges instead of *traditional mode* bridges. The bridge driver in Cumulus Linux is capable of VLAN filtering, which allows for configurations that are similar to incumbent network devices. For a comparison of traditional and VLAN-aware modes, read this knowledge base article.

NOTE

- Cumulus Linux does not put all ports into a bridge by default.
- You can configure both VLAN-aware and traditional mode bridges on the same network in Cumulus Linux; however you cannot have more than one VLAN-aware bridge on a given switch.

Create a VLAN-aware Bridge

To create a VLAN-aware bridge, see VLAN-aware Bridge Mode.

Create a Traditional Mode Bridge

To create a traditional mode bridge, see Traditional Bridge Mode.

Bridge MAC Addresses

The MAC address for a frame is learned when the frame enters the bridge through an interface. The MAC address is recorded in the bridge table and the bridge forwards the frame to its intended destination by looking up the destination MAC address. The MAC entry is then maintained for a period of time defined by the `bridge-ageing` configuration option. If the frame is seen with the same source MAC address before the MAC entry age is exceeded, the MAC entry age is refreshed; if the MAC entry age is exceeded, the MAC address is deleted from the bridge table.

The following example output shows a MAC address table for the bridge:

```
cumulus@switch:~$ net show bridge macs
VLAN      Master    Interface    MAC
TunnelDest  State      Flags    LastSeen
--------  --------  -----------  -----------------
------------  ---------  -------  -----------------
untagged  bridge    swp1
44:38:39:00:00:03                                  00:00:15
```

```
untagged  bridge    swp1
44:38:39:00:00:04                permanent           20 days,
01:14:03
```

By default, Cumulus Linux stores MAC addresses in the Ethernet switching table for 1800 seconds (30 minutes). To change the amount of time MAC addresses are stored in the table, configure *bridge ageing*.

The following example commands set MAC address ageing to 600 seconds.

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add bridge bridge ageing 600
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Configure a Switch Virtual Interface (SVI)

Bridges can be included as part of a routing topology after being assigned an IP address. This enables hosts within the bridge to communicate with other hosts outside of the bridge through a *switch virtual interface* (SVI), which provides layer 3 routing. The IP address of the bridge is typically from the same subnet as the member hosts of the bridge.

> **NOTE**
>
> When you add an interface to a bridge, it ceases to function as a router interface and the IP address on the interface becomes unreachable.

To configure the SVI:

NCLU Commands | Linux Commands

Run the `net add bridge` and `net add vlan` commands. The following example commands configure an SVI using swp1 and swp2, and VLAN ID 10.

```
cumulus@switch:~$ net add bridge bridge ports swp1-2
cumulus@switch:~$ net add vlan 10 ip address 10.100.100.1/24
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

When you configure a switch initially, all southbound bridge ports might be down; therefore, by default, the SVI is also down. You can force the SVI to always be up by disabling interface state tracking, which leaves the SVI in the UP state always, even if all member ports are down. Other

implementations describe this feature as *no autostate*. This is beneficial if you want to perform connectivity testing.

To keep the SVI perpetually UP, create a dummy interface, then make the dummy interface a member of the bridge.

▼ Example Configuration

IPv6 Link-local Address Generation

By default, Cumulus Linux automatically generates IPv6 link-local addresses on VLAN interfaces. If you want to use a different mechanism to assign link-local addresses, you can disable this feature. You can disable link-local automatic address generation for both regular IPv6 addresses and address-virtual (macvlan) addresses.

To disable automatic address generation for a regular IPv6 address on a VLAN:

NCLU Commands | Linux Commands

Run the `net add vlan <vlan> ipv6-addrgen off` command. The following example command disables automatic address generation for a regular IPv6 address on a VLAN 100.

```
cumulus@switch:~$ net add vlan 100 ipv6-addrgen off
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

To re-enable automatic link-local address generation for a VLAN:

NCLU Commands | Linux Commands

Run the `net del vlan <vlan> ipv6-addrgen off` command. The following example command re-enables automatic address generation for a regular IPv6 address on VLAN 100.

```
cumulus@switch:~$ net del vlan 100 ipv6-addrgen off
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

bridge fdb Command Output

The `bridge fdb` command in Linux interacts with the forwarding database table (FDB), which the bridge uses to store the MAC addresses it learns and the ports on which it learns those MAC addresses. The `bridge fdb show` command output contains some specific keywords:

Keyword	Description
self	The Linux kernel FDB entry flag that indicates the FDB entry belongs to the FDB on the device referenced by the device. For example, this FDB entry belongs to the VXLAN device `vx-1000`: `00:02:00:00:00:08 dev vx-1000 dst 27.0.0.10 self`
master	The Linux kernel FDB entry flag that indicates the FDB entry belongs to the FDB on the device's master and the FDB entry is pointing to a master's port. For example, this FDB entry is from the master device named bridge and is pointing to the VXLAN bridge port `vx-1001`: `02:02:00:00:00:08 dev vx-1001 vlan 1001 master bridge`

Keyword	Description
extern_learn	The Linux kernel FDB entry flag that indicates the FDB entry is managed (or offloaded) by an external control plane, such as the BGP control plane for EVPN.

The following example shows the `bridge fdb show` command output:

```
cumulus@switch:~$ bridge fdb show | grep 02:02:00:00:00:08
02:02:00:00:00:08 dev vx-1001 vlan 1001 extern_learn master
bridge
02:02:00:00:00:08 dev vx-1001 dst 27.0.0.10 self extern_learn
```

NOTE

- *02:02:00:00:00:08* is the MAC address learned with BGP EVPN.
- The first FDB entry points to a Linux bridge entry that points to the VXLAN device *vx-1001*.
- The second FDB entry points to the same entry on the VXLAN device and includes additional remote destination information.

- The VXLAN FDB augments the bridge FDB with additional remote destination information.
- All FDB entries that point to a VXLAN port appear as two entries. The second entry augments the remote destination information.

Considerations

- A bridge cannot contain multiple subinterfaces of the **same** port. Attempting this configuration results in an error.
- In environments where both VLAN-aware and traditional bridges are used, if a traditional bridge has a subinterface of a bond that is a normal interface in a VLAN-aware bridge, the bridge is flapped when the traditional bridge's bond subinterface is brought down.
- You cannot enslave a VLAN raw device to a different master interface (you cannot edit the `vlan-raw-device` setting in the `/etc/network/interfaces` file). You need to delete the VLAN and recreate it.
- Cumulus Linux supports up to 2000 VLANs. This includes the internal interfaces, bridge interfaces, logical interfaces, and so on.
- In Cumulus Linux, MAC learning is enabled by default on traditional or VLAN-aware bridge interfaces. Do not disable MAC learning unless you are using EVPN. See Ethernet Virtual Private Network - EVPN.

Related Information

- Linux Foundation - VLANs
- Linux Journal - Linux as an Ethernet Bridge
- Comparing Traditional Bridge Mode to VLAN-aware Bridge Mode

VLAN-aware Bridge Mode

The Cumulus Linux bridge driver supports two configuration modes, one that is VLAN-aware, and one that follows a more traditional Linux bridge model.

For traditional Linux bridges, the kernel supports VLANs in the form of VLAN subinterfaces. Enabling bridging on multiple VLANs means configuring a bridge for each VLAN and, for each member port on a bridge, creating one or more VLAN subinterfaces out of that port. This mode poses scalability challenges in terms of configuration size as well as boot time and run time state management, when the number of ports times the number of VLANs becomes large.

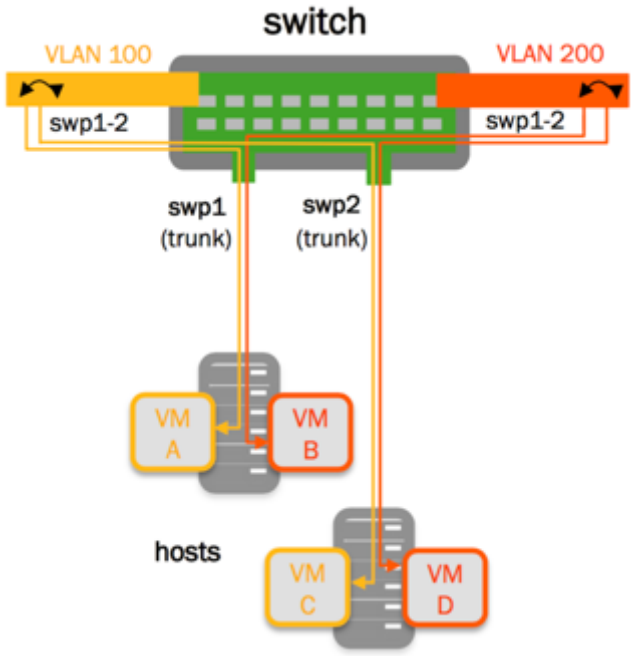
The VLAN-aware mode in Cumulus Linux implements a configuration model for large-scale layer 2 environments, with **one single instance** of spanning tree protocol. Each physical bridge member port is configured with the list of allowed VLANs as well as its port VLAN ID, either primary VLAN Identifier (PVID) or native VLAN. MAC address learning, filtering and forwarding are *VLAN-aware*. This significantly reduces the configuration size, and eliminates the large overhead of managing the port/VLAN instances as subinterfaces, replacing them with lightweight VLAN bitmaps and state updates.

TIP

You can configure both VLAN-aware and traditional mode bridges on thesame network in Cumulus Linux; however do not have more than one VLAN-aware bridge on a given switch.

Configure a VLAN-aware Bridge

The example below shows the commands required to create a VLAN-aware bridge configured for STP that contains two switch ports and includes 3 VLANs; the tagged VLANs 100 and 200 and the untagged (native) VLAN of 1.



NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add bridge bridge ports swp1-2
cumulus@switch:~$ net add bridge bridge vids 100,200
cumulus@switch:~$ net add bridge bridge pvid 1
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The above commands create the following code snippet in the `/etc/network/interfaces` file:

```
auto bridge
iface bridge
    bridge-ports swp1 swp2
    bridge-pvid 1
    bridge-vids 100 200
    bridge-vlan-aware yes
```

NOTE

The Primary VLAN Identifer (PVID) of the bridge defaults to 1. You

do *not* have to specify `bridge-pvid` for a bridge or a port. However, even though this does not affect the configuration, it helps other users for readability. The following configurations are identical to each other and the configuration above:

```
auto bridge
iface bridge
    bridge-ports swp1 swp2
    bridge-vids 1 100 200
    bridge-vlan-aware yes
```

```
auto bridge
iface bridge
    bridge-ports swp1 swp2
    bridge-pvid 1
    bridge-vids 1 100 200
    bridge-vlan-aware yes
```

```
auto bridge
iface bridge
```

```
bridge-ports swp1 swp2
bridge-vids 100 200
bridge-vlan-aware yes
```

TIP

If you specify `bridge-vids` or `bridge-pvid` at the bridge level, these configurations are inherited by all ports in the bridge. However, specifying any of these settings for a specific port overrides the setting in the bridge.

WARNING

Do not try to bridge the management port, eth0, with any switch ports (swp0, swp1 and so on). For example, if you create a bridge with eth0 and swp1, it will not work properly and might disrupt

access to the management interface.

Reserved VLAN Range

For hardware data plane internal operations, the switching silicon requires VLANs for every physical port, Linux bridge, and layer 3 subinterface. Cumulus Linux reserves a range of VLANs by default; the reserved range is 3600-3999.

TIP

You can modify the reserved range if it conflicts with any user-defined VLANs, as long the new range is a contiguous set of VLANs with IDs anywhere between 2 and 4094, and the minimum size of the range is 150 VLANs.

To configure the reserved range:

Edit the `/etc/cumulus/switchd.conf` file to uncomment the `resv_vlan_range` line and specify a new range, then restart `switchd`:

```
cumulus@switch:~$ sudo nano /etc/cumulus/switchd.conf
...
resv_vlan_range
```

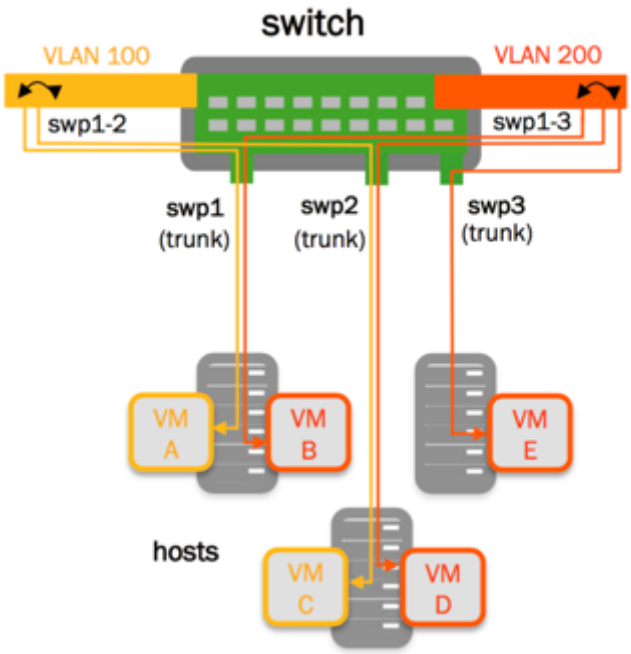
```
cumulus@switch:~$ sudo systemctl restart switchd.service
```

WARNING

Restarting the `switchd` service causes all network ports to reset, interrupting network services, in addition to resetting the switch hardware configuration.

VLAN Filtering (VLAN Pruning)

By default, the bridge port inherits the bridge VIDs. To configure a port to override the bridge VIDs:


switch
VLAN 100
VLAN 200
swp1-2
swp1-3
swp1
(trunk)
swp2
(trunk)
swp3
(trunk)
VM
A
VM
B
VM
E
hosts
VM
C
VM
D

NCLU Commands | Linux Commands

The following example commands configure swp3 to override the bridge VIDs:

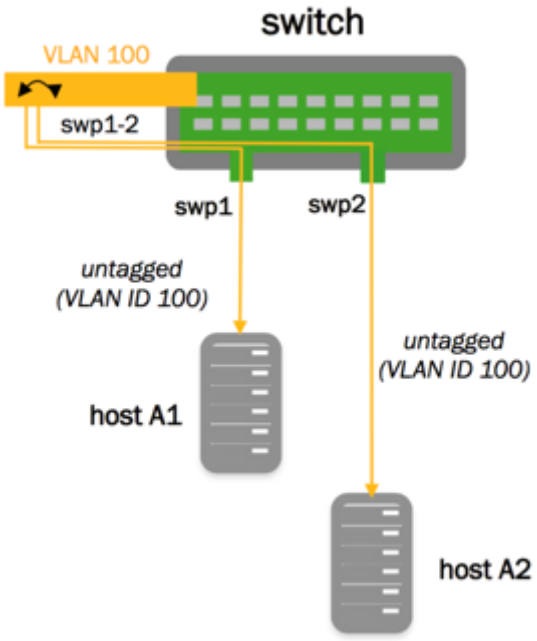
```
cumulus@switch:~$ net add bridge bridge ports swp1-3
cumulus@switch:~$ net add bridge bridge vids 100,200
cumulus@switch:~$ net add bridge bridge pvid 1
cumulus@switch:~$ net add interface swp3 bridge vids 200
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The above commands create the following code snippets in the `/etc/network/interfaces` file:

```
auto bridge
iface bridge
    bridge-ports swp1 swp2 swp3
    bridge-pvid 1
    bridge-vids 100 200
    bridge-vlan-aware yes

auto swp3
iface swp3
  bridge-vids 200
```

Untagged/Access Ports

Access ports ignore all tagged packets. In the configuration below, swp1 and swp2 are configured as access ports, while all untagged traffic goes to VLAN 100:



NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add bridge bridge ports swp1-2
cumulus@switch:~$ net add bridge bridge vids 100,200
cumulus@switch:~$ net add bridge bridge pvid 1
cumulus@switch:~$ net add interface swp1 bridge access 100
cumulus@switch:~$ net add interface swp2 bridge access 100
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The above commands create the following code snippets in the `/etc/network/interfaces` file:

```
auto bridge
iface bridge
    bridge-ports swp1 swp2
    bridge-pvid 1
    bridge-vids 100 200
    bridge-vlan-aware yes

auto swp1
iface swp1
    bridge-access 100

auto swp2
iface swp2
    bridge-access 100
```

Drop Untagged Frames

With VLAN-aware bridge mode, you can configure a switch port to drop any untagged frames. To do this, add `bridge-allow-untagged no` to the **switch port** (not to the bridge). This leaves the bridge port without a PVID and drops untagged packets.

NCLU Commands | Linux Commands

To configure a switch port to drop untagged frames, run the `net add interface swp2 bridge allow-untagged no` command. The following example command configures swp2 to drop untagged frames:

```
cumulus@switch:~$ net add interface swp2 bridge allow-
untagged no
```

When you check VLAN membership for that port, it shows that there is **no** untagged VLAN.

```
cumulus@switch:~$ net show bridge vlan

Interface      VLAN   Flags
-----------  ------   --------------------
swp1              1   PVID, Egress Untagged
                 10
                100
                200
swp2             10
                100
                200
```

VLAN Layer 3 Addressing

When configuring the VLAN attributes for the bridge, specify the attributes for each VLAN interface. If you are configuring the switch virtual interface (SVI) for the native VLAN, you must declare the native VLAN and specify its IP address. Specifying the IP address in the bridge stanza itself returns an error.

NCLU Commands | Linux Commands

The following example commands declare native VLAN 100 with IPv4 address 192.168.10.1/24 and IPv6 address 2001:db8::1/32.

```
cumulus@switch:~$ net add vlan 100 ip address 192.168.10.1/
24
cumulus@switch:~$ net add vlan 100 ipv6 address 2001:db8::1/
32
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

NOTE

In the above configuration, if your switch is configured for

multicast routing, you do not need to specify `bridge-igmp-querier-src`, as there is no need for a static IGMP querier configuration on the switch. Otherwise, the static IGMP querier configuration helps to probe the hosts to refresh their IGMP reports.

Configure ARP Timers

Cumulus Linux does not often interact directly with end systems as much as end systems interact with one another. Therefore, after a successful address resolution protocol (ARP) places a neighbor into a reachable state, Cumulus Linux might not interact with the client again for a long enough period of time for the neighbor to move into a stale state. To keep neighbors in the reachable state, Cumulus Linux includes a background process (`/usr/bin/neighmgrd`). The background process tracks neighbors that move into a stale, delay, or probe state, and attempts to refresh their state before they are removed from the Linux kernel and from hardware forwarding.

The ARP refresh timer defaults to 1080 seconds (18 minutes). To change this setting, follow the procedures outlined in Address Resolution Protocol - ARP.

Configure Multiple Ports in a Range

To save time, you can specify a range of ports or VLANs instead of enumerating each one individually.

To specify a range:

NCLU Commands | Linux Commands

In the example below, `swp1-52` indicates that swp1 through swp52 are part of the bridge.

```
cumulus@switch:~$ net add bridge bridge ports swp1-52
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Example Configurations

The following sections provide example VLAN-aware bridge configurations.

Access Ports and Pruned VLANs

The following example configuration contains an access port and switch port that are *pruned*; they only sends and receive traffic tagged to and from a specific set of VLANs declared by the `bridge-vids` attribute. It also contains other switch ports that send and receive traffic from all the

defined VLANs.

```
...
# ports swp3-swp48 are trunk ports which inherit vlans from the
'bridge'
# ie vlans 310,700,707,712,850,910
#
auto bridge
iface bridge
      bridge-ports swp1 swp2 swp3 ... swp51 swp52
      bridge-vids 310 700 707 712 850 910
      bridge-vlan-aware yes

auto swp1
iface swp1
      bridge-access 310
      mstpctl-bpduguard yes
      mstpctl-portadminedge yes

# The following is a trunk port that is "pruned".
# native vlan is 1, but only .1q tags of 707, 712, 850 are
# sent and received
#
auto swp2
iface swp2
```

```
      mstpctl-bpduguard yes
      mstpctl-portadminedge yes
      bridge-vids 707 712 850

# The following port is the trunk uplink and inherits all vlans
# from 'bridge'; bridge assurance is enabled using
'portnetwork' attribute
auto swp49
iface swp49
      mstpctl-portnetwork yes
      mstpctl-portpathcost 10

# The following port is the trunk uplink and inherits all vlans
# from 'bridge'; bridge assurance is enabled using
'portnetwork' attribute
 auto swp50
 iface swp50
      mstpctl-portnetwork yes
      mstpctl-portpathcost 0
...
```

Large Bond Set Configuration

The configuration below demonstrates a VLAN-aware bridge with a large set of bonds. The bond configurations are generated from a Mako template.

```
...
#
# vlan-aware bridge with bonds example
#
# uplink1, peerlink and downlink are bond interfaces.
# 'bridge' is a vlan aware bridge with ports uplink1, peerlink
# and downlink (swp2-20).
#
# native vlan is by default 1
#
# 'bridge-vids' attribute is used to declare vlans.
# 'bridge-pvid' attribute is used to specify native vlans if
other than 1
# 'bridge-access' attribute is used to declare access port
#
auto lo
iface lo


auto eth0
iface eth0 inet dhcp


# bond interface
auto uplink1
iface uplink1
```

```
    bond-slaves swp32
    bridge-vids 2000-2079

# bond interface
auto peerlink
iface peerlink
    bond-slaves swp30 swp31
    bridge-vids 2000-2079 4094

# bond interface
auto downlink
iface downlink
    bond-slaves swp1
    bridge-vids 2000-2079

#
# Declare vlans for all swp ports
# swp2-20 get vlans from 2004 to 2022.
# The below uses mako templates to generate iface sections
# with vlans for swp ports
#
%for port, vlanid in zip(range(2, 20), range(2004, 2022)) :
    auto swp${port}
    iface swp${port}
```

```
      bridge-vids ${vlanid}

%endfor

# svi vlan 2000
auto bridge.2000
iface bridge.2000
    address 11.100.1.252/24

# l2 attributes for vlan 2000
auto bridge.2000
vlan bridge.2000
    bridge-igmp-querier-src 172.16.101.1

#
# vlan-aware bridge
#
auto bridge
iface bridge
    bridge-ports uplink1 peerlink downlink swp1 swp2 swp49 swp50
    bridge-vlan-aware yes

# svi peerlink vlan
auto peerlink.4094
```

```
iface peerlink.4094
    address 192.168.10.1/30
    broadcast 192.168.10.3
...
```

VXLANs with VLAN-aware Bridges

Cumulus Linux supports using VXLANs with VLAN-aware bridge configuration. This provides improved scalability, as multiple VXLANs can be added to a single VLAN-aware bridge. A one to one association is used between the VXLAN VNI and the VLAN, with the bridge access VLAN definition on the VXLAN and the VLAN membership definition on the local bridge member interfaces.

The configuration example below shows the differences between a VXLAN configured for traditional bridge mode and one configured for VLAN-aware mode. The configurations use head end replication (HER) together with the VLAN-aware bridge to map VLANs to VNIs.

> **NOTE**
>
> See VXLAN Scale for information about the number of VXLANs you can configure simultaneously.

```
...
auto lo
iface lo inet loopback
    address 10.35.0.10/32

auto bridge
iface bridge
    bridge-ports uplink
    bridge-pvid 1
    bridge-vids 1-100
    bridge-vlan-aware yes
auto vni-10000
iface vni-10000
    alias CUSTOMER X VLAN 10
    bridge-access 10
    vxlan-id 10000
    vxlan-local-tunnelip 10.35.0.10
    vxlan-remoteip 10.35.0.34
...
```

Configure a Static MAC Address Entry

You can add a static MAC address entry to the layer 2 table for an interface within the VLAN-aware bridge by running a command similar to the following:

```
cumulus@switch:~$ sudo bridge fdb add 12:34:56:12:34:56 dev
swp1 vlan 150 master static
cumulus@switch:~$ sudo bridge fdb show
44:38:39:00:00:7c dev swp1 master bridge permanent
12:34:56:12:34:56 dev swp1 vlan 150 master bridge static
44:38:39:00:00:7c dev swp1 self permanent
12:12:12:12:12:12 dev swp1 self permanent
12:34:12:34:12:34 dev swp1 self permanent
12:34:56:12:34:56 dev swp1 self permanent
12:34:12:34:12:34 dev bridge master bridge permanent
44:38:39:00:00:7c dev bridge vlan 500 master bridge permanent
12:12:12:12:12:12 dev bridge master bridge permanent
```

Considerations

Spanning Tree Protocol (STP)

- Because STP is enabled on a per-bridge basis, VLAN-aware mode supports a single instance of STP across all VLANs. A common practice when using a single STP instance for all VLANs is to define every VLAN on every switch in the spanning tree instance.
- `mstpd` remains the user space protocol daemon.
- Cumulus Linux supports Rapid Spanning Tree Protocol (RSTP).

IGMP Snooping

IGMP snooping and group membership are supported on a per-VLAN basis; however, the IGMP snooping configuration (including enable, disable, and mrouter ports) is defined on a per-bridge port basis.

VLAN Translation

A bridge in VLAN-aware mode cannot have VLAN translation enabled. Only traditional mode bridges can utilize VLAN translation.

Convert Bridges between Supported Modes

You cannot convert traditional mode bridges automatically to and from a VLAN-aware bridge. You must delete the original configuration and bring down all member switch ports before creating a new bridge.

Traditional Bridge Mode

Use a VLAN-aware bridge on your switch. Use traditional mode bridges only if you need to run more than one bridge on the switch or if you need to use PVSTP+.

Configure a Traditional Mode Bridge

The following examples show how to create a simple traditional mode bridge configuration on the switch. The example also shows some optional elements:

- You can add an IP address to provide IP access to the bridge interface.
- You can specify a range of interfaces.

To configure spanning tree options for a bridge interface, refer to Spanning Tree and Rapid Spanning Tree - STP.

NCLU Commands | Linux Commands

The following example commands configure a traditional mode bridge called my_bridge with IP address 10.10.10.10/24. swp1, swp2, swp3, and swp4 are members of the bridge.

```
cumulus@switch:~$ net add bridge my_bridge ports swp1-4
cumulus@switch:~$ net add bridge my_bridge ip address
10.10.10.10/24
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

NOTE

The name of the bridge must be:

- Compliant with Linux interface naming conventions.
- Unique within the switch.
- Something other than *bridge*, as Cumulus Linux reserves that name for a single VLAN-aware bridge.

WARNING

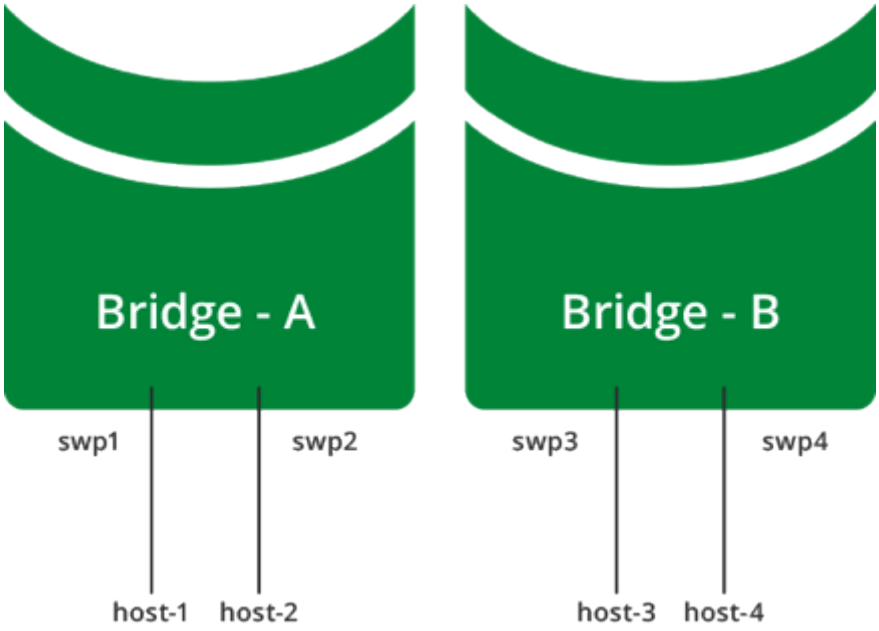
Do not try to bridge the management port, eth0, with any switch ports (swp0, swp1, and so on). For example, if you create a bridge with eth0 and swp1, it does **not** work.

Configure Multiple Traditional Mode Bridges

You can configure multiple bridges to logically divide a switch into multiple layer 2 domains. This allows for hosts to communicate with other hosts in the same domain, while separating them from hosts in other domains.

The diagram below shows a multiple bridge configuration, where host-1 and host-2 are connected to bridge-A, while host-3 and host-4 are connected to bridge-B:

- host-1 and host-2 can communicate with each other
- host-3 and host-4 can communicate with each other
- host-1 and host-2 cannot communicate with host-3 and host-4



This example configuration looks like this in the `/etc/network/interfaces` file:

```
...
auto bridge-A
iface bridge-A
    bridge-ports swp1 swp2
    bridge-vlan-aware no

auto bridge-B
iface bridge-B
    bridge-ports swp3 swp4
    bridge-vlan-aware no
...
```

Trunks in Traditional Bridge Mode

The standard for trunking is 802.1Q. The 802.1Q specification adds a 4 byte header within the Ethernet frame that identifies the VLAN of which the frame is a member.

802.1Q also identifies an *untagged* frame as belonging to the *native* VLAN (most network devices default their native VLAN to 1). The concept of native, non-native, tagged or untagged has generated confusion due to mixed terminology and vendor-specific implementations. In Cumulus Linux:

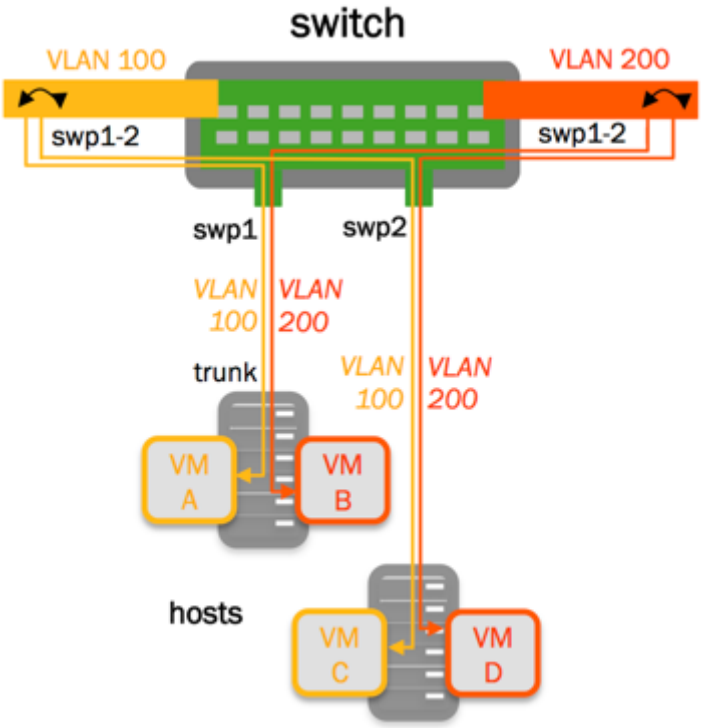
- A *trunk port* is a switch port configured to send and receive 802.1Q tagged frames.
- A switch sending an untagged (bare Ethernet) frame on a trunk port is sending from the native VLAN defined on the trunk port.
- A switch sending a tagged frame on a trunk port is sending to the VLAN identified by the 802.1Q tag.
- A switch receiving an untagged (bare Ethernet) frame on a trunk port places that frame in the native VLAN defined on the trunk port.
- A switch receiving a tagged frame on a trunk port places that frame in the VLAN identified by the 802.1Q tag.

A bridge in traditional mode has no concept of trunks, just tagged or untagged frames. With a trunk of 200 VLANs, there would need to be 199 bridges, each containing a tagged physical interface, and one bridge containing the native untagged VLAN. See the examples below for more information.

NOTE

The interaction of tagged and un-tagged frames on the same trunk often leads to undesired and unexpected behavior. A switch that uses VLAN 1 for the native VLAN may send frames to a switch that uses VLAN 2 for the native VLAN, thus merging those two VLANs and their spanning tree state.

Trunk Example



To create the above example:

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add bridge br-VLAN100 ports
swp1.100,swp2.100
cumulus@switch:~$ net add bridge br-VLAN200 ports
swp1.200,swp2.200
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

VLAN Tagging Examples

You can find more examples of VLAN tagging in the VLAN tagging chapter.

Configure ARP Timers

Cumulus Linux does not often interact directly with end systems as much as end systems interact with one another. Therefore, after a successful address resolution protocol (ARP) places a neighbor into a reachable state, Cumulus Linux might not interact with the client again for a long enough period of time for the neighbor to move into a stale state. To keep neighbors in the reachable state, Cumulus Linux includes a background process (`/usr/bin/neighmgrd`). The background process tracks neighbors that move into a stale, delay, or probe state, and attempts to refresh their state before they are removed from the Linux kernel and from hardware forwarding. The `neighmgrd` process only adds a neighbor if the sender's IP in the ARP packet is in one of the SVI's subnets (you can disable this check by

setting `subnet_checks` to *0* in the `/etc/cumulus/neighmgr.conf` file).

The ARP refresh timer defaults to 1080 seconds (18 minutes). To change this setting, follow the procedures outlined in this Address Resolution Protocol - ARP.

Considerations

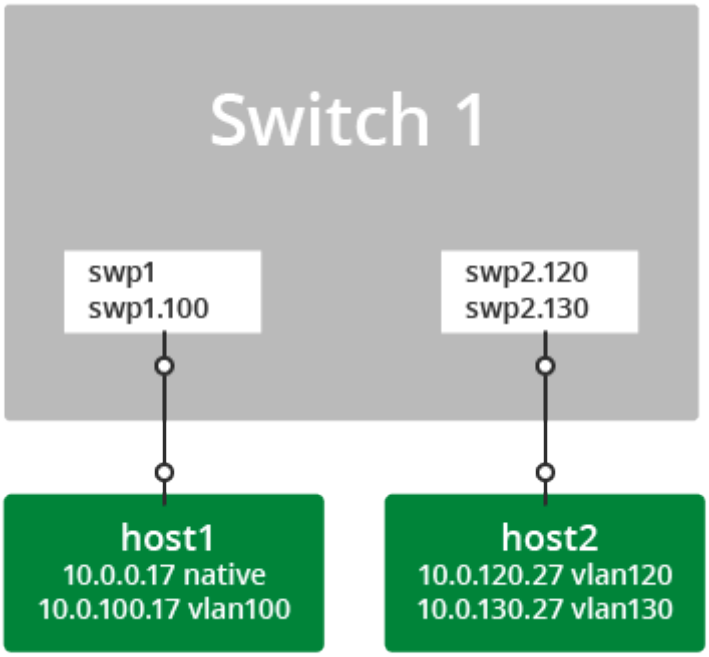
On Broadcom switches, when two VLAN subinterfaces are bridged to each other in a traditional mode bridge, `switchd` does not assign an internal resource ID to the subinterface, which is expected for each VLAN subinterface. To work around this issue, add a VXLAN on the bridge so that it does not require a real tunnel IP address.

VLAN Tagging

This topic shows two examples of VLAN tagging, one basic and one more advanced. They both demonstrate the streamlined interface configuration from `ifupdown2`.

VLAN Tagging, a Basic Example

A simple configuration demonstrating VLAN tagging involves two hosts connected to a switch.



- *host1* connects to swp1 with both untagged frames and with 802.1Q frames tagged for *vlan100*.
- *host2* connects to swp2 with 802.1Q frames tagged for *vlan120* and

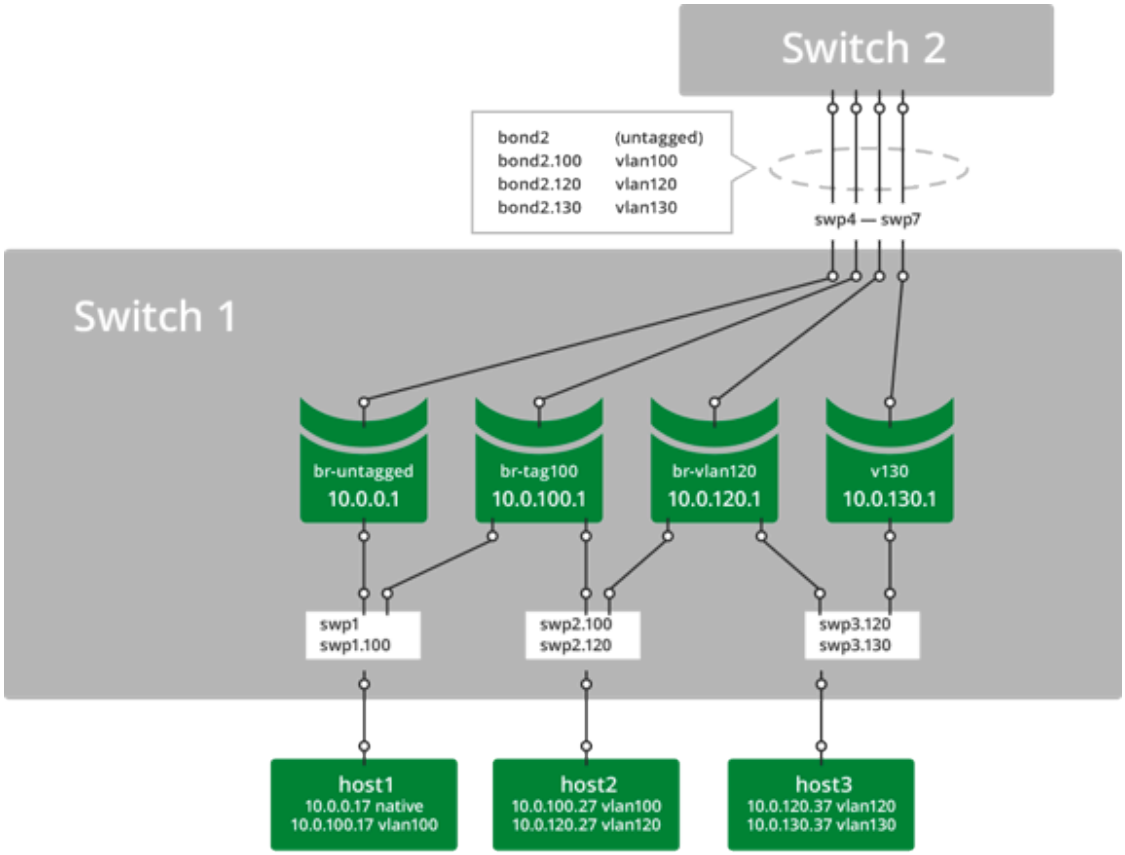
vlan130.

To configure the above example, edit the `/etc/network/interfaces` file and add a configuration like the following:

```
# Config for host1

auto swp1
iface swp1

auto swp1.100
iface swp1.100

# Config for host2
# swp2 must exist to create the .1Q subinterfaces, but it is
not assigned an address

auto swp2
iface swp2

auto swp2.120
iface swp2.120

auto swp2.130
iface swp2.130
```

VLAN Tagging, an Advanced Example

This example of VLAN tagging is more complex, involving three hosts and two switches, with a number of bridges and a bond connecting them all.



- *host1* connects to bridge *br-untagged* with bare Ethernet frames and to bridge *br-tag100* with 802.1q frames tagged for *vlan100*.
- *host2* connects to bridge *br-tag100* with 802.1q frames tagged for *vlan100* and to bridge *br-vlan120* with 802.1q frames tagged for *vlan120*.
- *host3* connects to bridge *br-vlan120* with 802.1q frames tagged for *vlan120* and to bridge *v130* with 802.1q frames tagged for *vlan130*.

- *bond2* carries tagged and untagged frames in this example.

Although not explicitly designated, the bridge member ports function as 802.1Q *access ports* and *trunk ports*. In the example above, comparing Cumulus Linux with a traditional Cisco device:

- *swp1* is equivalent to a trunk port with untagged and *vlan100*.
- *swp2* is equivalent to a trunk port with *vlan100* and *vlan120*.
- *swp3* is equivalent to a trunk port with *vlan120* and *vlan130*.
- *bond2* is equivalent to an EtherChannel in trunk mode with untagged, *vlan100*, *vlan120*, and *vlan130*.
- Bridges *br-untagged*, *br-tag100*, *br-vlan120*, and *v130* are equivalent to SVIs (switched virtual interfaces).

To create the above configuration, edit the `/etc/network/interfaces` file and add a configuration like the following:

```
# Config for host1


# swp1 does not need an iface section unless it has a specific
setting,
# it will be picked up as a dependent of swp1.100.
# And swp1 must exist in the system to create the .1q
subinterfaces..
# but it is not applied to any bridge..or assigned an address.
```

```
auto swp1.100
iface swp1.100

# Config for host2
# swp2 does not need an iface section unless it has a specific
setting,
# it will be picked up as a dependent of swp2.100 and swp2.120.
# And swp2 must exist in the system to create the .1q
subinterfaces..
# but it is not applied to any bridge..or assigned an address.

auto swp2.100
iface swp2.100

auto swp2.120
iface swp2.120

# Config for host3
# swp3 does not need an iface section unless it has a specific
setting,
# it will be picked up as a dependent of swp3.120 and swp3.130.
# And swp3 must exist in the system to create the .1q
subinterfaces..
# but it is not applied to any bridge..or assigned an address.
```

```
auto swp3.120
iface swp3.120

auto swp3.130
iface swp3.130

# Configure the bond

auto bond2
iface bond2
  bond-slaves glob swp4-7

# configure the bridges

auto br-untagged
iface br-untagged
    address 10.0.0.1/24
    bridge-ports swp1 bond2
    bridge-stp on

auto br-tag100
iface br-tag100
    address 10.0.100.1/24
```

```
    bridge-ports swp1.100 swp2.100 bond2.100
    bridge-stp on

auto br-vlan120
iface br-vlan120
    address 10.0.120.1/24
    bridge-ports swp2.120 swp3.120 bond2.120
    bridge-stp on

auto v130
iface v130
    address 10.0.130.1/24
    bridge-ports swp3.130 bond2.130
    bridge-stp on

#
```

To verify:

```
cumulus@switch:~$ sudo mstpctl showbridge br-tag100
br-tag100 CIST info
  enabled         yes
  bridge id       8.000.44:38:39:00:32:8B
```

```
  designated root 8.000.44:38:39:00:32:8B
  regional root   8.000.44:38:39:00:32:8B
  root port       none
  path cost     0          internal path cost   0
  max age       20         bridge max age       20
  forward delay 15         bridge forward delay 15
  tx hold count 6          max hops             20
  hello time    2          ageing time          300
  force protocol version     rstp
  time since topology change 333040s
  topology change count      1
  topology change            no
  topology change port       swp2.100
  last topology change port  None
```

```
cumulus@switch:~$ sudo mstpctl showportdetail br-tag100  | grep
-B 2 state
br-tag100:bond2.100 CIST info
  enabled            yes
role                 Designated
  port id            8.003
state                forwarding
--
```

```
br-tag100:swp1.100 CIST info
  enabled            yes
role                 Designated
  port id            8.001
state                forwarding
--
  br-tag100:swp2.100 CIST info
  enabled            yes
role                 Designated
  port id            8.002
state                forwarding
```

```
cumulus@switch:~$ cat /proc/net/vlan/config
VLAN Dev name    | VLAN ID
Name-Type: VLAN_NAME_TYPE_RAW_PLUS_VID_NO_PAD
bond2.100      | 100  | bond2
bond2.120      | 120  | bond2
bond2.130      | 130  | bond2
swp1.100       | 100  | swp1
swp2.100       | 100  | swp2
swp2.120       | 120  | swp2
swp3.120       | 120  | swp3
swp3.130       | 130  | swp3
```

```
cumulus@switch:~$ cat /proc/net/bonding/bond2
Ethernet Channel Bonding Driver: v3.7.1 (April 27, 2011)

Bonding Mode: IEEE 802.3ad Dynamic link aggregation
Transmit Hash Policy: layer3+4 (1)
MII Status: up
MII Polling Interval (ms): 100
Up Delay (ms): 0
Down Delay (ms): 0

802.3ad info
LACP rate: fast
Min links: 0
Aggregator selection policy (ad_select): stable
Active Aggregator Info:
    Aggregator ID: 3
    Number of ports: 4
    Actor Key: 33
    Partner Key: 33
    Partner Mac Address: 44:38:39:00:32:cf

Slave Interface: swp4
MII Status: up
Speed: 10000 Mbps
```

```
Duplex: full
Link Failure Count: 0
Permanent HW addr: 44:38:39:00:32:8e
Aggregator ID: 3
Slave queue ID: 0

Slave Interface: swp5
MII Status: up
Speed: 10000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 44:38:39:00:32:8f
Aggregator ID: 3
Slave queue ID: 0

Slave Interface: swp6
MII Status: up
Speed: 10000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 44:38:39:00:32:90
Aggregator ID: 3
Slave queue ID: 0
```

```
Slave Interface: swp7
MII Status: up
Speed: 10000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 44:38:39:00:32:91
Aggregator ID: 3
Slave queue ID: 0
```

WARNING

A single bridge cannot contain multiple subinterfaces of the **same** port as members. Attempting to apply such a configuration will result in an error:

```
cumulus@switch:~$ sudo brctl addbr another_bridge
cumulus@switch:~$ sudo brctl addif another_bridge swp9
swp9.100
bridge cannot contain multiple subinterfaces of the same
port: swp9, swp9.100
```

VLAN Translation

By default, Cumulus Linux does not allow VLAN subinterfaces associated with different VLAN IDs to be part of the same bridge. Base interfaces are not explicitly associated with any VLAN IDs and are exempt from this restriction.

In some cases, it may be useful to relax this restriction. For example, two servers might be connected to the switch using VLAN trunks, but the VLAN numbering provisioned on the two servers are not consistent. You can choose to just bridge two VLAN subinterfaces of different VLAN IDs from the servers. You do this by enabling the `sysctl net.bridge.bridge-allow-multiple-vlans`. Packets entering a bridge from a member VLAN subinterface will egress another member VLAN subinterface with the VLAN ID translated.

> **NOTE**
>
> A bridge in VLAN-aware mode cannot have VLAN translation enabled for it; only bridges configured in traditional mode can utilize VLAN translation.

The following example enables the VLAN translation `sysctl`:

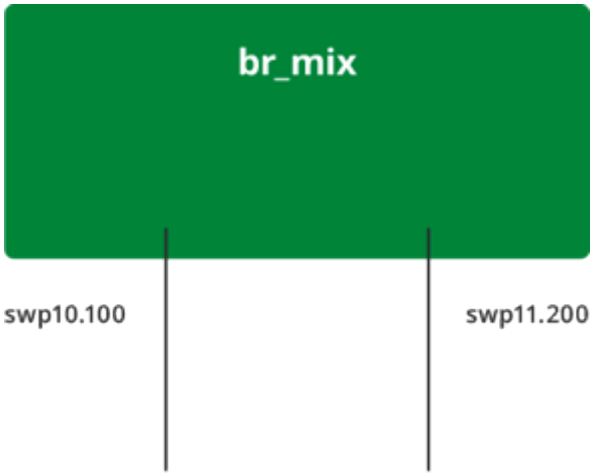
```
cumulus@switch:~$ echo net.bridge.bridge-allow-multiple-vlans =
1 | sudo tee /etc/sysctl.d/multiple_vlans.conf
net.bridge.bridge-allow-multiple-vlans = 1
cumulus@switch:~$ sudo sysctl -p /etc/sysctl.d/
multiple_vlans.conf
net.bridge.bridge-allow-multiple-vlans = 1
```

If the `sysctl` is enabled and you want to disable it, run the above example, setting the `sysctl net.bridge.bridge-allow-multiple-vlans` to *0*.

After `sysctl` is enabled, ports with different VLAN IDs can be added to the same bridge. In the following example, packets entering the bridge `br-mix` from swp10.100 will be bridged to swp11.200 with the VLAN ID translated from 100 to 200:

```
cumulus@switch:~$ sudo brctl addif br_mix swp10.100 swp11.200


cumulus@switch:~$ sudo brctl show br_mix
bridge name     bridge id               STP enabled
interfaces
br_mix          8000.4438390032bd       yes
swp10.100
```

```
swp11.200
```



Spanning Tree and Rapid Spanning Tree - STP

Spanning tree protocol (STP) identifies links in the network and shuts down redundant links, preventing possible network loops and broadcast radiation on a bridged network. STP also provides redundant links for automatic failover when an active link fails. STP is enabled by default in Cumulus Linux for both VLAN-aware and traditional bridges.

Cumulus Linux supports RSTP, PVST, and PVRST modes:

- *Traditional bridges* operate in both PVST and PVRST mode. The default is set to PVRST. Each traditional bridge has its own separate STP instance.
- *VLAN-aware bridges* operate **only** in RSTP mode.

STP for a Traditional Mode Bridge

Per VLAN Spanning Tree (PVST) creates a spanning tree instance for a bridge. Rapid PVST (PVRST) supports RSTP enhancements for each spanning tree instance. To use PVRST with a traditional bridge, you must create a bridge corresponding to the untagged native VLAN and all the physical switch ports must be part of the same VLAN.

> **NOTE**
>
> For maximum interoperability, when connected to a switch that has

a native VLAN configuration, the native VLAN **must** be configured to be VLAN 1 only.

STP for a VLAN-aware Bridge

VLAN-aware bridges operate in RSTP mode only. RSTP on VLAN-aware bridges works with other modes in the following ways:

RSTP and STP

If a bridge running RSTP (802.1w) receives a common STP (802.1D) BPDU, it falls back to 802.1D automatically.

RSTP and PVST

The RSTP domain sends BPDUs on the native VLAN, whereas PVST sends BPDUs on a per VLAN basis. For both protocols to work together, you need to enable the native VLAN on the link between the RSTP to PVST domain; the spanning tree is built according to the native VLAN parameters.

The RSTP protocol does not send or parse BPDUs on other VLANs, but floods BPDUs across the network, enabling the PVST domain to maintain its spanning-tree topology and provide a loop-free network.

- To enable proper BPDU exchange across the network, be sure to allow all VLANs participating in the PVST domain on the link between the RSTP

and PVST domains.

- When using RSTP together with an existing PVST network, you need to define the root bridge on the PVST domain. Either lower the priority on the PVST domain or change the priority of the RSTP switches to a higher number.

- When connecting a VLAN-aware bridge to a proprietary PVST+ switch using STP, you must allow VLAN 1 on all 802.1Q trunks that interconnect them, regardless of the configured *native* VLAN. Only VLAN 1 enables the switches to address the BPDU frames to the IEEE multicast MAC address. The proprietary switch might be configured like this:

```
switchport trunk allowed vlan 1-100
```

RSTP and MST

RSTP works with MST seamlessly, creating a single instance of spanning tree that transmits BPDUs on the native VLAN.

RSTP treats the MST domain as one giant switch, whereas MST treats the RSTP domain as a different region. To enable proper communication between the regions, MST creates a Common Spanning Tree (CST) that connects all the boundary switches and forms the overall view of the MST domain. Because changes in the CST need to be reflected in all regions, the RSTP tree is included in the CST to ensure that changes on the RSTP domain are reflected in the CST domain. This does cause topology changes

on the RSTP domain to impact the rest of the network but keeps the MST domain informed of every change occurring in the RSTP domain, ensuring a loop-free network.

Configure the root bridge within the MST domain by changing the priority on the relevant MST switch. When MST detects an RSTP link, it falls back into RSTP mode. The MST domain choses the switch with the lowest cost to the CST root bridge as the CIST root bridge.

RSTP with MLAG

More than one spanning tree instance enables switches to load balance and use different links for different VLANs. With RSTP, there is only one instance of spanning tree. To better utilize the links, you can configure MLAG on the switches connected to the MST or PVST domain and set up these interfaces as an MLAG port. The PVST or MST domain thinks it is connected to a single switch and utilizes all the links connected to it. Load balancing is based on the port channel hashing mechanism instead of different spanning tree instances and uses all the links between the RSTP to the PVST or MST domains. For information about configuring MLAG, see Multi-Chassis Link Aggregation - MLAG.

Optional Configuration

There are a number of ways to customize STP in Cumulus Linux. Exercise caution when changing the settings below to prevent malfunctions in STP loop avoidance.

Spanning Tree Priority

If you have a multiple spanning tree instance (MSTI 0, also known as a common spanning tree, or CST), you can set the *tree priority* for a bridge. The bridge with the lowest priority is elected the *root bridge*. The priority must be a number between *0* and *61440*, and must be a multiple of 4096. The default is *32768*.

To set the tree priority, run the following commands:

NCLU Commands | Linux Commands

The following example command sets the tree priority to 8192:

```
cumulus@switch:~$ net add bridge stp treeprio 8192
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

NOTE

Cumulus Linux supports MSTI 0 only. It does not support MSTI 1 through 15.

PortAdminEdge (PortFast Mode)

PortAdminEdge is equivalent to the PortFast feature offered by other vendors. It enables or disables the *initial edge state* of a port in a bridge.

All ports configured with PortAdminEdge bypass the listening and learning states to move immediately to forwarding.

> **WARNING**
>
> PortAdminEdge mode might cause loops if it is not used with the BPDU guard feature.

It is common for edge ports to be configured as access ports for a simple end host; however, this is not mandatory. In the data center, edge ports typically connect to servers, which might pass both tagged and untagged traffic.

To configure PortAdminEdge mode:

NCLU Commands | Linux Commands

The following example commands configure PortAdminEdge and BPDU guard for swp5.

```
cumulus@switch:~$ net add interface swp5 stp bpduguard
cumulus@switch:~$ net add interface swp5 stp portadminedge
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

PortAutoEdge

PortAutoEdge is an enhancement to the standard PortAdminEdge (PortFast) mode, which allows for the automatic detection of edge ports. PortAutoEdge enables and disables the *auto transition* to and from the edge state of a port in a bridge.

NOTE

Edge ports and access ports are not the same. Edge ports transition directly to the forwarding state and skip the listening and learning stages. Upstream topology change notifications are not generated when an edge port link changes state. Access ports only forward untagged traffic; however, there is no such restriction on

edge ports, which can forward both tagged and untagged traffic.

When a BPDU is received on a port configured with PortAutoEdge, the port ceases to be in the edge port state and transitions into a normal STP port. When BPDUs are no longer received on the interface, the port becomes an edge port, and transitions through the discarding and learning states before resuming forwarding.

PortAutoEdge is enabled by default in Cumulus Linux.

To disable PortAutoEdge for an interface:

NCLU Commands | Linux Commands

The following example commands disable PortAutoEdge on swp1:

```
cumulus@switch:~$ net add interface swp1 stp portautoedge no
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

To re-enable PortAutoEdge for an interface:

NCLU Commands | Linux Commands

The following example commands re-enable PortAutoEdge on swp1:

```
cumulus@switch:~$ net del interface swp1 stp portautoedge no
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

BPDU Guard

You can configure *BPDU guard* to protect the spanning tree topology from unauthorized switches affecting the forwarding path. For example, if you add a new switch to an access port off a leaf switch and this new switch is configured with a low priority, it might become the new root switch and affect the forwarding path for the entire layer 2 topology.

To configure BPDU guard:

NCLU Commands Linux Commands

The following example commands set BPDU guard for swp5:

```
cumulus@switch:~$ net add interface swp5 stp bpduguard
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

If a BPDU is received on the port, STP brings down the port and logs an error in `/var/log/syslog`. The following is a sample error:

```
mstpd: error, MSTP_IN_rx_bpdu: bridge:bond0 Recvd BPDU on BPDU
Guard Port - Port Down
```

To determine whether BPDU guard is configured, or if a BPDU has been received:

NCLU Commands Linux Commands

```
cumulus@switch:~$ net show bridge spanning-tree | grep bpdu
  bpdu guard port    yes               bpdu guard
error     yes
```

The only way to recover a port that has been placed in the disabled state is to manually bring up the port with the `sudo ifup <interface>` command. See Interface Configuration and Management for more information about `ifupdown`.

> **NOTE**
>
> Bringing up the disabled port does not correct the problem if the configuration on the connected end-station has not been resolved.

Bridge Assurance

On a point-to-point link where RSTP is running, if you want to detect unidirectional links and put the port in a discarding state, you can enable bridge assurance on the port by enabling a port type network. The port is then in a bridge assurance inconsistent state until a BPDU is received from the peer. You need to configure the port type network on both ends of the link for bridge assurance to operate properly.

Bridge assurance is disabled by default.

To enable bridge assurance on an interface:

NCLU Commands | Linux Commands

The following example commands enable bridge assurance on swp1:

```
cumulus@switch:~$ net add interface swp1 stp portnetwork
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

To monitor logs for bridge assurance messages, run the following command:

```
cumulus@switch:~$ sudo grep -in assurance /var/log/syslog |
grep mstp
  1365:Jun 25 18:03:17 mstpd: br1007:swp1.1007 Bridge assurance
inconsistent
```

BPDU Filter

You can enable `bpdufilter` on a switch port, which filters BPDUs in both directions. This disables STP on the port as no BPDUs are transiting.

WARNING

Using BDPU filter might cause layer 2 loops. Use this feature deliberately and with extreme caution.

To configure the BPDU filter on an interface:

NCLU Commands | Linux Commands

The following example commands configure the BPDU filter on swp6:

```
cumulus@switch:~$ net add interface swp6 stp portbpdufilter
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Parameter List

Spanning tree parameters are defined in the IEEE 802.1D and 802.1Q specifications.

The table below describes the STP configuration parameters available in Cumulus Linux. For a comparison of STP parameter configuration between

`mstpctl` and other vendors, read this knowledge base article.

> **NOTE**
>
> Most of these parameters are blacklisted in the `ifupdown_blacklist` section of the `/etc/netd.conf` file. Before you configure these parameters, you must edit the file to remove them from the blacklist.

Parameter	NCLU Command	Description
`mstpctl-maxage`	`net add bridge stp maxage <seconds>`	Sets the maximum age of the bridge in seconds. The default is 20. The maximum age must meet the condition 2 * (Bridge Forward Delay - 1 second) >= Bridge Max Age.
`mstpctl-ageing`	`net add bridge stp ageing <seconds>`	Sets the Ethernet (MAC) address ageing time for the bridge in seconds when the running version is STP, but not RSTP/MSTP. The default is 1800.
`mstpctl-fdelay`	`net add bridge`	Sets the bridge

Parameter	NCLU Command	Description
	`stp fdelay <seconds>`	forward delay time in seconds. The default value is 15. The bridge forward delay must meet the condition 2 * (Bridge Forward Delay - 1 second) >= Bridge Max Age.
`mstpctl-maxhops`	`net add bridge stp maxhops <max-hops>`	Sets the maximum hops for the bridge. The default is 20.
`mstpctl-txholdcount`	`net add bridge stp txholdcount <hold-count>`	Sets the bridge transmit hold count. The default value is 6.
`mstpctl-forcevers`	`net add bridge stp forcevers RSTP|STP`	Sets the force STP version of the bridge to either RSTP/STP. The default is RSTP.
`mstpctl-treeprio`	`net add bridge stp treeprio <priority>`	Sets the tree priority of the bridge for an MSTI (multiple spanning tree instance). The priority value is a number between 0 and 61440 and must be a multiple of

Parameter	NCLU Command	Description
		4096. The bridge with the lowest priority is elected the root bridge. The default is 32768. See Spanning Tree Priority above. **Note**: Cumulus Linux supports MSTI 0 only. It does not support MSTI 1 through 15.
`mstpctl-hello`	`net add bridge stp hello <seconds>`	Sets the bridge hello time in seconds. The default is 2.
`mstpctl-portpathcost`	`net add interface <interface> stp portpathcost <cost>`	Sets the port cost of the interface. The default is 0. `mstpd` supports only long mode; 32 bits for the path cost.
`mstpctl-treeportprio`	`net add interface <interface> stp treeportprio <priority>`	Sets the priority of the interface for the MSTI. The priority value is a number between 0 and 240 and must be a multiple of 16. The default is 128. **Note**: Cumulus

Parameter	NCLU Command	Description
		Linux supports MSTI 0 only. It does not support MSTI 1 through 15.
`mstpctl-portadminedge`	`net add interface <interface> stp portadminedge`	Enables or disables the initial edge state of the interface in the bridge. The default is no. In NCLU, to use a setting other than the default, you must specify this attribute without setting an option. See PortAdminEdge above.
`mstpctl-portautoedge`	`net add interface <interface> stp portautoedge`	Enables or disables the auto transition to and from the edge state of the interface in the bridge. PortAutoEdge is enabled by default. See PortAutoEdge above.
`mstpctl-portp2p`	`net add interface <interface> stp portp2p yes|no`	Enables or disables the point-to-point detection mode of the interface in the

Parameter	NCLU Command	Description
		bridge.
`mstpctl-portrestrrole`	`net add interface <interface> stp portrestrrole`	Enables or disables the ability of the interface in the bridge to take the root role. The default is no. To enable this feature with the NCLU command, you specify this attribute without an option (`portrestrrole`). To enable this feature by editing the `/etc/network/interfaces` file, you specify `mstpctl-portrestrrole yes`.
`mstpctl-portrestrtcn`	`net add interface <interface> stp portrestrtcn`	Enables or disables the ability of the interface in the bridge to propagate received topology change notifications. The default is no.
`mstpctl-portnetwork`	`net add interface`	Enables or disables the bridge

Parameter	NCLU Command	Description
	`<interface> stp portnetwork`	assurance capability for a network interface. The default is no. See Bridge Assurance above.
`mstpctl-bpduguard`	`net add interface <interface> stp bpduguard`	Enables or disables the BPDU guard configuration of the interface in the bridge. The default is no. See BPDU Guard above.
`mstpctl-portbpdufilter`	`net add interface <interface> stp portbpdufilter`	Enables or disables the BPDU filter functionality for an interface in the bridge. The default is no. See BPDU Filter above.
`mstpctl-treeportcost`	`net add interface <interface> stp treeportcost <port-cost>`	Sets the spanning tree port cost to a value from 0 to 255. The default is 0.

Troubleshooting

To check STP status for a bridge:

NCLU Commands | Linux Commands

Run the `net show bridge spanning-tree` command:

```
cumulus@switch:~$ net show bridge spanning-tree
Bridge info
  enabled           yes
  bridge id         8.000.44:38:39:FF:40:94
    Priority:       32768
    Address:        44:38:39:FF:40:94
  This bridge is root.


  designated root 8.000.44:38:39:FF:40:94
    Priority:       32768
    Address:        44:38:39:FF:40:94


  root port         none
  path cost      0           internal path cost    0
  max age        20          bridge max age        20
  forward delay 15           bridge forward delay 15
  tx hold count 6            max hops              20
  hello time     2           ageing time           300
  force protocol version      rstp


INTERFACE   STATE   ROLE   EDGE
---------   -----   ----   ----
peerlink    forw    Desg   Yes
vni13       forw    Desg   Yes
vni24       forw    Desg   Yes
vxlan4001   forw    Desg   Yes
```

Related Information

The source code for `mstpd` and `mstpctl` was written by Vitalii Demianets and is hosted at the URL below.

- GitHub - mstpd project
- brctl(8)
- bridge-utils-interfaces(5)
- ifupdown-addons-interfaces(5)
- mstpctl(8)
- mstpctl-utils-interfaces(5)

Storm Control

Storm control provides protection against excessive inbound BUM (broadcast, unknown unicast, multicast) traffic on layer 2 switch port interfaces, which can cause poor network performance.

NOTE

- Storm control is *not* supported on a switch with the Tomahawk2 ASIC.
- On Broadcom switches, ARP requests over layer 2 VXLAN bypass broadcast storm control; they are forwarded to the CPU and subjected to embedded control plane QoS instead.

Configure Storm Control

To configure storm control for physical ports, edit the `/etc/cumulus/switchd.conf` file. For example, to enable broadcast storm control for swp1 at 400 packets per second (pps), multicast storm control at 3000 pps, and unknown unicast at 500 pps, edit the `/etc/cumulus/switchd.conf` file and uncomment the `storm_control.broadcast`, `storm_control.multicast`, and `storm_control.unknown_unicast` lines:

```
cumulus@switch:~$ sudo nano /etc/cumulus/switchd.conf
...
# Storm Control setting on a port, in pps, 0 means disable
interface.swp1.storm_control.broadcast = 400
interface.swp1.storm_control.multicast = 3000
interface.swp1.storm_control.unknown_unicast = 500
...
```

When you update the `/etc/cumulus/switchd.conf` file, you must restart `switchd` for the changes to take effect.

```
cumulus@switch:~$ sudo systemctl restart switchd.service
```

WARNING

Restarting the `switchd` service causes all network ports to reset, interrupting network services, in addition to resetting the switch hardware configuration.

Alternatively, you can run the following commands. The configuration

below takes effect immediately, but does not persist if you reboot the switch. For a persistent configuration, edit the `/etc/cumulus/switchd.conf` file, as described above.

```
cumulus@switch:~$ sudo sh -c 'echo 400 > /cumulus/switchd/
config/interface/swp1/storm_control/broadcast'
cumulus@switch:~$ sudo sh -c 'echo 3000 > /cumulus/switchd/
config/interface/swp1/storm_control/multicast'
cumulus@switch:~$ sudo sh -c 'echo 500 > /cumulus/switchd/
config/interface/swp1/storm_control/unknown_unicast'
```

Bonding - Link Aggregation

Linux bonding provides a method for aggregating multiple network interfaces (*slaves*) into a single logical bonded interface (*bond*). Link aggregation is useful for linear scaling of bandwidth, load balancing, and failover protection.

Cumulus Linux supports two bonding modes:

- IEEE 802.3ad link aggregation mode that allows one or more links to be aggregated together to form a *link aggregation group* (LAG) so that a media access control (MAC) client can treat the group as if it were a single link. IEEE 802.3ad link aggregation is the default mode.
- Balance-xor mode, where the bonding of slave interfaces are static and all slave interfaces are active for load balancing and fault tolerance purposes. This is useful for MLAG deployments.

Cumulus Linux uses version 1 of the LAG control protocol (LACP).

To temporarily bring up a bond even when there is no LACP partner, use LACP Bypass.

Hash Distribution

Egress traffic through a bond is distributed to a slave based on a packet hash calculation, providing load balancing over the slaves; many conversation flows are distributed over all available slaves to load balance the total traffic. Traffic for a single conversation flow always hashes to the

same slave.

The hash calculation uses packet header data to choose to which slave to transmit the packet:

- For IP traffic, IP header source and destination fields are used in the calculation.
- For IP + TCP/UDP traffic, source and destination ports are included in the hash calculation.

> **NOTE**
>
> In a failover event, the hash calculation is adjusted to steer traffic over available slaves.

LAG Custom Hashing

On Mellanox switches, you can configure which fields are used in the LAG hash calculation. For example, if you do not want to use source or destination port numbers in the hash calculation, you can disable the source port and destination port fields.

You can configure the following fields:

- Source MAC
- Destination
- Source IP
- Destination IP

- Ether type
- VLAN ID
- Source port
- Destination port
- Layer 3 protocol

To configure custom hash, edit the `/etc/cumulus/datapath/traffic.conf` file:

1. To enable custom hashing, uncomment the `lag_hash_config.enable = true` line.
2. To enable a field, set the field to `true`. To disable a field, set the field to `false`.
3. Run the `echo 1 > /cumulus/switchd/ctrl/hash_config_reload` command. This command does not cause any traffic interruptions.

The following shows an example `/etc/cumulus/datapath/traffic.conf` file:

```
cumulus@switch:~$ sudo nano /etc/cumulus/datapath/traffic.conf
...
#LAG HASH config
#HASH config for LACP to enable custom fields
#Fields will be applicable for LAG hash
#calculation
#Uncomment to enable custom fields configured below
```

```
lag_hash_config.enable = true


lag_hash_config.smac = true
lag_hash_config.dmac = true
lag_hash_config.sip  = true
lag_hash_config.dip  = true
lag_hash_config.ether_type = true
lag_hash_config.vlan_id = true
lag_hash_config.sport = false
lag_hash_config.dport = false
lag_hash_config.ip_prot = true
...
```

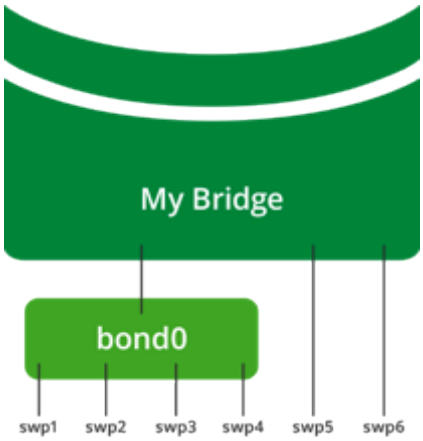
NOTE

Symmetric hashing is enabled by default on Mellanox switches. Make sure that the settings for the source IP (`lag_hash_config.sip`) and destination IP (`lag_hash_config.dip`) fields match, and that the settings for the source port (`lag_hash_config.sport`) and destination port (`lag_hash_config.dport`) fields match; otherwise symmetric hashing is disabled automatically. You can disable symmetric hashing manually in the `/etc/cumulus/datapath/traffic.conf` file by setting `symmetric_hash_enable = FALSE`.

You can set a unique hash seed for each switch to help avoid hash polarization. See Configure a Hash Seed to Avoid Hash Polarization.

Create a Bond

In the example below, the front panel port interfaces swp1 thru swp4 are slaves in bond0, while swp5 and swp6 are not part of bond0.



To create and configure a bond:

NCLU Commands | Linux Commands

Run the `net add bond` command. The example command below creates a bond called `bond0` with slaves swp1, swp2, swp3, and swp4:

```
cumulus@switch:~$ net add bond bond0 bond slaves swp1-4
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

NOTE

- The bond is configured by default in IEEE 802.3ad link aggregation mode. To configure the bond in balance-xor mode, see Configuration Parameters below.
- If the bond is *not* going to become part of a bridge, you need to specify an IP address.
- The name of the bond must be compliant with Linux interface naming conventions and unique within the switch.

When networking is started on the switch, bond0 is created as MASTER and interfaces swp1 thru swp4 come up in SLAVE mode, as seen in the `ip link show` command:

```
cumulus@switch:~$ ip link show
...

3: swp1: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc
pfifo_fast master bond0 state UP mode DEFAULT qlen 500
    link/ether 44:38:39:00:03:c1 brd ff:ff:ff:ff:ff:ff
4: swp2: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc
pfifo_fast master bond0 state UP mode DEFAULT qlen 500
    link/ether 44:38:39:00:03:c1 brd ff:ff:ff:ff:ff:ff
5: swp3: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc
pfifo_fast master bond0 state UP mode DEFAULT qlen 500
    link/ether 44:38:39:00:03:c1 brd ff:ff:ff:ff:ff:ff
6: swp4: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc
pfifo_fast master bond0 state UP mode DEFAULT qlen 500
    link/ether 44:38:39:00:03:c1 brd ff:ff:ff:ff:ff:ff
...

55: bond0: <BROADCAST,MULTICAST,MASTER,UP,LOWER_UP> mtu 1500
qdisc noqueue state UP mode DEFAULT
    link/ether 44:38:39:00:03:c1 brd ff:ff:ff:ff:ff:ff
```

NOTE

All slave interfaces within a bond have the same MAC address as the bond. Typically, the first slave added to the bond donates its MAC address as the bond MAC address, whereas the MAC addresses of the other slaves are set to the bond MAC address. The bond MAC address is used as the source MAC address for all traffic leaving the bond and provides a single destination MAC address to address traffic to the bond.

Configure Bond Options

The configuration options for a bond are are described in the table below. To configure a bond:

NCLU Commands | Linux Commands

Run `net add bond <bond-name> bond <option>`. The following example sets the bond mode for bond01 to `balance-xor`:

```
cumulus@switch:~$ net add bond bond1 bond mode balance-xor
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

> **NOTE**
>
> Each bond configuration option, except for `bond slaves`, is set to the recommended value by default in Cumulus Linux. Only configure an option if a different setting is needed. For more information on configuration values, refer to the Related Information section below.

Parameter	Description
`bond-mode 802.3ad|balance-xor`	Cumulus Linux supports IEEE 802.3ad link aggregation mode (802.3ad) and balance-xor mode. The default mode is 802.3ad. **Note:** When you enable balance-xor mode, the bonding of slave interfaces are static and all slave interfaces are active for load balancing and fault tolerance purposes. Packet transmission on the bond is based on the hash policy specified by xmit-hash-policy. When using balance-xor mode to dual-connect host-facing bonds in an MLAG environment, you must configure the clag-id

Parameter	Description
	parameter on the MLAG bonds and it must be the same on both MLAG switches. Otherwise, the bonds are treated by the MLAG switch pair as single-connected. Use balance-xor mode only if you cannot use LACP; LACP can detect mismatched link attributes between bond members and can even detect misconnections.
`bond-slaves <interface-list>`	The list of slaves in the bond.
`bond miimon <value>`	Defines how often the link state of each slave is inspected for failures. You can specify a value between 0 and 255. The default value is 100.
`bond-use-carrier no`	Determines the link state.
`bond-lacp-bypass-allow`	Enables LACP bypass.
`bond-lacp-rate slow`	Sets the rate to ask the link partner to transmit LACP control packets. slow is the only option.
`bond-min-links`	Defines the minimum number of links (between 0 and 255) that must be active before the bond is put into service. The default

Parameter	Description
	value is 1. A value greater than 1 is useful if higher level services need to ensure a minimum aggregate bandwidth level before activating a bond. Keeping bond-min-links set to 1 indicates the bond must have at least one active member. If the number of active members drops below the bond-min-links setting, the bond appears to upper-level protocols as link-down. When the number of active links returns to greater than or equal to bond-min-links, the bond becomes link-up.

Show Bond Information

To show information for a bond:

NCLU Commands | Linux Commands

Run the `net show interface <bond>` command:

```
cumulus@switch:~$ net show interface bond1
    Name    MAC                Speed    MTU    Mode
--  ------  -----------------  -------  -----  ------
UP  bond1   00:02:00:00:00:12  20G      1500   Bond



Bond Details
--------------  -------------
Bond Mode:       Balance-XOR
Load Balancing:  Layer3+4
Minimum Links:   1
In CLAG:         CLAG Inactive



    Port     Speed      TX    RX    Err    Link Failures
--  -------  -------  ----  ----  -----  ---------------
UP  swp3(P)  10G         0     0      0                0
UP  swp4(P)  10G         0     0      0                0



LLDP
-------  ----  ------------
swp3(P)  ====  swp1(p1c1h1)
swp4(P)  ====  swp2(p1c1h1)Routing
-------
  Interface bond1 (vrf mgmt) is up, line protocol is up
  Link ups:       3    last: 2017/04/26 21:00:38.26
```

> ⚠ **IMPORTANT**
>
> The detailed output in `/proc/net/bonding/<filename>` includes the actor/partner LACP information. This information is not necessary and requires you to use `sudo` to view the file.

Considerations

- An interface cannot belong to multiple bonds.
- A bond can have subinterfaces, but subinterfaces cannot have a bond.
- A bond cannot enslave VLAN subinterfaces.
- Set all slave ports within a bond to the same speed/duplex and make sure they match the link partner's slave ports.
- On a Cumulus RMP switch, if you create a bond with multiple 10G member ports, traffic gets dropped when the bond uses members of the same *unit* listed in the `/var/lib/cumulus/porttab` file. For example, traffic gets dropped if both swp49 and swp52 are in the bond because they both are in xe0 (or if both swp50 and swp51 are in the same bond because they are both in xe1):

```
swp49 xe0 0 0 -1 0
swp50 xe1 0 0 -1 0
swp51 xe1 1 0 -1 0
swp52 xe0 1 0 -1 0
```

Single port member bonds, bonds with different units (xe0 or xe1, as above), or layer 3 bonds do not have this issue.

NOTE

On Cumulus RMP switches, which are built with two Hurricane2 ASICs, you cannot form an LACP bond on links that terminate on different Hurricane2 ASICs.

Related Information

- Linux Foundation - Bonding
- 802.3ad (Accessible writeup)
- Wikipedia - Link aggregation

Multi-Chassis Link Aggregation - MLAG

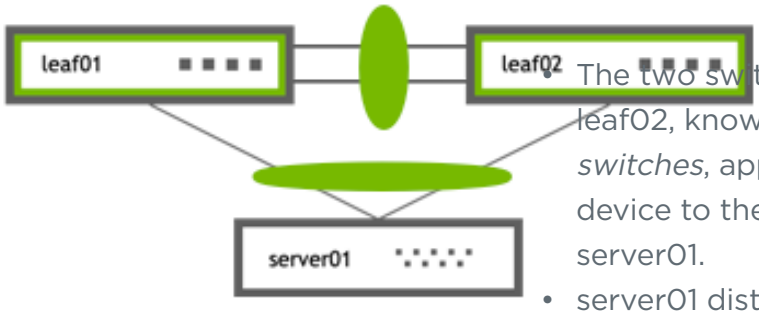
> **NOTE**
>
> **MLAG or CLAG**: The Cumulus Linux implementation of MLAG is referred to by other vendors as CLAG, MC-LAG or VPC. You will even see references to CLAG in Cumulus Linux, including the management daemon, named `clagd`, and other options in the code, such as `clag-id`, which exist for historical purposes. The Cumulus Linux implementation is truly a multi-chassis link aggregation protocol, so we call it MLAG.

Multi-Chassis Link Aggregation (MLAG) enables a server or switch with a two-port bond, such as a link aggregation group (LAG), EtherChannel, port group or trunk, to connect those ports to different switches and operate as if they are connected to a single, logical switch. This provides greater redundancy and greater system throughput.

Dual-connected devices can create LACP bonds that contain links to each physical switch; active-active links from the dual-connected devices are supported even though they are connected to two different physical switches.

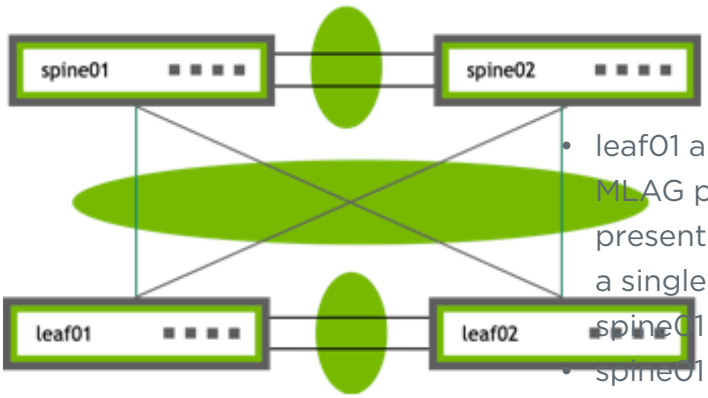
How Does MLAG Work?

A basic MLAG configuration looks like this:



- The two switches, leaf01 and leaf02, known as *peer switches*, appear as a single device to the bond on server01.
- server01 distributes traffic between the two links to leaf01 and leaf02 in the way you configure on the host.
- Traffic inbound to server01 can traverse leaf01 or leaf02 and arrive at server01.

More elaborate configurations are also possible. The number of links between the host and the switches can be greater than two and does not have to be symmetrical. Additionally, because the two peer switches appear as a single switch to other bonding devices, you can also connect pairs of MLAG switches to each other in a switch-to-switch MLAG configuration:

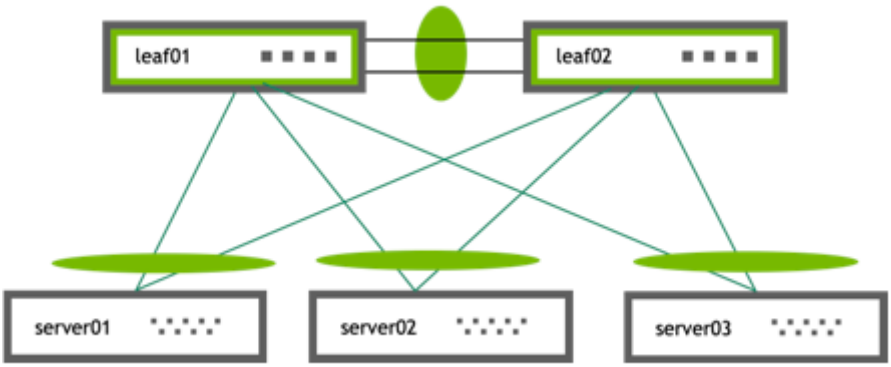


- leaf01 and leaf02 are also MLAG peer switches and present a two-port bond from a single logical system to spine01 and spine02.
- spine01 and spine02 do the same as far as leaf01 and leaf02 are concerned.

LACP and Dual-connected Links

Link Aggregation Control Protocol (LACP), the IEEE standard protocol for managing bonds, is used for verifying dual-connectedness. LACP runs on the dual-connected devices and on each of the MLAG peer switches. On a dual-connected device, the only configuration requirement is to create a bond that is managed by LACP.

On each of the peer switches, you must place the links that are connected to the dual-connected host or switch in the bond. This is true even if the links are a single port on each peer switch, where each port is placed into a bond, as shown below:



All of the dual-connected bonds on the peer switches have their system ID set to the MLAG system ID. Therefore, from the point of view of the hosts, each of the links in its bond is connected to the same system and so the host uses both links.

Each peer switch periodically makes a list of the LACP partner MAC addresses for all of their bonds and sends that list to its peer (using the `clagd` service). The LACP partner MAC address is the MAC address of the system at the other end of a bond (server01, server02, and server03 in the figure above). When a switch receives this list from its peer, it compares the list to the LACP partner MAC addresses on its switch. If any matches are found and the `clag-id` for those bonds match, then that bond is a dual-connected bond. You can find the LACP partner MAC address by the running `net show bridge macs` command.

Requirements

MLAG has these requirements:

- There must be a direct connection between the two peer switches configured with MLAG. This is typically a bond for increased reliability

and bandwidth.

- There must be only two peer switches in one MLAG configuration, but you can have multiple configurations in a network for *switch-to-switch MLAG*.
- Both switches in the MLAG pair must be running the same release of Cumulus Linux. See Upgrading Cumulus Linux.

Basic Configuration

To configure MLAG, you need to create a bond that uses LACP on the dual-connected devices and configure the interfaces (including bonds, VLANs, bridges, and peer links) on each peer switch.

Follow these steps on each peer switch in the MLAG pair:

1. On the dual-connected device, such as a host or server that sends traffic to and from the switch, create a bond that uses LACP. The method you use varies with the type of device you are configuring.

   > **NOTE**
   >
   > If you cannot use LACP in your environment, you can configure the bonds in balance-xor mode.

2. Place every interface that connects to the MLAG pair from a dual-connected device into a bond, even if the bond contains only a single link

on a single physical switch.

The following examples place swp1 in bond1 and swp2 in bond2. The examples also add a description for the bonds (an alias), which is optional.

NCLU Commands | Linux Commands

```
cumulus@leaf01:~$ net add bond bond1 bond slaves swp1
cumulus@leaf01:~$ net add bond bond1 alias bond1 on swp1
cumulus@leaf01:~$ net add bond bond2 bond slaves swp2
cumulus@leaf01:~$ net add bond bond2 alias bond2 on swp2
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

3. Add a unique MLAG ID (clag-id) to each bond.

   You must specify a unique MLAG ID (clag-id) for every dual-connected bond on each peer switch so that switches know which links are dual-connected or are connected to the same host or switch. The value must be between 1 and 65535 and must be the same on both peer switches. A value of 0 disables MLAG on the bond.

   The example commands below add an MLAG ID of 1 to bond1 and 2 to bond2:

NCLU Commands | Linux Commands

```
cumulus@leaf01:~$ net add bond bond1 clag id 1
cumulus@leaf01:~$ net add bond bond2 clag id 2
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

4. Add the bonds you created above to a bridge. The example commands below add bond1 and bond2 to a VLAN-aware bridge.

   On Mellanox switches, you must add **all** VLANs configured on the MLAG bond to the bridge so that traffic to the downstream device connected in MLAG is redirected successfully over the peerlink in case of an MLAG bond failure.

NCLU Commands | Linux Commands

```
cumulus@leaf01:~$ net add bridge bridge ports bond1,bond2
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

5. Create the inter-chassis bond and the peer link VLAN (as a VLAN subinterface). You also need to provide the peer link IP address, the MLAG bond interfaces, the MLAG system MAC address, and the backup

interface.

- By default, the NCLU command configures the inter-chassis bond with the name *peerlink* and the peer link VLAN with the name *peerlink.4094*. Use *peerlink.4094* to ensure that the VLAN is independent of the bridge and spanning tree forwarding decisions.
- The peer link IP address is an unrouteable link-local address that provides layer 3 connectivity between the peer switches.
- NVIDIA provides a reserved range of MAC addresses for MLAG (between 44:38:39:ff:00:00 and 44:38:39:ff:ff:ff). Use a MAC address from this range to prevent conflicts with other interfaces in the same bridged network.
  - Do not to use a multicast MAC address.
  - Do not use the same MAC address for different MLAG pairs; make sure you specify a different MAC address for each MLAG pair in the network.
- The backup IP address is any layer 3 backup interface for the peer link, which is used in case the peer link goes down. The backup IP address is **required** and **must** be different than the peer link IP address. It must be reachable by a route that does not use the peer link. Use the loopback or management IP address of the switch.

▼ Loopback or Management IP Address?

The following examples show commands for both MLAG peers (leaf01 and leaf02).

NCLU Commands | Linux Commands

The NCLU command is a macro command that:

- Automatically creates the inter-chassis bond (`peerlink`) and the peer link VLAN subinterface (`peerlink.4094`), and adds the `peerlink` bond to the bridge
- Configures the peer link IP address (`primary` is the link-local address)
- Adds the MLAG system MAC address, the MLAG bond interfaces, and the backup IP address you specify

leaf01 | leaf02

```
cumulus@leaf01:~$ net add clag peer sys-mac
44:38:39:BE:EF:AA interface swp49-50 primary backup-
ip 10.10.10.2
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

To configure the backup link to a VRF, include the name of the VRF with the `backup-ip` parameter. The following example configures the backup link to VRF RED:

```
cumulus@leaf01:~$ net add clag peer sys-mac
44:38:39:BE:EF:AA interface swp49-50 primary backup-
ip 10.10.10.2 vrf RED
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

NOTE

- Do *not* add VLAN 4094 to the bridge VLAN list; VLAN 4094 for the peer link subinterface **cannot** be configured as a bridged VLAN with bridge VIDs under the bridge.
- Do not use 169.254.0.1 as the MLAG peer link IP address; Cumulus Linux uses this address exclusively for BGP unnumbered interfaces.
- When you configure MLAG manually in the `/etc/network/interfaces` file, the changes take effect when you bring the peer link interface up with the `sudo ifreload -a` command. Do **not** use `systemctl restart clagd.service` to apply the new configuration.

MLAG synchronizes the dynamic state between the two peer switches but it does not synchronize the switch configurations. After modifying the configuration of one peer switch, you must make the same changes to the configuration on the other peer switch. This applies to all configuration changes, including:

- Port configuration, such as VLAN membership, MTU and bonding parameters.
- Bridge configuration, such as spanning tree parameters or bridge properties.

- Static address entries, such as static FDB entries and static IGMP entries.
- QoS configuration, such as ACL entries.

Optional Configuration

This section describes optional configuration procedures.

Set Roles and Priority

Each MLAG-enabled switch in the pair has a *role*. When the peering relationship is established between the two switches, one switch is put into the *primary* role and the other into the *secondary* role. When an MLAG-enabled switch is in the secondary role, it does not send STP BPDUs on dual-connected links; it only sends BPDUs on single-connected links. The switch in the primary role sends STP BPDUs on all single- and dual-connected links.

By default, the role is determined by comparing the MAC addresses of the two sides of the peering link; the switch with the lower MAC address assumes the primary role. You can override this by setting the `priority` option for the peer link:

NCLU Commands | Linux Commands

```
cumulus@leaf01:~$ net add interface peerlink.4094 clag
priority 2048
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The switch with the lower priority value is given the primary role; the default value is 32768 and the range is between 0 and 65535.

When the `clagd` service exits during switch reboot or if you stop the service on the primary switch, the peer switch that is in the secondary role becomes the primary.

However, if the primary switch goes down without stopping the `clagd` service for any reason, or if the peer link goes down, the secondary switch does **not** change its role. If the peer switch is determined to not be alive, the switch in the secondary role rolls back the LACP system ID to be the bond interface MAC address instead of the MLAG system MAC address (`clagd-sys-mac`) and the switch in primary role uses the MLAG system MAC address as the LACP system ID on the bonds.

Set clagctl Timers

The `clagd` service has a number of timers that you can tune for enhanced performance:

Timer	Description
`--reloadTimer <seconds>`	The number of seconds to wait for the peer switch to become active. If the peer switch does not become active after the timer expires, the MLAG bonds leave the initialization (protodown) state and become active. This provides `clagd` with sufficient time to determine whether the peer switch is coming up or if it is permanently unreachable. The default is 300 seconds.
`--peerTimeout <seconds>`	The number of seconds `clagd` waits without receiving any messages from the peer switch before it determines that the peer is no longer active. At this point, the switch reverts all configuration changes so that it operates as a standard non-MLAG switch. This includes removing all statically assigned MAC addresses, clearing the egress forwarding mask, and allowing addresses to move from any port to the peer port. After a message is again received from the peer, MLAG operation restarts. If this parameter is not specified, `clagd` uses ten times the local

Timer	Description
	`lacpPoll` value.
`--initDelay <seconds>`	The number of seconds `clagd` delays bringing up MLAG bonds and anycast IP addresses. The default is 180 seconds.
`--sendTimeout <seconds>`	The number of seconds `clagd` waits until the sending socket times out. If it takes longer than the `sendTimeout` value to send data to the peer, `clagd` generates an exception. The default is 30 seconds.
`--lacpPoll <seconds>`	The number of seconds `clagd` waits before obtaining local LACP information. The default is 2 seconds.

To set a timer:

NCLU Commands | Linux Commands

Run the `net add interface peerlink.4094 clag args <timer> <value>` command. The following example command sets the peerlink timer to 900 seconds:

```
cumulus@leaf01:~$ net add interface peerlink.4094 clag args
--peerTimeout 900
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

Configure MLAG with a Traditional Mode Bridge

To configure MLAG with a traditional mode bridge instead of a VLAN-aware mode bridge, you must configure the peer link and all dual-connected links as untagged (native) ports on a bridge (note the absence of any VLANs in the `bridge-ports` line and the lack of the `bridge-vlan-aware` parameter below):

```
...
auto br0
iface br0
    bridge-ports peerlink bond1 bond2
```

```
...
```

The following example shows you how to allow VLAN 10 across the peer link:

```
...
auto br0.10
iface br0.10
    bridge-ports peerlink.10 bond1.10 bond2.10
    bridge-stp on
...
```

Configure a Backup UDP Port

By default, Cumulus Linux uses UDP port 5342 with the backup IP address. To change the backup UDP port:

NCLU Commands | Linux Commands

```
cumulus@leaf01:~$ net add interface peerlink.4094 clag args
--backupPort 5400
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

Best Practices

Follow these best practices when configuring MLAG on your switches.

MTU and MLAG

The MTU in MLAG traffic is determined by the bridge MTU. Bridge MTU is determined by the lowest MTU setting of an interface that is a member of the bridge. If you want to set an MTU other than the default of 9216 bytes, you must configure the MTU on each physical interface and bond interface that is a member of every MLAG bridge in the entire bridged domain.

The following example commands set an MTU of 1500 for each of the bond interfaces (peerlink, uplink, bond1, bond2), which are members of bridge *bridge*:

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add bond peerlink mtu 1500
cumulus@switch:~$ net add bond uplink mtu 1500
cumulus@switch:~$ net add bond bond1 mtu 1500
cumulus@switch:~$ net add bond bond2 mtu 1500
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

STP and MLAG

Enabling STP in your layer 2 network and BPDU Guard on the host-facing bond interfaces is highly recommended.

- The STP global configuration must be the same on both peer switches.
- The STP configuration for dual-connected ports must be the same on both peer switches.
- The STP priority must be the same on both peer switches.
- To minimize convergence times when a link transitions to the forwarding state, configure the edge ports (for tagged and untagged frames) with PortAdminEdge and BPDU guard enabled.

Peer Link Sizing

The peer link carries very little traffic when compared to the bandwidth consumed by dataplane traffic. In a typical MLAG configuration, most every connection between the two switches in the MLAG pair is dual-connected

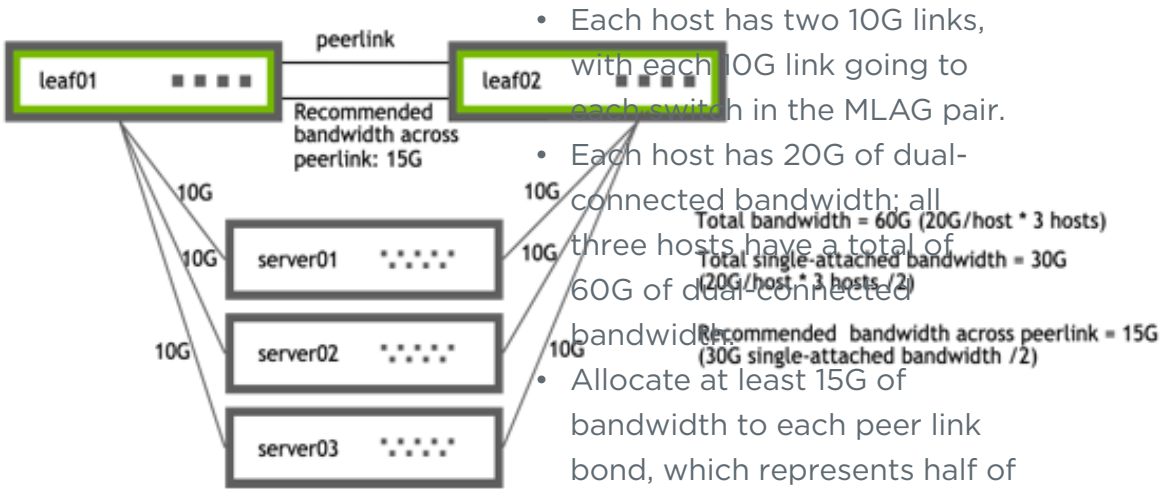
so the only traffic going across the peer link is traffic from the `clagd` process and some LLDP or LACP traffic; the traffic received on the peer link is not forwarded out of the dual-connected bonds.

However, there are some instances where a host is connected to only one switch in the MLAG pair; for example:

- You have a hardware limitation on the host where there is only one PCIE slot, and therefore, one NIC on the system, so the host is only single-connected across that interface.
- The host does not support 802.3ad and you cannot create a bond on it.
- You are accounting for a link failure, where the host becomes single connected until the failure is resolved.

Determine how much bandwidth is traveling across the single-connected interfaces and allocate half of that bandwidth to the peer link. On average, one half of the traffic destined to the single-connected host arrives on the switch directly connected to the single-connected host and the other half arrives on the switch that is not directly connected to the single-connected host. When this happens, only the traffic that arrives on the switch that is not directly connected to the single-connected host needs to traverse the peer link.

In addition, you might want to add extra links to the peer link bond to handle link failures in the peer link bond itself.



- Each host has two 10G links, with each 10G link going to each switch in the MLAG pair.
- Each host has 20G of dual-connected bandwidth; all three hosts have a total of 60G of dual-connected bandwidth.
- Allocate at least 15G of bandwidth to each peer link bond, which represents half of the single-connected bandwidth.

When planning for link failures for a full rack, you need only allocate enough bandwidth to meet your site strategy for handling failure scenarios. For example, for a full rack with 40 servers and two switches, you might plan for four to six servers to lose connectivity to a single switch and become single connected before you respond to the event. Therefore, if you have 40 hosts each with 20G of bandwidth dual-connected to the MLAG pair, you might allocate between 20G and 30G of bandwidth to the peer link, which accounts for half of the single-connected bandwidth for four to six hosts.

Peer Link Routing

When enabling a routing protocol in an MLAG environment, it is also necessary to manage the uplinks; by default MLAG is not aware of layer 3

uplink interfaces. If there is a peer link failure, MLAG does not remove static routes or bring down a BGP or OSPF adjacency unless you use a separate link state daemon such as `ifplugd`.

When you use MLAG with VRR, set up a routed adjacency across the peerlink.4094 interface. If a routed connection is not built across the peer link, during an uplink failure on one of the switches in the MLAG pair, egress traffic might not be forwarded if the destination is on the switch whose uplinks are down.

To set up the adjacency, configure a BGP or OSPF unnumbered peering, as appropriate for your network.

For BGP, use a configuration like this:

```
cumulus@switch:~$ net add bgp neighbor peerlink.4094 interface
remote-as internal
cumulus@switch:~$ net commit
```

For OSPF, use a configuration like this:

```
cumulus@switch:~$ net add interface peerlink.4094 ospf area
0.0.0.1
cumulus@switch:~$ net commit
```

If you are using EVPN and MLAG, you need to enable the EVPN address family across the peerlink.4094 interface as well:

```
cumulus@switch:~$ net add bgp neighbor peerlink.4094 interface
remote-as internal
cumulus@switch:~$ net add bgp l2vpn evpn neighbor peerlink.4094
activate
cumulus@switch:~$ net commit
```

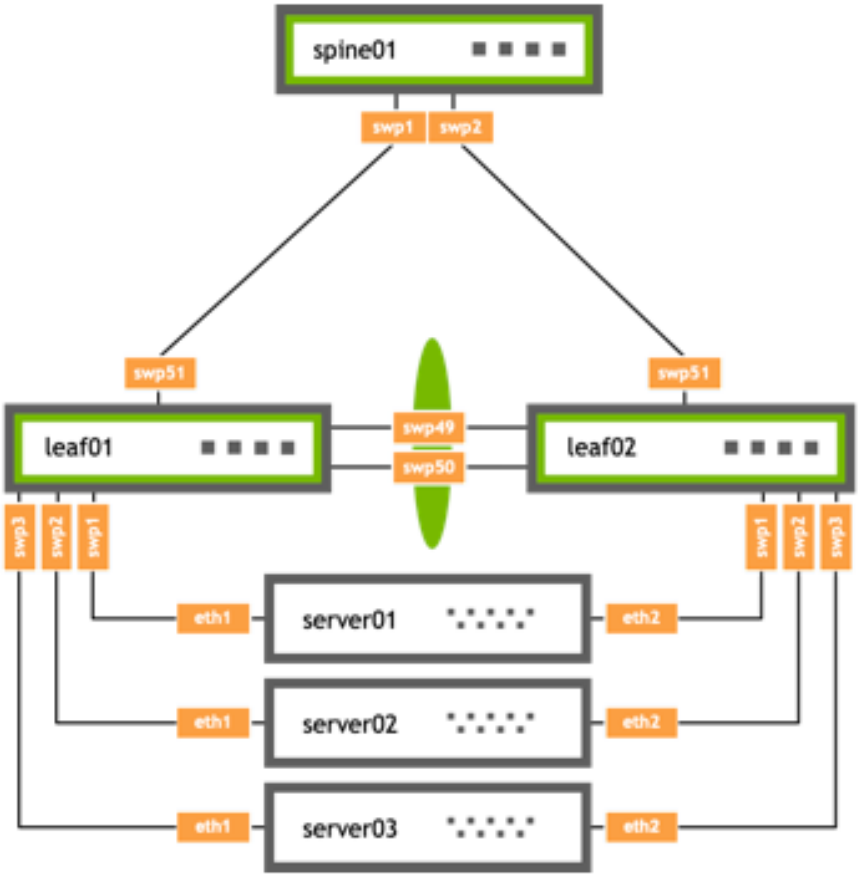
> **NOTE**
>
> If you use NCLU to create an iBGP peering across the peer link, the `net add bgp l2vpn evpn neighbor peerlink.4094 activate` command creates a new eBGP neighborship when one is already configured for iBGP. The existing iBGP configuration is still valid.

Configuration Examples

Basic Example

The example below shows a basic MLAG configuration, where:

- leaf01 and leaf02 are MLAG peers
- Three bonds are configured for MLAG, each with a single port, a peer link that is a bond with two member ports, and three VLANs on each port


spine01
swp1
swp2
swp51
swp51
leaf01
swp49
swp50
leaf02
swp3
swp2
swp1
swp1
swp2
swp3
eth1
server01
eth2
eth1
server02
eth2
eth1
server03
eth2

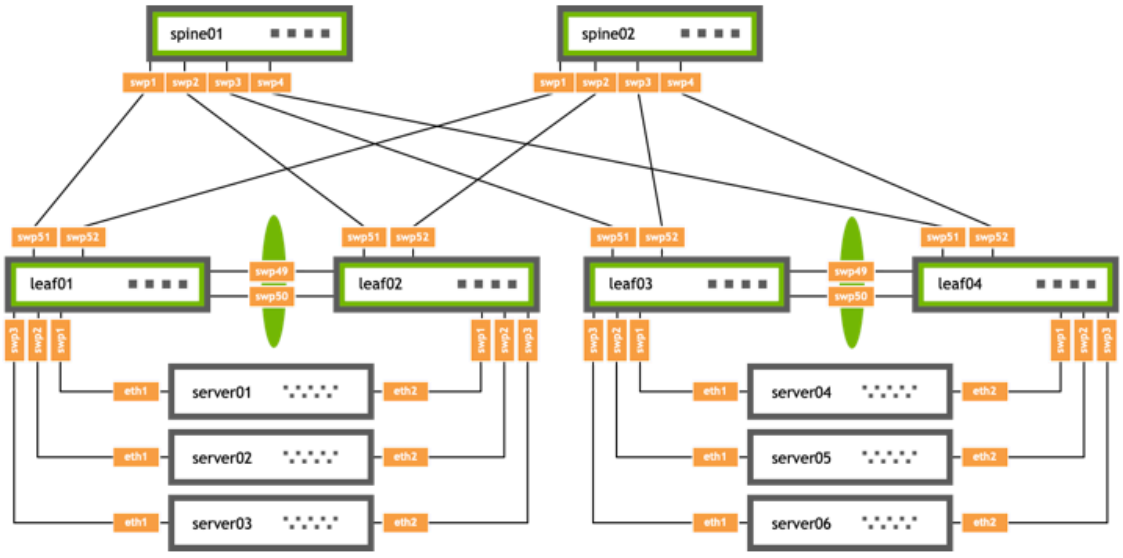
leaf01 leaf02 spine01

```
cumulus@leaf01:~$ cat /etc/network/interfaces

auto lo
iface lo inet loopback
    address 10.10.10.1/32

auto mgmt
iface mgmt
    vrf-table auto
    address 127.0.0.1/8
    address ::1/128

auto eth0
iface eth0 inet dhcp
    vrf mgmt

auto bridge
iface bridge
    bridge-ports peerlink
    bridge-ports bond1 bond2 bond3
    bridge-vids 10 20 30
    bridge-vlan-aware yes

auto vlan10
iface vlan10
    address 10.1.10.2/24
    vlan-raw-device bridge
    vlan-id 10
```

MLAG and BGP Example

The example configuration below shows an MLAG configuration where:

- leaf01 and leaf02 are MLAG peers, and leaf03 and leaf04 are are MLAG peers
- Three bonds are configured for MLAG, each with a single port, a peer link that is a bond with two member ports, and three VLANs on each port
- BGP unnumbered is configured on the leafs and spines with a routed adjacency across the `peerlink.4094` interface



/etc/network/interfaces

leaf01 leaf02 leaf03 leaf04 spine01 spine02

```
cumulus@leaf01:~$ cat /etc/network/interfaces
auto lo
iface lo inet loopback
    address 10.10.10.1/32

auto mgmt
iface mgmt
    vrf-table auto
    address 127.0.0.1/8
    address ::1/128

auto eth0
iface eth0 inet dhcp
    vrf mgmt

auto bridge
iface bridge
    bridge-ports peerlink
    bridge-ports bond1 bond2 bond3
    bridge-vids 10 20 30
    bridge-vlan-aware yes

auto vlan10
iface vlan10
    address 10.1.10.2/24
    vlan-raw-device bridge
    vlan-id 10

auto vlan20
```

/etc/frr/frr.conf

leaf01 leaf02 leaf03 leaf04 spine01 spine02

```
cumulus@leaf01:~$ cat /etc/frr/frr.conf
...
service integrated-vtysh-config
!
log syslog informational
!
router bgp 65101
 bgp router-id 10.10.10.1
 bgp bestpath as-path multipath-relax
 neighbor underlay peer-group
 neighbor underlay remote-as external
 neighbor peerlink.4094 interface remote-as internal
 neighbor swp51 interface peer-group underlay
 neighbor swp52 interface peer-group underlay
 !
 !
 address-family ipv4 unicast
  redistribute connected
 exit-address-family
 !
!
line vty
!
```

Troubleshooting

Use the following troubleshooting tips to check that MLAG is configured and working correctly.

Check MLAG Status

To check the status of your MLAG configuration, run the NCLU `net show clag` command or the Linux `clagctl` command. For example:

```
cumulus@switch:~$ net show clag
The peer is alive
    Peer Priority, ID, and Role: 4096 44:38:39:FF:00:01 primary
     Our Priority, ID, and Role: 8192 44:38:39:FF:00:02
secondary
          Peer Interface and IP: peerlink.4094 linklocal
                      Backup IP: 192.168.1.12 (inactive)
                    System MAC: 44:38:39:FF:00:01


CLAG Interfaces
Our Interface      Peer Interface     CLAG Id
Conflicts              Proto-Down Reason
----------------   ----------------   -------
--------------------   -----------------
           bond1   bond1              1
```

```
-                       -
           bond2    bond2              2
-                       -
```

Show All MLAG Settings

To see all MLAG settings, run the `clagctl params` command:

```
cumulus@leaf01:~$ clagctl params
clagVersion = 1.4.0
clagDataVersion = 1.4.0
clagCmdVersion = 1.1.0
peerIp = linklocal
peerIf = peerlink.4094
sysMac = 44:38:39:be:ef:aa
lacpPoll = 2
currLacpPoll = 2
peerConnect = 1
cmdConnect = 1
peerLinkPoll = 1
switchdReadyTimeout = 120
reloadTimer = 300
periodicRun = 4
```

```
priority = 1000
quiet = False
debug = 0x0
verbose = False
log = syslog
vm = True
peerPort = 5342
peerTimeout = 20
initDelay = 180
sendTimeout = 30
sendBufSize = 65536
forceDynamic = False
dormantDisable = False
redirectEnable = False
backupIp = 10.10.10.2
backupVrf = None
backupPort = 5342
vxlanAnycast = None
neighSync = True
permanentMacSync = True
cmdLine = /usr/sbin/clagd --daemon linklocal peerlink.4094
44:38:39:BE:EF:AA --priority 1000 --backupIp 10.10.10.2
peerlinkLearnEnable = False
```

View the MLAG Log File

By default, when running, the `clagd` service logs status messages to the `/var/log/clagd.log` file and to `syslog`. Example log file output is shown below:

```
cumulus@spine01:~$ sudo tail /var/log/clagd.log
2016-10-03T20:31:50.471400+00:00 spine01 clagd[1235]: Initial
config loaded
2016-10-03T20:31:52.479769+00:00 spine01 clagd[1235]: The peer
switch is active.
2016-10-03T20:31:52.496490+00:00 spine01 clagd[1235]: Initial
data sync to peer done.
2016-10-03T20:31:52.540186+00:00 spine01 clagd[1235]: Role is
now primary; elected
2016-10-03T20:31:54.250572+00:00 spine01 clagd[1235]:
HealthCheck: role via backup is primary
2016-10-03T20:31:54.252642+00:00 spine01 clagd[1235]:
HealthCheck: backup active
2016-10-03T20:31:54.537967+00:00 spine01 clagd[1235]: Initial
data sync from peer done.
2016-10-03T20:31:54.538435+00:00 spine01 clagd[1235]: Initial
handshake done.
2016-10-03T20:31:58.527464+00:00 spine01 clagd[1235]: leaf03-04
is now dual connected.
```

```
2016-10-03T22:47:35.255317+00:00 spine01 clagd[1235]: leaf01-02
is now dual connected.
```

Monitor the clagd Service

Due to the critical nature of the `clagd` service, `systemd` continuously monitors its status by receiving notify messages every 30 seconds. If the `clagd` service terminates or becomes unresponsive for any reason and `systemd` receives no messages after 60 seconds, `systemd` restarts the `clagd` service. `systemd` logs these failures in the `/var/log/syslog` file and, on the first failure, also generates a `cl-support`file.

Monitoring is configured and enabled automatically as long as the `clagd` service is enabled (the peer IP address (`clagd-peer-ip`), the MLAG system MAC address (`clagd-sys-mac`), and the backup IP address (`clagd-backup-ip`) are configured for an interface) and the `clagd` service is running. If you stop `clagd` with the `systemctl stop clagd.service` command, `clagd` monitoring also stops.

You can check if `clagd` is enabled and running with the `cl-service-summary` or the `systemctl status` command:

```
cumulus@switch:~$ cl-service-summary
```

```
Service cron               enabled    active
Service ssh                enabled    active
Service syslog             enabled    active
Service asic-monitor       enabled    inactive
Service clagd              enabled    active
...
```

```
cumulus@switch:~$ systemctl status clagd.service
 ● clagd.service - Cumulus Linux Multi-Chassis LACP Bonding
Daemon
    Loaded: loaded (/lib/systemd/system/clagd.service; enabled)
    Active: active (running) since Mon 2016-10-03 20:31:50 UTC;
4 days ago
        Docs: man:clagd(8)
    Main PID: 1235 (clagd)
    CGroup: /system.slice/clagd.service
            ├─1235 /usr/bin/python /usr/sbin/clagd --daemon
169.254.255.2 peerlink.4094 44:38:39:FF:40:90 --prior...
            └─15795 /usr/share/mgmt-vrf/bin/ping6 -L -c 1
ff02::1 -I peerlink.409

Feb 01 23:19:30 leaf01 clagd[1717]: Cleanup is executing.
Feb 01 23:19:31 leaf01 clagd[1717]: Cleanup is finished
```

```
Feb 01 23:19:31 leaf01 clagd[1717]: Beginning execution of
clagd version 1.3.0
Feb 01 23:19:31 leaf01 clagd[1717]: Invoked with: /usr/sbin/
clagd --daemon 169.254.255.2 peerlink.4094 44:38:39:FF:40:94 --
pri...168.0.12
Feb 01 23:19:31 leaf01 clagd[1717]: Role is now secondary
Feb 01 23:19:31 leaf01 clagd[1717]: Initial config loaded
Feb 01 23:19:31 leaf01 systemd[1]: Started Cumulus Linux Multi-
Chassis LACP Bonding Daemon.
Feb 01 23:24:31 leaf01 clagd[1717]: HealthCheck: reload timeout.
Feb 01 23:24:31 leaf01 clagd[1717]: Role is now primary; Reload
timeout
...
```

Large Packet Drops on the Peer Link Interface

A large volume of packet drops across one of the peer link interfaces can be expected. These drops serve to prevent looping of BUM (broadcast, unknown unicast, multicast) packets. When a packet is received across the peer link, if the destination lookup results in an egress interface that is a dual-connected bond, the switch does not forward the packet (to prevent loops). This results in a drop being recorded on the peer link.

To check packet drops across peer link interfaces, run the following command:

NCLU Commands | Linux Commands

Run the `net show counters` command. The number of dropped packets is displayed in the `RX_DRP` column.

```
cumulus@switch:~$ net show counters

Kernel Interface table
Iface            MTU    RX_OK    RX_ERR    RX_DRP
RX_OVR    TX_OK    TX_ERR    TX_DRP    TX_OVR  Flg
-------------  -----  -------  --------  --------
--------  -------  --------  --------  --------  -----
bond1           9216        0         0         0
0      542         0         0         0  BMmU
bond2           9216        0         0         0
0      542         0         0         0  BMmU
bridge          9216        0         0         0
0       17         0         0         0  BMRU
eth0            1500     5497         0         0
0      933         0         0         0  BMRU
lo             65536     1328         0         0
0     1328         0         0         0  LRU
mgmt           65536      790         0         0
0        0         0        33         0  OmRU
peerlink        9216    23626         0       520
0    23665         0         0         0  BMmRU
peerlink.4094   9216     8013         0         0
0     8017         0         0         0  BMRU
swp1            9216        5         0         0
0      553         0         0         0  BMsRU
swp2            9216        3         0         0
```

Peer Link Interfaces and the protodown State

In addition to the standard UP and DOWN administrative states, an interface that is a member of an MLAG bond can also be in a `protodown` state. When MLAG detects a problem that might result in connectivity issues, it can put that interface into `protodown` state. Such connectivity issues include:

- When the peer link goes down but the peer switch is up (the backup link is active).
- When the bond is configured with an MLAG ID but the `clagd` service is not running (either deliberately stopped or crashes).
- When an MLAG-enabled node is booted or rebooted, the MLAG bonds are placed in a `protodown` state until the node establishes a connection to its peer switch, or five minutes have elapsed.

When an interface goes into a `protodown` state, it results in a local OPER DOWN (carrier down) on the interface.

To show an interface in `protodown` state, run the NCLU `net show bridge link` command or the Linux `ip link show` command. For example:

```
cumulus@switch:~$ net show bridge link
3: swp1 state DOWN: <NO-CARRIER,BROADCAST,MULTICAST,MASTER,UP>
mtu 9216 master pfifo_fast master host-bond1 state DOWN mode
DEFAULT qlen 500 protodown on
```

```
    link/ether 44:38:39:00:69:84 brd ff:ff:ff:ff:ff:ff
```

LACP Partner MAC Address Duplicate or Mismatch

Cumulus Linux puts interfaces in a protodown state under the following conditions:

- When there is an LACP partner MAC address mismatch. For example if a bond comes up with a `clag-id` and the peer is using a bond with the same `clag-id` but a different LACP partner MAC address. The `clagctl` command output shows the protodown reason as a `partner-mac-mismatch`.
- When there is a duplicate LACP partner MAC address. For example, when there are multiple LACP bonds between the same two LACP endpoints. The `clagctl` command output shows the protodown reason as a `duplicate-partner-mac`.

  To prevent a bond from coming up when an MLAG bond with an LACP partner MAC address already in use comes up, use the `--clag-args --allowPartnerMacDup False` option. This option puts the slaves of that bond interface in a protodown state and the `clagctl` output shows the protodown reason as a `duplicate-partner-mac`.

After you make the necessary cable or configuration changes to avoid the protodown state and you want MLAG to reevaluate the LACP partners, use

the `clagctl clearconflictstate` command to remove `duplicate-partner-mac` or `partner-mac-mismatch` from the protodowned bonds, allowing them to come back up.

Related Information

- MLAG Redundancy Scenarios
- Compare Traditional Bridge Mode to VLAN-aware Bridge Mode

LACP Bypass

On Cumulus Linux, *LACP bypass* allows a bond configured in 802.3ad mode to become active and forward traffic even when there is no LACP partner. For example, you can enable a host that does not have the capability to run LACP to PXE boot while connected to a switch on a bond configured in 802.3ad mode. After the pre-boot process completes and the host is capable of running LACP, the normal 802.3ad link aggregation operation takes over.

LACP Bypass All-active Mode

In *all-active* mode, when a bond has multiple slave interfaces, each bond slave interface operates as an active link while the bond is in bypass mode. This is useful during PXE boot of a server with multiple NICs, when you cannot determine beforehand which port needs to be active.

NOTE

- All-active mode is *not* supported on bonds that are *not* specified as bridge ports on the switch.
- STP does not run on the individual bond slave interfaces when the LACP bond is in all-active mode. Only use all-active mode on host-facing LACP bonds. Configuring STP

BPDU guard together with all-active mode is highly recommended.

- In an MLAG deployment where bond slaves of a host are connected to two switches and the bond is in all-active mode, all the slaves of bond are active on both the primary and secondary MLAG nodes.
- LACP bypass is supported with EVPN multihoming.
- `priority mode`, `bond-lacp-bypass-period`, `bond-lacp-bypass-priority`, and `bond-lacp-bypass-all-active` are not supported.

Configure LACP Bypass

To enable LACP bypass on the host-facing bond, set `bond-lacp-bypass-allow` to *yes*.

NCLU Commands | Linux Commands

The following commands create a VLAN-aware bridge with LACP bypass enabled:

```
cumulus@switch:~$ net add bond bond1 bond slaves
swp51s2,swp51s3
cumulus@switch:~$ net add bond bond1 clag id 1
cumulus@switch:~$ net add bond bond1 bond lacp-bypass-allow
cumulus@switch:~$ net add bond bond1 stp bpduguard
cumulus@switch:~$ net add bridge bridge ports
bond1,bond2,bond3,bond4,peer5
cumulus@switch:~$ net add bridge bridge vids 100-105
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

To check the status of the configuration, run the following commands.

NCLU Commands | Linux Commands

Run the `net show interface <bond>` command on the bond and its slave interfaces:

```
cumulus@switch:~$ net show interface bond1

    Name    MAC                Speed   MTU   Mode
--  ------ ----------------- ------- ----- ----------
UP  bond1  44:38:39:00:00:5b 1G      1500  Bond/Trunk

Bond Details
----------------- -------------------------
Bond Mode:          LACP
Load Balancing:     Layer3+4
Minimum Links:      1
In CLAG:            CLAG Active
LACP Sys Priority:
LACP Rate:          Fast Timeout
LACP Bypass:        LACP Bypass Not Supported

    Port        Speed     TX    RX    Err    Link Failures
-- --------   ------- ---- ---- ----- ---------------
UP swp51s2(P) 1G          0     0      0                0
UP swp51s3(P) 1G          0     0      0                0



All VLANs on L2 Port
---------------------
100-105
```

To verify that LACP bypass is enabled on a bond and its slave interfaces, use the `cat` command:

```
cumulus@switch:~$ cat /sys/class/net/bond1/bonding/lacp_bypass
on 1
cumulus@switch:~$ cat /sys/class/net/bond1/bonding/slaves
swp51 swp52
cumulus@switch:~$ cat /sys/class/net/swp52/bonding_slave/
ad_rx_bypass
1
cumulus@switch:~$ cat /sys/class/net/swp51/bonding_slave/
ad_rx_bypass
1
```

Example LACP Bypass Configuration

The following configuration shows LACP bypass enabled for multiple active interfaces (all-active mode) with a bridge in traditional bridge mode:

```
...
auto bond1
iface bond1
    bond-slaves swp3 swp4
    bond-lacp-bypass-allow 1
```

```
auto br0
iface br0
    bridge-ports bond1 bond2 bond3 bond4 peer5
    mstpctl-bpduguard bond1=yes
...
```

Virtual Router Redundancy - VRR and VRRP

Cumulus Linux provides the option of using Virtual Router Redundancy (VRR) or Virtual Router Redundancy Protocol (VRRP).

- **VRR** enables hosts to communicate with any redundant router without reconfiguration, by running dynamic router protocols or router redundancy protocols. Redundant routers respond to Address Resolution Protocol (ARP) requests from hosts. Routers are configured to respond in an identical manner, but if one fails, the other redundant routers continue to respond, leaving the hosts with the impression that nothing has changed. VRR is typically used in an MLAG configuration.

  Use VRR when you have multiple devices connected to a single logical connection, such as an MLAG bond. A device connected to an MLAG bond believes there is a single device on the other end of the bond and only forwards one copy of the transit frames. If this frame is destined to the virtual MAC address and you are running VRRP, it is possible that the frame is sent to the link connected to the VRRP standby device, which will not forward the frame appropriately. By having the virtual MAC active on both MLAG devices, it ensures either MLAG device handles the frame it receives correctly.

- **VRRP** allows a single virtual default gateway to be shared between two or more network devices in an active/standby configuration. The physical VRRP router that forwards packets at any given time is called the master.

If this VRRP router fails, another VRRP standby router automatically takes over as master. VRRP is used in a non-MLAG configuration.

Use VRRP when you have multiple distinct devices that connect to a layer 2 segment through multiple logical connections (not through a single bond). VRRP elects a single active forwarder that *owns* the virtual MAC address while it is active. This prevents the forwarding database of the layer 2 domain from continuously updating in response to MAC flaps as frames sourced from the virtual MAC address are received from discrete logical connections.

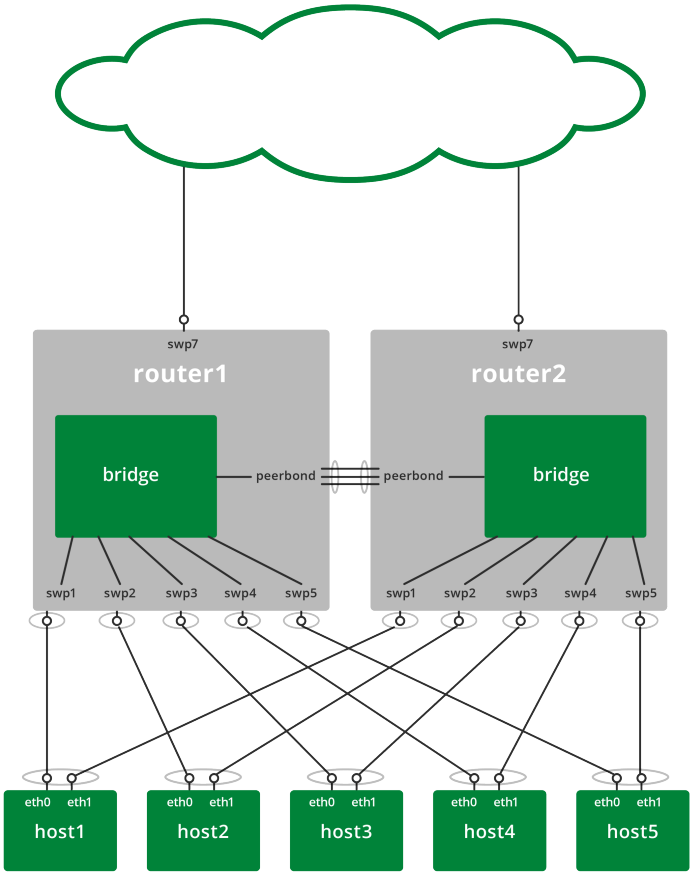
> **NOTE**
>
> You cannot configure both VRR and VRRP on the same switch.

VRR

The diagram below illustrates a basic VRR-enabled network configuration. The network includes several hosts and two routers running Cumulus Linux configured with multi-chassis link aggregation (MLAG).

> **NOTE**
>
> Cumulus Linux only supports VRR on switched virtual interfaces

(SVIs). VRR is not supported on physical interfaces or virtual subinterfaces.



A production implementation has many more server hosts and network connections than are shown here. However, this basic configuration provides a complete description of the important aspects of the VRR setup.

As the bridges in each of the redundant routers are connected, they each receive and reply to ARP requests for the virtual router IP address.

Each ARP request made by a host receives replies from each router; these replies are identical, and the host receiving the replies either ignores replies after the first, or accepts them and overwrites the previous identical reply.

A range of MAC addresses is reserved for use with VRR to prevent MAC address conflicts with other interfaces in the same bridged network. The reserved range is `00:00:5E:00:01:00` to `00:00:5E:00:01:ff`.

Use MAC addresses from the reserved range when configuring VRR.

The reserved MAC address range for VRR is the same as for the Virtual Router Redundancy Protocol (VRRP), as they serve similar purposes.

Configure the Routers

The routers implement the layer 2 network interconnecting the hosts and the redundant routers. To configure the routers, add a bridge with the following interfaces to each router:

- One bond interface or switch port interface to each host. For networks using MLAG, use bond interfaces. Otherwise, use switch port interfaces.
- One or more interfaces to each peer router. To accommodate higher bandwidth between the routers and to offer link redundancy, multiple inter-peer links are typically bonded interfaces. The VLAN interface must have unique IP addresses for both the physical (the `address` option below) and virtual (the `address-virtual` option below) interfaces; the unique address is used when the switch initiates an ARP request.

NCLU Commands | Linux Commands

The example NCLU commands below create a VLAN-aware bridge interface for a VRR-enabled network:

```
cumulus@switch:~$ net add bridge
cumulus@switch:~$ net add vlan 500 ip address 192.0.2.252/24
cumulus@switch:~$ net add vlan 500 ip address-virtual
00:00:5e:00:01:00 192.0.2.254/24
cumulus@switch:~$ net add vlan 500 ipv6 address 2001:db8::1/
32
cumulus@switch:~$ net add vlan 500 ipv6 address-virtual
00:00:5e:00:01:00 2001:db8::f/32
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Configure the Hosts

Each host must have two network interfaces. The routers configure the interfaces as bonds running LACP; the hosts must also configure the two interfaces using teaming, port aggregation, port group, or EtherChannel running LACP. Configure the hosts either statically or with DHCP, with a gateway address that is the IP address of the virtual router; this default gateway address never changes.

Configure the links between the hosts and the routers in *active-active* mode for First Hop Redundancy Protocol.

Example VRR Configuration with MLAG

To create an MLAG configuration that incorporates VRR, use a configuration similar to the following.

> **NOTE**
>
> The following examples uses a single virtual MAC address for all VLANs. You can add a unique MAC address for each VLAN, but this is not necessary.

leaf01 leaf02 server01 server02

```
cumulus@leaf01:~$ net add interface eth0 ip address
192.168.0.21
cumulus@leaf01:~$ net add bond server01 bond slaves swp1-2
cumulus@leaf01:~$ net add bond server01 clag id 1
cumulus@leaf01:~$ net add bond server01 mtu 9216
cumulus@leaf01:~$ net add bond server01 alias LACP
etherchannel to uplink on server01
cumulus@leaf01:~$ net add bond peerlink bond slaves swp49-50
cumulus@leaf01:~$ net add interface peerlink.4094
peerlink.4094
cumulus@leaf01:~$ net add interface peerlink.4094 ip
address 169.254.255.1/30
cumulus@leaf01:~$ net add interface peerlink.4094 clag peer-
ip 169.254.255.2
cumulus@leaf01:~$ net add interface peerlink.4094 clag
backup-ip 192.168.0.22
cumulus@leaf01:~$ net add interface peerlink.4094 clag sys-
mac 44:38:39:FF:40:90
cumulus@leaf01:~$ net add bridge bridge ports
server01,peerlink
cumulus@leaf01:~$ net add bridge stp treeprio 4096
cumulus@leaf01:~$ net add vlan 100 ip address 10.0.1.2/24
cumulus@leaf01:~$ net add vlan 100 ip address-virtual
00:00:5E:00:01:01 10.0.1.1/24
cumulus@leaf01:~$ net add vlan 200 ip address 10.0.2.2/24
cumulus@leaf01:~$ net add vlan 200 ip address-virtual
00:00:5E:00:01:01 10.0.2.1/24
cumulus@leaf01:~$ net add vlan 300 ip address 10.0.3.2/24
cumulus@leaf01:~$ net add vlan 300 ip address-virtual
```

VRRP

VRRP allows for a single virtual default gateway to be shared among two or more network devices in an active standby configuration. The VRRP router that forwards packets at any given time is called the master. If this VRRP router fails, another VRRP standby router automatically takes over as master. The master sends VRRP advertisements to other VRRP routers in the same virtual router group, which include the priority and state of the master. VRRP router priority determines the role that each virtual router plays and who becomes the new master if the master fails.

All virtual routers use 00:00:5E:00:01:XX for IPv4 gateways or 00:00:5E:00:02:XX for IPv6 gateways as their MAC address. The last byte of the address is the Virtual Router IDentifier (VRID), which is different for each virtual router in the network. This MAC address is used by only one physical router at a time, which replies with this address when ARP requests or neighbor solicitation packets are sent for the IP addresses of the virtual router.

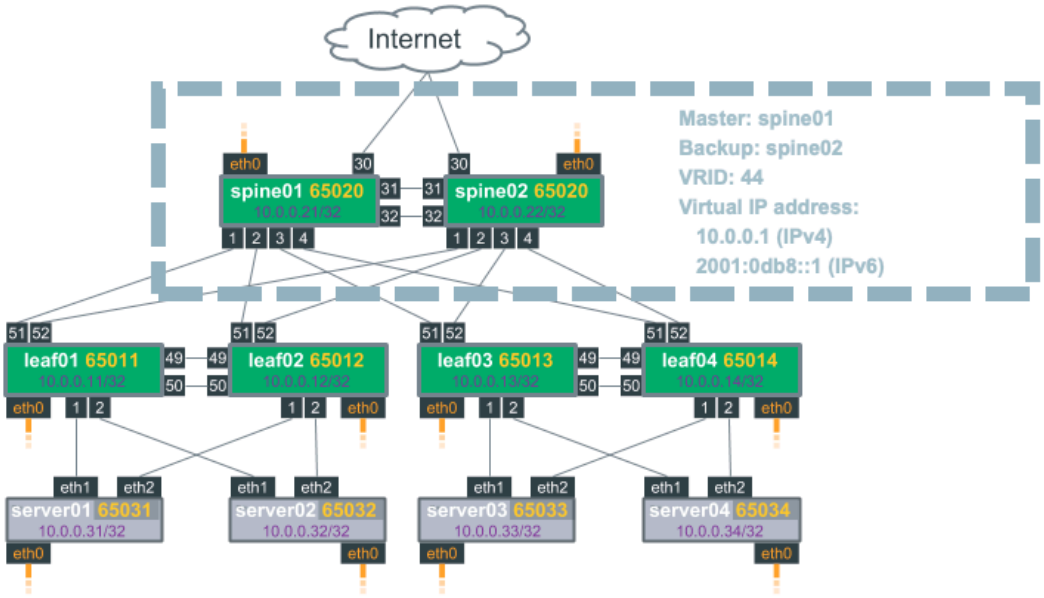
NOTE

- Cumulus Linux supports both VRRPv2 and VRRPv3. The default protocol version is VRRPv3.
- 255 virtual routers are supported per switch.

- VRRP is not supported currently in an MLAG environment or with EVPN.
- To configure VRRP on an SVI, you need to edit the `/etc/frr/frr.conf` file; The NCLU commands are not supported for SVIs.

RFC 5798 describes VRRP in detail.

The following example illustrates a basic VRRP configuration.



Configure VRRP

To configure VRRP, specify the following information on each switch:

- **A virtual router ID (VRID) that identifies the group of VRRP routers**. You must specify the same ID across all virtual routers in the group.
- **One or more virtual IP addresses that are assigned to the virtual router group**. These are IP addresses that do not directly connect to a specific interface. Inbound packets sent to a virtual IP address are redirected to a physical network interface.

You can also set these optional parameters. If you do not set these parameters, the defaults are used:

Optional Parameter	Default Value	Description
`priority`	100	The priority level of the virtual router within the virtual router group, which determines the role that each virtual router plays and what happens if the master fails. Virtual routers have a priority between 1 and 254; the router with the highest priority becomes the master.

Optional Parameter	Default Value	Description
`advertisement interval`	1000 milliseconds	The advertisement interval is the interval between successive advertisements by the master in a virtual router group. You can specify a value between 10 and 40950.
`preempt`	enabled	Preempt mode lets the router take over as master for a virtual router group if it has a higher priority than the current master. Preempt mode is enabled by default. To disable preempt mode, you need to edit the `/etc/frr/frr.conf` file and add the line `no vrrp <VRID> preempt` to the interface stanza, then restart the FRR service.

The NCLU commands write VRRP configuration to the `/etc/network/interfaces` file and the `/etc/frr/frr.conf` file.

The following example commands configure two switches (spine01 and spine02) that form one virtual router group (VRID 44) with IPv4 address 10.0.0.1/24 and IPv6 address 2001:0db8::1/64. *spine01* is the master; it has a priority of 254. *spine02* is the backup VRRP router.

NCLU Commands | Linux and vtysh Commands

spine01

```
cumulus@spine01:~$ net add interface swp1 vrrp 44 10.0.0.1/
24
cumulus@spine01:~$ net add interface swp1 vrrp 44
2001:0db8::1/64
cumulus@spine01:~$ net add interface swp1 vrrp 44 priority
254
cumulus@spine01:~$ net add interface swp1 vrrp 44
advertisement-interval 5000
cumulus@spine01:~$ net pending
cumulus@spine01:~$ net commit
```

spine02

```
cumulus@spine02:~$ net add interface swp1 vrrp 44 10.0.0.1/
24
cumulus@spine02:~$ net add interface swp1 vrrp 44
2001:0db8::1/64
cumulus@spine02:~$ net pending
cumulus@spine02:~$ net commit
```

The NCLU and vtysh commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
cumulus@spine01:~$ sudo cat /etc/frr/frr.conf
...
interface swp1
vrrp 44
vrrp 44 advertisement-interval 5000
vrrp 44 priority 254
vrrp 44 ip 10.0.0.1
vrrp 44 ipv6 2001:0db8::1
...
```

Show VRRP Configuration

To show virtual router information on a switch, run the NCLU `net show vrrp <VRID>` command or the vtysh `show vrrp <VRID>` command. For example:

```
cumulus@spine01:~$ net show vrrp 44
Virtual Router ID                    44
Protocol Version                     3
Autoconfigured                       No
Shutdown                             No
Interface                            swp1
```

```
 VRRP interface (v4)                vrrp4-3-1
VRRP interface (v6)                 vrrp6-3-1
Primary IP (v4)
Primary IP (v6)                     fe80::54df:e543:5c12:7762
Virtual MAC (v4)                    00:00:5e:00:01:01
Virtual MAC (v6)                    00:00:5e:00:02:01
Status (v4)                         Master
Status (v6)                         Master
Priority                            254
Effective Priority (v4)             254
Effective Priority (v6)             254
Preempt Mode                        Yes
Accept Mode                         Yes
Advertisement Interval              5000 ms
Master Advertisement Interval (v4)  0 ms
Master Advertisement Interval (v6)  5000 ms
Advertisements Tx (v4)              17
Advertisements Tx (v6)              17
Advertisements Rx (v4)              0
Advertisements Rx (v6)              0
Gratuitous ARP Tx (v4)              1
Neigh. Adverts Tx (v6)              1
State transitions (v4)              2
State transitions (v6)              2
```

```
Skew Time (v4)                      0 ms
Skew Time (v6)                      0 ms
Master Down Interval (v4)           0 ms
Master Down Interval (v6)           0 ms
IPv4 Addresses                      1
. . . . . . . . . . . . . . . . . . 10.0.0.1
IPv6 Addresses                      1
. . . . . . . . . . . . . . . . . . 2001:0db8::1
```

IGMP and MLD Snooping

IGMP (Internet Group Management Protocol) and MLD (Multicast Listener Discovery) snooping are implemented in the bridge driver in the Cumulus Linux kernel and are enabled by default. IGMP snooping processes IGMP v1/v2/v3 reports received on a bridge port in a bridge to identify the hosts which would like to receive multicast traffic destined to that group.

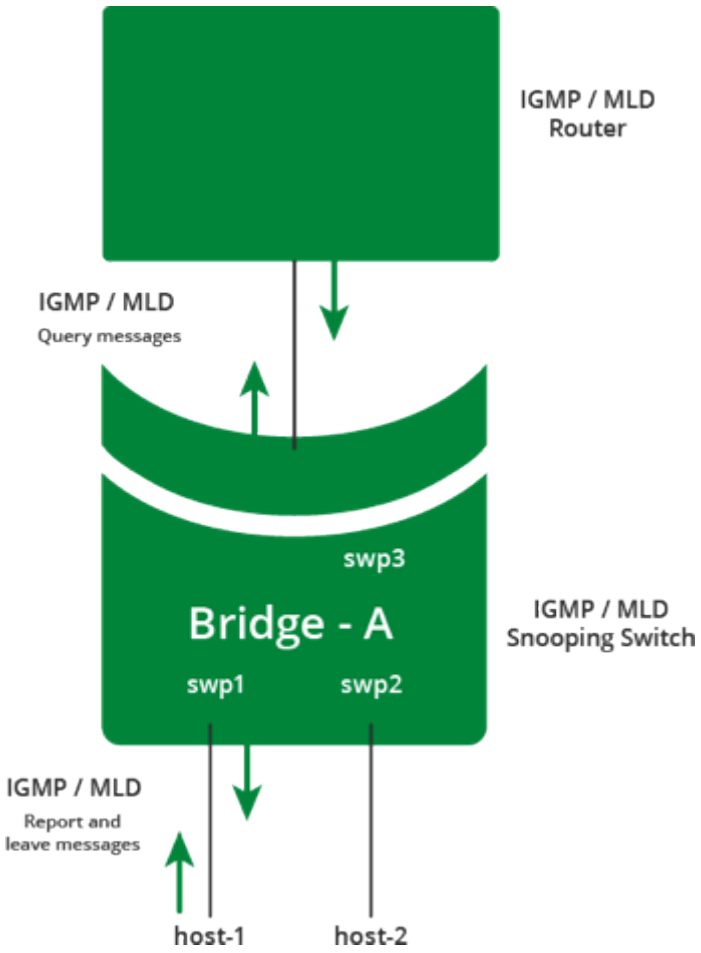
> **NOTE**
>
> IGMP and MLD snooping is supported over VXLAN bridges; however, this feature is *not* enabled by default. To enable IGMP and MLD over VXLAN, see Configure IGMP/MLD Snooping over VXLAN.

When an IGMPv2 leave message is received, a group specific query is sent to identify if there are any other hosts interested in that group, before the group is deleted.

An IGMP query message received on a port is used to identify the port that is connected to a router and is interested in receiving multicast traffic.

MLD snooping processes MLD v1/v2 reports, queries and v1 done messages for IPv6 groups. If IGMP or MLD snooping is disabled, multicast traffic gets flooded to all the bridge ports in the bridge. Similarly, in the absence of receivers in a VLAN, multicast traffic is flooded to all ports in the VLAN. The multicast group IP address is mapped to a multicast MAC address and a

forwarding entry is created with a list of ports interested in receiving multicast traffic destined to that group.



Configure IGMP/MLD Snooping over VXLAN

Cumulus Linux supports IGMP/MLD snooping over VXLAN bridges, where VXLAN ports are set as router ports, on Broadcom switches.

To enable IGMP/MLD snooping over VXLAN:

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add bridge mybridge mcsnoop yes
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Consider also configuring IGMP/MLD querier. See Configure IGMP/MLD Querier, below.

To disable IGMP/MLD snooping over VXLAN, run the `net add bridge <bridge> mcsnoop no` command.

Configure IGMP/MLD Querier

If no multicast router is sending queries to configure IGMP/MLD querier on the switch, you can add a configuration similar to the following in the `/etc/network/interfaces` file. To enable IGMP and MLD snooping for a bridge, set `bridge-mcquerier` to *1* in the bridge stanza. By default, the source IP address of IGMP queries is 0.0.0.0.

For an explanation of the relevant parameters, see the `ifupdown-addons-interfaces` man page.

For a VLAN-aware bridge, use a configuration like the following:

```
...
auto bridge.100
vlan bridge.100
  bridge-igmp-querier-src 123.1.1.1

auto bridge
iface bridge
  bridge-ports swp1 swp2 swp3
  bridge-vlan-aware yes
  bridge-vids 100 200
  bridge-pvid 1
  bridge-mcquerier 1
...
```

For a VLAN-aware bridge, like *bridge* in the above example, to enable querier functionality for VLAN 100 in the bridge, set `bridge-mcquerier` to *1* in the bridge stanza and set `bridge-igmp-querier-src` to *123.1.1.1* in the bridge.100 stanza. 123.1.1.1 is a typical loopback IP address.

You can specify a range of VLANs as well. For example:

```
...
auto bridge.[1-200]
```

```
vlan bridge.[1-200]
  bridge-igmp-querier-src 123.1.1.1
...
```

For a bridge in traditional mode, you can set the source IP address of the queries to be the bridge IP address — configure `bridge-mcqifaddr 1`. Use a configuration like the following:

```
...
auto br0
iface br0
  address 192.0.2.10/24
  bridge-ports swp1 swp2 swp3
  bridge-vlan-aware no
  bridge-mcquerier 1
  bridge-mcqifaddr 1
...
```

Disable IGMP and MLD Snooping

To disable IGMP and MLD snooping, set the `bridge-mcsnoop` value to *0*.

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add bridge bridge mcsnoop no
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Troubleshooting

To show the IGMP/MLD snooping bridge state, run the `brctl showstp <bridge>` command:

```
cumulus@switch:~$ sudo brctl showstp bridge
  bridge
  bridge id              8000.7072cf8c272c
  designated root        8000.7072cf8c272c
  root port                 0                    path
cost                  0
  max age                20.00                 bridge max
age            20.00
  hello time              2.00                 bridge hello
time          2.00
  forward delay          15.00                 bridge forward
delay      15.00
```

```
  ageing time             300.00
  hello timer               0.00                  tcn
timer                   0.00
  topology change timer     0.00                  gc
timer                 263.70
  hash elasticity        4096                     hash
max                4096
  mc last member count      2                     mc init query
count        2
  mc router                 1                     mc
snooping                1
  mc last member timer      1.00                  mc membership
timer      260.00
  mc querier timer        255.00                  mc query
interval        125.00
  mc response interval     10.00                  mc init query
interval    31.25
  mc querier                0                     mc query
ifaddr            0
  flags


swp1 (1)
  port id                 8001
state                forwarding
```

```
  designated root        8000.7072cf8c272c       path
cost                  2
  designated bridge      8000.7072cf8c272c       message age
timer          0.00
  designated port        8001                    forward delay
timer        0.00
  designated cost           0                    hold
timer                 0.00
  mc router                 1                    mc fast
leave              0
  flags

swp2 (2)
  port id                8002
state                forwarding
  designated root        8000.7072cf8c272c       path
cost                  2
  designated bridge      8000.7072cf8c272c       message age
timer          0.00
  designated port        8002                    forward delay
timer        0.00
  designated cost           0                    hold
timer                 0.00
  mc router                 1                    mc fast
```

```
leave              0
  flags

swp3 (3)
  port id               8003
state                forwarding
  designated root       8000.7072cf8c272c       path
cost                  2
  designated bridge     8000.7072cf8c272c       message age
timer          0.00
  designated port       8003                    forward delay
timer        8.98
  designated cost          0                    hold
timer                 0.00
  mc router                1                    mc fast
leave              0
  flags
```

To show the groups and bridge port state, run the NCLU `net show bridge mdb` command or the Linux `bridge mdb show` command. To show detailed router ports and group information, run the `bridge -d -s mdb show` command:

```
cumulus@switch:~$ sudo bridge -d -s mdb show
  dev bridge port swp2 grp 234.10.10.10 temp 241.67
  dev bridge port swp1 grp 238.39.20.86 permanent 0.00
  dev bridge port swp1 grp 234.1.1.1 temp 235.43
  dev bridge port swp2 grp ff1a::9 permanent 0.00
  router ports on bridge: swp3
```

Related Information

- Wikipedia entry for IGMP snooping
- RFC 2236
- RFC 2710
- RFC 3376
- RFC 3810
- RFC 4541

Network Virtualization

Cumulus Linux supports a few forms of network virtualization.

VXLAN (Virtual Extensible LAN) is a standard overlay protocol that abstracts logical virtual networks from the physical network underneath. You can deploy simple and scalable layer 3 Clos architectures while extending layer 2 segments over that layer 3 network.

VXLAN uses a VLAN-like encapsulation technique to encapsulate MAC-based layer 2 Ethernet frames within layer 3 UDP packets. Each virtual network is a VXLAN logical layer 2 segment. VXLAN scales to 16 million segments - a 24-bit VXLAN network identifier (VNI ID) in the VXLAN header - for multi-tenancy.

Hosts on a given virtual network are joined together through an overlay protocol that initiates and terminates tunnels at the edge of the multi-tenant network, typically the hypervisor vSwitch or top of rack. These edge points are the VXLAN tunnel end points (VTEP).

Cumulus Linux can initiate and terminate VTEPs in hardware and supports wire-rate VXLAN. VXLAN provides an efficient hashing scheme across the IP fabric during the encapsulation process; the source UDP port is unique, with the hash based on layer 2 through layer 4 information from the original frame. The UDP destination port is the standard port 4789.

Cumulus Linux includes the native Linux VXLAN kernel support and integrates with controller-based overlay solutions like VMware NSX and Midokura MidoNet.

VXLAN is supported only on switches in the Cumulus Linux HCL using the Broadcom Tomahawk, Trident II, Trident II+ and Trident3 chipsets, as well as the Mellanox Spectrum chipset.

> **NOTE**
>
> VXLAN encapsulation over layer 3 subinterfaces (for example, swp3.111) or SVIs is not supported as traffic transiting through the switch may get dropped; even if the subinterface is used only for underlay traffic and does not perform VXLAN encapsulation, traffic may still get dropped. Only configure VXLAN uplinks as layer 3 interfaces without any subinterfaces (for example, swp3).
> The VXLAN tunnel endpoints cannot share a common subnet; there must be at least one layer 3 hop between the VXLAN source and destination.

Considerations

Cut-through Mode and Store and Forward Switching

On switches using Broadcom Tomahawk, Trident II, Trident II+, and Trident3 ASICs, Cumulus Linux supports store and forward switching for VXLANs but does **not** support cut-through mode.

On switches using Mellanox Spectrum ASICs, Cumulus Linux supports cut-through mode for VXLANs but does **not** support store and forward

switching.

MTU Size for Virtual Network Interfaces

The maximum transmission unit (MTU) size for a virtual network interface should be 50 bytes smaller than the MTU for the physical interfaces on the switch. For more information on setting MTU, read Layer 1 and Switch Port Attributes.

Layer 3 and Layer 2 VNIs Cannot Share the Same ID

A layer 3 VNI and a layer 2 VNI cannot have the same ID. If the VNI IDs are the same, the layer 2 VNI does not get created.

Related Information

- VXLAN - RFC 7348
- ovsdb-server

Ethernet Virtual Private Network - EVPN

VXLAN is the de facto technology for implementing network virtualization in the data center, enabling layer 2 segments to be extended over an IP core (the underlay). The initial definition of VXLAN (RFC 7348) did not include any control plane and relied on a flood-and-learn approach for MAC address learning.

Ethernet Virtual Private Network (EVPN) is a standards-based control plane for VXLAN defined in RFC 7432 and draft-ietf-bess-evpn-overlay that allows for building and deploying VXLANs at scale. It relies on multi-protocol BGP (MP-BGP) to exchange information and is based on BGP-MPLS IP VPNs (RFC 4364). It enables not only bridging between end systems in the same layer 2 segment but also routing between different segments (subnets). There is also inherent support for multi-tenancy. EVPN is often referred to as the means of implementing *controller-less VXLAN*.

The routing control plane (including EVPN) is installed as part of the FRRouting (FRR) package. For more information about FRR, refer to FRRouting.

Key Features

Cumulus Linux fully supports EVPN as the control plane for VXLAN, including for both intra-subnet bridging and inter-subnet routing, and provides these key features:

- VNI membership exchange between VTEPs using EVPN type-3 (Inclusive multicast Ethernet tag) routes.
- Host MAC and IP address exchange using EVPN type-2 (MAC/IP advertisement) routes.
- Host/VM mobility support (MAC and IP moves) through exchange of the MAC Mobility Extended community.
- Dual-attached hosts via VXLAN active-active mode. MAC synchronization between the peer switches is done using MLAG.
- ARP/ND suppression, which enables VTEPs to suppress ARP flooding over VXLAN tunnels is enabled by default on VNIs in Cumulus Linux.
- Exchange of static MAC addresses through EVPN.
- Inter-subnet routing for both IPv4 and IPv6 hosts: Distributed symmetric routing between different subnets, distributed asymmetric routing between different subnets, and centralized routing.
- Prefix-based routing using EVPN type-5 routes (EVPN IP prefix route).
- Layer 3 multi-tenancy.
- IPv6 tenant routing.
- ECMP for overlay networks on RIOT-capable Broadcom ASICs (Trident 3, Maverick, Trident 2+) in addition to Tomahawk and Mellanox Spectrum-A1 ASICs. No configuration is needed, ECMP occurs in the overlay when there are multiple next hops.
- Head end replication is enabled by default in Cumulus Linux on Broadcom Tomahawk, Maverick, Trident3, Trident II+, and Trident II ASICs and switches with Mellanox Spectrum ASICs. Cumulus Linux supports up to 128 VTEPs with head end replication.

The EVPN address-family is supported with both eBGP and iBGP peering. If

the underlay routing is provisioned using eBGP, you can use the same eBGP session to carry EVPN routes. For example, in a typical 2-tier Clos network topology where the leaf switches are the VTEPs, if eBGP sessions are in use between the leaf and spine switches for the underlay routing, the same sessions can be used to exchange EVPN routes; the spine switches merely act as *route forwarders* and do not install any forwarding state as they are not VTEPs. When EVPN routes are exchanged over iBGP peering, OSPF can be used as the IGP or the next hops can also be resolved using iBGP.

For information about VXLAN routing, including platform and hardware limitations, see VXLAN Routing.

> **NOTE**
>
> Data plane MAC learning is disabled by default on VXLAN interfaces. Do *not* enable MAC learning on VXLAN interfaces: EVPN is responsible for installing remote MACs.

Basic Configuration

The following sections provide the basic configuration needed to use EVPN as the control plane for VXLAN. The steps provided assume you have already configured VXLAN interfaces, attached them to a bridge, and mapped VLANs to VNIs.

> **NOTE**
>
> In Cumulus Linux 4.0, MAC learning is disabled and ARP/ND suppression is enabled by default. This is a change from earlier Cumulus Linux releases, where MAC learning is *enabled* and ARP/ND suppression *disabled* by default. Be sure to update any configuration scripts, if necessary.

Enable EVPN between BGP Neighbors

To enable EVPN between BGP neighbors, add the address family *evpn* to the existing neighbor `address-family` activation command.

For a non-VTEP device that is merely participating in EVPN route exchange, such as a spine switch where the network deployment uses hop-by-hop eBGP or the switch is acting as an iBGP route reflector, activating the interface for the EVPN address family is the fundamental configuration needed in FRRouting.

The other BGP neighbor address family specific configurations supported for EVPN are `allowas-in` and `route-reflector-client`.

To configure an EVPN route exchange with a BGP peer, activate the peer or peer group within the EVPN address family. For example:

NCLU Commands | vtysh Commands

```
cumulus@leaf01:~$ net add bgp autonomous-system 65101
cumulus@leaf01:~$ net add bgp router-id 10.10.10.1
cumulus@leaf01:~$ net add bgp neighbor swp51 interface
remote-as external
cumulus@leaf01:~$ net add bgp l2vpn evpn neighbor swp51
activate
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The above commands create the following configuration snippet in the `/etc/frr/frr.conf` file.

```
...
router bgp 65101
  bgp router-id 10.10.10.1
  neighbor swp51 interface remote-as external
```

```
address-family l2vpn evpn
  neighbor swp51 activate
...
```

The above configuration does not result in BGP knowing about the local VNIs defined on the system and advertising them to peers. This requires additional configuration, described in Advertise All VNIs, below.

Advertise All VNIs

FRR is not aware of any local VNIs and MACs, or hosts (neighbors) associated with those VNIs until you enable the BGP control plane for all VNIs configured on the switch by setting the `advertise-all-vni` option.

> **NOTE**
>
> This configuration is only needed on leaf switches that are VTEPs. EVPN routes received from a BGP peer are accepted, even without this explicit EVPN configuration. These routes are maintained in the global EVPN routing table. However, they only become effective (imported into the per-VNI routing table and appropriate entries installed in the kernel) when the VNI corresponding to the received route is locally known.

NCLU Commands | vtysh Commands

```
cumulus@leaf01:~$ net add bgp l2vpn evpn advertise-all-vni
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The above commands create the following configuration snippet in the `/etc/frr/frr.conf` file.

```
...
router bgp 65101
  bgp router-id 10.10.10.1
  neighbor swp51 interface remote-as external
  address-family l2vpn evpn
neighbor swp51 activate
  advertise-all-vni
...
```

EVPN and VXLAN Active-active Mode

For EVPN in VXLAN active-active mode, both switches in the MLAG pair establish EVPN peering with other EVPN speakers (for example, with spine switches if using hop-by-hop eBGP) and inform about their locally known

VNIs and MACs. When MLAG is active, both switches announce this information with the shared anycast IP address.

For active-active configuration, make sure that:

- The `clagd-vxlan-anycast-ip` and `vxlan-local-tunnelip` parameters are under the loopback stanza on both peers.
- The anycast address is advertised to the routed fabric from both peers.
- The VNIs are configured identically on both peers.
- The peerlink must belong to the bridge.

MLAG synchronizes information between the two switches in the MLAG pair; EVPN does not synchronize.

For type-5 routes in an EVPN *symmetric* configuration with VXLAN active-active mode, Cumulus Linux uses Primary IP Address Advertisement. For information on configuring Primary IP Address Advertisement, see Advertise Primary IP Address.

For information about active-active VTEPs and anycast IP behavior, and for failure scenarios, see VXLAN Active-active Mode.

Considerations

- When EVPN is enabled on a VTEP, all locally defined VNIs on that switch and other information (such as MAC addresses) are advertised to EVPN peers. There is no provision to only announce certain VNIs.
- On switches with Spectrum ASICs, ND suppression only works with the Spectrum-A1 chip.

- ARP suppression is enabled by default in Cumulus Linux. However, in a VXLAN active-active configuration, ARPs are sometimes *not* suppressed. This is because the neighbor entries are not synchronized between the two switches operating in active-active mode by a control plane. This has no impact on forwarding.
- You must configure the overlay (tenants) in a specific VRF and separate from the underlay, which resides in the default VRF. Layer 3 VNI mapping for the default VRF is not supported.
- In an EVPN deployment, Cumulus Linux supports a single BGP ASN which represents the ASN of the core as well as the ASN for any tenant VRFs if they have BGP peerings. If you need to change the ASN, you must first remove the layer 3 VNI in the `/etc/frr/frr.conf` file, modify the BGP ASN, then add back the layer 3 VNI in the `/etc/frr/frr.conf` file.
- EVPN is not supported when Redistribute Neighbor is also configured. Enabling both features simultaneously causes instability in IPv4 and IPv6 neighbor entries.
- Cumulus Linux implements a stricter check on a received type-3 route to ensure that it has the PMSI attribute with the replication type set to *ingress-replication* in order to conform to RFC 6514.

EVPN Enhancements

This section describes EVPN enhancements.

Define RDs and RTs

When FRR learns about a local VNI and there is no explicit configuration for that VNI in FRR, the route distinguisher (RD), and import and export route targets (RTs) for this VNI are automatically derived. The RD uses *RouterId:VNI-Index* and the import and export RTs use *AS:VNI*. For routes that come from a layer 2 VNI (type-2 and type-3), the RD uses the `vxlan-local-tunnelip` from the layer 2 VNI interface instead of the RouterId (`vxlan-local-tunnelip:VNI`). The RD and RTs are used in the EVPN route exchange.

The RD disambiguates EVPN routes in different VNIs (as they may have the same MAC and/or IP address) while the RTs describe the VPN membership for the route. The *VNI-Index* used for the RD is a unique, internally generated number for a VNI. It only has local significance; on remote switches, its only role is for route disambiguation. This number is used instead of the VNI value itself because this number has to be less than or equal to 65535. In the RT, the AS is always encoded as a 2-byte value to allow room for a large VNI. If the router has a 4-byte AS, only the lower 2 bytes are used. This ensures a unique RT for different VNIs while having the same RT for the same VNI across routers in the same AS.

For eBGP EVPN peering, the peers are in a different AS so using an

automatic RT of *AS:VNI* does not work for route import. Therefore, the import RT is treated as **:VNI* to determine which received routes are applicable to a particular VNI. This only applies when the import RT is auto-derived and not configured.

If you do *not* want RDs and RTs to be derived automatically, you can define them manually. The following example commands are per VNI. You must specify these commands under `address-family l2vpn evpn` in BGP.

NCLU Commands | vtysh Commands

leaf01 | leaf03

```
cumulus@leaf01:~$ net add bgp l2vpn evpn vni 10 rd
10.10.10.1:20
cumulus@leaf01:~$ net add bgp l2vpn evpn vni 10 route-
target export 65101:10
cumulus@leaf01:~$ net add bgp l2vpn evpn vni 10 route-
target import 65102:10
cumulus@leaf01:~$ net add bgp l2vpn evpn advertise-all-
vni
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

These commands create the following configuration snippet in the `/etc/`

`frr/frr.conf` file.

leaf01 leaf03

```
...
address-family l2vpn evpn
  advertise-all-vni
  vni 10
   rd 10.10.10.1:20
   route-target export 65101:10
   route-target import 65102:10
...
```

> **NOTE**
>
> If you delete the RD or RT later, it reverts back to its corresponding default value.

You can configure multiple RT values. In addition, you can configure both the import and export route targets with a single command by using `route-target both`:

NCLU Commands | vtysh Commands

leaf01 | leaf03

```
cumulus@leaf01:~$ net add bgp l2vpn evpn vni 10 route-
target import 65102:10
cumulus@leaf01:~$ net add bgp l2vpn evpn vni 10 route-
target import 65102:20
cumulus@leaf01:~$ net add bgp l2vpn evpn vni 20 route-
target both 65101:10
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The above commands create the following configuration snippet in the `/etc/frr/frr.conf` file:

leaf01 leaf03

```
...
address-family l2vpn evpn
  vni 10
    route-target import 65102:10
    route-target import 65102:20
  vni 20
    route-target import 65101:10
    route-target export 65101:10
...
```

Enable EVPN in an iBGP Environment with an OSPF Underlay

You can use EVPN with an OSPF or static route underlay. This is a more complex configuration than using eBGP. In this case, iBGP advertises EVPN routes directly between VTEPs and the spines are unaware of EVPN or BGP.

The leaf switches peer with each other in a full mesh within the EVPN address family without using route reflectors. The leafs generally peer to their loopback addresses, which are advertised in OSPF. The receiving VTEP imports routes into a specific VNI with a matching route target community.

NCLU Commands vtysh Commands

```
cumulus@leaf01:~$ net add bgp autonomous-system 65101
cumulus@leaf01:~$ net add bgp l2vpn evpn neighbor
10.10.10.2 remote-as internal
cumulus@leaf01:~$ net add bgp l2vpn evpn neighbor
10.10.10.3 remote-as internal
cumulus@leaf01:~$ net add bgp l2vpn evpn neighbor
10.10.10.4 remote-as internal
cumulus@leaf01:~$ net add bgp l2vpn evpn neighbor
10.10.10.2 activate
cumulus@leaf01:~$ net add bgp l2vpn evpn neighbor
10.10.10.3 activate
cumulus@leaf01:~$ net add bgp l2vpn evpn neighbor
10.10.10.4 activate
cumulus@leaf01:~$ net add bgp l2vpn evpn advertise-all-vni
cumulus@leaf01:~$ net add ospf router-id 10.10.10.1
cumulus@leaf01:~$ net add loopback lo ospf area 0.0.0.0
cumulus@leaf01:~$ net add ospf passive-interface lo
cumulus@leaf01:~$ net add interface swp49 ospf area 0.0.0.0
cumulus@leaf01:~$ net add interface swp50 ospf area 0.0.0.0
cumulus@leaf01:~$ net add interface swp51 ospf area 0.0.0.0
cumulus@leaf01:~$ net add interface swp52 ospf area 0.0.0.0
cumulus@leaf01:~$ net add interface swp49 ospf network
point-to-point
cumulus@leaf01:~$ net add interface swp50 ospf network
point-to-point
cumulus@leaf01:~$ net add interface swp51 ospf network
point-to-point
cumulus@leaf01:~$ net add interface swp52 ospf network
point-to-point
```

These commands create the following configuration snippet in the `/etc/frr/frr.conf` file.

```
...
interface lo
  ip ospf area 0.0.0.0
!
interface swp49
  ip ospf area 0.0.0.0
  ip ospf network point-to-point
!
interface swp50
  ip ospf area 0.0.0.0
  ip ospf network point-to-point
!
interface swp51
  ip ospf area 0.0.0.0
  ip ospf network point-to-point
!
interface swp52
  ip ospf area 0.0.0.0
  ip ospf network point-to-point
!
router bgp 65101
  neighbor 10.10.10.2 remote-as internal
```

```
  neighbor 10.10.10.3 remote-as internal
  neighbor 10.10.10.4 remote-as internal
  !
  address-family l2vpn evpn
  neighbor 10.10.10.2 activate
  neighbor 10.10.10.3 activate
  neighbor 10.10.10.4 activate
  advertise-all-vni
  exit-address-family
  !
Router ospf
  Ospf router-id 10.10.10.1
  Passive-interface lo
...
```

ARP and ND Suppression

ARP suppression with EVPN allows a VTEP to suppress ARP flooding over VXLAN tunnels as much as possible. A local proxy handles ARP requests received from locally attached hosts for remote hosts. ARP suppression is the implementation for IPv4; ND suppression is the implementation for IPv6.

ARP/ND suppression is enabled by default on all VNIs in Cumulus Linux to reduce flooding of ARP/ND packets over VXLAN tunnels.

In a centralized routing deployment, you must configure layer 3 interfaces even if the switch is configured only for layer 2 (you are not using VXLAN routing). To avoid unnecessary layer 3 information from being installed, configure the `ip forward off` or `ip6 forward off` options as appropriate on the VLANs. See the example configuration below.

The following examples show a configuration using two VXLANs (10 and 20) and two VLANs (10 and 20).

NCLU Commands | Linux Commands

```
cumulus@leaf01:~$ net add bridge bridge ports vni10,vni20
cumulus@leaf01:~$ net add bridge bridge vids 10,20
cumulus@leaf01:~$ net add vxlan vni10 vxlan id 10
cumulus@leaf01:~$ net add vxlan vni20 vxlan id 20
cumulus@leaf01:~$ net add vxlan vni10 bridge access 10
cumulus@leaf01:~$ net add vxlan vni20 bridge access 20
cumulus@leaf01:~$ net add vxlan vni10 vxlan local-tunnelip
10.10.10.1
cumulus@leaf01:~$ net add vxlan vni20 vxlan local-tunnelip
10.10.10.1
cumulus@leaf01:~$ net add vlan 10 ip forward off
cumulus@leaf01:~$ net add vlan 10 ipv6 forward off
cumulus@leaf01:~$ net add vlan 20 ip forward off
cumulus@leaf01:~$ net add vlan 20 ipv6 forward off
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

For a bridge in traditional mode, you must edit the bridge configuration in the `/etc/network/interfaces` file using a text editor:

```
cumulus@leaf01:~$ sudo nano /etc/network/interfaces
```

```
...
auto bridge1
iface bridge1
    bridge-ports swp1.10 swp2.10 vni10
    ip6-forward off
    ip-forward off
...
```

When deploying EVPN and VXLAN using a hardware profile *other* than the default Forwarding Table Profile, ensure that the Linux kernel ARP `sysctl` settings `gc_thresh2` and `gc_thresh3` are both set to a value larger than the number of neighbor (ARP/ND) entries anticipated in the deployment. To configure these settings, edit the `/etc/sysctl.d/neigh.conf` file, then reboot the switch. If your network has more hosts than the values used in the example below, change the `sysctl` entries accordingly.

▼ Example /etc/sysctl.d/neigh.conf file

Keep ARP and ND suppression enabled to reduce flooding of ARP/ND packets over VXLAN tunnels. However, if you need to disable ARP and ND suppression, follow the example commands below.

NCLU Commands | Linux Commands

```
cumulus@leaf01:~$ net del vxlan vni10 bridge arp-nd-suppress
cumulus@leaf01:~$ net del vxlan vni20 bridge arp-nd-suppress
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

Configure Static MAC Addresses

MAC addresses that are intended to be pinned to a particular VTEP can be provisioned on the VTEP as a static bridge FDB entry. EVPN picks up these MAC addresses and advertises them to peers as remote static MACs. You configure static bridge FDB entries for MACs under the bridge configuration:

NCLU Commands | Linux Commands

```
cumulus@leaf01:~$ net add bridge post-up bridge fdb add
26:76:e6:93:32:78 dev bond1 vlan 10 master static
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

For a bridge in traditional mode, you must edit the bridge configuration in the `/etc/network/interfaces` file using a text editor:

```
cumulus@leaf01:~$ sudo nano /etc/network/interfaces
...
auto br10
iface br10
    bridge-ports swp1.10 vni10
    post-up bridge fdb add 26:76:e6:93:32:78 dev swp1.10
master static
...
```

Filter EVPN Routes

A common deployment scenario for large data centers is to sub divide the data center into multiple pods with full host mobility within a pod but only

do prefix-based routing across pods. You can achieve this by only exchanging EVPN type-5 routes across pods.

To filter EVPN routes based on the route type and allow only certain types of EVPN routes to be advertised in the fabric:

NCLU Commands | vtysh Commands

Use the `net add routing route-map <route_map_name> (deny|permit) <1-65535> match evpn default-route` command or the `net add routing route-map <route_map_name> (deny|permit) <1-65535> match evpn route-type (macip|prefix|multicast)` command.

The following example commands configure EVPN to advertise type-5 routes only:

```
cumulus@leaf01:~$ net add routing route-map map1 permit 1
match evpn route-type prefix
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

NOTE

You must apply the route map for the configuration to take effect.

See Route Maps for more information.

Advertise SVI IP Addresses

In a typical EVPN deployment, you *reuse* SVI IP addresses on VTEPs across multiple racks. However, if you use *unique* SVI IP addresses across multiple racks and you want the local SVI IP address to be reachable via remote VTEPs, you can enable the `advertise-svi-ip` option. This option advertises the SVI IP/MAC address as a type-2 route and eliminates the need for any flooding over VXLAN to reach the IP from a remote VTEP/rack.

NOTE

- When you enable the `advertise-svi-ip` option, the anycast IP/MAC address pair is not advertised. Be sure **not** to enable both the `advertise-svi-ip` option and the `advertise-default-gw` option at the same time. (The `advertise-default-gw` option configures the gateway VTEPs to advertise their IP/MAC address. See Advertising the Default Gateway.
- If your switch is in an MLAG configuration, refer to Advertise

Primary IP Address.

To advertise *all* SVI IP/MAC addresses on the switch, run these commands:

NCLU Commands | vtysh Commands

```
cumulus@leaf01:~$ net add bgp l2vpn evpn advertise-svi-ip
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

To advertise a *specific* SVI IP/MAC address, run these commands:

NCLU Commands | vtysh Commands

```
cumulus@leaf01:~$ net add bgp l2vpn evpn vni 10 advertise-
svi-ip
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The NCLU and vtysh commands save the configuration in the `/etc/frr/`

`frr.conf` file. For example:

```
cumulus@leaf01:~$ sudo cat /etc/frr/frr.conf
...
address-family l2vpn evpn
  vni 10
  advertise-svi-ip
exit-address-family
...
```

Disable BUM Flooding

By default, the VTEP floods all broadcast, and unknown unicast and multicast packets (such as ARP, NS, or DHCP) it receives to all interfaces (except for the incoming interface) and to all VXLAN tunnel interfaces in the same broadcast domain. When the switch receives such packets on a VXLAN tunnel interface, it floods the packets to all interfaces in the packet's broadcast domain.

You can disable BUM flooding over VXLAN tunnels so that EVPN does not advertise type-3 routes for each local VNI and stops taking action on received type-3 routes.

Disabling BUM flooding is useful in a deployment with a controller or orchestrator, where the switch is pre-provisioned and there is no need to flood any ARP, NS, or DHCP packets.

> **NOTE**
>
> For information on EVPN BUM flooding with PIM, refer to EVPN BUM Traffic with PIM-SM.

To disable BUM flooding, run the NCLU `net add bgp l2vpn evpn disable-flooding` command or the vtysh `flooding disable` command. For example:

NCLU Commands | vtysh Commands

```
cumulus@leaf01:~$ net add bgp l2vpn evpn disable-flooding
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The NCLU and vtysh commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router bgp 65101
 !
 address-family l2vpn evpn
```

```
  flooding disable
 exit-address-family
...
```

To re-enable BUM flooding, run the NCLU `net del bgp l2vpn evpn disable-flooding` command or the vtysh `flooding head-end-replication` command. For example:

NCLU Commands | vtysh Commands

```
cumulus@leaf01:~$ net del bgp l2vpn evpn disable-flooding
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

Verify Configuration

To show that BUM flooding is disabled, run the NCLU `net show bgp l2vpn evpn vni` command or the vtysh `show bgp l2vpn evpn vni` command. For example:

```
cumulus@leaf01:~$ net show bgp l2vpn evpn vni
Advertise Gateway Macip: Disabled
```

```
Advertise SVI Macip: Disabled
Advertise All VNI flag: Enabled
BUM flooding: Disabled
Number of L2 VNIs: 3
Number of L3 VNIs: 2
Flags: * - Kernel
  VNI        Type RD                    Import
RT                 Export RT                Tenant VRF
* 20         L2   10.10.10.1:3
65101:20                  65101:20                RED
* 30         L2   10.10.10.1:4
65101:30                  65101:30                BLUE
* 10         L2   10.10.10.1:6
65101:10                  65101:10                RED
* 4002       L3   10.1.30.2:2
65101:4002                65101:4002              BLUE
* 4001       L3   10.1.20.2:5
65101:4001                65101:4001              RED
```

Run the NCLU `net show bgp l2vpn evpn route type multicast` command to make sure no locally-originated EVPN type-3 routes are listed.

Extended Mobility

Cumulus Linux supports scenarios where the IP to MAC binding for a host or virtual machine changes across the move. In addition to the simple mobility scenario where a host or virtual machine with a binding of `IP1`, `MAC1` moves from one rack to another, Cumulus Linux supports additional scenarios where a host or virtual machine with a binding of `IP1`, `MAC1` moves and takes on a new binding of `IP2`, `MAC1` or `IP1`, `MAC2`. The EVPN protocol mechanism to handle extended mobility continues to use the MAC mobility extended community and is the same as the standard mobility procedures. Extended mobility defines how the sequence number in this attribute is computed when binding changes occur.

Extended mobility not only supports virtual machine *moves*, but also where one virtual machine shuts down and another is provisioned on a different rack that uses the IP address or the MAC address of the previous virtual machine. For example, in an EVPN deployment with OpenStack, where virtual machines for a tenant are provisioned and shut down very dynamically, a new virtual machine can use the same IP address as an earlier virtual machine but with a different MAC address.

The support for extended mobility is enabled by default and does not require any additional configuration.

You can examine the sequence numbers associated with a host or virtual machine MAC address and IP address with the NCLU `net show evpn mac vni <vni> mac <address>` command or the vtysh `show evpn mac vni <vni>`

`mac <address>` command. For example:

```
cumulus@switch:~$ net show evpn mac vni 10100 mac
00:02:00:00:00:42
MAC: 00:02:00:00:00:42
  Remote VTEP: 10.0.0.2
  Local Seq: 0 Remote Seq: 3
  Neighbors:
    10.1.1.74 Active

cumulus@switch:~$ net show evpn arp vni 10100 ip 10.1.1.74
IP: 10.1.1.74
  Type: local
  State: active
  MAC: 44:39:39:ff:00:24
  Local Seq: 2 Remote Seq: 3
```

Duplicate Address Detection

Cumulus Linux is able to detect duplicate MAC and IPv4/IPv6 addresses on hosts or virtual machines in a VXLAN-EVPN configuration. The Cumulus Linux switch (VTEP) considers a host MAC or IP address to be duplicate if the address moves across the network more than a certain number of times within a certain number of seconds (five moves within 180 seconds by default). In addition to legitimate host or VM mobility scenarios, address movement can occur when IP addresses are misconfigured on hosts or

when packet looping occurs in the network due to faulty configuration or behavior.

Duplicate address detection is enabled by default and triggers when:

- Two hosts have the same MAC address (the host IP addresses might be the same or different)
- Two hosts have the same IP address but different MAC addresses

By default, when a duplicate address is detected, Cumulus Linux flags the address as a duplicate and generates an error in syslog so that you can troubleshoot the reason and address the fault, then clear the duplicate address flag. No functional action is taken on the address.

> **NOTE**
>
> If a MAC address is flagged as a duplicate, all IP addresses associated with that MAC are flagged as duplicates. However, in an MLAG configuration, only one of the MLAG peers might flag the associated IP addresses as duplicates.

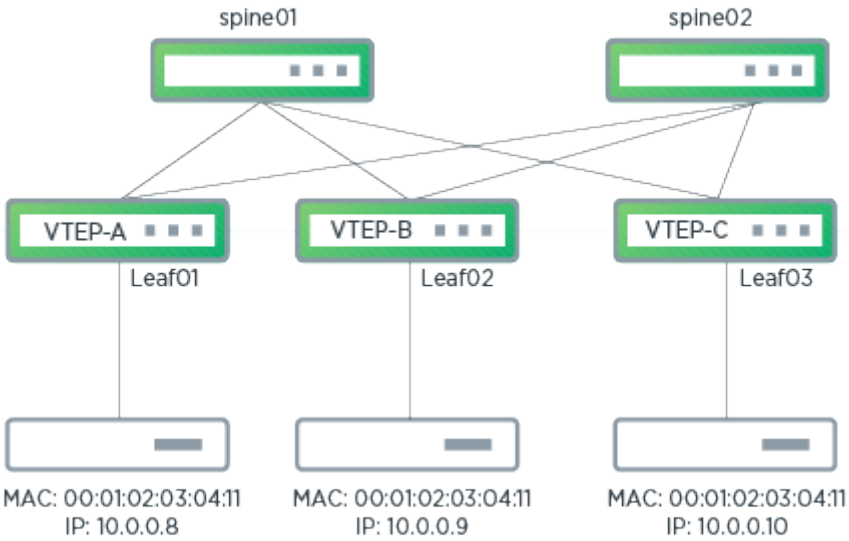
> **NOTE**
>
> In an MLAG configuration, MAC mobility detection runs independently on each switch in the MLAG pair. Based on the

sequence in which local learning and/or route withdrawal from the remote VTEP occurs, a type-2 route might have its MAC mobility counter incremented only on one of the switches in the MLAG pair. In rare cases, it is possible for neither VTEP to increment the MAC mobility counter for the type-2 prefix.

When Does Duplicate Address Detection Trigger?

The VTEP that sees an address move from remote to local begins the detection process by starting a timer. Each VTEP runs duplicate address detection independently. Detection always starts with the first mobility event from *remote* to *local*. If the address is initially remote, the detection count can start with the very first move for the address. If the address is initially local, the detection count starts only with the second or higher move for the address. If an address is undergoing a mobility event between remote VTEPs, duplicate detection is not started.

The following illustration shows VTEP-A, VTEP-B, and VTEP-C in an EVPN configuration. Duplicate address detection triggers on VTEP-A when there is a duplicate MAC address for two hosts attached to VTEP-A and VTEP-B. However, duplicate detection does *not* trigger on VTEP-A when mobility events occur between two remote VTEPs (VTEP-B and VTEP-C).



Configure Duplicate Address Detection

To change the threshold for MAC and IP address moves, run the `net add bgp l2vpn evpn dup-addr-detection max-moves <number-of-events> time <duration>` command. You can specify `max-moves` to be between 2 and 1000 and `time` to be between 2 and 1800 seconds.

The following example command sets the maximum number of address moves allowed to 10 and the duplicate address detection time interval to 1200 seconds.

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add bgp l2vpn evpn dup-addr-detection
max-moves 10 time 1200
```

To disable duplicate address detection, see Disable Duplicate Address Detection below.

Example syslog Messages

The following example shows the syslog message that is generated when Cumulus Linux detects a MAC address as a duplicate during a local update:

```
2018/11/06 18:55:29.463327 ZEBRA: [EC 4043309149] VNI 1001: MAC
00:01:02:03:04:11 detected as duplicate during local update,
last VTEP 172.16.0.16
```

The following example shows the syslog message that is generated when Cumulus Linux detects an IP address as a duplicate during a remote update:

```
2018/11/09 22:47:15.071381 ZEBRA: [EC 4043309151] VNI 1002: MAC
aa:22:aa:aa:aa:aa IP 10.0.0.9 detected as duplicate during
remote update, from VTEP 172.16.0.16
```

Freeze a Detected Duplicate Address

Cumulus Linux provides a *freeze* option that takes action on a detected duplicate address. You can freeze the address *permanently* (until you intervene) or for a *defined amount of time*, after which it is cleared

automatically.

When you enable the freeze option and a duplicate address is detected:

- If the MAC or IP address is learned from a remote VTEP at the time it is frozen, the forwarding information in the kernel and hardware is not updated, leaving it in the prior state. Any future remote updates are processed but they are not reflected in the kernel entry. If the remote VTEP sends a MAC-IP route withdrawal, the local VTEP removes the frozen remote entry. Then, if the local VTEP has a locally-learned entry already present in its kernel, FRRouting will originate a corresponding MAC-IP route and advertise it to all remote VTEPs.
- If the MAC or IP address is locally learned on this VTEP at the time it is frozen, the address is not advertised to remote VTEPs. Future local updates are processed but are not advertised to remote VTEPs. If FRR receives a local entry delete event, the frozen entry is removed from the FRR database. Any remote updates (from other VTEPs) change the state of the entry to remote but the entry is not installed in the kernel (until cleared).

To recover from a freeze, shut down the faulty host or VM or fix any other misconfiguration in the network. If the address is frozen *permanently,* issue the clear command on the VTEP where the address is marked as duplicate. If the address is frozen for a defined period of time, it is cleared automatically after the timer expires (you can clear the duplicate address before the timer expires with the clear command).

> **NOTE**
>
> If you issue the clear command or the timer expires before you address the fault, duplicate address detection might occur repeatedly.

After you clear a frozen address, if it is present behind a remote VTEP, the kernel and hardware forwarding tables are updated. If the address is locally learned on this VTEP, the address is advertised to remote VTEPs. All VTEPs get the correct address as soon as the host communicates . Silent hosts are learned only after the faulty entries age out, or you intervene and clear the faulty MAC and ARP table entries.

Configure the Freeze Option

To enable Cumulus Linux to *freeze* detected duplicate addresses, run the `net add bgp l2vpn evpn dup-addr-detection freeze <duration>|permanent` command. The duration can be any number of seconds between 30 and 3600.

The following example command freezes duplicate addresses for a period of 1000 seconds, after which it is cleared automatically:

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add bgp l2vpn evpn dup-addr-detection
freeze 1000
```

NOTE

Set the freeze timer to be three times the duplicate address detection window. For example, if the duplicate address detection window is set to the default of 180 seconds, set the freeze timer to 540 seconds.

The following example command freezes duplicate addresses permanently (until you issue the clear command):

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add bgp l2vpn evpn dup-addr-detection
freeze permanent
```

Clear Duplicate Addresses

You can clear a duplicate MAC or IP address (and unfreeze a frozen address). The following example command clears IP address 10.0.0.9 for VNI 101.

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net clear evpn dup-addr vni 101 ip
10.0.0.9
```

To clear duplicate addresses for all VNIs, run the following command:

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net clear evpn dup-addr vni all
```

NOTE

In an MLAG configuration, you need to run the clear command on both the MLAG primary and secondary switch.

> **NOTE**
>
> When you clear a duplicate MAC address, all its associated IP addresses are also cleared. However, you cannot clear an associated IP address if its MAC address is still in a duplicate state.

Disable Duplicate Address Detection

By default, duplicate address detection is enabled and a syslog error is generated when a duplicate address is detected. To disable duplicate address detection, run the following command.

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net del bgp l2vpn evpn dup-addr-detection
```

When you disable duplicate address detection, Cumulus Linux clears the configuration and all existing duplicate addresses.

Show Detected Duplicate Address Information

During the duplicate address detection process, you can see the start time and current detection count with the NCLU `net show evpn mac vni <vni_id> mac <mac_addr>` command or the vtysh `show evpn mac vni <vni_id> mac <mac_addr>` command. The following command example

shows that detection started for MAC address 00:01:02:03:04:11 for VNI 1001 on Tuesday, Nov 6 at 18:55:05 and the number of moves detected is 1.

```
cumulus@switch:~$ net show evpn mac vni 1001 mac
00:01:02:03:04:11
MAC: 00:01:02:03:04:11
  Intf: hostbond3(15) VLAN: 1001
  Local Seq: 1 Remote Seq: 0
  Duplicate detection started at Tue Nov  6 18:55:05 2018,
detection count 1
  Neighbors:
    10.0.1.26 Active
```

After the duplicate MAC address is cleared, the NCLU `net show evpn mac vni <vni_id> mac <mac_addr>` command or the vtysh `show evpn mac vni <vni_id> mac <mac_addr>` command shows:

```
MAC: 00:01:02:03:04:11
  Remote VTEP: 172.16.0.16
  Local Seq: 13 Remote Seq: 14
  Duplicate, detected at Tue Nov  6 18:55:29 2018
  Neighbors:
    10.0.1.26 Active
```

To display information for a duplicate IP address, run the NCLU `net show evpn arp-cache vni <vni_id> ip <ip_addr>` command or the vtysh `show evpn arp-cache vni <vni_id> ip <ip_addr>` command. The following command example shows information for IP address 10.0.0.9 for VNI 1001.

```
cumulus@switch:~$ net show evpn arp-cache vni 1001 ip 10.0.0.9
IP: 10.0.0.9
  Type: remote
  State: inactive
  MAC: 00:01:02:03:04:11
  Remote VTEP: 10.0.0.34
  Local Seq: 0 Remote Seq: 14
  Duplicate, detected at Tue Nov  6 18:55:29 2018
```

To show a list of MAC addresses detected as duplicate for a specific VNI or for all VNIs, run the NCLU `net show evpn mac vni <vni-id|all> duplicate` command or the vtysh `show evpn mac vni <vni-id|all> duplicate` command. The following example command shows a list of duplicate MAC addresses for VNI 1001:

```
cumulus@switch:~$ net show evpn mac vni 1001 duplicate
Number of MACs (local and remote) known for this VNI: 16
MAC               Type   Intf/Remote VTEP      VLAN
```

```
aa:bb:cc:dd:ee:ff local  hostbond3            1001
```

To show a list of IP addresses detected as duplicate for a specific VNI or for all VNIs, run the NCLU `net show evpn arp-cache vni <vni-id|all> duplicate` command or the vtysh `show evpn arp-cache vni <vni-id|all> duplicate` command. The following example command shows a list of duplicate IP addresses for VNI 1001:

```
cumulus@switch:~$ net show evpn arp-cache vni 1001 duplicate
Number of ARPs (local and remote) known for this VNI: 20
IP                Type   State    MAC               Remote VTEP
10.0.0.8          local  active   aa:11:aa:aa:aa:aa
10.0.0.9          local  active   aa:11:aa:aa:aa:aa
10.10.0.12        remote active   aa:22:aa:aa:aa:aa  172.16.0.16
```

To show configured duplicate address detection parameters, run the NCLU `net show evpn` command or the vtysh `show evpn` command:

```
cumulus@switch:~$ net show evpn
L2 VNIs: 4
L3 VNIs: 2
```

```
Advertise gateway mac-ip: No
Duplicate address detection: Enable
  Detection max-moves 7, time 300
  Detection freeze permanent
```

Inter-subnet Routing

There are multiple models in EVPN for routing between different subnets (VLANs), also known as inter-VLAN routing. The model you choose depends if every VTEP acts as a layer 3 gateway and performs routing or if only specific VTEPs perform routing, and if routing is performed only at the ingress of the VXLAN tunnel or both the ingress and the egress of the VXLAN tunnel.

Cumulus Linux supports these models:

- Centralized routing: Specific VTEPs act as designated layer 3 gateways and perform routing between subnets; other VTEPs just perform bridging.
- Distributed asymmetric routing: Every VTEP participates in routing, but all routing is done at the ingress VTEP; the egress VTEP only performs bridging.
- Distributed symmetric routing: Every VTEP participates in routing and routing is done at both the ingress VTEP and the egress VTEP.

Distributed routing (asymmetric or symmetric) is commonly deployed with the VTEPs configured with an *anycast IP/MAC address* for each subnet; each VTEP that has a particular subnet is configured with the same IP/MAC for that subnet. Such a model facilitates easy host/VM mobility as there is no need to change the host/VM configuration when it moves from one VTEP to another.

All routing occurs in the context of a tenant VRF (virtual routing and

forwarding). A VRF instance is provisioned for each tenant and the subnets of the tenant are associated with that VRF (the corresponding SVI is attached to the VRF). Inter-subnet routing for each tenant occurs within the context of the VRF for that tenant and is separate from the routing for other tenants.

Centralized Routing

In centralized routing, you configure a specific VTEP to act as the default gateway for all the hosts in a particular subnet throughout the EVPN fabric. It is common to provision a pair of VTEPs in active-active mode as the default gateway using an anycast IP/MAC address for each subnet. You need to configure all subnets on such a gateway VTEP. When a host in one subnet wants to communicate with a host in another subnet, it addresses the packets to the gateway VTEP. The ingress VTEP (to which the source host is attached) bridges the packets to the gateway VTEP over the corresponding VXLAN tunnel. The gateway VTEP performs the routing to the destination host and post-routing, the packet gets bridged to the egress VTEP (to which the destination host is attached). The egress VTEP then bridges the packet on to the destination host.

To enable centralized routing, you must configure the gateway VTEPs to advertise their IP/MAC address. Use the `advertise-default-gw` command:

NCLU Commands | vtysh Commands

```
cumulus@leaf01:~$ net add bgp autonomous-system 65101
cumulus@leaf01:~$ net add bgp l2vpn evpn advertise-default-
gw
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

These commands create the following configuration snippet in the `/etc/frr/frr.conf` file.

```
...
router bgp 65101
...
  address-family l2vpn evpn
    advertise-default-gw
  exit-address-family
...
```

NOTE

You can deploy centralized routing at the VNI level. Therefore, you

can configure the `advertise-default-gw` command per VNI so that centralized routing is used for some VNIs while distributed routing (described below) is used for other VNIs. This type of configuration is not recommended unless the deployment requires it.
When centralized routing is in use, even if the source host and destination host are attached to the same VTEP, the packets travel to the gateway VTEP to get routed and then come back.

Asymmetric Routing

In distributed asymmetric routing, each VTEP acts as a layer 3 gateway, performing routing for its attached hosts. The routing is called asymmetric because only the ingress VTEP performs routing, the egress VTEP only performs bridging. Asymmetric routing can be achieved with only host routing and does not involve any interconnecting VNIs. However, you must provision each VTEP with all VLANs/VNIs - the subnets between which communication can take place; this is required even if there are no locally-attached hosts for a particular VLAN.

TIP

The only additional configuration required to implement asymmetric routing beyond the standard configuration for a layer 2 VTEP described earlier is to ensure that each VTEP has all VLANs (and corresponding VNIs) provisioned on it and the SVI for each such VLAN is configured with an anycast IP/MAC address.

Symmetric Routing

In distributed symmetric routing, each VTEP acts as a layer 3 gateway, performing routing for its attached hosts; however, both the ingress VTEP and egress VTEP route the packets (similar to the traditional routing behavior of routing to a next hop router). In the VXLAN encapsulated packet, the inner destination MAC address is set to the router MAC address of the egress VTEP as an indication that the egress VTEP is the next hop and also needs to perform routing. All routing happens in the context of a tenant (VRF). For a packet received by the ingress VTEP from a locally attached host, the SVI interface corresponding to the VLAN determines the VRF. For a packet received by the egress VTEP over the VXLAN tunnel, the VNI in the packet has to specify the VRF. For symmetric routing, this is a VNI corresponding to the tenant and is different from either the source VNI or the destination VNI. This VNI is referred to as the layer 3 VNI or interconnecting VNI; it has to be provisioned by the operator and is

exchanged through the EVPN control plane. To make the distinction clear, the regular VNI, which is used to map a VLAN, is referred to as the layer 2 VNI.

NOTE

- There is a one-to-one mapping between a layer 3 VNI and a tenant (VRF).
- The VRF to layer 3 VNI mapping has to be consistent across all VTEPs. The layer 3 VNI has to be provisioned by the operator.
- A layer 3 VNI and a layer 2 VNI cannot have the same ID. If the VNI IDs are the same, the layer 2 VNI does not get created.
- In an MLAG configuration, the SVI used for the layer 3 VNI cannot be part of the bridge. This ensures that traffic tagged with that VLAN ID is not forwarded on the peer link or other trunks.

In an EVPN symmetric routing configuration, when a type-2 (MAC/IP) route is announced, in addition to containing two VNIs (the layer 2 VNI and the layer 3 VNI), the route also contains separate RTs for layer 2 and layer 3. The layer 3 RT associates the route with the tenant VRF. By default, this is auto-derived in a similar way to the layer 2 RT, using the layer 3 VNI instead of the layer 2 VNI; however you can also explicitly configure it.

For EVPN symmetric routing, additional configuration is required:

- Configure a per-tenant VXLAN interface that specifies the layer 3 VNI for the tenant. This VXLAN interface is part of the bridge and the router MAC address of the remote VTEP is installed over this interface.
- Configure an SVI (layer 3 interface) corresponding to the per-tenant VXLAN interface. This is attached to the VRF of the tenant. Remote host routes for symmetric routing are installed over this SVI.
- Specify the VRF to layer 3 VNI mapping. This configuration is for the BGP control plane.

Optional configuration includes configuring RD and RTs for the tenant VRF and advertising the locally-attached subnets.

Configure a Per-tenant VXLAN Interface

NCLU Commands | Linux Commands

```
cumulus@leaf01:~$ net add vxlan vni10 vxlan id 10
cumulus@leaf01:~$ net add vxlan vni10 bridge access 10
cumulus@leaf01:~$ net add vxlan vni10 vxlan local-tunnelip
10.10.10.1
cumulus@leaf01:~$ net add bridge bridge ports vni10
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

Configure an SVI for the Layer 3 VNI

NCLU Commands | Linux Commands

```
cumulus@leaf01:~$ net add vlan 4001 vrf RED
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

NOTE

When two VTEPs are operating in **VXLAN active-active** mode and performing **symmetric** routing, you need to configure the router MAC corresponding to each layer 3 VNI to ensure both VTEPs use the same MAC address. Specify the `address-virtual` (MAC address) for the SVI corresponding to the layer 3 VNI. Use the same address on both switches in the MLAG pair. Use the MLAG system MAC address. See Advertise Primary IP Address.

Configure the VRF to Layer 3 VNI Mapping

NCLU Commands | Linux Commands

```
cumulus@leaf01:~$ net add vrf RED vni 4001
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

Configure RD and RTs for the Tenant VRF

If you do not want the RD and RTs (layer 3 RTs) for the tenant VRF to be derived automatically, you can configure them manually by specifying them under the `l2vpn evpn` address family for that specific VRF.

NCLU Commands | vtysh Commands

```
cumulus@leaf01:~$ net add bgp vrf RED l2vpn evpn rd
10.1.20.2:5
cumulus@leaf01:~$ net add bgp vrf RED l2vpn evpn route-
target import 65102:4001
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

These commands create the following configuration snippet in the `/etc/`

`frr/frr.conf` file:

```
...
router bgp 65101 vrf RED
  address-family l2vpn evpn
  rd 10.1.20.2:5
  route-target import 65102:4001
...
```

> **NOTE**
>
> The tenant VRF RD and RTs are different from the RD and RTs for the layer 2 VNI. See Define RDs and RTs.

Advertise Locally-attached Subnets

Symmetric routing presents a problem in the presence of silent hosts. If the ingress VTEP does not have the destination subnet and the host route is not advertised for the destination host, the ingress VTEP cannot route the packet to its destination. You can overcome this problem by having VTEPs announce the subnet prefixes corresponding to their connected subnets in addition to announcing host routes. These routes are announced as EVPN prefix (type-5) routes.

To advertise locally attached subnets:

1. Enable advertisement of EVPN prefix (type-5) routes. Refer to Prefix-based Routing - EVPN Type-5 Routes, below.
2. Ensure that the routes corresponding to the connected subnets are known in the BGP VRF routing table by injecting them using the `network` command or redistributing them using the `redistribute connected` command.

> **NOTE**
>
> This configuration is recommended only if the deployment is known to have silent hosts. It is also recommended that you enable on only one VTEP per subnet, or two for redundancy.

Prefix-based Routing

EVPN in Cumulus Linux supports prefix-based routing using EVPN type-5 (prefix) routes. Type-5 routes (or prefix routes) are primarily used to route to destinations outside of the data center fabric.

EVPN prefix routes carry the layer 3 VNI and router MAC address and follow the symmetric routing model for routing to the destination prefix.

> **NOTE**

- When connecting to a WAN edge router to reach destinations outside the data center, deploy specific border/exit leaf switches to originate the type-5 routes.
- On switches with Spectrum ASICs, centralized routing, symmetric routing, and prefix-based routing only work with the Spectrum A1 chip.
- If you are using a Broadcom Trident II+ switch as a border/exit leaf, see Considerations below for a required workaround; the workaround only applies to Trident II+ switches, not Tomahawk or Spectrum.

Install EVPN Type-5 Routes

For a switch to be able to install EVPN type-5 routes into the routing table, you must configure it with the layer 3 VNI related information. This configuration is the same as for symmetric routing. You need to:

1. Configure a per-tenant VXLAN interface that specifies the layer 3 VNI for the tenant. This VXLAN interface is part of the bridge; router MAC addresses of remote VTEPs are installed over this interface.
2. Configure an SVI (layer 3 interface) corresponding to the per-tenant VXLAN interface. This is attached to the VRF of the tenant. The remote prefix routes are installed over this SVI.

3. Specify the mapping of the VRF to layer 3 VNI. This configuration is for the BGP control plane.

Announce EVPN Type-5 Routes

The following configuration is required in the tenant VRF to announce IP prefixes in the BGP RIB as EVPN type-5 routes.

NCLU Commands | vtysh Commands

```
cumulus@leaf01:~$ net add bgp vrf RED l2vpn evpn advertise
ipv4 unicast
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

These commands create the following snippet in the `/etc/frr/frr.conf` file:

```
...
router bgp 65101 vrf RED
  address-family l2vpn evpn
    advertise ipv4 unicast
  exit-address-family
end
...
```

EVPN Type-5 Routing in Asymmetric Mode

Asymmetric routing is an ideal choice when all VLANs (subnets) are configured on all leaf switches. It simplifies the routing configuration and eliminates the potential need for advertising subnet routes to handle silent hosts. However, most deployments need access to external networks to reach the Internet or global destinations, or to do subnet-based routing between pods or data centers; this requires EVPN type-5 routes.

Cumulus Linux supports EVPN type-5 routes for prefix-based routing in asymmetric configurations within the pod or data center by providing an option to use the layer 3 VNI only for type-5 routes; type-2 routes (host routes) only use the layer 2 VNI.

The following example commands show how to use the layer 3 VNI for type-5 routes only:

NCLU Commands | Linux Commands

```
cumulus@leaf01:~$ net add vrf RED vni 4001 prefix-routes-
only
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

NOTE

There is no command to delete the `prefix-routes-only` option. The `net del vrf <vrf> vni <vni> prefix-routes-only` command deletes the VNI.

Control RIB Routes

By default, when announcing IP prefixes in the BGP RIB as EVPN type-5 routes, all routes in the BGP RIB are picked for advertisement as EVPN type-5 routes. You can use a route map to allow selective advertisement of routes from the BGP RIB as EVPN type-5 routes.

The following commands add a route map filter to IPv4 EVPN type-5 route advertisement:

NCLU Commands vtysh Commands

```
cumulus@leaf01:~$ net add bgp vrf RED l2vpn evpn advertise
ipv4 unicast route-map map1
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

Originate Default EVPN Type-5 Routes

Cumulus Linux supports originating EVPN default type-5 routes. The default type-5 route is originated from a border (exit) leaf and advertised to all the other leafs within the pod. Any leaf within the pod follows the default route towards the border leaf for all external traffic (towards the Internet or a different pod).

To originate a default type-5 route in EVPN, you need to execute FRRouting commands. The following shows an example:

```
cumulus@leaf01:~$ sudo vtysh

leaf01# configure terminal
leaf01(config)# router bgp 65101 vrf RED
leaf01(config-router)# address-family l2vpn evpn
leaf01(config-router-af)# default-originate ipv4
```

```
leaf01(config-router-af)# default-originate ipv6
leaf01(config-router-af)# end
leaf01# write memory
```

Advertise Primary IP address (VXLAN Active-Active Mode)

With Cumulus Linux 3.7 and earlier, in EVPN symmetric routing configurations with VXLAN active-active (MLAG), all EVPN routes are advertised with the anycast IP address (clagd-vxlan-anycast-ip) as the next-hop IP address and the anycast MAC address as the router MAC address. In a failure scenario, this can lead to traffic being forwarded to a leaf switch that does not have the destination routes. Traffic has to traverse the peer link (with additional BGP sessions per VRF).

To prevent sub-optimal routing in Cumulus Linux 4.0 and later, the next hop IP address of the VTEP is conditionally handled depending on the route type: host type-2 (MAC/IP advertisement) or type-5 (IP prefix route).

- For host type-2 routes, the anycast IP address is used as the next hop IP address and the anycast MAC address is used as the router MAC address.
- For type-5 routes, the system IP address (the primary IP address of the VTEP) is used as the next hop IP address and the system MAC address of the VTEP is used as the router MAC address.

See EVPN and VXLAN Active-Active mode for information about EVPN and VXLAN active-active mode.

Configure Advertise Primary IP Address

Run the `address-virtual <anycast-mac>` command under the SVI, where `<anycast-mac>` is the MLAG system MAC address (clagd-sys-mac). Run these commands on both switches in the MLAG pair.

NCLU Commands | Linux Commands

Run the `net add vlan <vlan> ip address-virtual <anycast-mac>` command. For example:

```
cumulus@leaf01:~$ net add vlan 4001 ip address-virtual
44:38:39:BE:EF:AA
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

NOTE

- In Cumulus Linux 3.7 and earlier, the `hwaddress` command is used instead of the `address-virtual` command. If you upgrade from Cumulus Linux 3.7 to 4.0 or later and have a previous symmetric routing with VXLAN active-active

configuration, you must change `hwaddress` to `address-virtual`. Either run the NCLU `address-virtual <anycast-mac>` command or edit the `/etc/network/interfaces` file.

- When configuring third party networking devices using MLAG and EVPN for interoperability, you must configure and announce a single shared router MAC value per advertised next hop IP address.

Optional Configuration

If you do not want Cumulus Linux to derive the system IP address automatically, you can provide the system IP address and system MAC address under each BGP VRF instance.

The system MAC address must be the layer 3 SVI MAC address (not the `clad-sys-mac`).

The following example commands add the system IP address 10.10.10.1 and the system MAC address 44:38:39:be:ef:aa:

NCLU Commands vtysh Commands

```
cumulus@leaf01:~$ net add vlan 4001 hwaddress
44:38:39:ff:00:00
cumulus@leaf01:~$ net add bgp vrf vrf1 l2vpn evpn advertise-
pip ip 10.10.10.1 mac 44:38:39:be:ef:aa
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The system IP address and system MAC address you provide take precedence over the addresses that Cumulus Linux derives automatically.

Disable Advertise Primary IP Address

Each switch in the MLAG pair advertises type-5 routes with its own system IP, which creates an additional next hop at the remote VTEPs. In a large multi-tenancy EVPN deployment, where additional resources are a concern, you might prefer to disable this feature.

To disable Advertise Primary IP Address under each tenant VRF BGP instance:

NCLU Commands | vtysh Commands

```
cumulus@leaf01:~$ net del bgp vrf RED l2vpn evpn advertise-
pip
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

Show Advertise Primary IP Address Information

To show Advertise Primary IP Address parameters, run the NCLU `net show bgp l2vpn evpn vni <vni>` command or the vtysh `show bgp l2vpn evpn vni <vni>` command. For example:

```
cumulus@leaf01:~$ net show bgp l2vpn evpn vni 4001
VNI: 4001 (known to the kernel)
  Type: L3
  Tenant VRF: RED
  RD: 10.1.20.2:5
  Originator IP: 10.0.1.1
  Advertise-gw-macip : n/a
  Advertise-svi-macip : n/a
  Advertise-pip: Yes
  System-IP: 10.10.10.1
```

```
  System-MAC: 44:38:39:be:ef:aa
  Router-MAC: 44:38:39:be:ef:aa
  Import Route Target:
    65101:4001
  Export Route Target:
    65101:4001
leaf01#
```

To show EVPN routes with Primary IP Advertisement, run the NCLU `net show bgp l2vpn evpn route` command or the vtysh `show bgp l2vpn evpn route` command. For example:

```
cumulus@leaf01:~$ net show bgp l2vpn evpn route
...
Route Distinguisher: 10.10.10.1:3
*> [2]:[0]:[48]:[00:60:08:69:97:ef]
                    10.0.1.1                           32768 i
                    ET:8 RT:65101:10 RT:65101:4001
Rmac:44:38:39:be:ef:aa
*> [2]:[0]:[48]:[26:76:e6:93:32:78]
                    10.0.1.1                           32768 i
                    ET:8 RT:65101:10 RT:65101:4001
```

```
Rmac:44:38:39:be:ef:aa
*> [2]:[0]:[48]:[26:76:e6:93:32:78]:[32]:[10.1.10.101]
                    10.0.1.1                            32768 i
                    ET:8 RT:65101:10 RT:65101:4001
Rmac:44:38:39:be:ef:aa
...
```

To show the learned route from an external router injected as a type-5 route, run the NCLU `net show bgp vrf <vrf> ipv4 unicast` command or the vtysh `show bgp vrf <vrf> ipv4 unicast` command.

Considerations

VXLAN Decapsulation on Maverick and Broadcom Trident II Switches

On the Broadcom Trident II+ and Maverick-based switch, when a lookup is done after VXLAN decapsulation on the external-facing switch (the exit or border leaf), the switch does not rewrite the MAC addresses or TTL. For through traffic, packets are dropped by the next hop instead of correctly routing from a VXLAN overlay network into a non-VXLAN external network (such as the Internet). This applies to all forms of VXLAN routing (centralized, asymmetric, and symmetric) and affects all traffic from VXLAN overlay hosts that need to be routed after VXLAN decapsulation on an exit

or border leaf. This includes traffic destined to external networks (through traffic) and traffic destined to the exit leaf SVI address. To work around this issue, modify the external-facing interface for each VLAN sub-interface on the exit leaf by creating a temporary VNI and associating it with the existing VLAN ID.

▼ Example Workaround

Centralized Routing with ARP Suppression Enabled on the Gateway

In an EVPN centralized routing configuration, where the layer 2 network extends beyond VTEPs, (for example, a host with bridges), the gateway MAC address is not refreshed in the network when ARP suppression is enabled on the gateway. To work around this issue, disable ARP suppression on the centralized gateway.

Type-5 Routes and ECMP

For VXLAN type-5 routes, ECMP does not work when the VTEP is directly connected to remote VTEPs. To work around this issue, add an additional device in the VXLAN fabric between the local and remote VTEPs, so that local and remote VTEPs are not directly connected.

Symmetric Routing and the Same SVI IP Address Across Racks

In EVPN symmetric routing, if you use the same SVI IP address across racks; for example, if the SVI IP address for a specific VLAN interface (such as vlan100) is the same on all VTEPs where this SVI is present, be aware of

the following:

- You cannot use ping between SVI IP adresses to verify connectivity between VTEPs because either the local rack itself uses the ping destination IP address or many remote racks use the ping destination IP address.
- If you use ping from a host to the SVI IP address, the local VTEP (gateway) might not reply if the host has an ARP entry from a remote gateway.

There are no issues with host-to-host traffic.

EVPN Multihoming

EVPN multihoming (EVPN-MH) provides support for all-active server redundancy. It is a standards-based replacement for MLAG in data centers deploying Clos topologies. Replacing MLAG:

- Eliminates the need for peerlinks or inter-switch links between the top of rack switches
- Allows more than two TOR switches to participate in a redundancy group
- Provides a single BGP-EVPN control plane
- Allows multi-vendor interoperability

EVPN-MH uses BGP-EVPN type-1, type-2 and type-4 routes for discovering Ethernet segments (ES) and for forwarding traffic to those Ethernet segments. The MAC and neighbor databases are synced between the Ethernet segment peers via these routes as well. An *Ethernet segment* is a group of switch links that are attached to the same server. Each Ethernet segment has an unique Ethernet segment ID (`es-id`) across the entire PoD.

Configuring EVPN-MH involves setting an Ethernet segment system MAC address (`es-sys-mac`) and a local Ethernet segment ID (`local-es-id`) on a static or LACP bond. The `es-sys-mac` and `local-es-id` are used to build a type-3 `es-id`. This `es-id` must be globally unique across all the EVPN VTEPs. The same `es-sys-mac` can be configured on multiple interfaces.

While you can specify a different `es-sys-mac` on different Ethernet segments attached to the same switch, the `es-sys-mac` must be the same

on the downlinks attached to the same server.

> ⚠ **IMPORTANT**
>
> When using Spectrum 2 or Spectrum 3 switches, an Ethernet segment can span more than two switches. Each Ethernet segment is a distinct redundancy group.
>
> However, when using Spectrum A1 switches, a maximum of two switches can participate in a redundancy group or Ethernet segment.

Supported Features

- Known unicast traffic multihoming via type-1/EAD (Ethernet auto discovery) routes and type-2 (non-zero ESI) routes. Includes all-active redundancy via aliasing and support for fast failover.
- EVPN BUM traffic handling with EVPN-PIM on multihomed sites via Type-4/ESR routes, which includes split-horizon-filtering and designated forwarder election.

> **WARNING**
>
> Head-end replication is not supported with multihoming, so you must use EVPN-PIM for BUM traffic handling.

- VLAN-aware bridge mode only.
- LACP Bypass is supported.
  - When an EVPN-MH bond enters LACP bypass state, BGP stops advertising EVPN type-1 and type-4 routes for that bond. Split-horizon and designated forwarder filters are disabled.
  - When an EVPN-MH bond exits the LACP bypass state, BGP starts advertising EVPN type-1 and type-4 routes for that bond. Split-horizon and designated forwarder filters are enabled.
- Distributed symmetric routing.
- ARP suppression must be enabled.
- EVI (*EVPN virtual instance*). Cumulus Linux supports VLAN-based service only, so the EVI is just a layer 2 VNI.
- Supported ASICs include Mellanox Spectrum A1, Spectrum 2 and Spectrum 3.

WARNING

In order to use EVPN-MH, you must remove any MLAG configuration on the switch. This entails:

- Removing the `clag-id` from all interfaces in the `/etc/network/interfaces` file.
- Removing the peerlink interfaces in the `/etc/network/interfaces` file.
- Then running `ifreload` to reload the configuration:

```
cumulus@switch:~$ sudo ifreload
```

Configure EVPN-MH

To configure EVPN-MH, first you need to enable the `evpn.multihoming.enable` variable in `switchd.conf`. Then you need to specify the following required settings:

- The Ethernet segment ID (`es-id`)
- The Ethernet segment system MAC address (`es-sys-mac`)

These settings are applied to interfaces, typically bonds.

An Ethernet segment configuration has these characteristics:

- The `es-id` is a 24-bit integer (1-16777215).
- Each interface (bond) needs its own `es-id`.
- Static and LACP bonds can be associated with an `es-id`.

A *designated forwarder* (DF) is elected for each Ethernet segment. The DF is responsible for forwarding flooded traffic received via the VXLAN overlay to the locally attached Ethernet segment. We recommend you specify a preference (using the `es-df-pref` option) on an Ethernet segment for the DF election, as this leads to predictable failure scenarios. The EVPN VTEP with the highest `es-df-pref` setting becomes the DF. The `es-df-pref` setting defaults to *32767*.

NCLU generates the EVPN-MH configuration and reloads FRR and `ifupdown2`. The configuration appears in both the `/etc/network/interfaces` file and in `/etc/frr/frr.conf` file.

> ⚠ **IMPORTANT**
>
> In addition to the `es-id` and the `es-sys-mac`, you need to specify a unique SVI IP address for each VTEP across the racks. These IP addresses must be reachable from remote VTEPs. You enable the

advertisement of these IP addresses using the `advertise-svi-ip` option, under the BGP EVPN address family. See the leaf configurations in the example configuration below.

Enable EVPN-MH in switchd

To enable EVPN-MH in `switchd`, set the `evpn.multihoming.enable` variable in `switchd.conf` to *TRUE*, then restart the `switchd` service. The variable is disabled by default.

```
cumulus@switch:~$ sudo nano /etc/cumulus/switchd.conf
...

evpn.multihoming.enable = TRUE

...

cumulus@switch:~$ sudo systemctl restart switchd.service
```

Configure the EVPN-MH Bonds

To configure bond interfaces for EVPN multihoming, run commands similar to the following:

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add bond hostbond1 bond slaves swp5
cumulus@switch:~$ net add bond hostbond2 bond slaves swp6
cumulus@switch:~$ net add bond hostbond3 bond slaves swp7
cumulus@switch:~$ net add bond hostbond1 evpn mh es-id 1
cumulus@switch:~$ net add bond hostbond2 evpn mh es-id 2
cumulus@switch:~$ net add bond hostbond3 evpn mh es-id 3
cumulus@switch:~$ net add bond hostbond1-3 evpn mh es-sys-
mac 44:38:39:ff:ff:01
cumulus@switch:~$ net add bond hostbond1-3 evpn mh es-df-
pref 50000
cumulus@switch:~$ net commit
```

The NCLU commands create the following configuration in the `/etc/network/interfaces` file. If you are editing the `/etc/network/interfaces` file directly, apply a configuration like the following:

```
interface hostbond1
```

```
  bond-slaves swp5
  es-sys-mac 44:38:39:ff:ff:01

interface hostbond2
  bond-slaves swp6
  es-sys-mac 44:38:39:ff:ff:01

interface hostbond3
  bond-slaves swp7
  es-sys-mac 44:38:39:ff:ff:01
```

These commands also create the following configuration in the `/etc/frr/frr.conf` file.

```
!
interface hostbond1
 evpn mh es-df-pref 50000
 evpn mh es-id 1
 evpn mh es-sys-mac 44:38:39:ff:ff:01
!
interface hostbond2
 evpn mh es-df-pref 50000
```

```
 evpn mh es-id 2
 evpn mh es-sys-mac 44:38:39:ff:ff:01
!
interface hostbond3
 evpn mh es-df-pref 50000
 evpn mh es-id 3
 evpn mh es-sys-mac 44:38:39:ff:ff:01
!
```

EVPN MH Global Settings

There are a few global settings for EVPN multihoming you can set, including:

- `mac-holdtime`: MAC hold time, in seconds. This is the duration for which a switch maintains SYNC MAC entries after the Ethernet segment peer's EVPN type-2 route is deleted. During this time, the switch attempts to independently establish reachability of the MAC on the local Ethernet segment. The hold time can be between 0 and 86400 seconds. The default is 1080 seconds.
- `neigh-holdtime`: Neighbor entry hold time, in seconds. The duration for which a switch maintains SYNC neigh entries after the Ethernet segment peer's EVPN type-2 route is deleted. During this time, the switch attempts to independently establish reachability of the host on the local Ethernet segment. The hold time can be between 0 and 86400 seconds.

The default is 1080 seconds.

- `redirect-off`: **Cumulus VX only.** Disables fast failover of traffic destined to the access port via the VXLAN overlay. This knob only applies to Cumulus VX, since fast failover is only supported on the ASIC.
- `startup-delay`: Startup delay. The duration for which a switch holds the Ethernet segment-bond in a protodown state after a reboot or process restart. This allows the initialization of the VXLAN overlay to complete. The delay can be between 0 and 216000 seconds. The default is 180 seconds.

To configure a MAC hold time for 1000 seconds, run the following commands:

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add evpn mh mac-holdtime 1000
cumulus@switch:~$ net commit
```

This creates the following configuration in the `/etc/frr/frr.conf` file:

```
evpn mh mac-holdtime 1200
```

To configure a neighbor hold time for 600 seconds, run the following commands:

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add evpn mh neigh-holdtime 600
cumulus@switch:~$ net commit
```

This creates the following configuration in the `/etc/frr/frr.conf` file:

```
evpn mh neigh-holdtime 600
```

To configure a startup delay for 1800 seconds, run the following commands:

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add evpn mh startup-delay 1800
cumulus@switch:~$ net commit
```

This creates the following configuration in the `/etc/frr/frr.conf` file:

```
evpn mh startup-delay 1800
```

Enable Uplink Tracking

When all the uplinks go down, the VTEP loses connectivity to the VXLAN overlay. To prevent traffic loss in this state, the uplinks' oper-state is tracked. When all the uplinks are down, the Ethernet segment bonds on the switch are put into a protodown or error-disabled state. You can configure a link as an MH uplink to enable this tracking.

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add interface swp1-4 evpn mh uplink
cumulus@switch:~$ net add interface swp1-4 pim
cumulus@switch:~$ net commit
```

These commands create the following configuration in the `/etc/frr/frr.conf` file:

```
...
!
interface swp1
 evpn mh uplink
 ip pim
!
interface swp2
```

```
 evpn mh uplink
 ip pim
!
interface swp3
 evpn mh uplink
 ip pim
!
interface swp4
 evpn mh uplink
 ip pim
!
...
```

Enable FRR Debugging

You can add debug statements to the `/etc/frr/frr.conf` file to debug the Ethernet segments, routes and routing protocols (via Zebra).

NCLU Commands vtysh Commands

To debug Ethernet segments and routes, use the `net add bgp debug evpn mh (es|route)` command. To debug the routing protocols, use `net add evpn mh debug zebra (es|mac|neigh|nh)`.

```
cumulus@switch:~$ net add bgp debug evpn mh es
cumulus@switch:~$ net add bgp debug evpn mh route
cumulus@switch:~$ net add evpn mh debug zebra
cumulus@switch:~$ net add evpn mh debug zebra es
cumulus@switch:~$ net add evpn mh debug zebra mac
cumulus@switch:~$ net add evpn mh debug zebra neigh
cumulus@switch:~$ net add evpn mh debug zebra nh
cumulus@switch:~$ net commit
```

These commands create the following configuration in the `/etc/frr/frr.conf` file:

```
cumulus@switch:~$ cat /etc/frr/frr.conf
frr version 7.4+cl4u1
frr defaults datacenter

...
```

```
!
debug bgp evpn mh es
debug bgp evpn mh route
debug bgp zebra
debug zebra evpn mh es
debug zebra evpn mh mac
debug zebra evpn mh neigh
debug zebra evpn mh nh
debug zebra vxlan
!
...
```

Fast Failover

When an Ethernet segment link goes down, the attached VTEP notifies all other VTEPs via a single EAD-ES withdraw. This is done by way of an Ethernet segment bond redirect.

Fast failover is also triggered by:

- Rebooting a leaf switch or VTEP.
- Uplink failure. When all uplinks are down, the Ethernet segment bonds on the switch are protodowned or error disabled.

Disable Next Hop Group Sharing in the ASIC

Container sharing for both layer 2 and layer 3 next hop groups is enabled by default when EVPN-MH is configured. These settings are stored in the `evpn.multihoming.shared_l2_groups` and `evpn.multihoming.shared_l3_groups` variables.

Disabling container sharing allows for faster failover when an Ethernet segment link flaps.

To disable either setting, edit `switchd.conf`, set the variable to *FALSE*, then restart the `switchd` service. For example, to disable container sharing for layer 3 next hop groups, do the following:

```
cumulus@switch:~$ sudo nano /etc/cumulus/switchd.conf
...

evpn.multihoming.shared_l3_groups = FALSE

...

cumulus@switch:~$ sudo systemctl restart switchd.service
```

Disable EAD-per-EVI Route Advertisements

RFC 7432 requires type-1/EAD (Ethernet Auto-discovery) routes to be

advertised two ways:

- As EAD-per-ES (Ethernet Auto-discovery per Ethernet segment) routes
- As EAD-per-EVI (Ethernet Auto-discovery per EVPN instance) routes

Some third party switch vendors don't advertise EAD-per-EVI routes; they only advertise EAD-per-ES routes. To interoperate with these vendors, you need to disable EAD-per-EVI route advertisements.

To remove the dependency on EAD-per-EVI routes and activate the VTEP upon receiving the EAD-per-ES route, run:

```
cumulus@switch:~$ net add bgp l2vpn evpn disable-ead-evi-rx
cumulus@switch:~$ net commit
```

To suppress the advertisement of EAD-per-EVI routes, run:

```
cumulus@switch:~$ net add bgp l2vpn evpn disable-ead-evi-tx
cumulus@switch:~$ net commit
```

Troubleshooting

You can use the following `net show` commands to troubleshoot your EVPN multihoming configuration.

Show Ethernet Segment Information

The `net show evpn es` command displays the Ethernet segments across all VNIs.

```
cumulus@switch:~$ net show evpn es
Type: L local, R remote, N non-DF
ESI                             Type ES-IF                 VTEPs
03:44:38:39:ff:ff:01:00:00:01  R    -
172.0.0.22,172.0.0.23
03:44:38:39:ff:ff:01:00:00:02  LR   hostbond2
172.0.0.22,172.0.0.23
03:44:38:39:ff:ff:01:00:00:03  LR   hostbond3
172.0.0.22,172.0.0.23
03:44:38:39:ff:ff:01:00:00:05  L    hostbond1
03:44:38:39:ff:ff:02:00:00:01  R    -
172.0.0.24,172.0.0.25,172.0.0.26
03:44:38:39:ff:ff:02:00:00:02  R    -
172.0.0.24,172.0.0.25,172.0.0.26
03:44:38:39:ff:ff:02:00:00:03  R    -
172.0.0.24,172.0.0.25,172.0.0.26
```

Show Ethernet Segment per VNI Information

The `net show evpn es-evi` command displays the Ethernet segments learned for each VNI.

```
cumulus@switch:~$ net show evpn es-evi
Type: L local, R remote
VNI      ESI                             Type
...
1002     03:44:38:39:ff:ff:01:00:00:02   L
1002     03:44:38:39:ff:ff:01:00:00:03   L
1002     03:44:38:39:ff:ff:01:00:00:05   L
1001     03:44:38:39:ff:ff:01:00:00:02   L
1001     03:44:38:39:ff:ff:01:00:00:03   L
1001     03:44:38:39:ff:ff:01:00:00:05   L
...
```

Show BGP Ethernet Segment Information

The `net show bgp l2vpn evpn es` command displays the Ethernet segments across all VNIs learned via type-1 and type-4 routes.

```
cumulus@switch:~$ net show bgp l2vpn evpn es
ES Flags: L local, R remote, I inconsistent
VTEP Flags: E ESR/Type-4, A active nexthop
ESI                             Flags RD
#VNIs    VTEPs
03:44:38:39:ff:ff:01:00:00:01   LR    172.0.0.9:3
```

```
10       172.0.0.10(EA),172.0.0.11(EA)
03:44:38:39:ff:ff:01:00:00:02  LR    172.0.0.9:4
10       172.0.0.10(EA),172.0.0.11(EA)
03:44:38:39:ff:ff:01:00:00:03  LR    172.0.0.9:5
10       172.0.0.10(EA),172.0.0.11(EA)
cumulus@switch:~$
```

Show BGP Ethernet Segment per VNI Information

The `net show bgp l2vpn evpn es-evi` command displays the Ethernet segments per VNI learned via type-1 and type-4 routes.

```
cumulus@switch:~$ net show bgp l2vpn evpn es-evi
Flags: L local, R remote, I inconsistent
VTEP-Flags: E EAD-per-ES, V EAD-per-EVI
VNI      ESI                             Flags VTEPs
...
1002     03:44:38:39:ff:ff:01:00:00:01  R
172.0.0.22(EV),172.0.0.23(EV)
1002     03:44:38:39:ff:ff:01:00:00:02  LR
172.0.0.22(EV),172.0.0.23(EV)
1002     03:44:38:39:ff:ff:01:00:00:03  LR
172.0.0.22(EV),172.0.0.23(EV)
```

```
1002     03:44:38:39:ff:ff:01:00:00:05  L
1002     03:44:38:39:ff:ff:02:00:00:01  R
172.0.0.24(EV),172.0.0.25(EV),172.0.0.26(EV)
1002     03:44:38:39:ff:ff:02:00:00:02  R
172.0.0.24(EV),172.0.0.25(EV),172.0.0.26(EV)
1002     03:44:38:39:ff:ff:02:00:00:03  R
172.0.0.24(EV),172.0.0.25(EV),172.0.0.26(EV)
1001     03:44:38:39:ff:ff:01:00:00:01  R
172.0.0.22(EV),172.0.0.23(EV)
1001     03:44:38:39:ff:ff:01:00:00:02  LR
172.0.0.22(EV),172.0.0.23(EV)
1001     03:44:38:39:ff:ff:01:00:00:03  LR
172.0.0.22(EV),172.0.0.23(EV)
1001     03:44:38:39:ff:ff:01:00:00:05  L
1001     03:44:38:39:ff:ff:02:00:00:01  R
172.0.0.24(EV),172.0.0.25(EV),172.0.0.26(EV)
1001     03:44:38:39:ff:ff:02:00:00:02  R
172.0.0.24(EV),172.0.0.25(EV),172.0.0.26(EV)
1001     03:44:38:39:ff:ff:02:00:00:03  R
172.0.0.24(EV),172.0.0.25(EV),172.0.0.26(EV)
...
cumulus@switch:~$
```

Show EAD Route Types

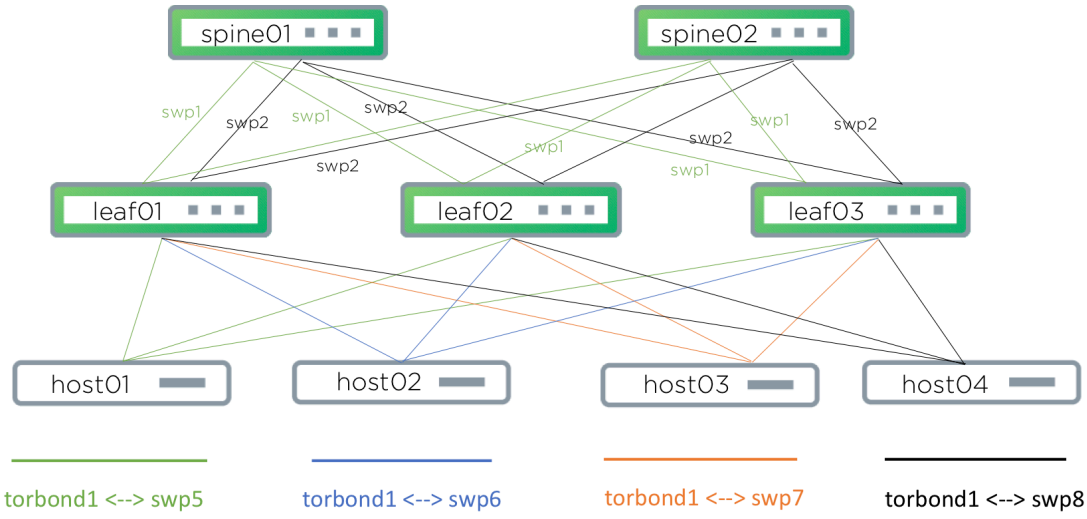
You can use the `net show bgp l2vpn evpn route` command to view type-1 EAD routes. Just include the `ead` route type option.

```
cumulus@switch:~$ net show bgp evpn l2vpn route type ead
BGP table version is 30, local router ID is 172.16.0.21
Status codes: s suppressed, d damped, h history, * valid, >
best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete
EVPN type-1 prefix: [1]:[ESI]:[EthTag]:[IPlen]:[VTEP-IP]
EVPN type-2 prefix: [2]:[EthTag]:[MAClen]:[MAC]:[IPlen]:[IP]
EVPN type-3 prefix: [3]:[EthTag]:[IPlen]:[OrigIP]
EVPN type-4 prefix: [4]:[ESI]:[IPlen]:[OrigIP]
EVPN type-5 prefix: [5]:[EthTag]:[IPlen]:[IP]

   Network          Next Hop            Metric LocPrf Weight
Path
                    Extended Community
Route Distinguisher: 172.16.0.21:2
*> [1]:[0]:[03:44:38:39:ff:ff:01:00:00:01]:[128]:[0.0.0.0]
                    172.16.0.21                           32768 i
                    ET:8 RT:5556:1005
*> [1]:[0]:[03:44:38:39:ff:ff:01:00:00:02]:[128]:[0.0.0.0]
                    172.16.0.21                           32768 i
```

```
                    ET:8 RT:5556:1005
*> [1]:[0]:[03:44:38:39:ff:ff:01:00:00:03]:[128]:[0.0.0.0]
                    172.16.0.21                            32768 i
                    ET:8 RT:5556:1005

...

Displayed 198 prefixes (693 paths) (of requested type)
cumulus@switch:~$
```

Example Configuration

The following example uses the topology illustrated here. It shows one rack for simplicity, but multiple racks can be added to this topology.



Configuration Commands

This section lists the NCLU commands to configure the switches and the network as well as the `vtysh` commands to configure FRRouting.

If you are not using NCLU to configure the `/etc/network/interfaces` file, go to /etc/network/interfaces below and copy the configurations directly into the `interfaces` file on each switch and server in the topology.

leaf01 leaf02 leaf03 spine01 spine02

NCLU Commands

```
cumulus@leaf01:~$ net show configuration commands
net del all
net add dns nameserver ipv4 192.168.0.3 vrf mgmt
net add time ntp server 0.cumulusnetworks.pool.ntp.org
iburst
net add time ntp server 1.cumulusnetworks.pool.ntp.org
iburst
net add time ntp server 2.cumulusnetworks.pool.ntp.org
iburst
net add time ntp server 3.cumulusnetworks.pool.ntp.org
iburst
net add time ntp source eth0
net add snmp-server listening-address localhost
net add bgp autonomous-system 5556
net add bond hostbond1-3 evpn mh es-df-pref 50000
net add bond hostbond1-3 evpn mh es-sys-mac
44:38:39:ff:ff:01
net add interface ipmr-lo,lo,swp1-4 pim
net add interface swp1-4 evpn mh uplink
net add bond hostbond1 evpn mh es-id 1
net add bond hostbond2 evpn mh es-id 2
net add bond hostbond3 evpn mh es-id 3
net add interface lo igmp
net add routing defaults datacenter
net add routing log file /var/log/frr/bgpd.log
net add routing [illegible]
net add routing line vty exec-timeout 0 0
```

/etc/network/interfaces

If you are using the NCLU commands listed above, they create the following configurations in the `/etc/network/interfaces` files for the leaf and spine switches.

If you are not using NCLU and are configuring the topology on the command line, copy the configurations below to the appropriate switches or servers. For the leaf and spine switch configurations, reload the new configuration by running `ifreload -a`:

```
cumulus@switch:~$ sudo ifreload -a
```

leaf01 leaf02 leaf03 spine01 spine02 host01

host02 host03 host04

```
cumulus@leaf01:~$ cat /etc/network/interfaces
# This file describes the network interfaces available on
your system
# and how to activate them. For more information, see
interfaces(5)

# The primary network interface
auto eth0
iface eth0
    address 192.168.0.15/24
    gateway 192.168.0.2
    vrf mgmt

#Enabling Mgmt VRF interface
auto mgmt
iface mgmt
    address 172.16.0.1/8
    address ::1/128
    vrf-table auto

auto lo
iface lo
    address 172.16.0.21/32
    alias BGP un-numbered Use for Vxlan Src Tunnel

auto swp1
iface swp1
    alias Local Node leaf01 and Ports swp1 <==> Remote
```

/etc/frr/frr.conf

These `vtysh` commands create the following configuration in the `/etc/frr/frr.conf` file:

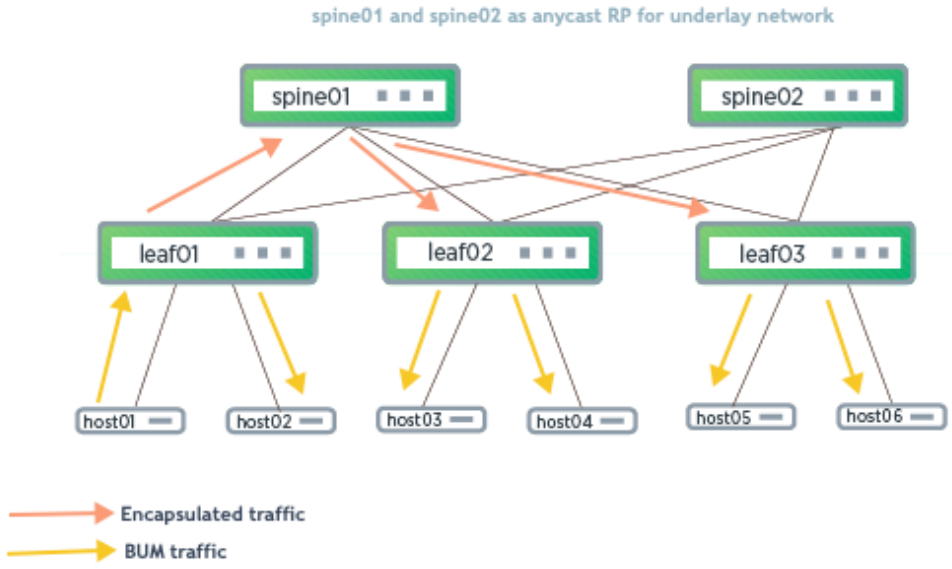
leaf01 leaf02 leaf03 spine01 spine02

```
cumulus@leaf01:~$ cat /etc/frr/frr.conf
frr version 7.4+cl4u1
frr defaults datacenter
hostname leaf01
log file /var/log/frr/bgpd.log
log timestamp precision 6
evpn mh startup-delay 30
zebra nexthop proto only
ip pim rp 192.0.2.5 239.1.1.0/24
ip pim spt-switchover infinity-and-beyond
service integrated-vtysh-config
!
debug bgp evpn mh es
debug bgp evpn mh route
debug bgp zebra
debug zebra evpn mh es
debug zebra evpn mh mac
debug zebra evpn mh neigh
debug zebra evpn mh nh
debug zebra vxlan
!
enable password cn321
password cn321
!
vrf vrf1
 vni 4001
 exit-vrf
!
vrf vrf2
```

EVPN BUM Traffic with PIM-SM

Without EVPN and PIM-SM, HER is the default way to replicate BUM traffic to remote VTEPs, where the ingress VTEP generates as many copies as VTEPs for each overlay BUM packet. This might not be optimal in certain deployments.

The following example shows a EVPN-PIM configuration, where underlay multicast is used to distribute BUM traffic. A multicast distribution tree (MDT) optimizes the flow of overlay BUM in the underlay network.



In the above example, host01 sends an ARP request to resolve host03. leaf01 (in addition to flooding the packet to host02) sends an encapsulated packet over the underlay network, which is forwarded using the MDT to

leaf02 and leaf03.

For PIM-SM, type-3 routes do not result in any forwarding entries. Cumulus Linux does **not** advertise type-3 routes for a layer 2 VNI when BUM mode for that VNI is PIM-SM.

> **NOTE**
> EVPN-PIM is supported on Broadcom Trident3 and Trident 2+ switches, and Mellanox Spectrum, Spectrum-2 and Spectrum-3 switches.

Configure Multicast VXLAN Tunnels

To configure multicast VXLAN tunnels, you need to configure PIM-SM in the underlay:

- Enable PIM-SM on the appropriate layer 3 interfaces.
- Configure static RP on all the PIM routers.
- Configure MSDP on the RPs for RP redundancy.

The configuration steps needed to configure PIM-SM in the underlay are provided in Protocol Independent Multicast - PIM.

In addition to the PIM-SM configuration, you need to run the following commands on each VTEP to provide the layer 2 VNI to MDT mapping.

NCLU Commands | Linux Commands

Run the `net add vxlan <interface> vxlan mcastgrp <ip-address>` command. For example:

```
cumulus@switch:~$ net add vxlan vxlan1000111 vxlan mcastgrp
239.1.1.111
```

NOTE

One multicast group per layer 2 VNI is optimal configuration for underlay bandwidth utilization. However, you can specify the same multicast group for more than one layer 2 VNI.

Verify EVPN-PIM

Run the NCLU `net show mroute` command or the vtysh `show ip mroute` command to review the multicast route information in FRRouting. When using EVPN-PIM, every VTEP acts as both source and destination for a VNI-MDT group, therefore, mroute entries on each VTEP should look like this:

```
cumulus@switch:~$ net show mroute
Source          Group           Proto  Input
Output           TTL  Uptime
*               239.1.1.111     IGMP   swp2
pimreg           1    21:37:36
                                PIM                    ipmr-
lo          1    21:37:36
10.0.0.28       239.1.1.111     STAR   lo              ipmr-
lo          1    21:36:41
                                PIM
swp2             1    21:36:41
*               239.1.1.112     IGMP   swp2
pimreg           1    21:37:36
                                PIM                    ipmr-
lo          1    21:37:36
10.0.0.28       239.1.1.112     STAR   lo              ipmr-
lo          1    21:36:41
                                PIM
swp2             1    21:36:41
```

(*,G) entries should show `ipmr-lo` in the OIL (Outgoing Interface List) and (S,G) entries should show `lo` as the Source interface or incoming interface and `ipmr-lo` in the OIL.

Run the `ip mroute` command to review the multicast route information in

the kernel. The kernel information should match the FRR information.

```
cumulus@switch:~$ ip mroute
(10.0.0.28,239.1.1.112)      Iif: lo     Oifs: swp2   State:
resolved
(10.0.0.28,239.1.1.111)      Iif: lo     Oifs: swp2   State:
resolved
(0.0.0.0,239.1.1.111)        Iif: swp2   Oifs: pimreg ipmr-lo
swp2  State: resolved
(0.0.0.0,239.1.1.112)        Iif: swp2   Oifs: pimreg ipmr-lo
swp2  State: resolved
```

Run the `bridge fdb show | grep 00:00:00:00:00:00` command to verify that all zero MAC addresses for every VXLAN device point to the correct multicast group destination.

```
cumulus@switch:~$ bridge fdb show | grep 00:00:00:00:00:00
00:00:00:00:00:00 dev vxlan1000112 dst 239.1.1.112 self
permanent
00:00:00:00:00:00 dev vxlan1000111 dst 239.1.1.111 self
permanent
```

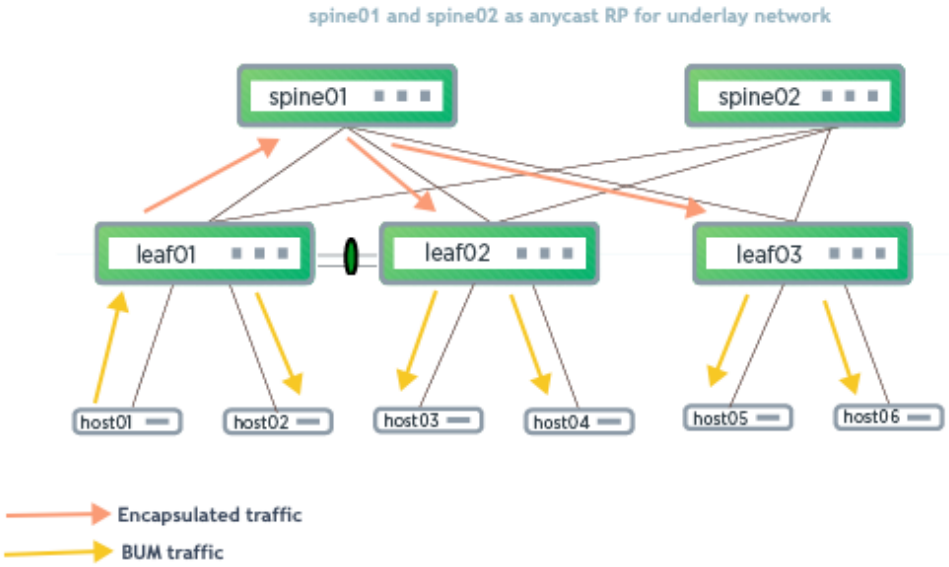
> **NOTE**
>
> The `show ip mroute count` command, often used to check multicast packet counts does *not* update for encapsulated BUM traffic originating or terminating on the VTEPs.

Run the NCLU `net show evpn vni <vni>` command or the vtysh `show evpn vni <vni>` command to ensure that your layer 2 VNI has the correct flooding information:

```
cumulus@switch:~$ net show evpn vni 10
VNI: 10
 Type: L2
 Tenant VRF: default
 VxLAN interface: vni10
 VxLAN ifIndex: 18
 Local VTEP IP: 10.0.0.28
 Mcast group: 239.1.1.112   <<<<<<<
 Remote VTEPs for this VNI:
  10.0.0.26 flood: -
  10.0.0.27 flood: -
 Number of MACs (local and remote) known for this VNI: 9
 Number of ARPs (IPv4 and IPv6, local and remote) known for
```

```
this VNI: 14

 Advertise-gw-macip: No
```

Configure EVPN-PIM in VXLAN Active-active Mode



To configure EVPN-PIM in VXLAN active-active mode, enable PIM on the peer link on each MLAG peer switch (**in addition to** the configuration described in Configure Multicast VXLAN Tunnels, above).

NCLU Commands | vtysh Commands

Run the `net add interface <peerlink> pim` command. For example:

```
cumulus@switch:~$ net add interface peerlink.4094 pim
cumulus@switch:~$ net commit
cumulus@switch:~$ net pending
```

Example Configuration

The following example shows an EVPN-PIM configuration on the VTEP, where:

- PIM is enabled on swp1, swp2, and the loopback interface (shown in the example `/etc/frr/frr.conf` file below).
- The group mapping 192.168.0.1 is configured for a static RP (shown at the top of the `/etc/frr/frr.conf` file example below).
- Multicast group 239.1.1.111 is mapped to VXLAN1000111. Multicast group 239.1.1.112 is mapped to VXLAN1000112 (shown in the example `/etc/network/interfaces` file below).

/etc/frr/frr.conf file /etc/network/interfaces file

```
cumulus@switch:~$ sudo cat /etc/frr/frr.conf
...
ip pim rp 192.168.0.1
ip pim keep-alive-timer 3600
...
vrf vrf1
 vni 104001
 exit-vrf
!
vrf vrf2
 vni 104002
 exit-vrf
!
interface swp1
 description swp1 > leaf-11's swp3
 ip ospf network point-to-point
 ip pim
!
interface swp2
 description swp2 > leaf-12's swp3
 ip ospf network point-to-point
 ip pim
!
interface swp3
 description swp3 > host-111's swp1
!
interface swp6
 description swp6 > host-112's swp1
!
```

Troubleshooting

This section provides various commands to help you examine your EVPN configuration and provides troubleshooting tips.

All of the following outputs are from the EVPN Symmetric Cumulus in the Cloud demo.

General Linux Commands

You can use various `iproute2` and `NCLU` commands to examine links, VLAN mappings and the bridge MAC forwarding database known to the Linux kernel. You can also use these commands to examine the neighbor cache and the routing table (for the underlay or for a specific tenant VRF). Some of the key commands are:

- `ip [-d] link show`
- `bridge link show`
- `bridge vlan show`
- `bridge [-s] fdb show`
- `ip neighbor show`
- `ip route show [table <vrf-name>]`

A sample output of `ip -d link show type vxlan` is shown below for one VXLAN interface. Relevant parameters are the VNI value, the state, the local IP address for the VXLAN tunnel, the UDP port number (4789) and the bridge of which the interface is part (*bridge* in the example below). The

output also shows that MAC learning is disabled (*off*) on the VXLAN interface.

```
cumulus@leaf01:~$ ip -d link show type vxlan
14: vni10: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9216 qdisc
noqueue master bridge state UP mode DEFAULT group default qlen
1000
    link/ether 42:83:73:20:46:ba brd ff:ff:ff:ff:ff:ff
promiscuity 1 minmtu 68 maxmtu 65535
    vxlan id 10 local 10.0.1.1 srcport 0 0 dstport 4789
nolearning ttl 64 ageing 300 udpcsum noudp6zerocsumtx
noudp6zerocsumrx
    bridge_slave state forwarding priority 8 cost 100 hairpin
off guard off root_block off fastleave off learning off flood
on port_id 0x8005 port_no 0x5 designated_port 32773
designated_cost 0 designated_bridge 8000.76:ed:2a:8a:67:24
designated_root 8000.76:ed:2a:8a:67:24 hold_timer    0.00
message_age_timer    0.00 forward_delay_timer    0.00
topology_change_ack 0 config_pending 0 proxy_arp off
proxy_arp_wifi off mcast_router 1 mcast_fast_leave off
mcast_flood on neigh_suppress on group_fwd_mask 0x0
group_fwd_mask_str 0x0 group_fwd_maskhi 0x0
group_fwd_maskhi_str 0x0 vlan_tunnel off isolated off
addrgenmode eui64 numtxqueues 1 numrxqueues 1 gso_max_size
65536 gso_max_segs 65535
```

```
...
```

The following example output for the `net show bridge macs` command shows:

- bond1 is an access ports with VLAN ID 10. This is mapped to VXLAN interface vni10.
- 26:76:e6:93:32:78 is the server01 host MAC learned on bond1.
- A remote VTEP that participate in VLAN ID 10 is 10.0.1.2 (the FDB entries have a MAC address of 00:00:00:00:00:00). These entries are used for BUM traffic replication.
- 68:0f:31:ae:3d:7a is a remote host MAC of server04 reachable over the VXLAN tunnel via VTEP 10.0.1.2.

```
cumulus@leaf01:mgmt:~$ net show bridge macs

VLAN      Master  Interface  MAC               TunnelDest
State      Flags              LastSeen
--------  ------  ---------  -----------------  ----------
---------  ------------------  --------
10        bridge  bond1
00:60:08:69:97:ef
00:01:40
```

```
10        bridge  bond1
26:76:e6:93:32:78
<1 sec
10        bridge  bridge     00:00:00:00:00:1a
permanent                      00:13:08
10        bridge  bridge     76:ed:2a:8a:67:24
permanent                      00:13:08
10        bridge  peerlink   c0:8a:e6:03:96:d0
static     sticky              00:12:58
10        bridge  vni10      50:88:b2:3c:08:f9
static     sticky              00:10:01
10        bridge  vni10
68:0f:31:ae:3d:7a                        extern_learn
00:09:58
10        bridge  vni10
94:8e:1c:0d:77:93                        extern_learn
00:09:58
10        bridge  vni10      c8:7d:bc:96:71:f3
static     sticky              00:10:01
20        bridge  bond2
cc:6e:fa:8d:ff:92
00:00:26
20        bridge  bond2
f0:9d:d0:59:60:5d
```

```
00:00:08
20        bridge  bridge     00:00:00:00:00:1b
permanent                      00:36:50
20        bridge  bridge     76:ed:2a:8a:67:24
permanent                      00:36:50
20        bridge  peerlink   c0:8a:e6:03:96:d0
static     sticky              00:36:41
20        bridge  vni20
12:15:9a:9c:f2:e1                        extern_learn
00:33:41
20        bridge  vni20      50:88:b2:3c:08:f9
static     sticky              00:33:44
20        bridge  vni20      c8:7d:bc:96:71:f3
static     sticky              00:33:44
20        bridge  vni20
f8:4f:db:ef:be:8b                        extern_learn
00:33:40
untagged          vni10      00:00:00:00:00:00  10.0.1.2
permanent  self                00:10:01
untagged          vni10      50:88:b2:3c:08:f9  10.0.1.2
static     self, sticky        00:10:01
untagged          vni10      68:0f:31:ae:3d:7a
10.0.1.2              self, extern_learn  00:09:58
untagged          vni10      94:8e:1c:0d:77:93
```

```
10.0.1.2               self, extern_learn  00:09:58
untagged          vni10      c8:7d:bc:96:71:f3  10.0.1.2
static     self, sticky        00:10:01
...
```

The following example output for the `net show neighbor` command shows:

- 10.1.10.101 is a locally-attached host server01 on VLAN 10. Interface `vlan10-v0` is the virtual VRR address for VLAN10.
- 10.1.10.104 is remote-host, server04 on VLAN10. This is indicated by the STATE `zebra` indicating it's an EVPN learned entry. Use `net show bridge macs` to see information about which VTEP the host is behind.
- 10.1.20.105 is remote-host, server05 on VLAN 20.

```
cumulus@leaf01:mgmt:~$ net show neigh
Neighbor                   MAC                Interface
AF    STATE
-------------------------  -----------------  -------------
----  ---------
10.1.10.104                68:0f:31:ae:3d:7a  vlan10
IPv4  zebra
10.1.10.101                26:76:e6:93:32:78  vlan10-v0
IPv4  REACHABLE
```

```
169.254.0.1               c0:8a:e6:03:96:d0  peerlink.4094
IPv4  zebra
10.0.1.2                  44:38:39:be:ef:bb  vlan4001
IPv4  zebra
169.254.0.1               c0:99:6b:c0:e1:ca  swp52
IPv4  zebra
10.1.20.3                 c0:8a:e6:03:96:d0  vlan20
IPv4  PERMANENT
169.254.0.1               ac:56:f0:f3:59:0c  swp54
IPv4  zebra
10.1.20.105               12:15:9a:9c:f2:e1  vlan20
IPv4  zebra
169.254.0.1               2c:f3:45:f4:6f:5f  swp53
IPv4  zebra
192.168.200.1             12:72:bc:4c:e1:83  eth0
IPv4  REACHABLE
169.254.0.1               f0:08:5f:12:cc:8c  swp51
IPv4  zebra
192.168.200.250           44:38:39:00:01:80  eth0
IPv4  REACHABLE
10.1.30.3                 c0:8a:e6:03:96:d0  vlan30
IPv4  PERMANENT
192.168.200.2             02:7a:19:45:66:48  eth0
IPv4  STALE
```

```
10.1.10.101               26:76:e6:93:32:78  vlan10
IPv4  REACHABLE
10.1.10.3                 c0:8a:e6:03:96:d0  vlan10
IPv4  PERMANENT
...
```

General BGP Commands

If you use BGP for the underlay routing, run the NCLU `net show bgp summary` command or the vtysh `show bgp summary` command to view a summary of the layer 3 fabric connectivity:

```
cumulus@leaf01:mgmt:~$ net show bgp summary
show bgp ipv4 unicast summary
=============================
BGP router identifier 10.10.10.1, local AS number 65101 vrf-id 0
BGP table version 13
RIB entries 25, using 4800 bytes of memory
Peers 5, using 106 KiB of memory
Peer groups 1, using 64 bytes of memory

Neighbor              V         AS   MsgRcvd   MsgSent
TblVer  InQ OutQ  Up/Down State/PfxRcd
```

```
spine01(swp51)        4      65199       814       805
0    0    0 00:37:34            7
spine02(swp52)        4      65199       814       805
0    0    0 00:37:34            7
spine03(swp53)        4      65199       814       805
0    0    0 00:37:34            7
spine04(swp54)        4      65199       814       805
0    0    0 00:37:34            7
leaf02(peerlink.4094) 4      65101       766       768
0    0    0 00:37:35           12

Total number of neighbors 5



show bgp ipv6 unicast summary
=============================
% No BGP neighbors found



show bgp l2vpn evpn summary
===========================
BGP router identifier 10.10.10.1, local AS number 65101 vrf-id 0
BGP table version 0
RIB entries 23, using 4416 bytes of memory
```

```
Peers 4, using 85 KiB of memory
Peer groups 1, using 64 bytes of memory

Neighbor        V         AS   MsgRcvd   MsgSent   TblVer  InQ
OutQ  Up/Down State/PfxRcd
spine01(swp51)  4      65199       814       805        0
0    0 00:37:35           34
spine02(swp52)  4      65199       814       805        0
0    0 00:37:35           34
spine03(swp53)  4      65199       814       805        0
0    0 00:37:35           34
spine04(swp54)  4      65199       814       805        0
0    0 00:37:35           34

Total number of neighbors 4
```

Run the NCLU `net show route` command or the vtysh `show route` command to examine the underlay routing and determine how remote VTEPs are reached. The following example shows output from a leaf switch:

This is the routing table of the global (underlay) routing table. Use

NOTE

the `vrf` keyword to see routes for specific VRFs where the hosts reside.

```
cumulus@leaf01:mgmt:~$ net show route
show ip route
=============
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r -
rejected route

C>* 10.0.1.1/32 is directly connected, lo, 00:40:02
B>* 10.0.1.2/32 [20/0] via fe80::2ef3:45ff:fef4:6f5f, swp53,
weight 1, 00:40:04
  *                    via fe80::ae56:f0ff:fef3:590c, swp54,
weight 1, 00:40:04
  *                    via fe80::c299:6bff:fec0:e1ca, swp52,
weight 1, 00:40:04
  *                    via fe80::f208:5fff:fe12:cc8c, swp51,
```

```
weight 1, 00:40:04
B>* 10.0.1.254/32 [20/0] via fe80::2ef3:45ff:fef4:6f5f, swp53,
weight 1, 00:35:18
  *                       via fe80::ae56:f0ff:fef3:590c, swp54,
weight 1, 00:35:18
  *                       via fe80::c299:6bff:fec0:e1ca, swp52,
weight 1, 00:35:18
  *                       via fe80::f208:5fff:fe12:cc8c, swp51,
weight 1, 00:35:18
C>* 10.10.10.1/32 is directly connected, lo, 00:42:58
B>* 10.10.10.2/32 [200/0] via fe80::c28a:e6ff:fe03:96d0,
peerlink.4094, weight 1, 00:42:56
B>* 10.10.10.3/32 [20/0] via fe80::2ef3:45ff:fef4:6f5f, swp53,
weight 1, 00:42:55
  *                       via fe80::ae56:f0ff:fef3:590c, swp54,
weight 1, 00:42:55
  *                       via fe80::c299:6bff:fec0:e1ca, swp52,
weight 1, 00:42:55
  *                       via fe80::f208:5fff:fe12:cc8c, swp51,
weight 1, 00:42:55
B>* 10.10.10.4/32 [20/0] via fe80::2ef3:45ff:fef4:6f5f, swp53,
weight 1, 00:42:55
  *                       via fe80::ae56:f0ff:fef3:590c, swp54,
weight 1, 00:42:55
```

```
  *                      via fe80::c299:6bff:fec0:e1ca, swp52,
weight 1, 00:42:55
  *                      via fe80::f208:5fff:fe12:cc8c, swp51,
weight 1, 00:42:55
B>* 10.10.10.63/32 [20/0] via fe80::2ef3:45ff:fef4:6f5f, swp53,
weight 1, 00:42:55
  *                       via fe80::ae56:f0ff:fef3:590c, swp54,
weight 1, 00:42:55
  *                       via fe80::c299:6bff:fec0:e1ca, swp52,
weight 1, 00:42:55
  *                       via fe80::f208:5fff:fe12:cc8c, swp51,
weight 1, 00:42:55
B>* 10.10.10.64/32 [20/0] via fe80::2ef3:45ff:fef4:6f5f, swp53,
weight 1, 00:38:07
  *                       via fe80::ae56:f0ff:fef3:590c, swp54,
weight 1, 00:38:07
  *                       via fe80::c299:6bff:fec0:e1ca, swp52,
weight 1, 00:38:07
  *                       via fe80::f208:5fff:fe12:cc8c, swp51,
weight 1, 00:38:07
B>* 10.10.10.101/32 [20/0] via fe80::f208:5fff:fe12:cc8c,
swp51, weight 1, 00:42:56
B>* 10.10.10.102/32 [20/0] via fe80::c299:6bff:fec0:e1ca,
swp52, weight 1, 00:42:56
```

```
B>* 10.10.10.103/32 [20/0] via fe80::2ef3:45ff:fef4:6f5f,
swp53, weight 1, 00:42:56
B>* 10.10.10.104/32 [20/0] via fe80::ae56:f0ff:fef3:590c,
swp54, weight 1, 00:42:56
```

Show EVPN address-family Peers

Run the NCLU `net show bgp l2vpn evpn summary` command or the vtysh `show bgp l2vpn evpn summary` command to see the BGP peers participating in the layer 2 VPN/EVPN address-family and their states. The following example output from a leaf switch shows eBGP peering with four spine switches to exchange EVPN routes; all peering sessions are in the *established* state.

```
cumulus@leaf01:mgmt:~$ net show bgp l2vpn evpn summary
BGP router identifier 10.10.10.1, local AS number 65101 vrf-id 0
BGP table version 0
RIB entries 23, using 4416 bytes of memory
Peers 4, using 85 KiB of memory
Peer groups 1, using 64 bytes of memory

Neighbor        V         AS   MsgRcvd   MsgSent   TblVer  InQ
OutQ  Up/Down State/PfxRcd
```

```
spine01(swp51)  4      65199       958       949        0
0    0 00:44:46           34
spine02(swp52)  4      65199       958       949        0
0    0 00:44:46           34
spine03(swp53)  4      65199       958       949        0
0    0 00:44:46           34
spine04(swp54)  4      65199       958       949        0
0    0 00:44:46           34

Total number of neighbors 4
```

Show EVPN VNIs

Run the NCLU `net show bgp l2vpn evpn vni` command or the vtysh `show bgp l2vpn evpn vni` command to display the configured VNIs on a network device participating in BGP EVPN. This command is only relevant on a VTEP. If you have configured symmetric routing, this command displays the special layer 3 VNIs that are configured per tenant VRF.

The following example from leaf01 shows three layer 2 VNIs (10, 20 and 30) as well as two layer 3 VNIs (4001, 4002).

```
cumulus@leaf01:mgmt:~$ net show bgp l2vpn evpn vni
```

```
Advertise Gateway Macip: Disabled
Advertise SVI Macip: Disabled
Advertise All VNI flag: Enabled
BUM flooding: Head-end replication
Number of L2 VNIs: 3
Number of L3 VNIs: 2
Flags: * - Kernel
  VNI        Type RD                    Import
RT                 Export RT                Tenant VRF
* 20         L2   10.10.10.1:4
65101:20                  65101:20                RED
* 30         L2   10.10.10.1:6
65101:30                  65101:30                BLUE
* 10         L2   10.10.10.1:3
65101:10                  65101:10                RED
* 4002       L3   10.1.30.2:2
65101:4002                65101:4002              BLUE
* 4001       L3   10.1.20.2:5
65101:4001                65101:4001              RED
```

Run the NCLU `net show evpn vni` command to see a summary of VNIs and the number of MAC or ARP entries associated with each VNI.

```
cumulus@leaf01:mgmt:~$ net show evpn vni
VNI        Type VxLAN IF              # MACs   # ARPs   #
Remote VTEPs  Tenant VRF
20         L2   vni20                8        5
1               RED
30         L2   vni30                8        4
1               BLUE
10         L2   vni10                8        6
1               RED
4001       L3   vniRED                1        1        n/
a             RED
4002       L3   vniBLUE               0        0        n/
a             BLUE
```

Run the NCLU `net show evpn vni <vni>` command or the vtysh `show evpn vni <vni>` command to examine EVPN information for a specific VNI in detail. The following example output shows details for the layer 2 VNI 10 as well as for the layer 3 VNI 4001. For the layer 2 VNI, the remote VTEPs that contain that VNI are shown. For the layer 3 VNI, the router MAC and associated layer 2 VNIs are shown. The state of the layer 3 VNI depends on the state of its associated VRF as well as the states of its underlying VXLAN interface and SVI.

```
cumulus@leaf01:mgmt:~$ net show evpn vni 10
VNI: 10
 Type: L2
 Tenant VRF: RED
 VxLAN interface: vni10
 VxLAN ifIndex: 14
 Local VTEP IP: 10.0.1.1
 Mcast group: 0.0.0.0
 Remote VTEPs for this VNI:
  10.0.1.2 flood: HER
 Number of MACs (local and remote) known for this VNI: 8
 Number of ARPs (IPv4 and IPv6, local and remote) known for
this VNI: 6
 Advertise-gw-macip: No
cumulus@leaf01:mgmt:~$
cumulus@leaf01:mgmt:~$ net show evpn vni 4001
VNI: 4001
  Type: L3
  Tenant VRF: RED
  Local Vtep Ip: 10.0.1.1
  Vxlan-Intf: vniRED
  SVI-If: vlan4001
  State: Up
  VNI Filter: none
```

```
  System MAC: 44:38:39:be:ef:aa
  Router MAC: 44:38:39:be:ef:aa
  L2 VNIs: 10 20
```

Examine Local and Remote MAC Addresses for a VNI

Run the NCLU `net show evpn mac vni <vni>` command or the vtysh `show evpn mac vni <vni>` command to examine all local and remote MAC addresses for a VNI. This command is only relevant for a layer 2 VNI:

```
cumulus@leaf01:mgmt:~$ net show evpn mac vni 10
Number of MACs (local and remote) known for this VNI: 8
Flags: B=bypass N=sync-neighs, I=local-inactive, P=peer-active,
X=peer-proxy
MAC               Type   Flags Intf/Remote ES/VTEP
VLAN  Seq #'s
26:76:e6:93:32:78 local        bond1
10    0/0
94:8e:1c:0d:77:93 remote
10.0.1.2                             0/0
50:88:b2:3c:08:f9 remote
10.0.1.2                             0/0
```

```
68:0f:31:ae:3d:7a remote
10.0.1.2                              1/0
c8:7d:bc:96:71:f3 remote
10.0.1.2                              0/0
c0:8a:e6:03:96:d0 local        peerlink
10    0/0
76:ed:2a:8a:67:24 local        vlan10
10    0/0
00:60:08:69:97:ef local        bond1
10    0/0
```

Run the NCLU `net show evpn mac vni all` command or the vtysh `show evpn mac vni all` command to examine MAC addresses for all VNIs.

You can examine the details for a specific MAC addresse or query all remote MAC addresses behind a specific VTEP:

```
cumulus@leaf01:mgmt:~$ net show evpn mac vni 10 mac
94:8e:1c:0d:77:93
MAC: 94:8e:1c:0d:77:93
 Remote VTEP: 10.0.1.2
 Sync-info: neigh#: 0
 Local Seq: 0 Remote Seq: 0
```

```
 Neighbors:
    No Neighbors
cumulus@leaf01:mgmt:~$
cumulus@leaf01:mgmt:~$ net show evpn mac vni 20 mac
94:8e:1c:0d:77:93
% Requested MAC does not exist in VNI 20
cumulus@leaf01:mgmt:~$
cumulus@leaf01:mgmt:~$ net show evpn mac vni 20 vtep 10.0.1.2
VNI 20

MAC               Type   FlagsIntf/Remote ES/VTEP
VLAN  Seq #'s
12:15:9a:9c:f2:e1 remote
10.0.1.2                              1/0
50:88:b2:3c:08:f9 remote
10.0.1.2                              0/0
f8:4f:db:ef:be:8b remote
10.0.1.2                              0/0
c8:7d:bc:96:71:f3 remote
10.0.1.2                              0/0
```

Examine Local and Remote Neighbors for a VNI

Run the NCLU `net show evpn arp-cache vni <vni>` command or the vtysh `show evpn arp-cache vni <vni>` command to examine all local and remote

neighbors (ARP entries) for a VNI. This command is only relevant for a layer 2 VNI and the output shows both IPv4 and IPv6 neighbor entries:

```
cumulus@leaf01:mgmt:~$ net show evpn arp-cache vni 10
Number of ARPs (local and remote) known for this VNI: 6
Flags: I=local-inactive, P=peer-active, X=peer-proxy
Neighbor                  Type   Flags State
MAC               Remote ES/VTEP                 Seq #'s
10.1.10.2                 local        active
76:ed:2a:8a:67:24                              0/0
fe80::968e:1cff:fe0d:7793 remote       active
68:0f:31:ae:3d:7a 10.0.1.2                     0/0
10.1.10.101               local        active
26:76:e6:93:32:78                              0/0
fe80::9465:45ff:fe6d:4890 local        active
26:76:e6:93:32:78                              0/0
10.1.10.104               remote       active
68:0f:31:ae:3d:7a 10.0.1.2                     0/0
fe80::74ed:2aff:fe8a:6724 local        active
76:ed:2a:8a:67:24                              0/0
...
```

Run the NCLU `net show evpn arp-cache vni all` command or the vtysh `show evpn arp-cache vni all` command to examine neighbor entries for all VNIs.

Examine Remote Router MACs

For symmetric routing, run the NCLU `net show evpn rmac vni <vni>` command or the vtysh `show evpn rmac vni <vni>` command to examine the router MACs corresponding to all remote VTEPs. This command is only relevant for a layer 3 VNI:

```
cumulus@leaf01:mgmt:~$ net show evpn rmac vni 4001
Number of Remote RMACs known for this VNI: 1
MAC                Remote VTEP
44:38:39:be:ef:bb 10.0.1.2
```

Run the NCLU `net show evpn rmac vni all` command or the vtysh `show evpn rmac vni all` command to examine router MACs for all layer 3 VNIs.

Examine Gateway Next Hops

For symmetric routing, you can run the NCLU `net show evpn next-hops vni <vni>` command or the vtysh `show evpn next-hops vni <vni>` command to examine the gateway next hops. This command is only relevant for a layer 3 VNI. In general, the gateway next hop IP addresses correspond to the remote VTEP IP addresses. Remote host and prefix routes are installed sing these next hops:

```
cumulus@leaf01:mgmt:~$ net show evpn next-hops vni 4001
Number of NH Neighbors known for this VNI: 1
IP              RMAC
10.0.1.2        44:38:39:be:ef:bb
```

Run the NCLU `net show evpn next-hops vni all` command or the vtysh `show evpn next-hops vni all` command to examine gateway next hops for all layer 3 VNIs.

You can query a specific next hop; the output displays the remote host and prefix routes through this next hop:

```
cumulus@leaf01:mgmt:~$ net show evpn next-hops vni 4001 ip
10.0.1.2
Ip: 10.0.1.2
  RMAC: 44:38:39:be:ef:bb
  Refcount: 2
  Prefixes:
    10.1.10.104/32
    10.1.20.105/32
```

Show the VRF Routing Table in FRRouting

Run the `net show route vrf <vrf-name>` command to examine the VRF routing table. In the context of EVPN, this command is relevant for symmetric routing to verify that remote host and prefix routes are installed in the VRF routing table and point to the appropriate gateway next hop.

```
cumulus@leaf01:mgmt:~$ net show route vrf RED
show ip route vrf RED
======================
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r -
rejected route



VRF RED:
K>* 0.0.0.0/0 [255/8192] unreachable (ICMP unreachable),
00:53:46
C * 10.1.10.0/24 [0/1024] is directly connected, vlan10-v0,
00:53:46
C>* 10.1.10.0/24 is directly connected, vlan10, 00:53:46
B>* 10.1.10.104/32 [20/0] via 10.0.1.2, vlan4001 onlink, weight
```

```
1, 00:43:55
C * 10.1.20.0/24 [0/1024] is directly connected, vlan20-v0,
00:53:46
C>* 10.1.20.0/24 is directly connected, vlan20, 00:53:46
B>* 10.1.20.105/32 [20/0] via 10.0.1.2, vlan4001 onlink, weight
1, 00:20:07
...
```

In the output above, the next hops for these routes are specified by EVPN to be *onlink*, or reachable over the specified SVI. This is necessary because this interface is not required to have an IP address. Even if the interface is configured with an IP address, the next hop is not on the same subnet as it is usually the IP address of the remote VTEP (part of the underlay IP network).

Show the Global BGP EVPN Routing Table

Run the NCLU `net show bgp l2vpn evpn route` command or the vtysh `show bgp l2vpn evpn route` command to display all EVPN routes, both local and remote. The routes displayed here are based on RD as they are across VNIs and VRFs:

```
cumulus@leaf01:mgmt:~$ net show bgp l2vpn evpn route
BGP table version is 6, local router ID is 10.10.10.1
Status codes: s suppressed, d damped, h history, * valid, >
best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete
EVPN type-1 prefix: [1]:[ESI]:[EthTag]:[IPlen]:[VTEP-IP]
EVPN type-2 prefix: [2]:[EthTag]:[MAClen]:[MAC]:[IPlen]:[IP]
EVPN type-3 prefix: [3]:[EthTag]:[IPlen]:[OrigIP]
EVPN type-4 prefix: [4]:[ESI]:[IPlen]:[OrigIP]
EVPN type-5 prefix: [5]:[EthTag]:[IPlen]:[IP]

   Network          Next Hop            Metric LocPrf Weight
Path
                    Extended Community
Route Distinguisher: 10.10.10.1:3
*> [2]:[0]:[48]:[00:60:08:69:97:ef]
                    10.0.1.1                           32768 i
                    ET:8 RT:65101:10 RT:65101:4001
Rmac:44:38:39:be:ef:aa
*> [2]:[0]:[48]:[26:76:e6:93:32:78]
                    10.0.1.1                           32768 i
                    ET:8 RT:65101:10 RT:65101:4001
Rmac:44:38:39:be:ef:aa
*> [2]:[0]:[48]:[26:76:e6:93:32:78]:[32]:[10.1.10.101]
```

```
                    10.0.1.1                            32768 i
                    ET:8 RT:65101:10 RT:65101:4001
Rmac:44:38:39:be:ef:aa
*>
[2]:[0]:[48]:[26:76:e6:93:32:78]:[128]:[fe80::9465:45ff:fe6d:4890]
                    10.0.1.1                            32768 i
                    ET:8 RT:65101:10
*> [2]:[0]:[48]:[c0:8a:e6:03:96:d0]
                    10.0.1.1                            32768 i
                    ET:8 RT:65101:10 RT:65101:4001 MM:0, sticky
MAC Rmac:44:38:39:be:ef:aa
*> [3]:[0]:[32]:[10.0.1.1]
                    10.0.1.1                            32768 i
                    ET:8 RT:65101:10
Route Distinguisher: 10.10.10.1:4
*> [2]:[0]:[48]:[c0:8a:e6:03:96:d0]
                    10.0.1.1                            32768 i
                    ET:8 RT:65101:20 RT:65101:4001 MM:0, sticky
MAC Rmac:44:38:39:be:ef:aa
*> [2]:[0]:[48]:[cc:6e:fa:8d:ff:92]
                    10.0.1.1                            32768 i
                    ET:8 RT:65101:20 RT:65101:4001
Rmac:44:38:39:be:ef:aa
*> [2]:[0]:[48]:[f0:9d:d0:59:60:5d]
```

```
                    10.0.1.1                            32768 i
                    ET:8 RT:65101:20 RT:65101:4001
Rmac:44:38:39:be:ef:aa
*>
[2]:[0]:[48]:[f0:9d:d0:59:60:5d]:[128]:[fe80::ce6e:faff:fe8d:ff92]
                    10.0.1.1                            32768 i
                    ET:8 RT:65101:20
*> [3]:[0]:[32]:[10.0.1.1]
                    10.0.1.1                            32768 i
                    ET:8 RT:65101:20
Route Distinguisher: 10.10.10.1:6
*> [2]:[0]:[48]:[c0:8a:e6:03:96:d0]
                    10.0.1.1                            32768 i
                    ET:8 RT:65101:30 RT:65101:4002 MM:0, sticky
MAC Rmac:44:38:39:be:ef:aa
*> [2]:[0]:[48]:[de:02:3b:17:c9:6d]
                    10.0.1.1                            32768 i
                    ET:8 RT:65101:30 RT:65101:4002
Rmac:44:38:39:be:ef:aa
*>
[2]:[0]:[48]:[de:02:3b:17:c9:6d]:[128]:[fe80::dc02:3bff:fe17:c96d]
                    10.0.1.1                            32768 i
                    ET:8 RT:65101:30
*> [2]:[0]:[48]:[ea:77:bb:f1:a7:ca]
```

```
                    10.0.1.1                            32768 i
                    ET:8 RT:65101:30 RT:65101:4002
Rmac:44:38:39:be:ef:aa
*> [3]:[0]:[32]:[10.0.1.1]
                    10.0.1.1                            32768 i
                    ET:8 RT:65101:30
Route Distinguisher: 10.10.10.3:3
*> [2]:[0]:[48]:[12:15:9a:9c:f2:e1]
                    10.0.1.2                                 0
65199 65102 i
                    RT:65102:20 RT:65102:4001 ET:8
Rmac:44:38:39:be:ef:bb
*  [2]:[0]:[48]:[12:15:9a:9c:f2:e1]
                    10.0.1.2                                 0
65199 65102 i
                    RT:65102:20 RT:65102:4001 ET:8
Rmac:44:38:39:be:ef:bb
...
```

You can filter the routing table based on EVPN route type. The available options are shown below:

```
cumulus@leaf01:mgmt:~$ net show bgp l2vpn evpn route type
```

```
    ead        :  EAD (Type-1) route
    es         :  Ethernet Segment (type-4) route
    macip      :  MAC-IP (Type-2) route
    multicast  :  Multicast
    prefix     :  An IPv4 or IPv6 prefix
```

Show a Specific EVPN Route

To drill down on a specific route for more information, run the NCLU `net show bgp l2vpn evpn route rd <rd-value>` command or the vtysh `show bgp l2vpn evpn route rd <rd-value>` command. This command displays all EVPN routes with that RD and with the path attribute details for each path. Additional filtering is possible based on route type or by specifying the MAC and/or IP address. The following example shows the specific MAC/IP route of server05. The output shows that this remote host is behind VTEP 10.10.10.3 and is reachable through four paths; one through each spine switch. This example is from a symmetric routing configuration, so the route shows both the layer 2 VNI (20) and the layer 3 VNI (4001), as well as the EVPN route target attributes corresponding to each and the associated router MAC address.

```
cumulus@leaf01:mgmt:~$ net show bgp l2vpn evpn route rd
10.10.10.3:3 mac 12:15:9a:9c:f2:e1 ip 10.1.20.105
```

```
BGP routing table entry for
10.10.10.3:3:[2]:[0]:[48]:[12:15:9a:9c:f2:e1]:[32]:[10.1.20.105]
Paths: (4 available, best #1)
  Advertised to non peer-group peers:
  spine01(swp51) spine02(swp52) spine03(swp53) spine04(swp54)
  Route [2]:[0]:[48]:[12:15:9a:9c:f2:e1]:[32]:[10.1.20.105] VNI
20/4001
  65199 65102
    10.0.1.2 from spine01(swp51) (10.10.10.101)
      Origin IGP, valid, external, bestpath-from-AS 65199, best
(Router ID)
      Extended Community: RT:65102:20 RT:65102:4001 ET:8
Rmac:44:38:39:be:ef:bb
      Last update: Fri Jan 15 08:16:24 2021
  Route [2]:[0]:[48]:[12:15:9a:9c:f2:e1]:[32]:[10.1.20.105] VNI
20/4001
  65199 65102
    10.0.1.2 from spine04(swp54) (10.10.10.104)
      Origin IGP, valid, external
      Extended Community: RT:65102:20 RT:65102:4001 ET:8
Rmac:44:38:39:be:ef:bb
      Last update: Fri Jan 15 08:16:24 2021
  Route [2]:[0]:[48]:[12:15:9a:9c:f2:e1]:[32]:[10.1.20.105] VNI
20/4001
```

```
  65199 65102
    10.0.1.2 from spine02(swp52) (10.10.10.102)
      Origin IGP, valid, external
      Extended Community: RT:65102:20 RT:65102:4001 ET:8
Rmac:44:38:39:be:ef:bb
      Last update: Fri Jan 15 08:16:24 2021
  Route [2]:[0]:[48]:[12:15:9a:9c:f2:e1]:[32]:[10.1.20.105] VNI
20/4001
  65199 65102
    10.0.1.2 from spine03(swp53) (10.10.10.103)
      Origin IGP, valid, external
      Extended Community: RT:65102:20 RT:65102:4001 ET:8
Rmac:44:38:39:be:ef:bb
      Last update: Fri Jan 15 08:16:24 2021

Displayed 4 paths for requested prefix
```

NOTE

- Only global VNIs are supported. Even though VNI values are exchanged in the type-2 and type-5 routes, the received values are not used when installing the routes into the

forwarding plane; the local configuration is used. You must ensure that the VLAN to VNI mappings and the layer 3 VNI assignment for a tenant VRF are uniform throughout the network.

- If the remote host is dual attached, the next hop for the EVPN route is the anycast IP address of the remote MLAG pair, when MLAG is active.

Show the per-VNI EVPN Routing Table

Received EVPN routes are maintained in the global EVPN routing table (described above), even if there are no appropriate local VNIs to **import** them into. For example, a spine switch maintains the global EVPN routing table even though there are no VNIs present on it. When local VNIs are present, received EVPN routes are imported into the per-VNI routing tables based on the route target attributes. You can examine the per-VNI routing table with the `net show bgp l2vpn evpn route vni <vni>` command:

```
cumulus@leaf01:mgmt:~$ net show bgp l2vpn evpn route vni 10
BGP table version is 16, local router ID is 10.10.10.1
Status codes: s suppressed, d damped, h history, * valid, >
```

```
best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete
EVPN type-1 prefix: [1]:[ESI]:[EthTag]:[IPlen]:[VTEP-IP]
EVPN type-2 prefix: [2]:[EthTag]:[MAClen]:[MAC]:[IPlen]:[IP]
EVPN type-3 prefix: [3]:[EthTag]:[IPlen]:[OrigIP]
EVPN type-4 prefix: [4]:[ESI]:[IPlen]:[OrigIP]
EVPN type-5 prefix: [5]:[EthTag]:[IPlen]:[IP]

   Network          Next Hop            Metric LocPrf Weight
Path
*> [2]:[0]:[48]:[00:60:08:69:97:ef]
                    10.0.1.1                           32768 i
                    ET:8 RT:65101:10 RT:65101:4001
Rmac:44:38:39:be:ef:aa
*> [2]:[0]:[48]:[26:76:e6:93:32:78]
                    10.0.1.1                           32768 i
                    ET:8 RT:65101:10 RT:65101:4001
Rmac:44:38:39:be:ef:aa
*> [2]:[0]:[48]:[26:76:e6:93:32:78]:[32]:[10.1.10.101]
                    10.0.1.1                           32768 i
                    ET:8 RT:65101:10 RT:65101:4001
Rmac:44:38:39:be:ef:aa
*>
[2]:[0]:[48]:[26:76:e6:93:32:78]:[128]:[fe80::9465:45ff:fe6d:4890]
```

```
                    10.0.1.1                            32768 i
                    ET:8 RT:65101:10
*  [2]:[0]:[48]:[50:88:b2:3c:08:f9]
                    10.0.1.2                                0
65199 65102 i
                    RT:65102:10 RT:65102:4001 ET:8 MM:0, sticky
MAC Rmac:44:38:39:be:ef:bb
*  [2]:[0]:[48]:[50:88:b2:3c:08:f9]
                    10.0.1.2                                0
65199 65102 i
                    RT:65102:10 RT:65102:4001 ET:8 MM:0, sticky
MAC Rmac:44:38:39:be:ef:bb
*  [2]:[0]:[48]:[50:88:b2:3c:08:f9]
                    10.0.1.2                                0
65199 65102 i
                    RT:65102:10 RT:65102:4001 ET:8 MM:0, sticky
MAC Rmac:44:38:39:be:ef:bb
*> [2]:[0]:[48]:[50:88:b2:3c:08:f9]
                    10.0.1.2                                0
65199 65102 i
                    RT:65102:10 RT:65102:4001 ET:8 MM:0, sticky
MAC Rmac:44:38:39:be:ef:bb
*  [2]:[0]:[48]:[68:0f:31:ae:3d:7a]
                    10.0.1.2                                0
```

```
65199 65102 i
                    RT:65102:10 RT:65102:4001 ET:8
Rmac:44:38:39:be:ef:bb
...
```

To display the VNI routing table for all VNIs, run the `net show bgp l2vpn evpn route vni all` command.

Show the per-VRF BGP Routing Table

For symmetric routing, received type-2 and type-5 routes are imported into the VRF routing table (against the corresponding address-family: IPv4 unicast or IPv6 unicast) based on a match on the route target attributes. Run the NCLU `net show bgp vrf <vrf-name> ipv4 unicast` command or the `net show bgp vrf <vrf-name> ipv6 unicast` command to examine the BGP VRF routing table. The equivalent vtysh commands are `show bgp vrf <vrf-name> ipv4 unicast` and `show bgp vrf <vrf-name> ipv6 unicast`.

```
cumulus@leaf01:mgmt:~$ net show bgp vrf RED ipv4 unicast
BGP table version is 2, local router ID is 10.1.20.2, vrf id 24
Default local pref 100, local AS 65101
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
*  10.1.10.104/32   10.0.1.2<                              0
65199 65102 i
*                   10.0.1.2<                              0
65199 65102 i
*                   10.0.1.2<                              0
65199 65102 i
*                   10.0.1.2<                              0
65199 65102 i
*>                  10.0.1.2<                              0
65199 65102 i
*                   10.0.1.2<                              0
65199 65102 i
*                   10.0.1.2<                              0
65199 65102 i
*                   10.0.1.2<                              0
65199 65102 i
*  10.1.20.105/32   10.0.1.2<                              0
```

```
65199 65102 i
*>                  10.0.1.2<                               0
65199 65102 i
*                   10.0.1.2<                               0
65199 65102 i
*                   10.0.1.2<                               0
65199 65102 i
*                   10.0.1.2<                               0
65199 65102 i
*                   10.0.1.2<                               0
65199 65102 i
*                   10.0.1.2<                               0
65199 65102 i
*                   10.0.1.2<                               0
65199 65102 i

Displayed  2 routes and 16 total paths
```

Support for EVPN Neighbor Discovery (ND) Extended Community

In an EVPN VXLAN deployment with ARP and ND suppression where the VTEPs are only configured for layer 2, EVPN needs to carry additional information for the attached devices so proxy ND can provide the correct information to attached hosts. Without this information, hosts might not be

able to configure their default routers or might lose their existing default router information. Cumulus Linux supports the EVPN Neighbor Discovery (ND) Extended Community with a type field value of 0x06, a sub-type field value of 0x08 (ND Extended Community), and a router flag; this enables the switch to determine if a particular IPv6-MAC pair belongs to a host or a router.

The **router flag** (R-bit) is used in:

- A centralized VXLAN routing configuration with a gateway router.
- A layer 2 switch deployment with ARP/ND suppression.

When the MAC/IP (type-2) route contains the IPv6-MAC pair and the R-bit is set, the route belongs to a router. If the R-bit is set to zero, the route belongs to a host. If the router is in a local LAN segment, the switch implementing the proxy ND function learns of this information by snooping on neighbor advertisement messages for the associated IPv6 address. This information is then exchanged with other EVPN peers by using the ND extended community in BGP updates.

To show the EVPN arp-cache that gets populated by the neighbor table and see if the IPv6-MAC entry belongs to a router, run either the NCLU `net show evpn arp-cache vni <vni> ip <address>` command or the vtysh `show evpn arp-cache vni <vni> ip <address>` command. For example:

```
cumulus@leaf01:mgmt:~$ net show evpn arp-cache vni 20 ip
10.1.20.105
```

```
IP: 10.1.20.105
 Type: remote
 State: active
 MAC: 12:15:9a:9c:f2:e1
 Sync-info: -
 Remote VTEP: 10.0.1.2
 Local Seq: 0 Remote Seq: 0
```

Examine MAC Moves

The first time a MAC moves from behind one VTEP to behind another, BGP associates a MAC Mobilit (MM) extended community attribute of sequence number 1, with the type-2 route for that MAC. From there, each time this MAC moves to a new VTEP, the MM sequence number increments by 1. You can examine the MM sequence number associated with a MAC's type-2 route with the NCLU `net show bgp l2vpn evpn route vni <vni> mac <mac>` command or the vtysh `show bgp l2vpn evpn route vni <vni> mac <mac>` command. The example output below shows the type-2 route for a MAC that has moved three times:

```
cumulus@switch:~$ net show bgp l2vpn evpn route vni 10109 mac
00:02:22:22:22:02
BGP routing table entry for [2]:[0]:[0]:[48]:[00:02:22:22:22:02]
```

```
Paths: (1 available, best #1)
Not advertised to any peer
Route [2]:[0]:[0]:[48]:[00:02:22:22:22:02] VNI 10109
Local
6.0.0.184 from 0.0.0.0 (6.0.0.184)
Origin IGP, localpref 100, weight 32768, valid, sourced, local,
bestpath-from-AS Local, best
Extended Community: RT:650184:10109 ET:8 MM:3
AddPath ID: RX 0, TX 10350121
Last update: Tue Feb 14 18:40:37 2017

Displayed 1 paths for requested prefix
```

Examine Static MAC Addresses

You can identify static or *sticky* MACs in EVPN by the presence of `MM:0, sticky MAC` in the Extended Community line of the output from the NCLU `net show bgp l2vpn evpn route vni <vni> mac <mac>` command or the vtysh `show bgp l2vpn evpn route vni <vni> mac <mac>` command.

```
cumulus@switch:~$ net show bgp l2vpn evpn route vni 10101 mac
00:02:00:00:00:01
BGP routing table entry for [2]:[0]:[0]:[48]:[00:02:00:00:00:01]
```

```
Paths: (1 available, best #1)
  Not advertised to any peer
  Route [2]:[0]:[0]:[48]:[00:02:00:00:00:01] VNI 10101
  Local
    172.16.130.18 from 0.0.0.0 (172.16.130.18)
      Origin IGP, localpref 100, weight 32768, valid, sourced,
local, bestpath-from-AS Local, best
      Extended Community: ET:8 RT:60176:10101 MM:0, sticky MAC
      AddPath ID: RX 0, TX 46
      Last update: Tue Apr 11 21:44:02 2017

Displayed 1 paths for requested prefix
```

Enable FRRouting Debug Logs

To troubleshoot EVPN, enable FRR debug logs. The relevant debug options are:

Option	Description
`debug zebra vxlan`	Traces VNI addition and deletion (local and remote) as well as MAC and neighbor addition and deletion (local and remote).
`debug zebra kernel`	Traces actual netlink messages exchanged with the kernel,

Option	Description
	which includes everything, not just EVPN.
`debug bgp updates`	Traces BGP update exchanges, including all updates. Output is extended to show EVPN specific information.
`debug bgp zebra`	Traces interactions between BGP and zebra for EVPN (and other) routes.

ICMP echo Replies and the ping Command

When you run the `ping -I` command and specify an interface, you don't get an ICMP echo reply. However, when you run the `ping` command without the `-I` option, everything works as expected.

`ping -I` command example:

```
cumulus@switch:default:~:# ping -I swp2 10.0.10.1
PING 10.0.10.1 (10.0.10.1) from 10.0.0.2 swp1.5: 56(84) bytes
of data.
```

`ping` command example:

```
cumulus@switch:default:~:# ping 10.0.10.1
PING 10.0.10.1 (10.0.10.1) 56(84) bytes of data.
64 bytes from 10.0.10.1: icmp_req=1 ttl=63 time=4.00 ms
64 bytes from 10.0.10.1: icmp_req=2 ttl=63 time=0.000 ms
64 bytes from 10.0.10.1: icmp_req=3 ttl=63 time=0.000 ms
64 bytes from 10.0.10.1: icmp_req=4 ttl=63 time=0.000 ms
^C
--- 10.0.10.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 0.000/1.000/4.001/1.732 ms
```

This is expected behavior with Cumulus Linux; when you send an ICMP echo request to an IP address that is not in the same subnet using the `ping -I` command, Cumulus Linux creates a failed ARP entry for the destination IP address.

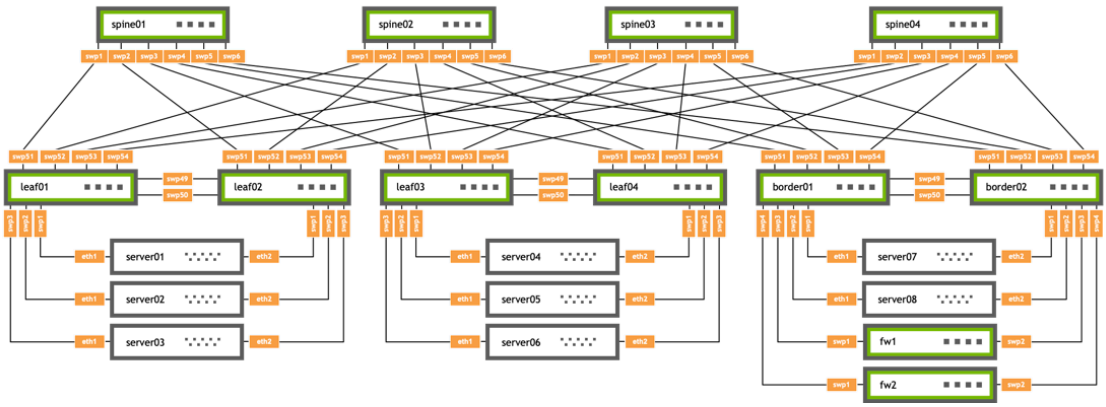
For more information, refer to this article.

Configuration Examples

This section shows the following EVPN configuration examples:

- Layer 2 EVPN with external routing
- EVPN centralized routing
- EVPN symmetric routing

The configuration examples are based on the reference topology below:

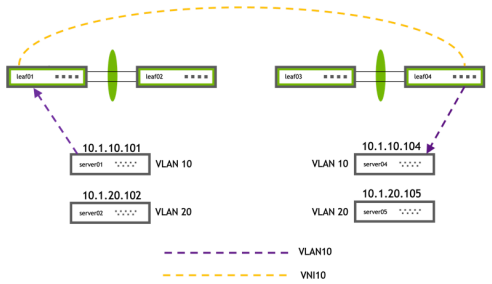


Layer 2 EVPN with External Routing

The following example configures a network infrastructure that creates a layer 2 extension between racks. Inter-VXLAN routed traffic routes between VXLANs on an external device.

- MLAG is configured between leaf01 and leaf02, and leaf03 and leaf04
- BGP unnumbered is in the underlay (configured on all leafs and spines)
- Server gateways are outside the VXLAN fabric

The following images shows traffic flow between tenants. The spines and other devices are omitted for simplicity.

Traffic Flow between server01 and server04	
	server01 and server04 are in the same VLAN but are located across different leafs. 1. server01 makes a LACP hash decision and forwards traffic to leaf01. 2. leaf01 does a layer 2 lookup, has the MAC address for server04, and forwards the packet out VNI10, towards leaf04. 3. The VXLAN encapsulated frame arrives on leaf04, which does a layer 2 lookup and has the MAC address for server04 in VLAN10.

/etc/network/interfaces

leaf01 leaf02 leaf03 leaf04 spine01 spine02
spine03 spine04 border01 border02

```
cumulus@leaf01:~$ cat /etc/network/interfaces

auto lo
iface lo inet loopback
    address 10.10.10.1/32
    clagd-vxlan-anycast-ip 10.0.1.1
    vxlan-local-tunnelip 10.10.10.1

auto mgmt
iface mgmt
    vrf-table auto
    address 127.0.0.1/8
    address ::1/128

auto eth0
iface eth0 inet dhcp
    vrf mgmt

auto bridge
iface bridge
    bridge-ports peerlink bond1 bond2 vni10 vni20
    bridge-vids 10 20
    bridge-vlan-aware yes

auto vni10
iface vni10
    bridge-access 10
    vxlan-id 10
```

/etc/frr/frr.conf

leaf01 leaf02 leaf03 leaf04 spine01 spine02
spine03 spine04 border01 border02

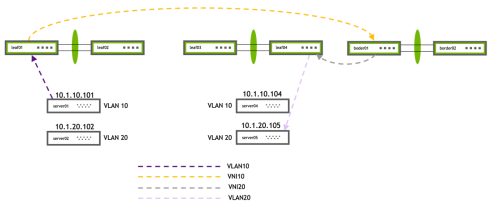
```
cumulus@leaf01:~$ cat /etc/frr/frr.conf
...
log syslog informational
!
router bgp 65101
 bgp router-id 10.10.10.1
 bgp bestpath as-path multipath-relax
 neighbor underlay peer-group
 neighbor underlay remote-as external
 neighbor peerlink.4094 interface remote-as internal
 neighbor swp51 interface peer-group underlay
 neighbor swp52 interface peer-group underlay
 neighbor swp53 interface peer-group underlay
 neighbor swp54 interface peer-group underlay
 !
 address-family ipv4 unicast
  redistribute connected
 exit-address-family
 !
 address-family l2vpn evpn
  neighbor underlay activate
  advertise-all-vni
 exit-address-family
!
line vty
```

EVPN Centralized Routing

The following example shows an EVPN centralized routing configuration:

- MLAG is configured between leaf01 and leaf02, leaf03 and leaf04, and border01 and border02
- BGP unnumbered is in the underlay (configured on all leafs and spines)
- SVIs are configured as gateways on the border leafs

The following images shows traffic flow between tenants. The spines and other devices are omitted for simplicity.

Traffic Flow between server01 and server05	
	server01 and server05 are in a different VLAN and are located across different leafs. 1. server01 makes a LACP hash decision and forwards traffic to leaf01. 2. leaf01 does a layer 2 lookup and forwards traffic to server01's default gateway (border01) out VNI10. 3. border01 does a layer 3 lookup and routes the packet out VNI20 towards leaf04. 4. The VXLAN encapsulated frame arrives on leaf04, which does a layer 2 lookup and has

Traffic Flow between server01 and server05	
	the MAC address for server05 in VLAN20.

/etc/network/interfaces

leaf01 leaf02 leaf03 leaf04 spine01 spine02
spine03 spine04 border01 border02

```
cumulus@leaf01:~$ cat /etc/network/interfaces

auto lo
iface lo inet loopback
    address 10.10.10.1/32
    clagd-vxlan-anycast-ip 10.0.1.1
    vxlan-local-tunnelip 10.10.10.1

auto mgmt
iface mgmt
    vrf-table auto
    address 127.0.0.1/8
    address ::1/128

auto eth0
iface eth0 inet dhcp
    vrf mgmt

auto bridge
iface bridge
    bridge-ports peerlink bond1 bond2 vni10 vni20
    bridge-vids 10 20
    bridge-vlan-aware yes

auto vni10
iface vni10
    bridge-access 10
    vxlan-id 10
```

/etc/frr/frr.conf

leaf01 leaf02 leaf03 leaf04 spine01 spine02
spine03 spine04 border01 border02

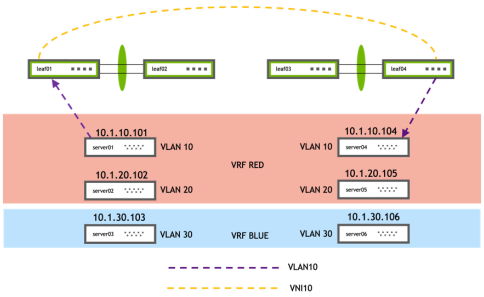
```
cumulus@leaf01:~$ cat /etc/frr/frr.conf
...
log syslog informational
!
router bgp 65101
 bgp router-id 10.10.10.1
 bgp bestpath as-path multipath-relax
 neighbor underlay peer-group
 neighbor underlay remote-as external
 neighbor peerlink.4094 interface remote-as internal
 neighbor swp51 interface peer-group underlay
 neighbor swp52 interface peer-group underlay
 neighbor swp53 interface peer-group underlay
 neighbor swp54 interface peer-group underlay
 !
 address-family ipv4 unicast
  redistribute connected
 exit-address-family
 !
 address-family l2vpn evpn
  neighbor underlay activate
 exit-address-family
!
line vty
```

EVPN Symmetric Routing

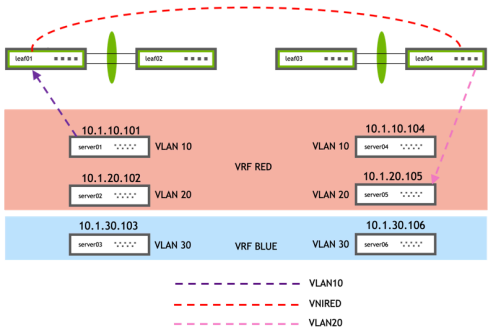
The following example shows an EVPN symmetric routing configuration, where:

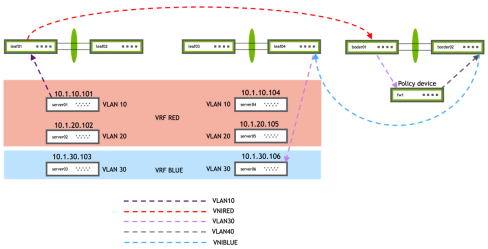
- MLAG is configured between leaf01 and leaf02, leaf03 and leaf04, and border01 and border02
- BGP unnumbered is in the underlay (configured on all leafs and spines)
- VRF BLUE and VRF RED are configured on the leafs for traffic flow between tenants for traffic isolation

The following images shows traffic flow between tenants. The spines and other devices are omitted for simplicity.

Traffic Flow between server01 and server04	
	server01 and server04 are in the same VRF and the same VLAN but are located across different leafs. 1. server01 makes a LACP hash decision and forwards traffic to leaf01. 2. leaf01 does a layer 2 lookup and has the MAC address for server04, it then forwards the packet out VNI10, through leaf04. 3. The VXLAN encapsulated

Traffic Flow between server01 and server04	
	frame arrives on leaf04, which does a layer 2 lookup and has the MAC address for server04 in VLAN10.

Traffic Flow between server01 and server05	
	server01 and server05 are in the same VRF, different VLANs, and are located across different leafs. 1. server01 makes an LACP hash decision to reach the default gateway and forwards traffic to leaf01. 2. leaf01 does a layer 3 lookup in VRF RED and has a route out VNIRED through leaf04. 3. The VXLAN encapsulated packet arrives on leaf04, which does a layer 3 lookup in VRF RED and has a route through VLAN20 to server05.

Traffic Flow between server01 and server06	
	server01 and server06 are in different VRFs, different VLANs, and are located across different leafs. 1. server01 makes an LACP hash decision to reach the default gateway and forwards traffic to leaf01. 2. leaf01 does a layer 3 lookup in VRF RED and has a route out VNIRED through border01. 3. The VXLAN encapsulated packet arrives on border01, which does a layer 3 lookup in VRF RED and has a route through VLAN30 to fw01 (the policy device). 4. fw01 does a layer 3 lookup (without any VRFs) and has a route in VLAN40, through border02. 5. border02 does a layer 3 lookup in VRF BLUE and has a route out VNIBLUE, through leaf04. 6. The VXLAN encapsulated packet arrives on leaf04, which does a layer 3 lookup in VRF BLUE and has a route in VLAN30 to server06.

/etc/network/interfaces

leaf01 leaf02 leaf03 leaf04 spine01 spine02
spine03 spine04 border01 border02

```
cumulus@leaf01:~$ cat /etc/network/interfaces

auto lo
iface lo inet loopback
    address 10.10.10.1/32
    clagd-vxlan-anycast-ip 10.0.1.1
    vxlan-local-tunnelip 10.10.10.1

auto mgmt
iface mgmt
    vrf-table auto
    address 127.0.0.1/8
    address ::1/128

auto eth0
iface eth0 inet dhcp
    vrf mgmt

auto RED
iface RED
  vrf-table auto

auto BLUE
iface BLUE
  vrf-table auto

auto bridge
iface bridge
```

/etc/frr/frr.conf

leaf01 leaf02 leaf03 leaf04 spine01 spine02
spine03 spine04 border01 border02

```
cumulus@leaf01:~$ cat /etc/frr/frr.conf
...
log syslog informational
!
vrf RED
  vni 4001
vrf BLUE
  vni 4002
!
router bgp 65101
 bgp router-id 10.10.10.1
 bgp bestpath as-path multipath-relax
 neighbor underlay peer-group
 neighbor underlay remote-as external
 neighbor peerlink.4094 interface remote-as internal
 neighbor swp51 interface peer-group underlay
 neighbor swp52 interface peer-group underlay
 neighbor swp53 interface peer-group underlay
 neighbor swp54 interface peer-group underlay
 !
 address-family ipv4 unicast
  redistribute connected
 exit-address-family
 !
 address-family l2vpn evpn
  neighbor underlay activate
  advertise-all-vni
 exit-address-family
```

VXLAN Active-active Mode

VXLAN active-active mode enables a pair of MLAG switches to act as a single VTEP, providing active-active VXLAN termination for bare metal as well as virtualized workloads.

Terminology

Term	Definition
VTEP	The virtual tunnel endpoint. This is an encapsulation and decapsulation point for VXLANs.
active-active VTEP	A pair of switches acting as a single VTEP.
ToR	The top of rack switch; also referred to as a leaf or access switch.
spine	The aggregation switch for multiple leafs. Specifically used when a data center is using a Clos network architecture. Read more about spine-leaf architecture in this white paper.
exit leaf	A switch dedicated to peering the Clos network to an outside network; also referred to as a border leaf, service leaf, or edge leaf.

Term	Definition
anycast	An IP address that is advertised from multiple locations. Anycast enables multiple devices to share the same IP address and effectively load balance traffic across them. With VXLAN, anycast is used to share a VTEP IP address between a pair of MLAG switches.
RIOT	Routing in and out of tunnels. A Broadcom feature for routing in and out of tunnels. Allows a VXLAN bridge to have a switch VLAN interface associated with it, and traffic to exit a VXLAN into the layer 3 fabric. Also called VXLAN Routing.
VXLAN routing	The industry standard term for the ability to route in and out of a VXLAN. Equivalent to the Broadcom RIOT feature.
clagd-vxlan-anycast-ip	The anycast address for the MLAG pair to share and bind to when MLAG is up and running.

Configure VXLAN Active-active Mode

VXLAN active-active mode requires the following underlying technologies to work correctly.

Technology	More Information
MLAG	Refer to MLAG for more detailed configuration information. Example configurations are provided below.
OSPF or BGP	Refer to OSPF or BGP for more detailed configuration information. Example configurations for BGP are provided below.
STP	You must enable BPDU filter and BPDU guard in the VXLAN interfaces if STP is enabled in the bridge that is connected to the VXLAN. Example configurations are provided below.

Active-active VTEP Anycast IP Behavior

You must provision each individual switch within an MLAG pair with a virtual IP address in the form of an anycast IP address for VXLAN data-path termination. The VXLAN termination address is an anycast IP address that you configure as a `clagd` parameter (`clagd-vxlan-anycast-ip`) under the loopback interface. `clagd` dynamically adds and removes this address as the loopback interface address as follows:

1. When the switches boot up, `ifupdown2` places all VXLAN interfaces in a PROTO_DOWN state. The configured anycast addresses are not configured yet.

2. MLAG peering takes place and a successful VXLAN interface consistency check between the switches occurs.
3. `clagd` (the daemon responsible for MLAG) adds the anycast address to the loopback interface as a second address. It then changes the local IP address of the VXLAN interface from a unique address to the anycast virtual IP address and puts the interface in an UP state.

> **TIP**
>
> For the anycast address to activate, you must configure a VXLAN interface on each switch in the MLAG pair.

Failure Scenario Behaviors

Scenario	Behavior
The peer link goes down.	The primary MLAG switch continues to keep all VXLAN interfaces up with the anycast IP address while the secondary switch brings down all VXLAN interfaces and places them in a PROTO_DOWN state. The secondary MLAG switch removes the anycast IP address from the loopback interface and changes the local IP address of

Scenario	Behavior
	the VXLAN interface to the configured unique IP address.
One of the switches goes down.	The other operational switch continues to use the anycast IP address.
`clagd` is stopped.	All VXLAN interfaces are put in a PROTO_DOWN state. The anycast IP address is removed from the loopback interface and the local IP addresses of the VXLAN interfaces are changed from the anycast IP address to unique non-virtual IP addresses.
MLAG peering could not be established between the switches.	`clagd` brings up all the VXLAN interfaces after the reload timer expires with the configured anycast IP address. This allows the VXLAN interface to be up and running on both switches even though peering is not established.
When the peer link goes down but the peer switch is up (the backup link is active).	All VXLAN interfaces are put into a PROTO_DOWN state on the secondary switch.
A configuration mismatch between the MLAG switches	The VXLAN interface is placed into a PROTO_DOWN state on the secondary switch.

Check VXLAN Interface Configuration Consistency

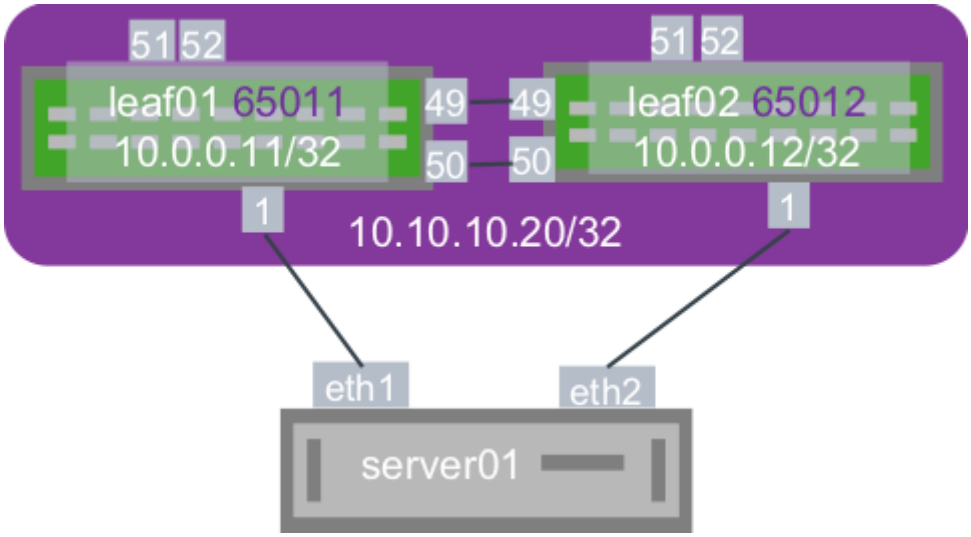
The active-active configuration for a given VXLAN interface must be consistent between the MLAG switches for correct traffic behavior. MLAG ensures that the configuration consistency is met before bringing up the VXLAN interfaces:

- The anycast virtual IP address for VXLAN termination must be the same on each pair of switches.
- A VXLAN interface with the same VXLAN ID must be configured and administratively up on both switches.

Run the `clagctl` command to check if any VXLAN switches are in a PROTO_DOWN state.

Configure the Anycast IP Address

With MLAG peering, both switches use an anycast IP address for VXLAN encapsulation and decapsulation. This enables remote VTEPs to learn the host MAC addresses attached to the MLAG switches against one logical VTEP, even though the switches independently encapsulate and decapsulate layer 2 traffic originating from the host. You can configure the anycast address under the loopback interface, as shown below.

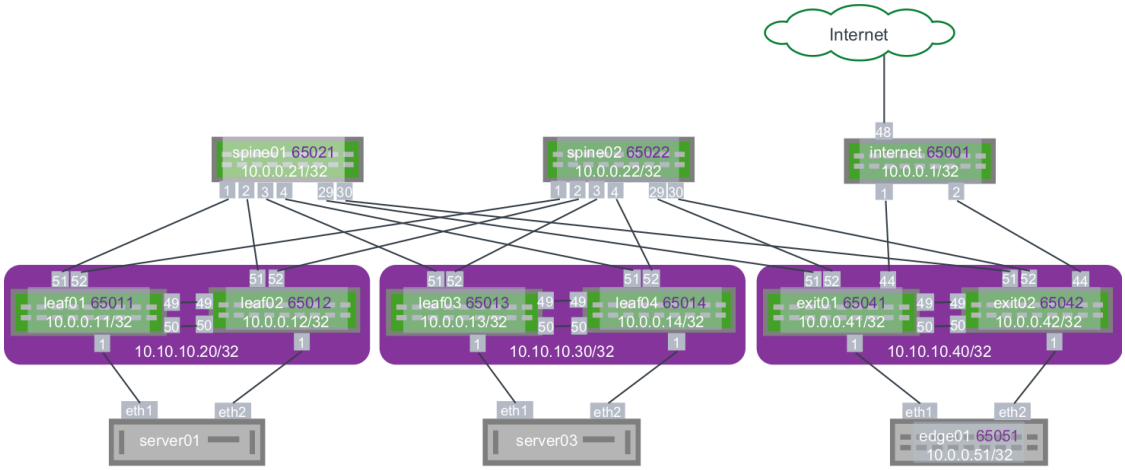


leaf01 /etc/network/interfaces snippet

leaf02 /etc/network/interfaces snippet

```
auto lo
iface lo inet loopback
  address 10.0.0.11/32
  clagd-vxlan-anycast-ip 10.10.10.20
```

Example VXLAN Active-Active Configuration



The VXLAN interfaces are configured with individual IP addresses, which `clagd` changes to anycast upon MLAG peering.

FRRouting Configuration

You can configure the layer 3 fabric using BGP or OSPF. The following example uses BGP unnumbered. The MLAG switch configuration for the topology above is:

Layer 3 IP Addressing

The IP address configuration for this example:

spine01 spine02 leaf01 leaf02 leaf03 leaf04

```
auto lo
iface lo inet loopback
  address 10.0.0.21/32

auto eth0
iface eth0 inet dhcp

# downlinks
auto swp1
iface swp1

auto swp2
iface swp2

auto swp3
iface swp3

auto swp4
iface swp4

auto swp29
iface swp29

auto swp30
iface swp30
```

Host Configuration

In this example, the servers are running Ubuntu 14.04. A layer2 bond must be mapped from server01 and server03 to the respective switch. In Ubuntu, you use subinterfaces.

server01 server03

```
auto lo
iface lo inet loopback


auto lo
iface lo inet static
  address 10.0.0.31/32


auto eth0
iface eth0 inet dhcp


auto eth1
iface eth1 inet manual
    bond-master bond0


auto eth2
iface eth2 inet manual
    bond-master bond0


auto bond0
iface bond0 inet static
  bond-slaves none
  bond-miimon 100
  bond-min-links 1
  bond-mode 802.3ad
  bond-xmit-hash-policy layer3+4
  bond-lacp-rate 1
  address 172.16.1.101/24

auto bond0.10
```

Troubleshooting

Run the `clagctl` command to show MLAG behavior and any inconsistencies that might arise between a MLAG pair.

```
cumulus@leaf01$ clagctl
The peer is alive
      Our Priority, ID, and Role: 32768 44:38:39:00:00:35
primary
    Peer Priority, ID, and Role: 32768 44:38:39:00:00:36
secondary
          Peer Interface and IP: peerlink.4094 169.254.1.2
                VxLAN Anycast IP: 10.10.10.30
                      Backup IP: 10.0.0.14 (inactive)
                      System MAC: 44:38:39:ff:40:95
CLAG Interfaces
Our Interface      Peer Interface     CLAG Id   Conflicts
Proto-Down Reason
----------------   ----------------   -------   -----------
-----------------
           bond0   bond0              1         -             -
         vxlan20   vxlan20            -         -             -
          vxlan1   vxlan1             -         -             -
         vxlan10   vxlan10            -         -             -
```

The additions to normal MLAG behavior are:

Output	Explanation
`VXLAN Anycast IP: 10.10.10.30`	The anycast IP address being shared by the MLAG pair for VTEP termination is in use and is 10.10.10.30.
`Conflicts: -`	There are no conflicts for this MLAG Interface.
`Proto-Down Reason: -`	The VXLAN is up and running (there is no Proto-Down).

In the following example the `vxlan-id` on VXLAN10 is switched to the wrong `vxlan-id`. When you run the `clagctl` command, VXLAN10 is down because this switch is the secondary switch and the peer switch takes control of VXLAN. The reason code is `vxlan-single` indicating that there is a `vxlan-id` mis-match on VXLAN10.

```
cumulus@leaf02$ clagctl
The peer is alive
    Peer Priority, ID, and Role: 32768 44:38:39:00:00:11 primary
      Our Priority, ID, and Role: 32768 44:38:39:00:00:12
secondary
          Peer Interface and IP: peerlink.4094 169.254.1.1
                VxLAN Anycast IP: 10.10.10.20
```

```
                    Backup IP: 10.0.0.11 (inactive)
                    System MAC: 44:38:39:ff:40:94

CLAG Interfaces
Our Interface      Peer Interface     CLAG Id   Conflicts
Proto-Down Reason
---------------    ---------------    -------   ------------
-----------------
           bond0   bond0              1         -              -
         vxlan20   vxlan20            -         -              -
          vxlan1   vxlan1             -         -              -
         vxlan10   -                  -         -
vxlan-single
```

Considerations

Use VLAN for Peer Link Only Once

Do not reuse the VLAN for the peer link layer 3 subinterface for any other interface in the system. A high VLAN ID value is recommended. For more information on VLAN ID ranges, refer to the VLAN-aware bridge chapter.

Bonds with Vagrant in Cumulus VX

Bonds (or LACP Etherchannels) fail to work in a Vagrant configuration unless the link is set to *promiscuous* mode. This is a limitation on virtual topologies only and is not needed on real hardware.

```
auto swp49
iface swp49
  #for vagrant so bonds work correctly
  post-up ip link set $IFACE promisc on

auto swp50
iface swp50
  #for vagrant so bonds work correctly
  post-up ip link set $IFACE promisc on
```

For more information on using Cumulus VX and Vagrant, refer to the Cumulus VX documentation.

VXLAN Routing

VXLAN routing, sometimes referred to as *inter-VXLAN routing*, provides IP routing between VXLAN VNIs in overlay networks. The routing of traffic is based on the inner header or the overlay tenant IP address.

Because VXLAN routing is fundamentally routing, it is most commonly deployed with a control plane, such as Ethernet Virtual Private Network (EVPN). You can also set up static routing for MAC distribution and BUM handling.

This topic describes the platform and hardware considerations for VXLAN routing. For a detailed description of different VXLAN routing models and configuration examples, refer to EVPN.

VXLAN routing supports full layer 3 multi-tenancy; all routing occurs in the context of a VRF. Also, VXLAN routing is supported for dual-attached hosts where the associated VTEPs function in active-active mode.

Supported Platforms

The following ASICs support VXLAN routing:

- Broadcom Trident II+, Trident3, and Maverick
- Broadcom Tomahawk and Tomahawk+, using an internal loopback on one or more switch ports
- Mellanox Spectrum

NOTE

- Using ECMP with VXLAN routing is supported only on RIOT-capable Broadcom ASICs (Trident 3, Maverick, Trident 2+) in addition to Tomahawk, Tomahawk+ and Mellanox Spectrum-A1 ASICs.
- For additional restrictions and considerations for VXLAN routing with EVPN, refer to Ethernet Virtual Private Network - EVPN.

VXLAN Routing Data Plane and Broadcom Switches

Trident II+, Trident3, and Maverick

The Trident II+, Trident3, and Maverick ASICs provide native support for VXLAN routing, also referred to as Routing In and Out of Tunnels (RIOT).

You can specify a VXLAN routing profile in the `vxlan_routing_overlay.profile` field of the `/usr/lib/python2.7/dist-packages/cumulus/__chip_config/bcm/datapath.conf` file to control the maximum number of overlay next hops (adjacency entries). The profile is one of the following:

- *default*: 15% of the underlay next hops are set apart for overlay (8k next hops are reserved)
- *mode-1*: 25% of the underlay next hops are set apart for overlay
- *mode-2*: 50% of the underlay next hops are set apart for overlay
- *mode-3*: 80% of the underlay next hops are set apart for overlay
- *disable*: disables VXLAN routing

The following shows an example of the *VXLAN Routing Profile* section of the `datapath.conf` file where the *default* profile is enabled.

```
...
# Specify a VxLan Routing Profile - the profile selected
determines the
# maximum number of overlay next hops that can be allocated.
# This is supported only on TridentTwoPlus and Maverick
#
# Profile can be one of {'default', 'mode-1', 'mode-2',
'mode-3', 'disable'}
# default: 15% of the overall nexthops are for overlay.
# mode-1:  25% of the overall nexthops are for overlay.
# mode-2:  50% of the overall nexthops are for overlay.
# mode-3:  80% of the overall nexthops are for overlay.
# disable: VxLan Routing is disabled
#
# By default VxLan Routing is enabled with the default profile.
```

```
vxlan_routing_overlay.profile = default
```

The Trident II+ and Trident3 ASICs support a maximum of 48k underlay next hops.

For any profile you specify, you can allocate a *maximum* of 2K (2048) VXLAN SVI interfaces.

To disable VXLAN routing on a Trident II+ or Trident3 switch, set the `vxlan_routing_overlay.profile` field to *disable*.

Trident II

VXLAN routing is not supported on Trident II switches, and the external hyperloop workaround for RIOT on Trident II switches has been removed in Cumulus Linux 4.0.0. Use native VXLAN routing platforms and EVPN for network virtualization.

Tomahawk and Tomahawk+

The Tomahawk and Tomahawk+ ASICs do not support RIOT natively; you must configure the switch ports for VXLAN routing to use internal loopback (also referred to as *internal hyperloop)*. The internal loopback facilitates the recirculation of packets through the ingress pipeline to achieve VXLAN routing.

For routing **into** a VXLAN tunnel, the first pass of the ASIC performs

routing and routing rewrites of the packet MAC source, destination address, and VLAN, then packets recirculate through the internal hyperloop for VXLAN encapsulation and underlay forwarding on the second pass.

For routing **out of** a VXLAN tunnel, the first pass performs VXLAN decapsulation, then packets recirculate through the hyperloop for routing on the second pass.

You only need to configure the switch ports that must be in internal loopback mode based on the amount of bandwidth required. No additional configuration is necessary.

To configure one or more switch ports for loopback mode, edit the `/etc/cumulus/ports.conf` file and change the port speed to *loopback*. In the example below, swp8 and swp9 are configured for loopback mode:

```
cumulus@switch:~$ sudo nano /etc/cumulus/ports.conf
...
7=4x10G
8=loopback
9=loopback
10=100G
...
```

Restart `switchd` for the changes to take effect.

> **NOTE**
>
> VXLAN routing with internal loopback is supported only with VLAN-aware bridges; you cannot use a bridge in traditional mode.

Tomahawk+ and 25G Ports for Loopback

For VXLAN routing on a switch with the Tomahawk+ ASIC, if you use 25G ports as the internal loopback, you must configure all four ports in the same port group.

VXLAN Routing Data Plane and Mellanox Spectrum ASICs

There is no special configuration required for VXLAN routing on Mellanox Spectrum ASICs.

Bridge Layer 2 Protocol Tunneling

A VXLAN connects layer 2 domains across a layer 3 fabric; however, layer 2 protocol packets, such as LLDP, LACP, STP, and CDP are normally terminated at the ingress VTEP. If you want the VXLAN to behave more like a wire or hub, where protocol packets are tunneled instead of being terminated locally, you can enable *bridge layer 2 protocol tunneling*.

Configure Bridge Layer 2 Protocol Tunneling

To configure bridge layer 2 protocol tunneling for all protocols:

```
cumulus@switch:~$ net add interface swp1 bridge l2protocol-
tunnel all
cumulus@switch:~$ net add interface vni13 bridge l2protocol-
tunnel all
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

To configure bridge layer 2 protocol tunneling for a **specific** protocol, such as LACP:

```
cumulus@switch:~$ net add interface swp1 bridge l2protocol-
tunnel lacp
cumulus@switch:~$ net add interface vni13 bridge l2protocol-
tunnel lacp
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

NOTE

You must enable layer 2 protocol tunneling on the VXLAN link also so that the packets get bridged and correctly forwarded.

The above commands create the following configuration in the `/etc/network/interfaces` file:

```
auto swp1
iface swp1
    bridge-access 10
    bridge-l2protocol-tunnel lacp

auto swp2
```

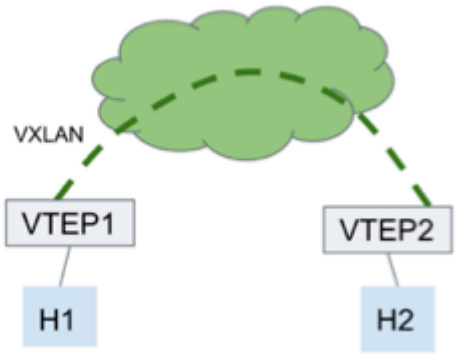
```
iface swp2


auto swp3
iface swp3


auto swp4
iface swp4


...


interface vni13
    bridge-access 13
    bridge-l2protocol-tunnel all
    bridge-learning off
    mstpctl-bpduguard yes
    mstpctl-portbpdufilter yes
    vxlan-id 13
    vxlan-local-tunnelip 10.0.0.4
```

LLDP Example

Here is another example configuration for Link Layer Discovery Protocol.

You can verify the configuration with `lldpcli`.

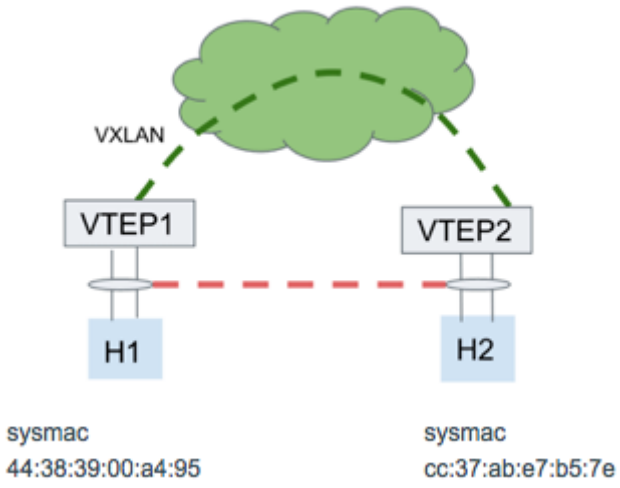
```
cumulus@switch:~$ sudo lldpcli show neighbors
-------------------------------------------------------------------------------
LLDP neighbors:
-------------------------------------------------------------------------------
Interface: swp23, via LLDP, RID: 13, TIme: 0 day, 00:58:20
  Chassis:
    ChassisID: mac e4:1d:2d:f7:d5:52
    SysName: H1
    MgmtIP: 10.0.2.207
    MgmtIP: fe80::e61d:2dff:fef7:d552
    Capability: Bridge, off
    Capability: Router, on
  Port:
    PortID: ifname swp14
    PortDesc: swp14
    TTL: 120
    PMD autoneg: support: yes, enabled: yes
      Adv: 1000Base-T, HD: no, FD: yes
      MAU oper type: 40GbaseCR4 - 40GBASE-R PCS/PMA over 4 lane
shielded copper balanced cable
...
```



LACP Example

```
H2 bond0:
Bonding Mode: IEEE 802.3ad Dynamic link aggregation
Transmit Hash Policy: layer 3+4(1)

802.3ad: info
LACP rate: fast
Min links: 1
Aggregator selection policy (ad_select): stable
System priority: 65535
System MAC address: cc:37:ab:e7:b5:7e
Active Aggregator Info:
    Aggregator ID: 1
    Number of ports: 2

Slave Interface: eth0
```

```
...
details partner lacp pdu:
    system priority: 65535
    system MAC address: 44:38:39:00:a4:95
...
Slave Interface: eth1
...
details partner lacp pdu:
    system priority: 65535
    system MAC address: 44:38:39:00:a4:95
```



Pseudo-wire Example

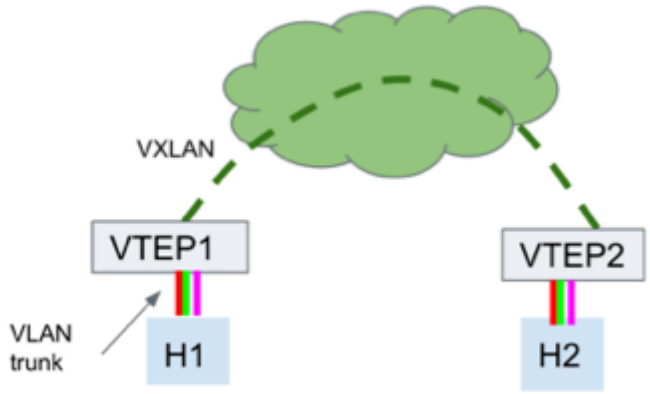
In this example, there are only 2 VTEPs in the VXLAN. VTEP1 and VTEP2 point to each other as the only remote VTEP.

The bridge on each VTEP is configured in 802.1ad mode.

The host interface is an 802.1Q VLAN trunk.

The `bridge-l2protocol-tunnel` is set to `all`.

The VTEP host-facing port is in access mode, and the PVID is mapped to the VNI.



Considerations

Use caution when enabling bridge layer 2 protocol tunneling. Keep the following issues in mind:

- Layer 2 protocol tunneling is not a full-featured pseudo-wire solution; there is no end-to-end link status tracking or feedback.
- Layer 2 protocols typically run on a link local scope. Running the protocols through a tunnel across a layer 3 fabric incurs significantly higher latency, which might require you to tune protocol timers.
- The lack of end to end link/tunnel status feedback and the higher

protocol timeout values make for a higher protocol convergence time in case of change.

- If the remote endpoint is a Cisco endpoint using LACP, you must configure `etherchannel misconfig guard` on the Cisco device.

Static VXLAN Tunnels

In VXLAN-based networks, there are a range of complexities and challenges in determining the destination *virtual tunnel endpoints* (VTEPs) for any given VXLAN. At scale, various solutions, including controller-based options like Midokura MidoNet or VMware NSX and even new standards like EVPN try to address these complexities, however, they also have their own complexities.

Static VXLAN tunnels serve to connect two VTEPs in a given environment. Static VXLAN tunnels are the simplest deployment mechanism for small scale environments and are interoperable with other vendors that adhere to VXLAN standards. Because you simply map which VTEPs are in a particular VNI, you can avoid the tedious process of defining connections to every VLAN on every other VTEP on every other rack.

Requirements

Cumulus Linux supports static VXLAN tunnels only on switches in the Cumulus Linux HCL that use the Mellanox Spectrum ASICs or the Broadcom Tomahawk, Trident II+, Trident II, Trident3, and Maverick ASICs.

For a basic VXLAN configuration, make sure that:

- The VXLAN has a network identifier (VNI). Do not use VNI ID 0 or 16777215; these are reserved values under Cumulus Linux.
- Bridge learning must be enabled on the VNI (bridge learning is disabled

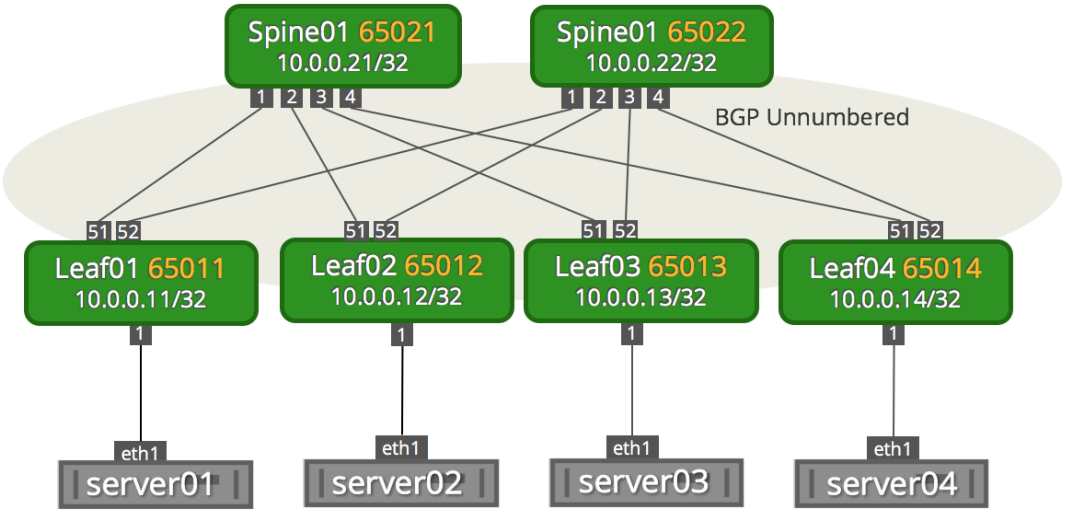
by default).

- The VXLAN link and local interfaces are added to the bridge to create the association between the port, VLAN and VXLAN instance.
- Each traditional mode bridge on the switch has only one VXLAN interface. Cumulus Linux does not support more than one VXLAN ID per traditional bridge.

> **NOTE**
>
> This limitation only affects a traditional mode bridge configuration. Cumulus Linux supports *more* than one VXLAN ID in VLAN-aware bridge mode.

Example Topology

The following topology is used in this chapter. Each IP address corresponds to the loopback address of the switch.



Configure Static VXLAN Tunnels

To configure static VXLAN tunnels, do the following on each leaf:

- Specify an IP address for the loopback.
- Create a VXLAN interface using the loopback address for the local tunnel IP address.
- Enable bridge learning on the VNI.
- Create the tunnels by configuring the remote IP address to each other leaf switch's loopback address.

For example, to configure static VXLAN tunnels on the four leafs in the topology shown above:

NCLU Commands | Linux Commands

Run the following commands on **leaf01**:

```
cumulus@leaf01:~$ net add loopback lo ip address 10.0.0.11/
32
cumulus@leaf01:~$ net add vxlan vni-10 vxlan id 10
cumulus@leaf01:~$ net add vxlan vni-10 bridge learning on
cumulus@leaf01:~$ net add vxlan vni-10 vxlan local-tunnelip
10.0.0.11
cumulus@leaf01:~$ net add vxlan vni-10 vxlan remoteip
10.0.0.12
cumulus@leaf01:~$ net add vxlan vni-10 vxlan remoteip
10.0.0.13
cumulus@leaf01:~$ net add vxlan vni-10 vxlan remoteip
10.0.0.14
cumulus@leaf01:~$ net add vxlan vni-10 bridge access 10
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

Run these commands on leaf02, leaf03, and leaf04:

leaf02

```
cumulus@leaf02:~$ net add loopback lo ip address 10.0.0.12/
32
cumulus@leaf02:~$ net add vxlan vni-10 vxlan id 10
cumulus@leaf02:~$ net add vxlan vni-10 bridge learning on
cumulus@leaf02:~$ net add vxlan vni-10 vxlan local-tunnelip
10.0.0.12
```

Verify the Configuration

After you configure all the leaf switches, run the following command to check for replication entries:

```
cumulus@leaf01:~$ sudo bridge fdb show | grep 00:00:00:00:00:00
00:00:00:00:00:00 dev vni-10 dst 10.0.0.14 self permanent
00:00:00:00:00:00 dev vni-10 dst 10.0.0.12 self permanent
00:00:00:00:00:00 dev vni-10 dst 10.0.0.13 self permanent
```

NOTE

In Cumulus Linux, bridge learning is disabled and ARP suppression is enabled by default. You can change the default behavior to set bridge learning on and ARP suppression off for all VNIs by creating a policy file called `bridge.json` in the `/etc/network/ifupdown2/policy.d/` directory. For example:

```
cumulus@leaf01:~$ sudo cat /etc/network/ifupdown2/
policy.d/bridge.json
{
    "bridge": {
```

```
        "module_globals": {
            "bridge_vxlan_port_learning" : "on",
            "bridge-vxlan-arp-nd-suppress" : "off"
        }
    }
}
```

After you create the file, run `ifreload -a` to load the new configuration.

Control Link-local Multicast across a Static VXLAN Tunnel

By default, when you configure static VXLAN tunnels, Cumulus Linux forwards link-local multicast packets to the CPU and floods the ASIC. Cumulus Linux 3.7.12 and later provides a configuration option on Broadcom switches to disable forwarding of link-local multicast packets to the CPU so that such packets only flood the ASIC, which reduces CPU usage.

To disable forwarding of link local multicast packets to the CPU on a Broadcom switch, run the following command:

```
cumulus@switch:~$ echo TRUE > /cumulus/switchd/config/hal/bcm/
ll_mcast_punt_disable
```

The configuration above takes effect immediately, but does not persist if you reboot the switch.

To apply the configuration so that it is persistent, edit the `/etc/cumulus/switchd.conf` file and uncomment the `hal.bcm.ll_mcast_punt_disable = TRUE` option. For example:

```
cumulus@switch:~$ sudo nano /etc/cumulus/switchd.conf
#
# /etc/cumulus/switchd.conf - switchd configuration file
#
...
# Bridge L2MC Link-Local multicast punt disable
hal.bcm.ll_mcast_punt_disable = TRUE
...
```

A `switchd` restart is **not** required.

VXLAN Scale

On Broadcom Trident II and Tomahawk switches running Cumulus Linux, there is a limit to the number of VXLANs you can configure simultaneously. The limit most often given is 2000 VXLANs, but you might want to get more specific and know exactly the limit for your specific design.

> **NOTE**
>
> While this limitation does apply to Trident II+, Trident3, or Maverick ASICs, Cumulus Linux supports the same number of VXLANs on these ASICs as it does for Trident II or Tomahawk ASICs.
> Mellanox Spectrum ASICs do not have a limitation on the number of VXLANs that they can support.

The limit is a physical to virtual mapping where a switch can hold 15000 mappings in hardware before you encounter hash collisions. There is also an upper limit of around 3000 VLANs you can configure before you hit the reserved range (Cumulus Linux uses 3600-3999 by default). Cumulus Linux typically uses a soft number because the math is unique to each environment. An internal VLAN is consumed by each layer 3 port, subinterface, traditional bridge, and the VLAN-aware bridge. Therefore, the number of configurable VLANs is:

(total configurable 802.1q VLANs) - (reserved VLANS) - (physical or logical interfaces) =

4094-999-eth0-loopback = **3093** by default (without any other configuration)

The equation for the number of configurable VXLANs looks like this:

(number of trunks) * (VXLAN/VLANs per trunk) = 15000 - (Linux logical and physical interfaces)

For example, on a 10Gb switch with 48 * 10 G ports and 6 * 40G uplinks, you can calculate for X, the amount of configurable VXLANs:

48 * X = 15000 - (48 downlinks + 6 uplinks + 1 loopback + 1 eth0 + 1 bridge)

48 * X = 14943

X = **311** VXLANs

Similarly, you can apply this logic to a 32 port 100G switch where 16 ports are broken up to 4 * 25 Gbps ports, for a total of 64 * 25 Gbps ports:

64 * X = 15000 - (64 downlinks + 16 uplinks + 1 loopback + 1 eth0 + 1 bridge)

64 * X = 14917

X = **233** VXLANs

However, not all ports are trunks for all VXLANs (or at least not all the time). It is much more common for subsets of ports to be used for different VXLANs. For example, a 10G (48 * 10G + 6 * 40G uplinks) can have the following configuration:

Ports	Trunks
swp1-20	100 VXLAN/VLANs
swp21-30	100 VXLAN/VLANs
swp31-48	X VXLAN/VLANs

The equation now looks like this:

20 swps * 100 VXLANs + 10 swps * 100 VXLANs + 18 swps * X VXLANs + (48 downlinks + 6 uplinks + loopback + 1 eth0 + 1 bridge) = 15000

20 swps * 100 VXLANs + 10 swps * 100 VXLANs + 18 swps * X VXLANs = 14943

18 * X = 11943

663 = VXLANS (still configurable) for a total of **863**

VXLAN Tunnel DSCP Operations

Cumulus Linux provides configuration options to control DSCP operations during VXLAN encapsulation and decapsulation, specifically for solutions that require end-to-end quality of service, such as RDMA over Converged Ethernet.

The configuration options propagate explicit congestion notification (ECN) between the underlay and overlay and are based on RFC 6040, which describes how to construct the IP header of an ECN field on both ingress to and egress from an IP-in-IP tunnel.

> **NOTE**
>
> VXLAN Tunnel DSCP operations are supported on Mellanox Spectrum switches only.

Configure DSCP Operations

You can set the following DSCP operations by editing the `/etc/cumulus/switchd.conf` file.

Option	Description
`vxlan.def_encap_dscp_action`	Sets the VXLAN outer DSCP

Option	Description
	action during encapsulation. You can specify one of the following options: - `copy` (if the inner packet is IP) - `set` (to a specific value) - `derive` (from the switch priority). The default setting is `derive`.
`vxlan.def_encap_dscp_value`	If the `vxlan.def_encap_dscp_action` option is `set`, you must specify a value.
`vxlan.def_decap_dscp_action`	Sets the VXLAN decapsulation DSCP/COS action. You can specify one of the following options: - `copy` (if the inner packet is IP) - `preserve` (the inner DSCP is unchanged) - `derive` (from the switch priority)

After you modify `/etc/cumulus/switchd.conf` file, you must restart `switchd` for the changes to take effect.

```
cumulus@switch:~$ sudo systemctl restart switchd.service
```

WARNING

Restarting the `switchd` service causes all network ports to reset, interrupting network services, in addition to resetting the switch hardware configuration.

The following example shows that the VXLAN outer DSCP action during encapsulation is `set` with a value of 16.

```
cumulus@switch:~$ sudo nano /etc/cumulus/switchd.conf
...
# default vxlan outer dscp action during encap
# {copy | set | derive}
# copy: only if inner packet is IP
# set: to specific value
# derive: from switch priority
vxlan.def_encap_dscp_action = set

# default vxlan encap dscp value, only applicable if action is
'set'
vxlan.def_encap_dscp_value = 16
```

```
# default vxlan decap dscp/cos action
# {copy | preserve | derive}
# copy: only if inner packet is IP
# preserve: inner dscp unchanged
# derive: from switch priority
#vxlan.def_decap_dscp_action = derive
...
```

You can also set the DSCP operations from the command line. Use the `echo` command to change the settings in the `/etc/cumulus/switchd.conf` file. For example, to change the encapsulation action to `copy`:

```
cumulus@switch:~$ echo "copy" > /cumulus/switchd/config/vxlan/
def_encap_dscp_action
```

To change the VXLAN outer DSCP action during encapsulation to `set` with a value of 32:

```
cumulus@switch:~$ echo "32" > /cumulus/switchd/config/vxlan/
def_encap_dscp_value
```

```
cumulus@switch:~$ echo "set" > /cumulus/switchd/config/vxlan/
def_encap_dscp_action
```

Considerations

Cumulus Linux supports only the default global settings. Per-VXLAN and per-tunnel granularity are not supported.

Hybrid Cloud Connectivity with QinQ and VXLANs

QinQ is an amendment to the IEEE 802.1Q specification that provides the capability for multiple VLAN tags to be inserted into a single Ethernet frame.

QinQ with VXLAN is typically used by a service provider who offers multi-tenant layer 2 connectivity between different customer data centers (private clouds) and also needs to connect those data centers to public cloud providers. Public clouds often has a mandatory QinQ handoff interface, where the outer tag is for the customer and the inner tag is for the service.

In Cumulus Linux, you map QinQ packets to VXLANs through:

- *Single tag translation*, where you map a customer to a VNI and preserve the service as an inner VLAN inside a VXLAN packet.
- *Double tag translation*, where you map a customer and service to a VNI.

QinQ is available on switches with the following ASICs:

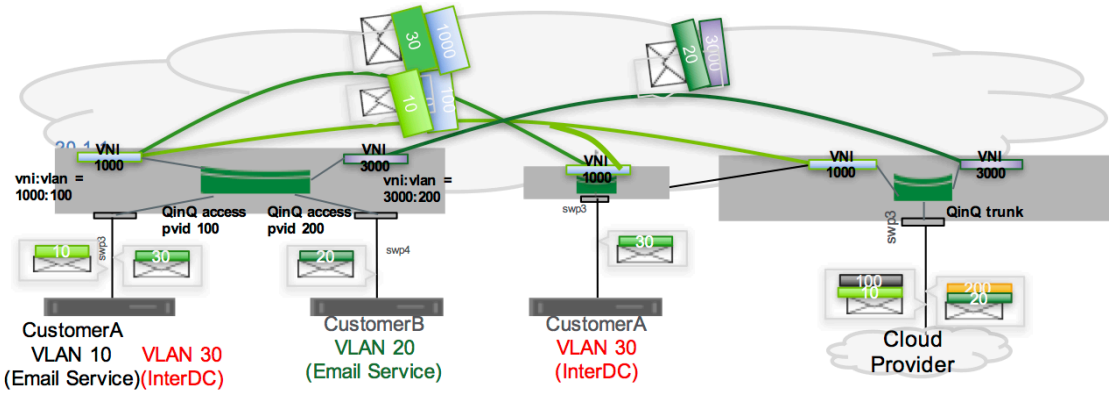
- Broadcom Tomahawk 2, Tomahawk+, Tomahawk, Trident3, Trident II+ and Trident II.
- Mellanox Spectrum 1, Spectrum 2 and Spectrum 3, only with VLAN-aware bridges with 802.1ad and only with single tag translation.

Configure Single Tag Translation

Single tag translation adheres to the traditional QinQ service model. The customer-facing interface is a QinQ access port with the outer S-tag being the PVID, representing the customer. The S-tag is translated to a VXLAN VNI. The inner C-tag, which represents the service, is transparent to the provider. The public cloud handoff interface is a QinQ trunk where packets on the wire carry both the S-tag and the C-tag.

Single tag translation works with both VLAN-aware bridge mode and traditional bridge mode. However, single tag translation with *VLAN-aware bridge mode* is more scalable.

An example configuration in VLAN-aware bridge mode looks like this:



You configure two switches: one at the service provider edge that faces the customer (the switch on the left above), and one on the public cloud handoff edge (the switch on the right above).

> **NOTE**
>
> To correctly interoperate, all edges must support QinQ with VXLANs.

Configure the Public Cloud-facing Switch

For the switch facing the public cloud:

- Configure the bridge with `vlan_protocol` set to *802.1ad*.
- The VNI maps back to S-tag (customer).
- A trunk port connected to the public cloud is the QinQ trunk and packets are double tagged, where the S-tag is for the customer and the C-tag is for the service.

To configure the public cloud-facing switch:

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add vxlan vni-1000 vxlan id 1000
cumulus@switch:~$ net add vxlan vni-1000 vxlan local-
tunnelip 10.0.0.1
cumulus@switch:~$ net add vxlan vni-1000 bridge access 100
cumulus@switch:~$ net add vxlan vni-3000 vxlan id 3000
cumulus@switch:~$ net add vxlan vni-3000 vxlan local-
tunnelip 10.0.0.1
cumulus@switch:~$ net add vxlan vni-3000 bridge access 200
cumulus@switch:~$ net add bridge bridge vlan-protocol
802.1ad
cumulus@switch:~$ net add bridge bridge ports
swp3,vni-1000,vni-3000
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Configure the Customer-facing Edge Switch

For the switch facing the customer:

- Configure the bridge with `vlan_protocol` set to *802.1ad*.
- The customer interface is the QinQ access port, the PVID is the S-tag (customer) and is mapped to a VNI.
- The service VLAN tags (C-tags) are preserved during VXLAN encapsulation.

To configure the customer-facing switch:

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add interface swp3 bridge access 100
cumulus@switch:~$ net add interface swp4 bridge access 200
cumulus@switch:~$ net add vxlan vni-1000 vxlan id 1000
cumulus@switch:~$ net add vxlan vni-1000 vxlan local-
tunnelip 10.0.0.1
cumulus@switch:~$ net add vxlan vni-1000 bridge access 100
cumulus@switch:~$ net add vxlan vni-3000 vxlan id 3000
cumulus@switch:~$ net add vxlan vni-3000 vxlan local-
tunnelip 10.0.0.1
cumulus@switch:~$ net add vxlan vni-3000 bridge access 200
cumulus@switch:~$ net add bridge bridge ports
swp3,swp4,vni-1000,vni-3000
cumulus@switch:~$ net add bridge bridge vlan-protocol
802.1ad
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Example Configuration in Traditional Bridge Mode

An example configuration for single tag translation in traditional bridge mode on a leaf switch is shown below.

▼ Example /etc/network/interfaces File

Configure Double Tag Translation

Double tag translation involves a bridge with double-tagged member interfaces, where a combination of the C-tag and S-tag map to a VNI. You create the configuration only at the edge facing the public cloud. The VXLAN configuration at the customer-facing edge doesn't need to change.

The double tag is always a cloud connection. The customer-facing edge is either single-tagged or untagged. At the public cloud handoff point, the VNI maps to double VLAN tags, with the S-tag indicating the customer and the C-tag indicating the service.

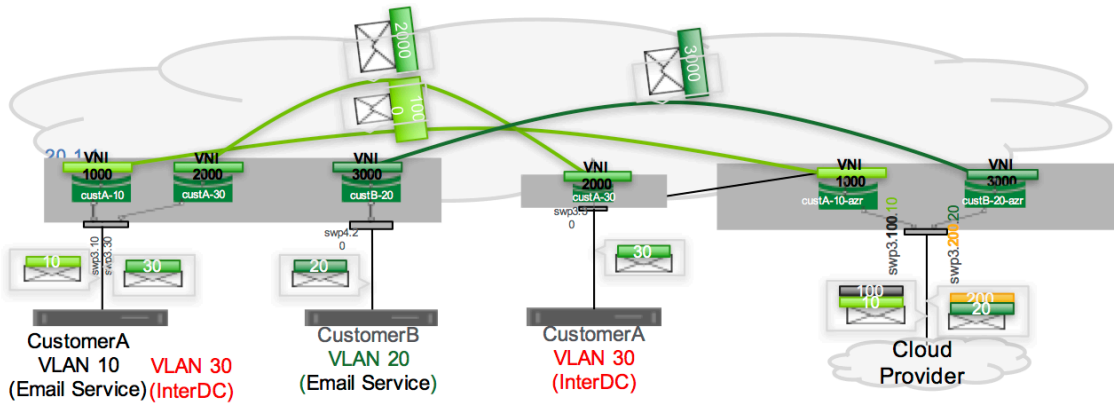
> **NOTE**
>
> The configuration in Cumulus Linux uses the outer tag for the customer and the inner tag for the service.

You configure a double-tagged interface by stacking the VLANs in the following manner: `<port>.<outer tag>.<inner tag>`. For example, consider swp1.100.10: the outer tag is VLAN 100, which represents the customer, and the inner tag is VLAN 10, which represents the service.

The outer tag or *TPID* (tagged protocol identifier) needs the `vlan_protocol` to be specified. It can be either *802.1Q* or *802.1ad*. If 802.1ad is used, it must be specified on the lower VLAN device, such as swp3.100 in the example below.

> **NOTE**
>
> Double tag translation only works with bridges in traditional mode (not VLAN-aware mode).

An example configuration:



To configure the switch for double tag translation using the above example, edit the `/etc/network/interfaces` file in a text editor and add the following:

```
auto swp3.100
iface swp3.100
    vlan_protocol 802.1ad

auto swp3.100.10
```

```
iface swp3.100.10
    mstpctl-portbpdufilter yes
    mstpctl-bpduguard yes

auto vni1000
iface vni1000
    vxlan-local-tunnelip  10.0.0.1
    mstpctl-portbpdufilter yes
    mstpctl-bpduguard yes
    vxlan-id 1000

auto custA-10-azr
iface custA-10-azr
    bridge-ports swp3.100.10 vni1000
    bridge-vlan-aware no
```

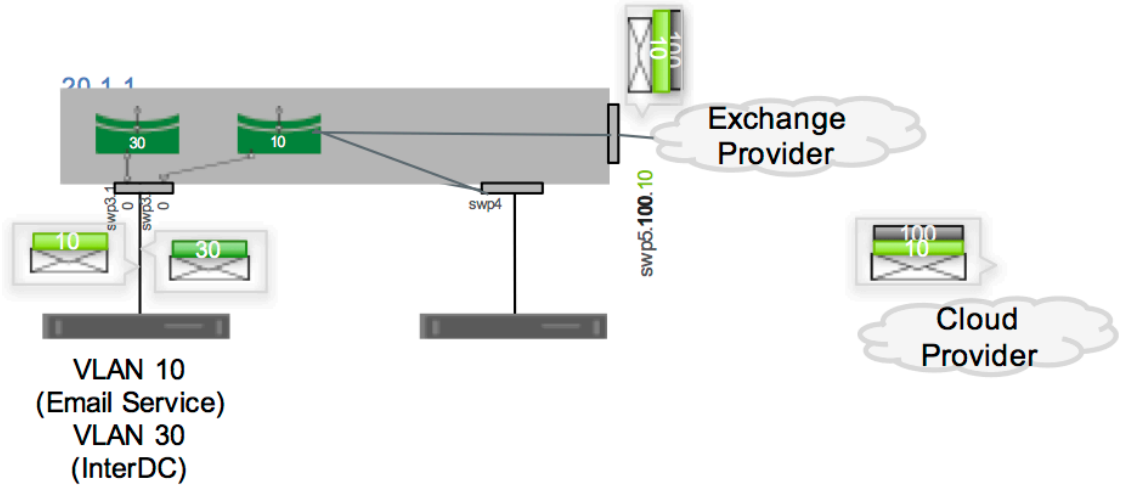
To check the configuration, run the `brctl show` command:

```
cumulus@switch:~$ sudo brctl show
bridge name     bridge id               STP enabled
interfaces
custA-10-azr    8000.00020000004b       yes
swp3.100.10
```

```
                                                          vni1000
custB-20-azr    8000.00020000004b       yes
swp3.200.20
                                                          vni3000
```

If the bridge is *not* VXLAN-enabled, the configuration looks like this:

```
auto swp5.100
iface swp5.100
    vlan-protocol 802.1ad

auto swp5.100.10
iface swp5.100.10
    mstpctl-portbpdufilter yes
    mstpctl-bpduguard yes

auto br10
iface br10
    bridge-ports swp3.10  swp4  swp5.100.10
    bridge-vlan-aware no
```



Considerations

Feature Limitations

- `iptables` match on double-tagged interfaces is not supported.
- MLAG is only supported with single-tagged translation.
- Mixing 802.1Q and 802.1ad subinterfaces on the same switch port is not supported.
- When configuring bridges in traditional mode, all VLANs that are members of the same switch port must use the same `vlan_protocol`.
- When using switches with Mellanox Spectrum ASICs in an MLAG pair:
  - Configure the peerlink (peerlink.4094) between the MLAG pair for VLAN protocol 802.1ad.
  - You cannot use the peerlink as a backup datapath in case one of the MLAG peers loses all uplinks.
- For switches with any type of Spectrum ASIC, when the bridge VLAN protocol is 802.1ad and is VXLAN-enabled, either:

- All bridge ports are access ports, except for the MLAG peerlink.
- All bridge ports are VLAN trunks. This means the switch terminating the cloud provider connections (double-tagged) cannot have local clients; these clients must be on a separate switch.

Long Interface Names

The Linux kernel limits interface names to 15 characters in length. For QinQ interfaces, you can reach this limit easily.

To work around this issue, create two VLANs as nested VLAN raw devices, one for the outer tag and one for the inner tag. For example, you cannot create an interface called swp50s0.1001.101 because it contains 16 characters. Instead, edit the `/etc/network/interfaces` file to create VLANs with IDs 1001 and 101. For example:

```
cumulus@switch:~$ sudo nano /etc/network/interfaces
...
auto vlan1001
iface vlan1001
      vlan-id 1001
       vlan-raw-device swp50s0
       vlan-protocol 802.1ad

auto vlan1001-101
iface vlan1001-101
```

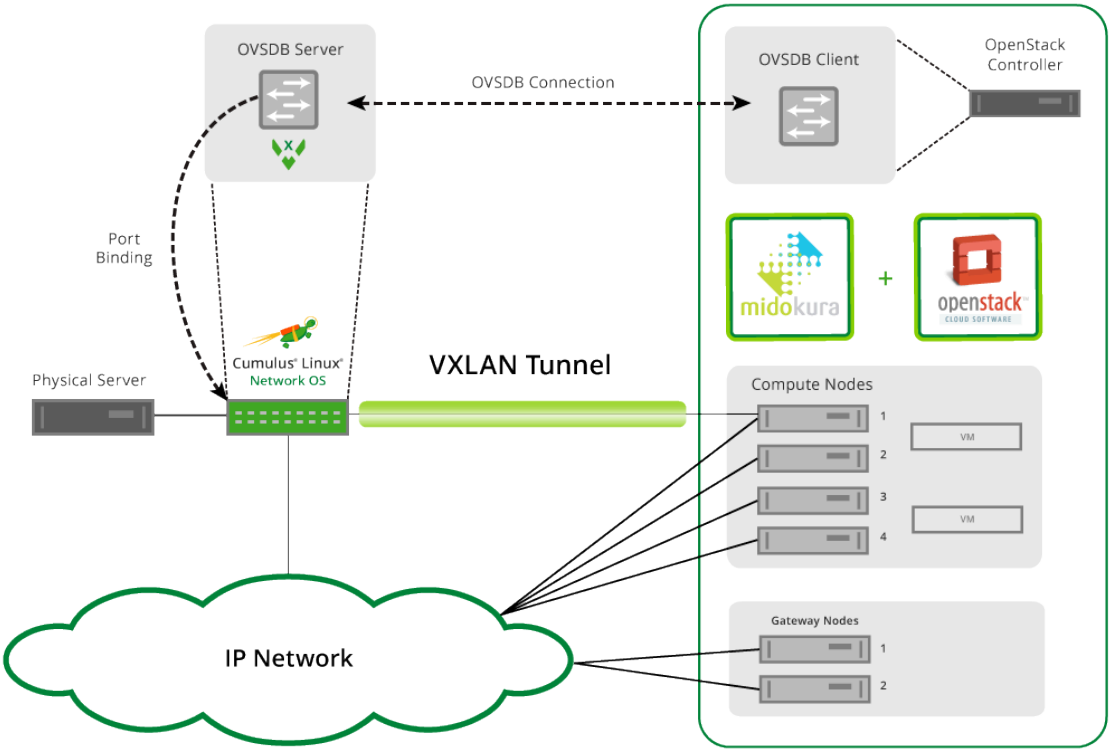
```
        vlan-id 101
        vlan-raw-device vlan1001

auto bridge101
iface bridge101
    bridge-ports vlan1001-101 vxlan1000101
...
```

Virtualization Integrations

Cumulus Linux integrates with a number of VXLAN controller-based virtualization solutions. You can integrate hardware VTEPs with Midokura-MidoNet and OpenStack, and with VMware NSX.

Integrating Hardware VTEPs with Midokura MidoNet and OpenStack

Cumulus Linux seamlessly integrates with the MidoNet OpenStack infrastructure, where the switches provide the VTEP gateway for terminating VXLAN tunnels from within MidoNet. MidoNet connects to the OVSDB server running on the Cumulus Linux switch, and exchanges information about the VTEPs and MAC addresses associated with the OpenStack Neutron networks. This provides seamless Ethernet connectivity between virtual and physical server infrastructures.



Get Started

Make sure you have a layer 2 gateway; a Tomahawk, Trident II+ or Trident II switch running Cumulus Linux. Cumulus Linux includes OVSDB server (`ovsdb-server`) and VTEPd (`ovs-vtepd`), which support VLAN-aware bridges.

To integrate a VXLAN with MidoNet, you need to:

- Configure the MidoNet integration on the switch
- Configure the MidoNet VTEP and port bindings
- Verify the VXLAN configuration

For more information about MidoNet, see the MidoNet Operations Guide, version 1.8 or later.

> **NOTE**
>
> There is no support for VXLAN routing in the Trident II chipset.

Configure the MidoNet Integration on the Switch

Before you start to configure the MidoNet tunnel zones and VTEP binding, and connect virtual ports to the VXLAN, you need to enable and start the `openvswitch-vtep` service, and configure the MidoNet integration on the switch. This creates the VTEP gateway and initializes the OVS database server.

Start the openvswitch-vtep Service

To enable and start the `openvswitch-vtep` service, run the following command:

```
cumulus@switch:~$ sudo systemctl enable openvswitch-vtep.service
cumulus@switch:~$ sudo systemctl start openvswitch-vtep.service
```

> **NOTE**
>
> In previous versions of Cumulus Linux, you had to edit the `/etc/default/openvswitch-vtep` file and then start the `openvswitch-vtep` service . Now, you just have to enable and start the `openvswitch-vtep` service.

Configure the MidoNet Integration Using the Configuration Script

The `vtep-bootstrap` script is available so you can perform the configuration automatically. For information, read `man vtep-bootstrap`. This script requires three parameters, in this order:

- The switch name (the name of the switch that is the VTEP gateway)
- The tunnel IP address (the datapath IP address of the VTEP)
- The management IP address (the IP address of the management

interface on the switch)

```
cumulus@switch:~$ sudo vtep-bootstrap sw11 10.111.1.1
10.50.20.21 --no_encryption
Executed:
  define physical switch
  ().
Executed:
  define local tunnel IP address on the switch
  ().
Executed:
  define management IP address on the switch
  ().
Executed:
  restart a service
  (Killing ovs-vtepd (28170).
Killing ovsdb-server (28146).
Starting ovsdb-server.
Starting ovs-vtepd.).
```

NOTE

Because MidoNet does not have a controller, you need to use a

dummy IP address (for example, 1.1.1.1) for the controller parameter in the script. After the script completes, delete the VTEP manager, as it is not needed and will otherwise fill the logs with inconsequential error messages:

```
cumulus@switch:~$ sudo vtep-ctl del-manager
```

Configure the MidoNet Integration Manually

If you do not use the configuration script, you must initialize the OVS database instance manually and create the VTEP.

Perform the following commands in order (see the automated script example above for values):

1. Define the switch in OVSDB:

```
cumulus@switch:~$ sudo vtep-ctl add-ps <switch_name>
```

2. Define the VTEP tunnel IP address:

```
cumulus@switch:~$ sudo vtep-ctl set Physical_switch
<switch_name> tunnel_ips=<tunnel_ip>
```

3. Define the management interface IP address:

```
cumulus@switch:~$ sudo vtep-ctl set Physical_switch
<switch_name> management_ips=<management_ip>
```

4. Restart the OVSDB server and `vtepd`:

```
cumulus@switch:~$ sudo systemctl restart openvswitch-
vtep.service
```

The switch is now ready to connect to MidoNet. The rest of the configuration is performed from the MidoNet Manager GUI or using the MidoNet API.

Configure MidoNet VTEP and Port Bindings

This part of the configuration sets up MidoNet and OpenStack to connect the virtualization environment to the Cumulus Linux switch. The `midonet-`

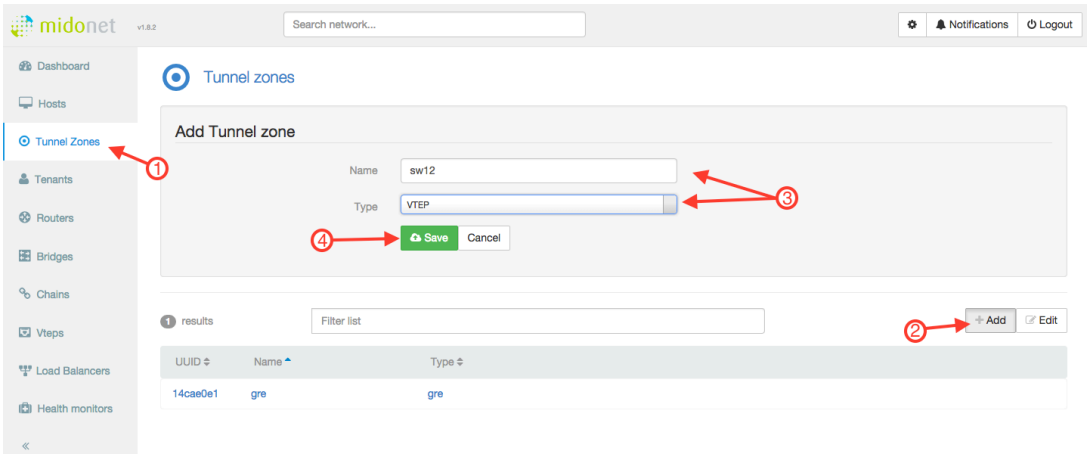
`agent` is the networking component that manages the VXLAN, while the Open Virtual Switch (OVS) client on the OpenStack controller node communicates MAC address information between the `midonet-agent` and the Cumulus Linux OVS database (OVSDB) server.

You can configure the MidoNet VTEP and port bindings from the MidoNet Manager GUI or the MidoNet CLI.

From the MidoNet Manager GUI

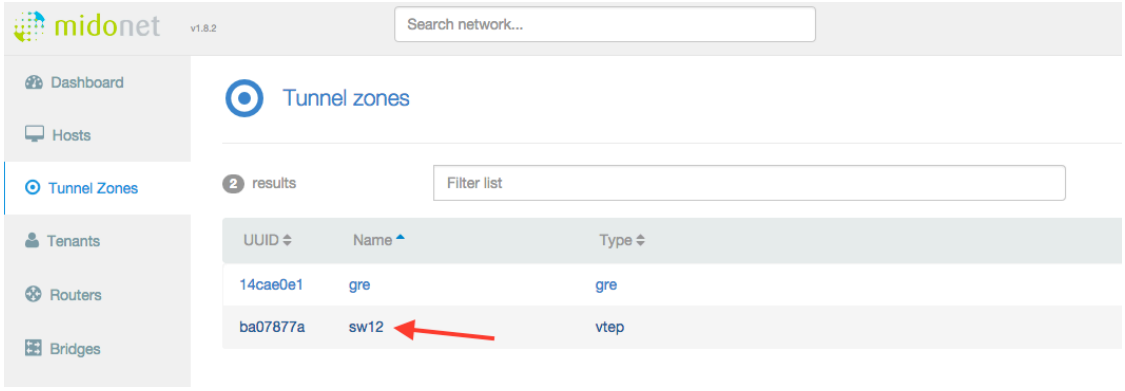
Create a Tunnel Zone

1. Click **Tunnel Zones** in the menu on the left side.
2. Click **Add**.
3. Give the tunnel zone a **Name** and select **VTEP** for the **Type**.
4. Click **Save**.



Add Hosts to a Tunnel Zone

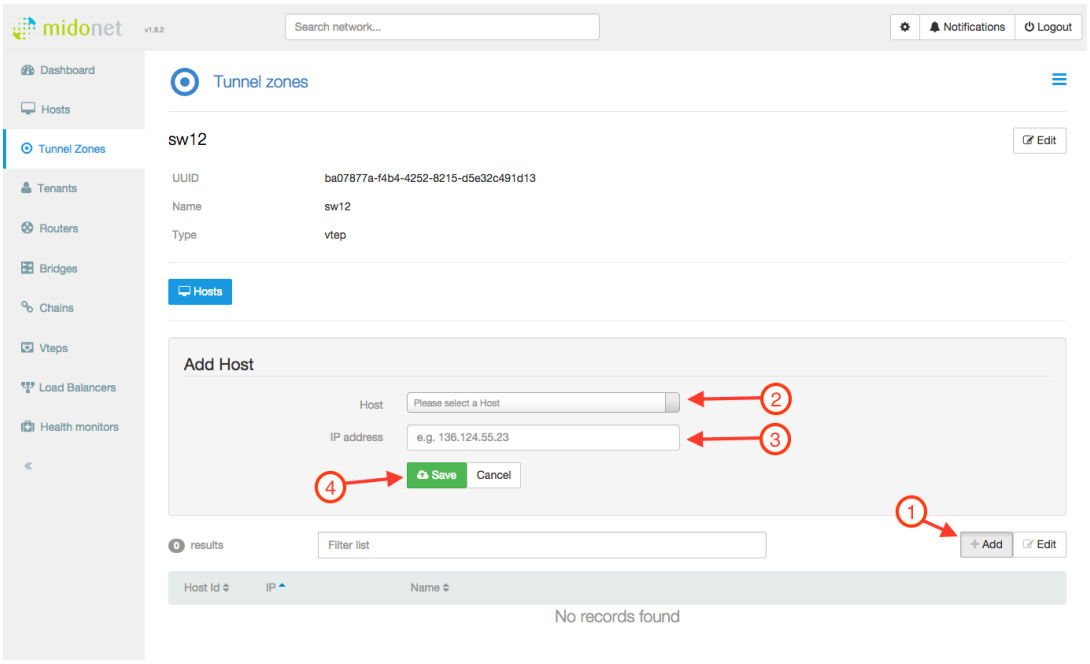
After you create the tunnel zone, click the name of the tunnel zone to view the hosts table.



The tunnel zone is a construct used to define the VXLAN source address used for the tunnel. The address of this host is used for the source of the VXLAN encapsulation and traffic transits into the routing domain from this point. Therefore, the host must have layer 3 reachability to the Cumulus Linux switch tunnel IP.

Next, add a host entry to the tunnel zone:

1. Click **Add**.
2. Select a host from the **Host** list.
3. Provide the tunnel source **IP Address** to use on the selected host.
4. Click **Save**.



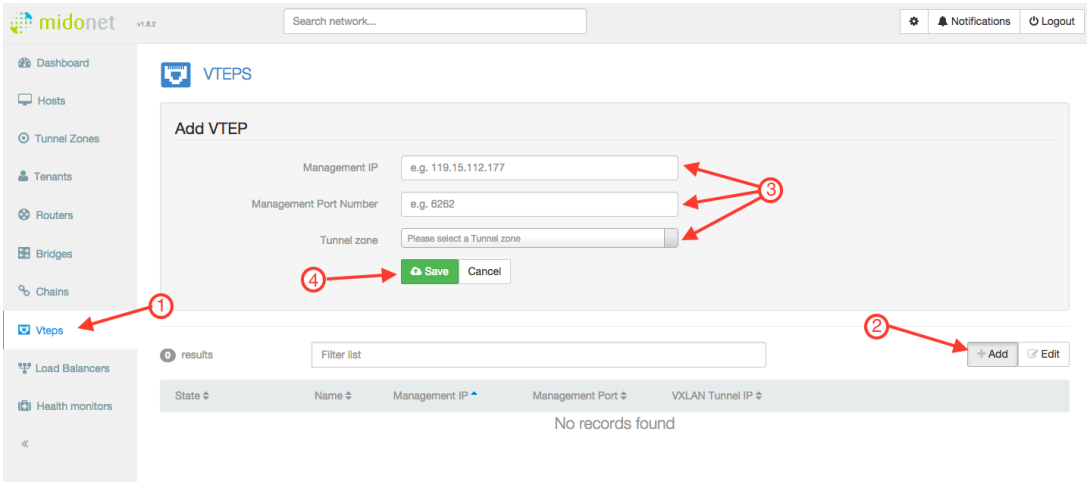
The host list now displays the new entry:

Create the VTEP

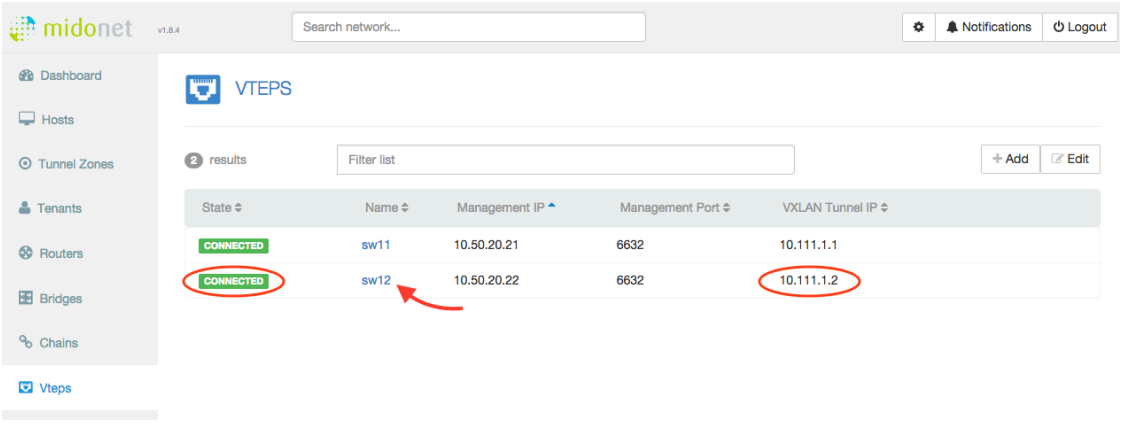
1. Click the **Vteps** menu on the left side.
2. Click **Add**.
3. Fill out the fields using the same information you used earlier on the switch for the bootstrap procedure:

- **Management IP** is typically the eth0 address of the switch. This tells the OVS-client to connect to the OVSDB-server on the Cumulus Linux switch.
- **Management Port Number** is the PTCP port you configured in the `ovs-ctl-vtep` script earlier (the example uses 6632).
- **Tunnel Zone** is the name of the zone you created in the previous procedure.

4. Click **Save**.



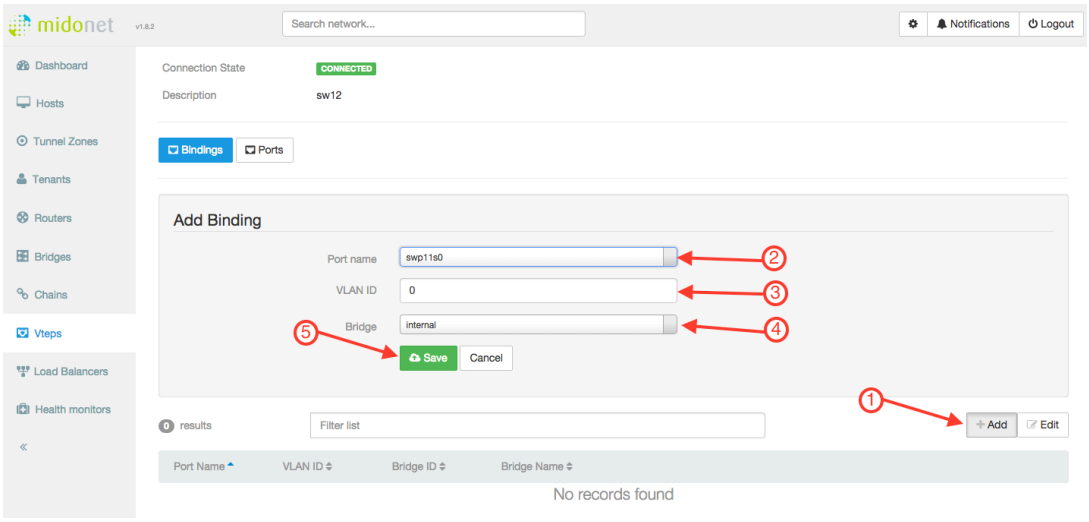
The new VTEP appears in the list below. MidoNet then initiates a connection between the OpenStack Controller and the Cumulus Linux switch. If the OVS client successfully connects to the OVSDB server, the VTEP entry displays the switch name and VXLAN tunnel IP address, which you specified during the bootstrapping process.



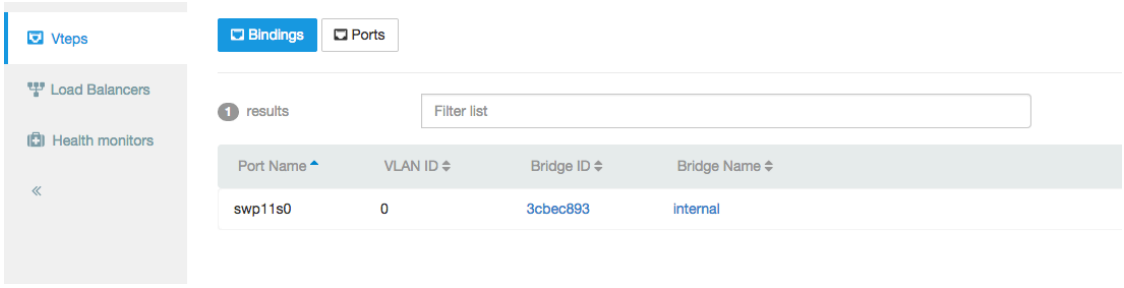
Bind Ports to the VTEP

Now that connectivity is established to the switch, you need to add a physical port binding to the VTEP on the Cumulus Linux switch:

1. Click **Add**.
2. In the **Port Name** list, select the port on the Cumulus Linux switch that you are using to connect to the VXLAN segment.
3. Specify the **VLAN ID** (enter 0 for untagged).
4. In the **Bridge** list, select the MidoNet bridge that the instances (VMs) are using in OpenStack.
5. Click **Save**.



You see the port binding displayed in the binding table under the VTEP.



After the port is bound, this automatically configures a VXLAN bridge interface, and includes the VTEP interface and the port bound to the bridge. Now the OpenStack instances (VMs) are able to ping the hosts connected to the bound port on the Cumulus switch. The Troubleshooting section below demonstrates the verification of the VXLAN data and control planes.

From the MidoNet CLI

To get started with the MidoNet CLI, you can access the CLI prompt on the

OpenStack Controller:

```
root@os-controller:~# midonet-cli
midonet>
```

From the MidoNet CLI, the commands explained in this section perform the same operations depicted in the previous section with the MidoNet Manager GUI.

1. Create a tunnel zone with a name and type *vtep*:

```
midonet> tunnel-zone create name sw12 type vtep
tzone1
```

2. The tunnel zone is a construct used to define the VXLAN source address used for the tunnel. The address of this host is used for the source of the VXLAN encapsulation and traffic transits into the routing domain from this point. Therefore, the host must have layer 3 reachability to the Cumulus Linux switch tunnel IP.

   - First, obtain the list of available hosts connected to the Neutron network and the MidoNet bridge.
   - Next, get a listing of all the interfaces.
   - Finally, add a host entry to the tunnel zone ID returned in the previous step and specify which interface address to use.

```
midonet> list host
host host0 name os-compute1 alive true
host host1 name os-network alive true
midonet> host host0 list interface
iface midonet host_id host0 status 0 addresses [] mac
02:4b:38:92:dd:ce mtu 1500 type Virtual endpoint DATAPATH
iface lo host_id host0 status 3 addresses [u'127.0.0.1',
u'169.254.169.254', u'0:0:0:0:0:0:0:1'] mac 00:00:00:00:00:00
mtu 65536 type Virtual endpoint LOCALHOST
iface virbr0 host_id host0 status 1 addresses
[u'192.168.122.1'] mac 22:6e:63:90:1f:69 mtu 1500 type
Virtual endpoint UNKNOWN
iface tap7cfcf84c-26 host_id host0 status 3 addresses
[u'fe80:0:0:0:e822:94ff:fee2:d41b'] mac ea:22:94:e2:d4:1b mtu
65000 type Virtual endpoint DATAPATH
iface eth1 host_id host0 status 3 addresses [u'10.111.0.182',
u'fe80:0:0:0:5054:ff:fe85:acd6'] mac 52:54:00:85:ac:d6 mtu
1500 type Physical endpoint PHYSICAL
iface tapfd4abcea-df host_id host0 status 3 addresses
[u'fe80:0:0:0:14b3:45ff:fe94:5b07'] mac 16:b3:45:94:5b:07 mtu
65000 type Virtual endpoint DATAPATH
iface eth0 host_id host0 status 3 addresses [u'10.50.21.182',
u'fe80:0:0:0:5054:ff:feef:c5dc'] mac 52:54:00:ef:c5:dc mtu
1500 type Physical endpoint PHYSICAL
```

```
midonet> tunnel-zone tzone0 add member host host0 address
10.111.0.182
zone tzone0 host host0 address 10.111.0.182
```

Repeat this procedure for each OpenStack host connected to the Neutron network and the MidoNet bridge.

3. Create a VTEP and assign it to the tunnel zone ID returned in the previous step. The management IP address (the destination address for the VXLAN or remote VTEP) and the port must be the same ones you configure in the `vtep-bootstrap` script or the manual bootstrapping:

```
midonet> vtep add management-ip 10.50.20.22 management-port
6632 tunnel-zone tzone0
name sw12 description sw12 management-ip 10.50.20.22
management-port 6632 tunnel-zone tzone0 connection-state
CONNECTED
```

In this step, MidoNet initiates a connection between the OpenStack Controller and the Cumulus Linux switch. If the OVS client successfully connects to the OVSDB server, the returned values should show the name and description matching the `switch-name` parameter specified in the bootstrap process.

> **NOTE**
>
> Verify the connection-state as CONNECTED. If ERROR is returned, you must debug. Typically this only fails if the `management-ip` and or the `management-port` settings are incorrect.

4. The VTEP binding uses the information provided to MidoNet from the OVSDB server, providing a list of ports that the hardware VTEP can use for layer 2 attachment. This binding virtually connects the physical interface to the overlay switch, and joins it to the Neutron bridged network.

   First, get the UUID of the Neutron network behind the MidoNet bridge:

   ```
   midonet> list bridge
   bridge bridge0 name internal state up
   bridge bridge1 name internal2 state up
   midonet> show bridge bridge1 id
   6c9826da-6655-4fe3-a826-4dcba6477d2d
   ```

   Next, create the VTEP binding using the UUID and the switch port being bound to the VTEP on the remote end. If there is no VLAN ID, set `vlan` to 0:

```
midonet> vtep name sw12 binding add network-id
6c9826da-6655-4fe3-a826-4dcba6477d2d physical-port swp11s0
vlan 0
management-ip 10.50.20.22 physical-port swp11s0 vlan 0
network-id 6c9826da-6655-4fe3-a826-4dcba6477d2d
```

At this point, the VTEP is connected and the layer 2 overlay is operational. From the openstack instance (VM), you can ping a physical server connected to the port bound to the hardware switch VTEP.

Troubleshooting

As with any complex system, there is a control plane and data plane.

Control Plane Troubleshooting

In this solution, the control plane consists of the connection between the OpenStack Controller and each Cumulus Linux switch running the `ovsdb-server` and `vtepd` daemons.

Verify VTEP and OVSDB Services

First, it is important that the OVSDB server and `ovs-vtep` daemon are running. Verify this is the case:

```
cumulus@switch12:~$ systemctl status openvswitch-vtep.service
ovsdb-server is running with pid 17440
ovs-vtepd is running with pid 17444
```

Verify OVSDB-server Connections

From the OpenStack Controller host, verify that it can connect to the `ovsdb-server`. Telnet to the switch IP address on port 6632:

```
root@os-controller:~# telnet 10.50.20.22 6632
Trying 10.50.20.22...
Connected to 10.50.20.22.
Escape character is '^]'.
<Ctrl+c>
Connection closed by foreign host.
```

If the connection fails, verify IP reachability from the host to the switch. If that succeeds, it is likely that the bootstrap process did not set up port 6632. Redo the bootstrapping procedures above.

```
root@os-controller:~# ping -c1 10.50.20.22
PING 10.50.20.22 (10.50.20.22) 56(84) bytes of data.
```

```
64 bytes from 10.50.20.22: icmp_seq=1 ttl=63 time=0.315 ms
--- 10.50.20.22 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.315/0.315/0.315/0.000 ms
```

Verify the VXLAN Bridge and VTEP Interfaces

After creating the VTEP in MidoNet and adding an interface binding, you see the **br-vxln** and **vxln** interfaces on the switch. Verify that the VXLAN bridge and VTEP interface are created and UP:

```
cumulus@switch12:~$ sudo brctl show
bridge name  bridge id          STP     enabled interfaces
bridge       8000.00e0ec2749a2  no      swp11s0
                                        vxln10006
```

```
cumulus@switch12:~$ sudo ip -d link show vxln10006
55: vxln10006: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
noqueue master br-vxln10006 state UNKNOWN mode DEFAULT
link/ether 72:94:eb:b6:6c:c3 brd ff:ff:ff:ff:ff:ff
vxlan id 10006 local 10.111.1.2 port 32768 61000 nolearning
ageing 300 svcnode 10.111.0.182
```

```
bridge_slave
```

Next, look at the bridging table for the VTEP and the forwarding entries. The bound interface and the VTEP are listed along with the MAC addresses of those interfaces. When the hosts attached to the bound port send data, those MACs are learned and entered into the bridging table, as well as the OVSDB.

```
cumulus@switch12:~$ brctl showmacs br-vxln10006
port name     mac addr            vlan    is local?   ageing
timer
swp11s0     00:e0:ec:27:49:a2       0         yes         0.00
swp11s0     64:ae:0c:32:f1:41       0         no          0.01
vxln10006   72:94:eb:b6:6c:c3       0         yes         0.00
```

```
cumulus@switch12:~$ sudo bridge fdb show br-vxln10006
fa:16:3e:14:04:2e dev vxln10004 dst 10.111.0.182 vlan 65535
self permanent
00:e0:ec:27:49:a2 dev swp11s0 vlan 0 master br-vxln10004
permanent
b6:71:33:3b:a7:83 dev vxln10004 vlan 0 master br-vxln10004
```

```
permanent
64:ae:0c:32:f1:41 dev swp11s0 vlan 0 master br-vxln10004
```

Datapath Troubleshooting

If you have verified the control plane is correct, and you still cannot get data between the OpenStack instances and the physical nodes on the switch, there might be something wrong with the data plane. The data plane consists of the actual VXLAN encapsulated path, between one of the OpenStack nodes running the `midolman` service. This is typically the compute nodes, but can include the MidoNet gateway nodes. If the OpenStack instances can ping the tenant router address but cannot ping the physical device connected to the switch (or vice versa), then something is wrong in the data plane.

Verify IP Reachability

First, there must be IP reachability between the encapsulating node, and the address you bootstrapped as the tunnel IP on the switch. Verify the OpenStack host can ping the tunnel IP. If this does not work, check the routing design and fix the layer 3 problem first.

```
root@os-compute1:~# ping -c1 10.111.1.2
PING 10.111.1.2 (10.111.1.2) 56(84) bytes of data.
```

```
64 bytes from 10.111.1.2: icmp_seq=1 ttl=62 time=0.649 ms
--- 10.111.1.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.649/0.649/0.649/0.000 ms
```

MidoNet VXLAN Encapsulation

If the instance (VM) cannot ping the physical server or the reply is not returning, look at the packets on the OpenStack node. Initiate a ping from the OpenStack instance, then use `tcpdump` to see the VXLAN data. This example displays a successful tcpdump.

```
root@os-compute1:~# tcpdump -i eth1 -l -nnn -vvv -X -e port 4789
52:54:00:85:ac:d6 > 00:e0:ec:26:50:36, ethertype IPv4 (0x0800),
length 148: (tos 0x0, ttl 255, id 7583, offset 0, flags [none],
proto UDP (17), length 134)
  10.111.0.182.41568 > 10.111.1.2.4789: [no cksum] VXLAN, flags
[I] (0x08), vni 10008
fa:16:3e:14:04:2e > 64:ae:0c:32:f1:41, ethertype IPv4 (0x0800),
length 98: (tos 0x0, ttl 64, id 64058, offset 0, flags [DF],
proto ICMP (1), length 84)
  10.111.102.104 > 10.111.102.2: ICMP echo request, id 15873,
seq 0, length 64
```

```
  0x0000: 4500 0086 1d9f 0000 ff11 8732 0a6f 00b6
E..........2.o..
  0x0010: 0a6f 0102 a260 12b5 0072 0000 0800 0000
.o...`...r......
  0x0020: 0027 1800 64ae 0c32 f141 fa16 3e14 042e
.'..d..2.A..>...
  0x0030: 0800 4500 0054 fa3a 4000 4001 5f26 0a6f
..E..T.:@.@._&.o
  0x0040: 6668 0a6f 6602 0800 f9de 3e01 0000 4233
fh.of.....>...B3
  0x0050: 7dec 0000 0000 0000 0000 0000 0000 0000
}...............
  0x0060: 0000 0000 0000 0000 0000 0000 0000 0000
................
  0x0070: 0000 0000 0000 0000 0000 0000 0000 0000
................
  0x0080: 0000 0000 0000 ......
00:e0:ec:26:50:36 > 52:54:00:85:ac:d6, ethertype IPv4 (0x0800),
length 148: (tos 0x0, ttl 62, id 2689, offset 0, flags [none],
proto UDP (17), length 134)
  10.111.1.2.63385 > 10.111.0.182.4789: [no cksum] VXLAN, flags
[I] (0x08), vni 10008
64:ae:0c:32:f1:41 > fa:16:3e:14:04:2e, ethertype IPv4 (0x0800),
length 98: (tos 0x0, ttl 255, id 64058, offset 0, flags [DF],
```

```
proto ICMP (1), length 84)
  10.111.102.2 > 10.111.102.104: ICMP echo reply, id 15873, seq
0, length 64
  0x0000: 4500 0086 0a81 0000 3e11 5b51 0a6f 0102
E.......>.[Q.o..
  0x0010: 0a6f 00b6 f799 12b5 0072 0000 0800 0000
.o.......r......
  0x0020: 0027 1800 fa16 3e14 042e 64ae 0c32 f141
.'....>...d..2.A
  0x0030: 0800 4500 0054 fa3a 4000 ff01 a025 0a6f
..E..T.:@....%.o
  0x0040: 6602 0a6f 6668 0000 01df 3e01 0000 4233
f..ofh....>...B3
  0x0050: 7dec 0000 0000 0000 0000 0000 0000 0000
}...............
  0x0060: 0000 0000 0000 0000 0000 0000 0000 0000
................
  0x0070: 0000 0000 0000 0000 0000 0000 0000 0000
................
  0x0080: 0000 0000 0000 ......
```

Inspect the OVSDB

These commands show you the information installed in the OVSDB. This database is structured using the *physical switch* ID, with one or more

logical switch IDs associated with it. The bootstrap process creates the physical switch and MidoNet creates the logical switch after the control session is established.

List the Physical Switch

```
cumulus@switch12:~$ vtep-ctl list-ps
sw12
```

List the Logical Switch

```
cumulus@switch12:~$ vtep-ctl list-ls
mn-6c9826da-6655-4fe3-a826-4dcba6477d2d
```

List Local or Remote MAC Addresses

These commands show the MAC addresses learned from the connected port bound to the logical switch or the MAC addresses advertised from MidoNet. The *unknown-dst* entries are installed to satisfy the ethernet flooding of unknown unicast and are important for learning.

```
cumulus@switch12:~$ vtep-ctl list-local-macs
mn-6c9826da-6655-4fe3-a826-4dcba6477d2d
```

```
ucast-mac-local
  64:ae:0c:32:f1:41 -> vxlan_over_ipv4/10.111.1.2
mcast-mac-local
  unknown-dst -> vxlan_over_ipv4/10.111.1.2
```

```
cumulus@switch12:~$ vtep-ctl list-remote-macs
mn-6c9826da-6655-4fe3-a826-4dcba6477d2d
ucast-mac-remote
  fa:16:3e:14:04:2e -> vxlan_over_ipv4/10.111.0.182
mcast-mac-remote
  unknown-dst -> vxlan_over_ipv4/10.111.0.182oh
```

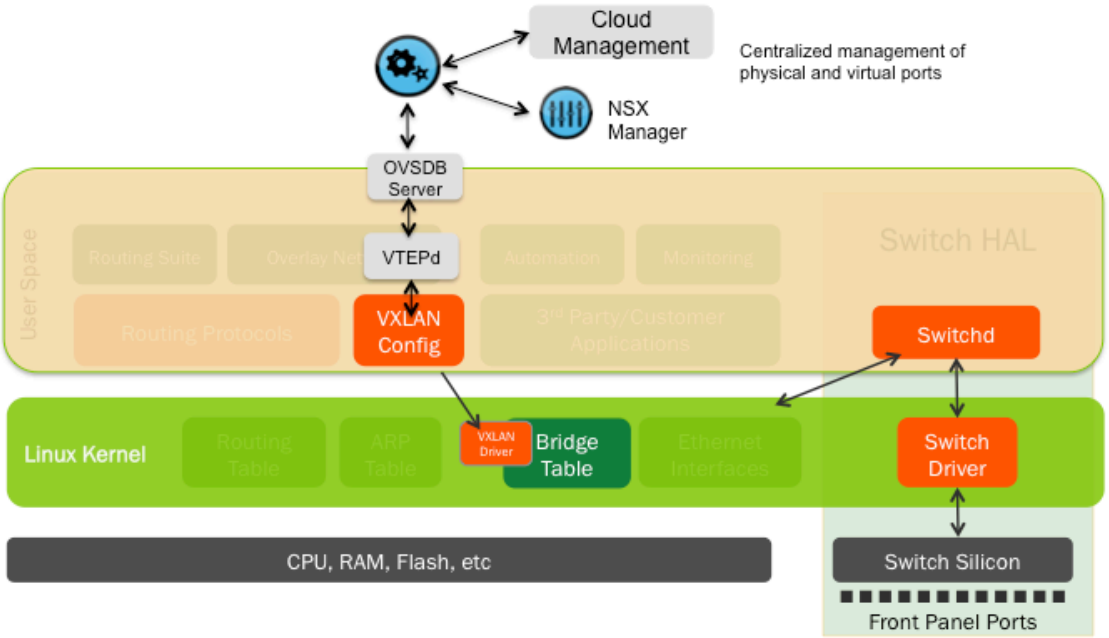
Show Open Vswitch Database (OVSDB) Data

The `ovsdb-client dump` command is large but shows all of the information and tables used in communication between the OVS client and server.

▼ Click to expand the output ...

Integrating Hardware VTEPs with VMware NSX-V

Switches running Cumulus Linux can integrate with VMware NSX-V to act as hardware VTEP gateways. The VMware NSX-V controller provides consistent provisioning across virtual and physical server infrastructures.



Cumulus Linux also supports integration with VMware NSX in high availability mode. Refer to OVSDB Server High Availability.

Get Started

Before you integrate VXLANs with NSX-V, make sure you have a layer 2 gateway; a switch with Broadcom Tomahawk, Trident II+, Trident II, or

Trident III, Maverick, or Mellanox Spectrum or Spectrum A1 running Cumulus Linux. Cumulus Linux includes OVSDB server (`ovsdb-server`) and VTEPd (`ovs-vtepd`), which support VLAN-aware bridges.

To integrate a VXLAN with NSX-V, you need to:

- Configure the NSX-V integration on the switch.
- Configure the transport and logical layers from the NSX Manager.
- Verify the VXLAN configuration.

> **NOTE**
>
> Cumulus Linux supports security protocol version TLSv1.2 for SSL connections between the OVSDB server and the NSX controller. The OVSDB server cannot select the loopback interface as the source IP address, causing top of rack registration to the controller to fail. To work around this issue, run the `net add bgp redistribute connected` command followed by the `net commit` command.

Configure the Switch for NSX-V Integration

Before you start configuring the gateway service, and logical switches and ports that comprise the VXLAN, you need to enable and start the `openvswitch-vtep` service, and configure the NSX integration on the switch, either using the script or performing the manual configuration.

Start the openvswitch-vtep Service

To enable and start the `openvswitch-vtep` service, run the following command:

```
cumulus@switch:~$ sudo systemctl enable openvswitch-vtep.service
cumulus@switch:~$ sudo systemctl start openvswitch-vtep.service
```

> **NOTE**
>
> In previous versions of Cumulus Linux, you had to edit the `/etc/default/openvswitch-vtep` file and then start the `openvswitch-vtep` service . Now, you just have to enable and start the `openvswitch-vtep` service .

Configure the NSX-V Integration Using the Configuration Script

A script is available so you can configure the NSX-V integration on the switch automatically.

In a terminal session connected to the switch, run the `vtep-bootstrap` command with these options:

- `controller_ip` is the IP address of the NSX controller (192.168.100.17 in

the example command below).

- The ID for the VTEP (`vtep7` in the example command below).
- The datapath IP address of the VTEP (`172.16.20.157` in the example command below). This is the VXLAN anycast IP address.
- The IP address of the management interface on the switch (`192.168.100.157` in the example command below). This interface is used for control traffic.

```
cumulus@switch:~$ vtep-bootstrap vtep7 --controller_ip
192.168.100.17 172.16.20.157 192.168.100.157
Executed:
    create certificate on a switch, to be used for
authentication with controller
        ().
Executed:
    sign certificate
        (vtep7-req.pem   Tue Sep 11 21:11:27 UTC 2018
        fingerprint a4cda030fe5e458c0d7ba44e22f52650f01bcd75).
Executed:
        define physical switch
         ().
Executed:
    define NSX controller IP address in OVSDB
        ().
```

```
Executed:
    define local tunnel IP address on the switch
        ().
Executed:
    define management IP address on the switch
        ().
Executed:
    restart a service
        ().
```

Run the following commands in the order shown to complete the configuration process:

```
cumulus@switch:~$ sudo systemctl restart openvswitch-
vtep.service
cumulus@switch:~$ sudo ifreload -a
cumulus@switch:~$ sudo systemctl restart networking.service
```

Configure the NSX-V Integration Manually

ⓘ NOTE

You can configure the NSX-V integration manually for standalone mode only; manual configuration for OVSDB server high availability is not supported.

If you do not want to use the configuration script to configure the NSX-V integration on the switch automatically, you can configure the integration manually, which requires you to perform the following steps:

- Generate a certificate and key pair for authentication by NSX-V.
- Configure a switch as a VTEP gateway.

Generate the Credentials Certificate

In Cumulus Linux, generate a certificate that the NSX controller uses for authentication.

1. In a terminal session connected to the switch, run the following commands:

```
cumulus@switch:~$ sudo ovs-pki init
Creating controllerca...
Creating switchca...
cumulus@switch:~$ sudo ovs-pki req+sign cumulus
```

```
cumulus-req.pem Wed Oct 23 05:32:49 UTC 2013
        fingerprint b587c9fe36f09fb371750ab50c430485d33a174a

cumulus@switch:~$ ls -l
total 12
-rw-r--r-- 1 root root 4028 Oct 23 05:32 cumulus-cert.pem
-rw------- 1 root root 1679 Oct 23 05:32 cumulus-privkey.pem
-rw-r--r-- 1 root root 3585 Oct 23 05:32 cumulus-req.pem
```

2. In the `/usr/share/openvswitch/scripts/ovs-ctl-vtep` file, make sure the lines containing **private-key**, **certificate**, and **bootstrap-ca-cert** point to the correct files; **bootstrap-ca-cert** is obtained dynamically the first time the switch talks to the controller:

```
# Start ovsdb-server.
set ovsdb-server "$DB_FILE"
set "$@" -vANY:CONSOLE:EMER -vANY:SYSLOG:ERR -vANY:FILE:INFO
set "$@" --remote=punix:"$DB_SOCK"
set "$@" --remote=db:Global,managers
set "$@" --remote=ptcp:6633:$LOCALIP
set "$@" --private-key=/root/cumulus-privkey.pem
set "$@" --certificate=/root/cumulus-cert.pem
```

```
set "$@" --bootstrap-ca-cert=/root/controller.cacert
set "$@ " --ssl-protocols=TLSv1,TLSv1.1,TLSv1.2
```

If files have been moved or regenerated, restart the OVSDB server and VTEPd:

```
cumulus@switch:~$ sudo systemctl restart openvswitch-
vtep.service
```

3. Define the NSX Controller Cluster IP address in OVSDB. This causes the OVSDB server to start contacting the NSX controller:

```
cumulus@switch:~$ sudo vtep-ctl set-manager
ssl:192.168.100.17:6632
```

4. Define the local IP address on the VTEP for VXLAN tunnel termination. First, find the physical switch name as recorded in OVSDB:

```
cumulus@switch:~$ sudo vtep-ctl list-ps
vtep7
```

Then set the tunnel source IP address of the VTEP. This is the datapath address of the VTEP, which is typically an address on a loopback interface on the switch that is reachable from the underlying layer 3 network:

```
cumulus@switch:~$ sudo vtep-ctl set Physical_Switch vtep7
tunnel_ips=172.16.20.157
```

After you generate the certificate, keep the terminal session active; you need to paste the certificate into NSX Manager when you configure the VTEP gateway.

Enable ovs-vtepd to Use the VLAN-aware Bridge

By default, in stand-alone mode, the ovs-vtep daemon creates traditional bridges for each VXLAN VTEP. To use the VLAN-aware bridge with the VTEPs, edit the `/usr/share/openvswitch/scripts/ovs-ctl-vtep` file and uncomment the `--enable-vlan-aware-mode` line:

```
# Start ovs-vtepd
set ovs-vtepd unix:"$DB_SOCK "
set "$@ " -vconsole:emer -vsyslog:err -vfile:info
#set "$@ " --enable-vlan-aware-mode
```

Then restart the OVSDB server and VTEPd:

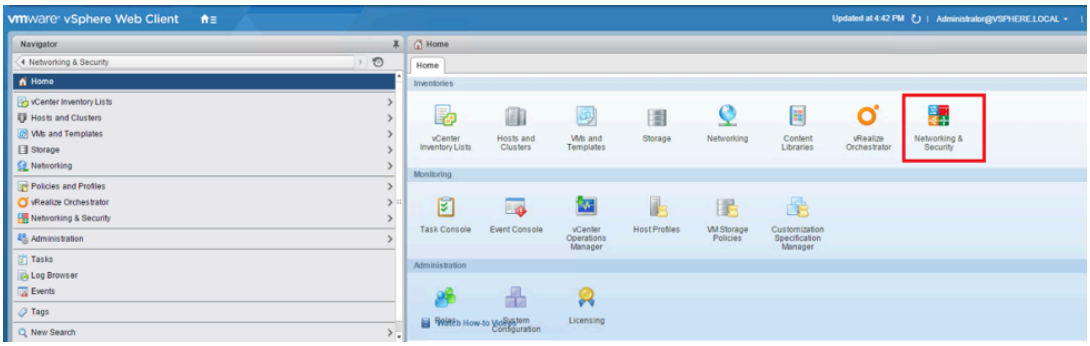
```
cumulus@switch:~$ sudo systemctl restart openvswitch-
vtep.service
```

Provision VMware NSX-V

Configure the Switch as a VTEP Gateway

After you create a certificate, connect to NSX Manager in a browser to configure a Cumulus Linux switch as a hardware VTEP gateway. In this example, the IP address of the NSX Manager is 192.168.110.23.

1. In NSX Manager, add a new HW VTEP gateway. Click the **Network & Security** icon, **Service Definitions** category, then the **Hardware Devices** tab. Under **Hardware Devices**, click **+**. The Create Add Hardware Devices window opens.

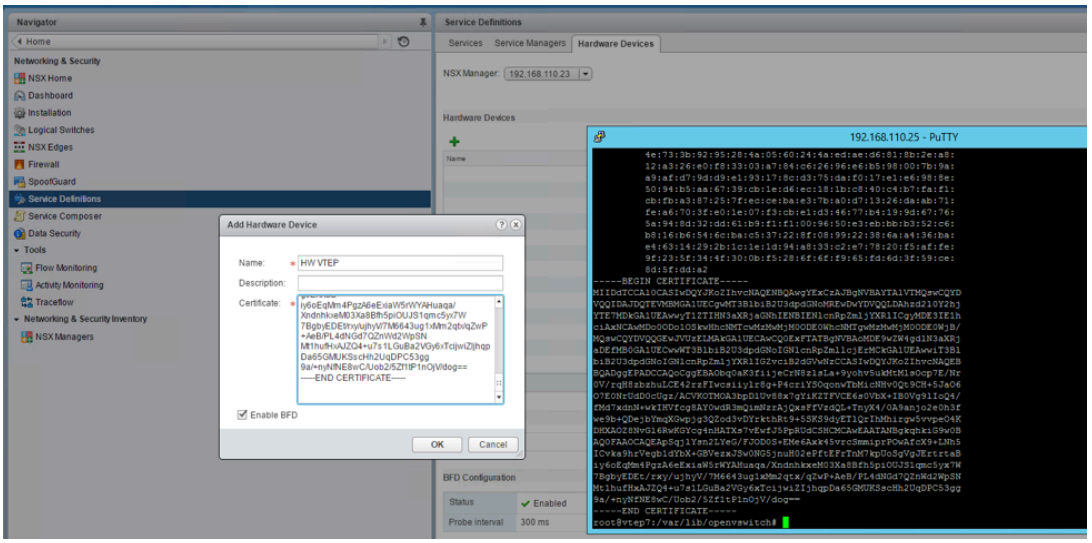


2. In the **Name** field, provide a name for the HW VTEP gateway.

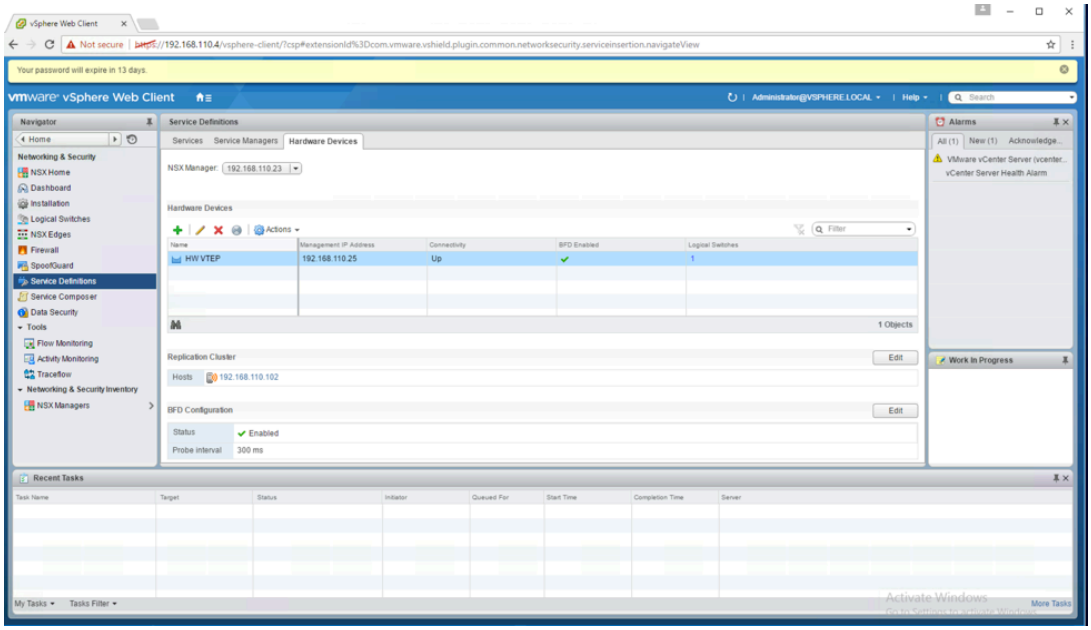
3. Enable the BFD service to the service nodes. Select the **Enable BFD** check box.

4. From the terminal session connected to the switch where you generated the certificate, copy the certificate and paste it into the **Certificate** text field. Copy only the bottom portion, including the `BEGIN CERTIFICATE` and `END CERTIFICATE` lines. For example, copy all the highlighted text in the terminal terminal and paste it into NSX Manager:

```
cumulus@switch:~$ cd /var/lib/openvswitch
cumulus@switch:/var/lib/openvswitch$ ls
conf.db  pki  vtep7-cert.pem  vtep7-privkey.pem  vtep7-req.pem
cumulus@switch:/var/lib/openvswitch$ cat vtep7-cert.pem
```



5. Click **OK** to save the gateway.



After communication is established between the switch and the controller, a `controller.cacert` file is downloaded onto the switch.

Verify that the controller and switch handshake is successful. In a terminal connected to the switch, run this command:

```
cumulus@switch:~$ sudo ovsdb-client dump -f list | grep -A 7
"Manager"
Manager table
_uuid               : 2693ea2e-306-4c23-ac03-934ala304077
inactivity_probe    : []
is_connected        : true
max_backoff         : []
```

```
other_config        : {}
status              : {sec_since_connect="557", state=ACTIVE}
target              : "ssl:192.168.110.110:6640"
```

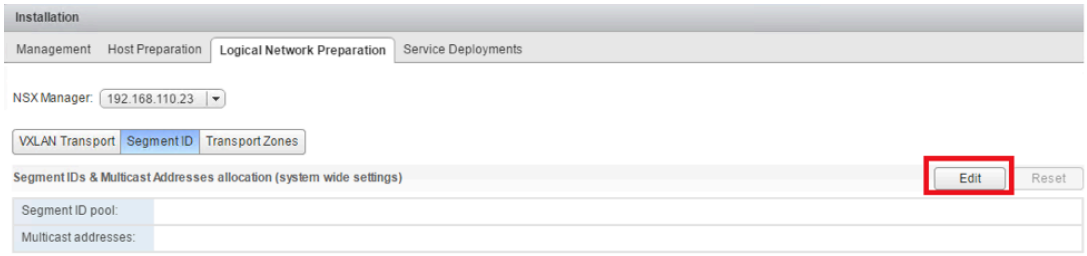
Configure the Transport and Logical Layers

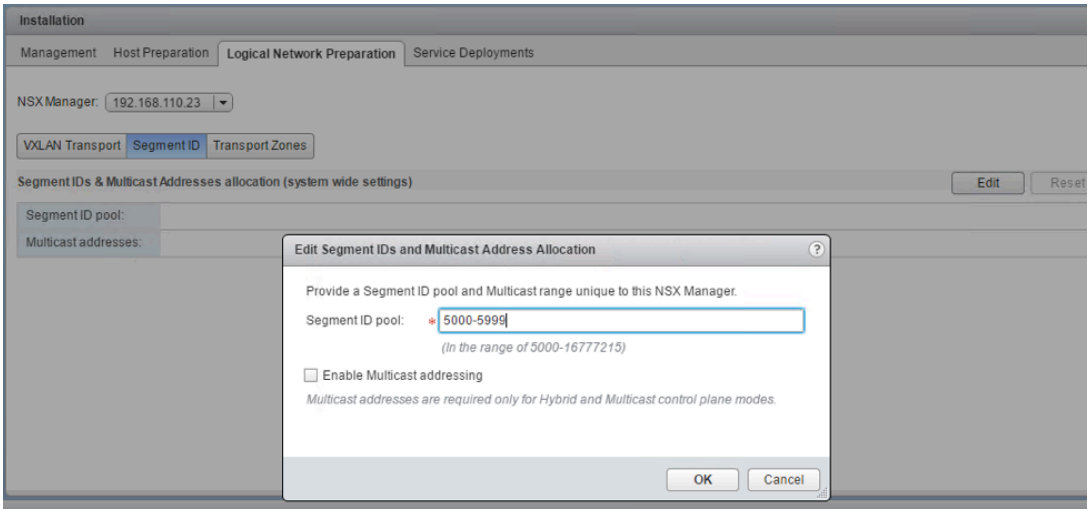
Configure the Transport Layer

After you finish configuring NSX-V integration on the switch, configure the transport zone and segment ID.

1. In NSX Manager, click the **Logical Network Preparation** tab in the **Installation** category, then click the **Segment ID** tab.

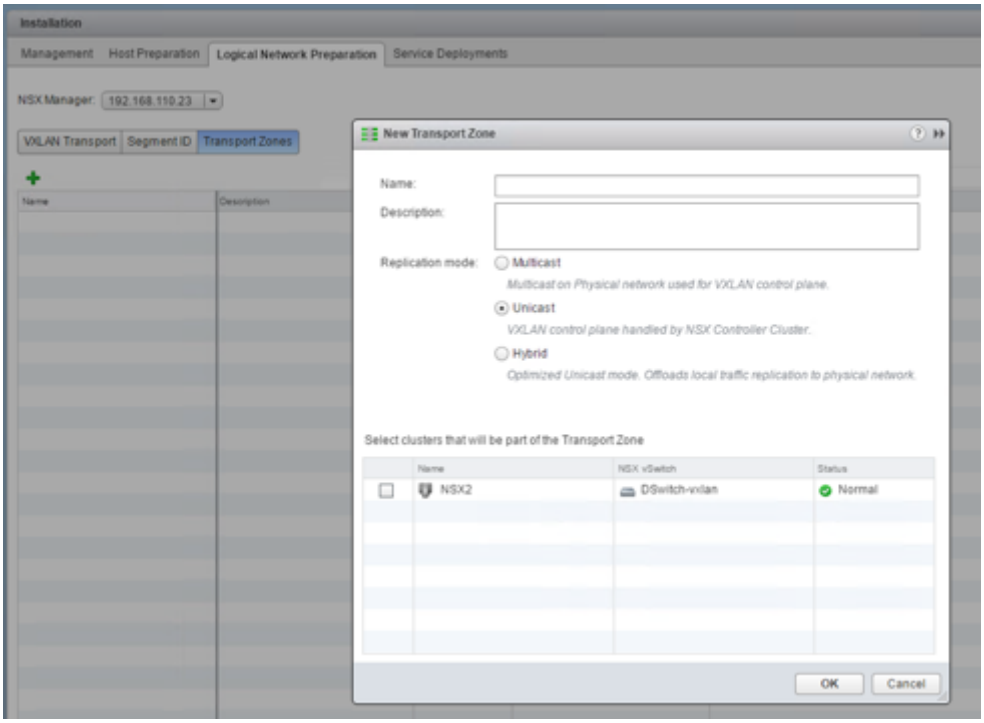
2. Click **Edit** and add the segment IDs (VNIDs) to be used. Here VNIs 5000-5999 are configured.





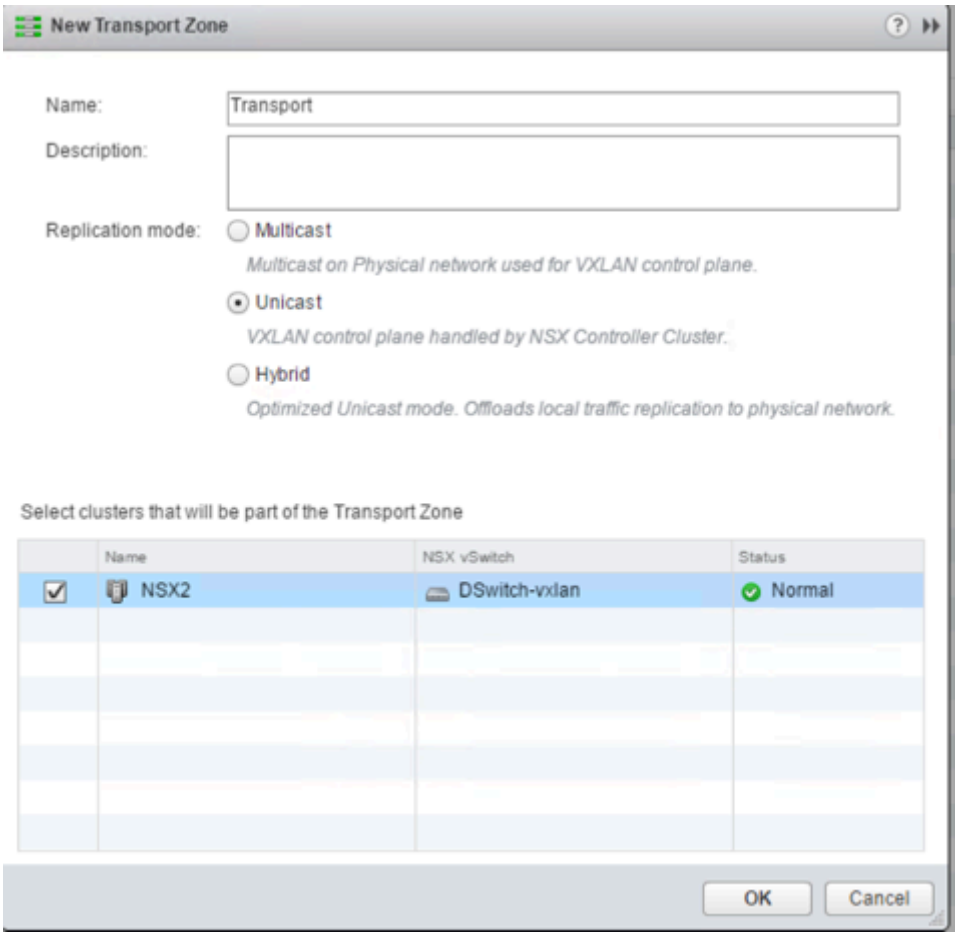
3. Click **OK** to save and provision the segment IDs.

4. Click the **Transport Zones** tab, choose the name of the transport zone.



5. Select **Unicast** to choose the NSX-V Controller Cluster to handle the

VXLAN control plane.



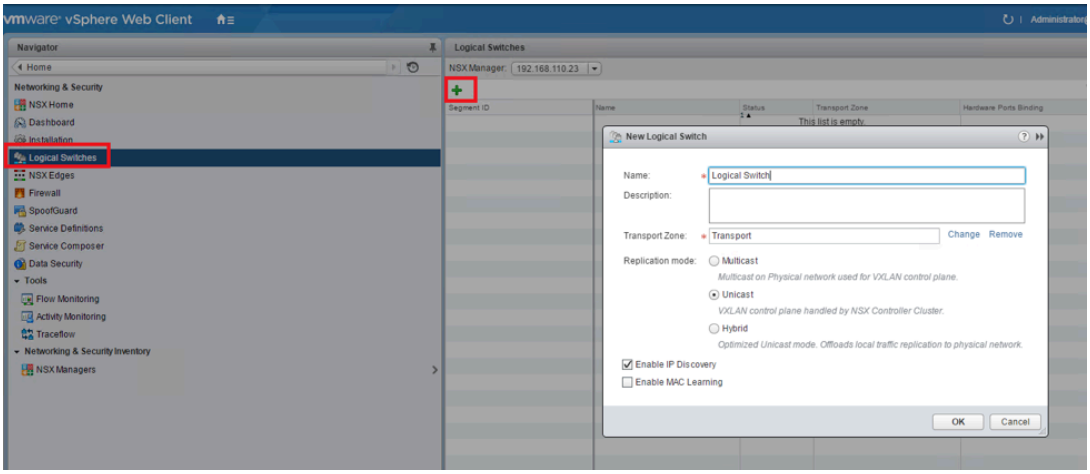
6. Click **OK** to save the new transport zone.

Configure the Logical Layer

To complete the integration with NSX-V, you need to configure the logical layer, which requires defining a logical switch (the VXLAN instance) and all the logical ports needed.

To define the logical switch:

1. In NSX Manager, select the **Logical Switches** category. Click **+** to add a logical switch instance.

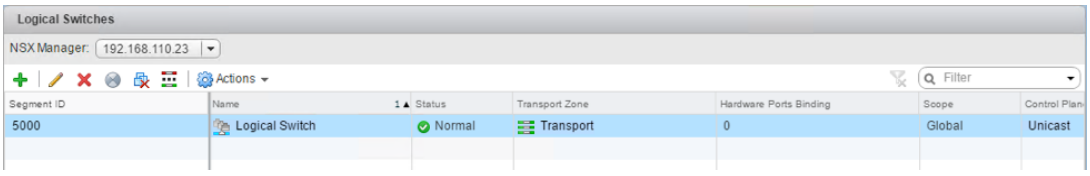


2. In the **Name** field, enter a name for the logical switch.

3. In the **Transport Zone** field, add the transport zone that you created earlier.

4. In the **Replication Mode** field, select **Unicast** for replication by the service node. Then check the **Enable IP Discovery** check box.

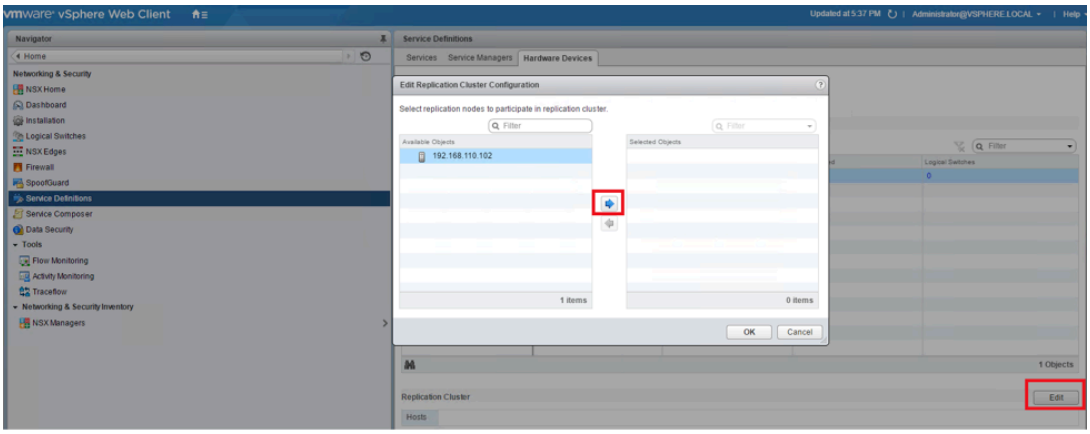
5. Click **OK**.



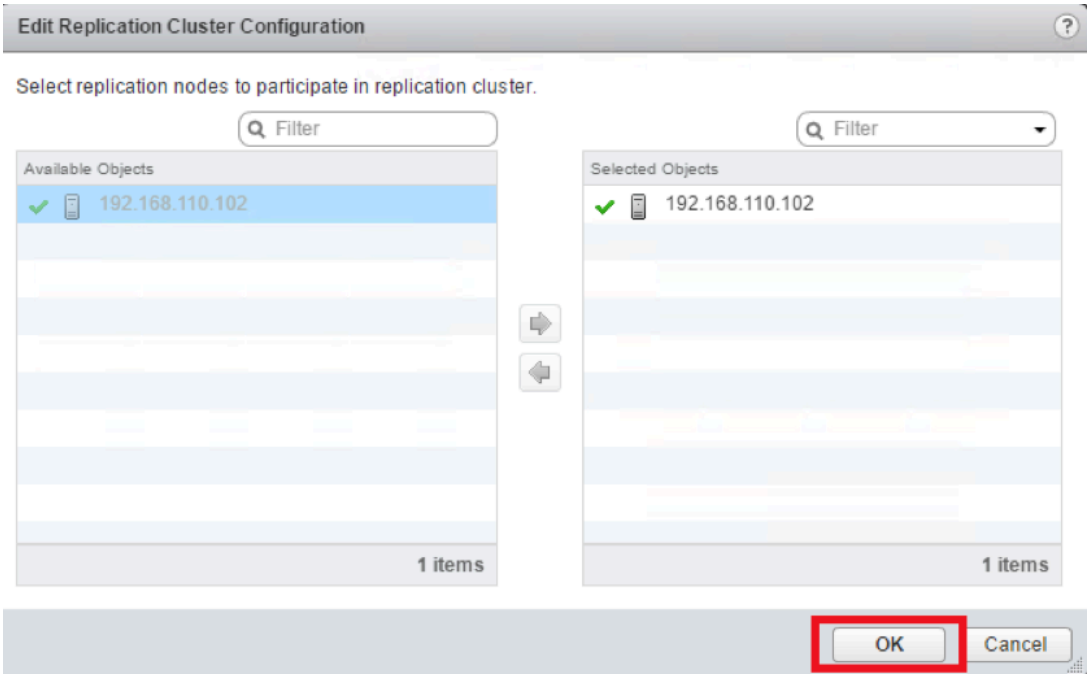
To configure the Replication Cluster:

1. Select the **Service Definitions** category, then click the **Hardware Devices**

tab. Next to the **Replication Cluster** field, click **Edit**.



2. Hypervisors connected to the NSX controller for replication appear in the **Available Objects** list. Select the required service nodes, then click the green arrow to move them to the **Selected Objects** list.

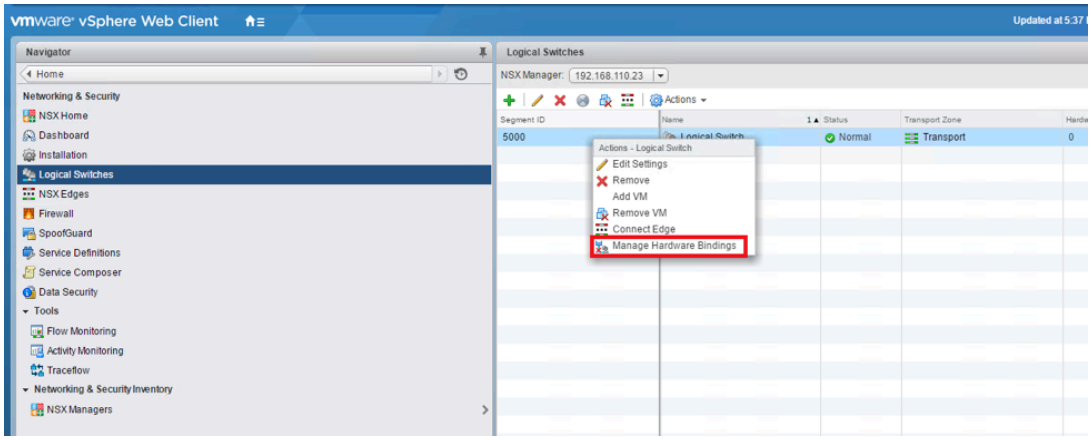


3. Click **OK** to save the replication node configuration.

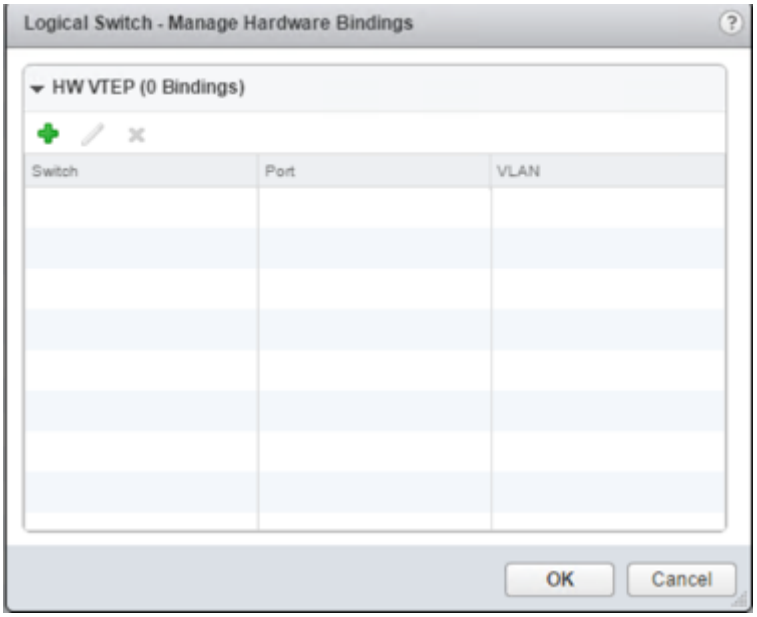
Define Logical Switch Ports

To define the logical switch ports (you can define a VLAN-to-VNI binding for each switch port associated with a particular logical switch):

1. In NSX Manager, add a new logical switch port. Click the **Logical Switches** category. Under **Actions**, click **Manage Hardware Bindings**. The Manage Hardware Binding wizard appears.



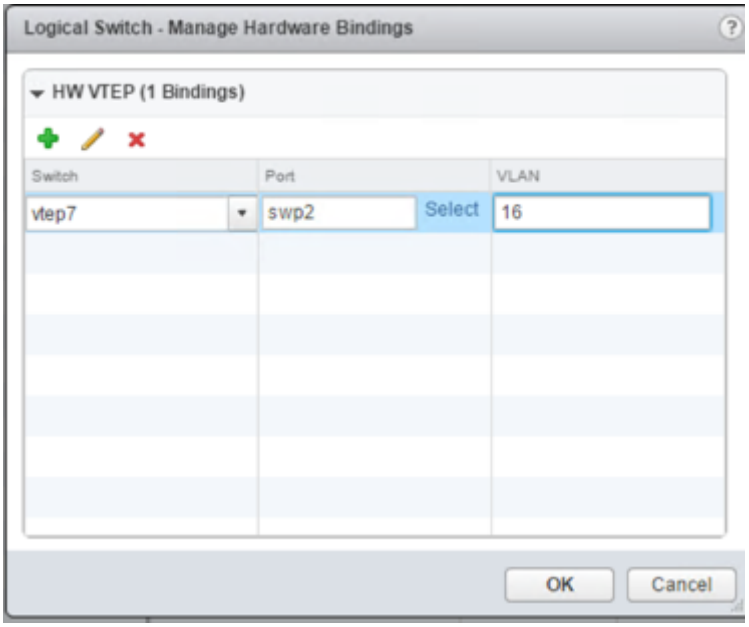
2. Click **+** to add a logical port to the logical switch.



3. Select the logical switch that you created earlier (5000).

4. Select the switch port and the corresponding VLAN binding for logical switch 5000. This creates the logical switch port and also maps VLAN 16 of switch port swp2 to VNI 5000.

5. Click **OK** to save the logical switch port. Connectivity is established. Repeat this procedure for each logical switch port you want to define.



Verify the VXLAN Configuration

After configuration is complete, you can verify the VXLAN configuration using either or both of these Cumulus Linux commands in a terminal connected to the switch:

```
cumulus@switch:/var/lib/openvswitch$ ip -d link show vxln5000
65: vxln5000: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9152 qdisc
noqueue master br-vxln5000 state UNKNOWN mode DEFAULT group
default
    link/ether da:d1:23:44:c4:5e brd ff:ff:ff:ff:ff:ff
promiscuity 1
    vxlan id 5000 local 172.16.20.157 srcport 0 0 dstport 4789
```

```
ageing 300
    bridge_slave state forwarding priority 8 cost 100 hairpin
off guard off root_block off fastleave off learning on flood on
port_id 0x8006 port_no 0x6 designated_port 32774
designated_cost 0 designated_bridge 8000.16:28:56:cc:97:e5
designated_root 8000.16:28:56:cc:97:e5 hold_timer    0.00
message_age_timer    0.00 forward_delay_timer    0.00
topology_change_ack 0 config_pending 0 proxy_arp off
proxy_arp_wifi off mcast_router 1 mcast_fast_leave off
mcast_flood on neigh_suppress off addrgenmode eui64
```

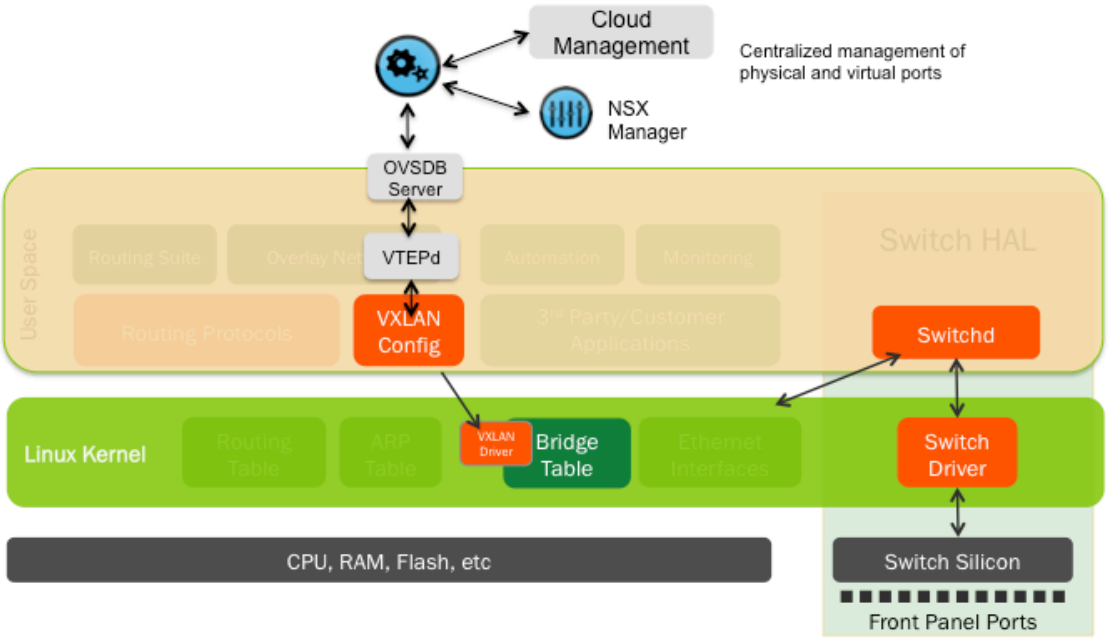
```
cumulus@switch:/var/lib/openvswitch$ bridge fdb show
b6:fb:be:89:99:65 dev vxln5000 master br-vxln5000 permanent
00:50:56:b5:3f:d2 dev vxln5000 master br-vxln5000 static
00:00:00:00:00:00 dev vxln5000 dst 172.16.1.11 self permanent
00:50:56:b5:3f:d2 dev vxln5000 dst 172.16.1.11 self static
36:cc:7a:bc:b9:e1 dev vxln0 master br-vxln0 permanent
00:23:20:00:00:01 dev dummy0 master br-vxln0 permanent
00:23:20:00:00:01 dev dummy 5000 master br-vxln5000 permanent
7c:fe:90:0b:c5:7e dev swp2.16 master br-vxln5000 permanent
```

To check that the active OVSDB server is connected to the NSX controller, run the `ovsdb-client dump Manager` command:

```
cumulus@switch:~$ sudo ovsdb-client dump Manager
Manager table
_uuid                                inactivity_probe
is_connected max_backoff other_config
status                                 target
------------------------------------ ----------------
------------ ----------- ------------
---------------------------------------- -------------------
e700ad21-8fd8-4f09-96dc-fa7cc6e498d8 30000
true         []          {}           {sec_since_connect="68 ",
state=ACTIVE} "ssl:54.0.0.2:6632"
```

Integrating Hardware VTEPs with VMware NSX-MH

Switches running Cumulus Linux can integrate with VMware NSX Multi-Hypervisor (MH) to act as hardware VTEP gateways. The VMware NSX-MH controller provides consistent provisioning across virtual and physical server infrastructures.



Cumulus Linux also supports integration with VMware NSX in high availability mode. Refer to OVSDB Server High Availability.

Get Started

Before you integrate VXLANs with NSX-MH, make sure you have a layer 2

gateway; a switch with Broadcom Tomahawk, Trident II+, Trident II, or Trident III, Maverick, or Mellanox Spectrum or Spectrum A1 running Cumulus Linux. Cumulus Linux includes OVSDB server (`ovsdb-server`) and VTEPd (`ovs-vtepd`), which support VLAN-aware bridges.

To integrate a VXLAN with NSX-MH, you need to:

- Configure the NSX-MH integration on the switch.
- Configure the transport and logical layers from the NSX Manager.
- Verify the VXLAN configuration.

> **NOTE**
>
> Cumulus Linux supports security protocol version TLSv1.2 for SSL connections between the OVSDB server and the NSX controller. The OVSDB server cannot select the loopback interface as the source IP address, causing top of rack registration to the controller to fail. To work around this issue, run the `net add bgp redistribute connected` command followed by the `net commit` command.

Configure the Switch for NSX-MH Integration

Before you start configuring the gateway service, logical switches, and ports that comprise the VXLAN, you need to enable and start the `openvswitch-vtep` service, and configure the NSX integration on the switch, either using the script or performing the manual configuration.

Start the openvswitch-vtep Service

To enable and start the `openvswitch-vtep` service, run the following command:

```
cumulus@switch:~$ sudo systemctl enable openvswitch-vtep.service
cumulus@switch:~$ sudo systemctl start openvswitch-vtep.service
```

> **NOTE**
>
> In previous versions of Cumulus Linux, you had to edit the `/etc/default/openvswitch-vtep` file and then start the `openvswitch-vtep` service . Now, you just have to enable and start the `openvswitch-vtep` service.

Configure the NSX-MH Integration Using the Configuration Script

A script is available so you can configure the NSX-MH integration on the switch automatically.

In a terminal session connected to the switch, run the `vtep-bootstrap` command with these options:

- `controller_ip` is the IP address of the NSX controller (192.168.100.17 in

the example command below).

- The ID for the VTEP (`vtep7` in the example command below).
- The datapath IP address of the VTEP (`172.16.20.157` in the example command below). This is the VXLAN anycast IP address.
- The IP address of the management interface on the switch (`192.168.100.157` in the example command below). This interface is used for control traffic.

```
cumulus@switch:~$ vtep-bootstrap --controller_ip 192.168.100.17
vtep7 172.16.20.157 192.168.100.157
Executed:
       create certificate on a switch, to be used for
authentication with controller
        ().
Executed:
       sign certificate
        (vtep-req.pem    Tue Sep 11 21:11:27 UTC 2018
        fingerprint a4cda030fe5e458c0d7ba44e22f52650f01bcd75).
Executed:
       deine physical switch
          ().
Executed:
       define NSX controller IP address in OVSDB
        ().
```

```
Executed:
       define local tunnel IP address on the switch
        ().
Executed:
       define management IP address on the switch
        ().
Executed:
       restart a service
        ().
```

Run the following commands in the order shown to complete the configuration process:

```
cumulus@switch:~$ sudo systemctl restart openvswitch-
vtep.service
cumulus@switch:~$ sudo ifreload -a
cumulus@switch:~$ sudo systemctl restart networking.service
```

Configure the NSX-MH Integration Manually

NOTE

You can configure the NSX-V integration manually for standalone mode only; manual configuration for OVSDB server high availability is not supported.

If you do *not* want to use the configuration script to configure the NSX-MH integration on the switch automatically, you can configure the integration manually, which requires you to perform the following steps:

- Generate a certificate and key pair for authentication by NSX.
- Configure the switch as a VTEP gateway.

Generate the Credentials Certificate

In Cumulus Linux, generate a certificate that the NSX controller uses for authentication.

1. In a terminal session connected to the switch, run the following commands:

```
cumulus@switch:~$ sudo ovs-pki init
Creating controllerca...
Creating switchca...
cumulus@switch:~$ sudo ovs-pki req+sign cumulus
```

```
cumulus-req.pem Wed Oct 23 05:32:49 UTC 2013
            fingerprint
b587c9fe36f09fb371750ab50c430485d33a174a

cumulus@switch:~$ ls -l
total 12
-rw-r--r-- 1 root root 4028 Oct 23 05:32 cumulus-cert.pem
-rw------- 1 root root 1679 Oct 23 05:32 cumulus-privkey.pem
-rw-r--r-- 1 root root 3585 Oct 23 05:32 cumulus-req.pem
```

2. In the `/usr/share/openvswitch/scripts/ovs-ctl-vtep` file, make sure the lines containing **private-key**, **certificate**, and **bootstrap-ca-cert** point to the correct files **bootstrap-ca-cert** is obtained dynamically the first time the switch talks to the controller:

```
# Start ovsdb-server.
set ovsdb-server "$DB_FILE"
set "$@" -vANY:CONSOLE:EMER -vANY:SYSLOG:ERR -vANY:FILE:INFO
set "$@" --remote=punix:"$DB_SOCK"
set "$@" --remote=db:Global,managers
set "$@" --remote=ptcp:6633:$LOCALIP
set "$@" --private-key=/root/cumulus-privkey.pem
```

```
set "$@" --certificate=/root/cumulus-cert.pem
set "$@" --bootstrap-ca-cert=/root/controller.cacert
```

If files have been moved or regenerated, restart the OVSDB server and VTEPd:

```
cumulus@switch:~$ sudo systemctl restart openvswitch-
vtep.service
```

3. Define the NSX controller cluster IP address in OVSDB. This causes the OVSDB server to start contacting the NSX controller:

```
cumulus@switch:~$ sudo vtep-ctl set-manager
ssl:192.168.100.17:6632
```

4. Define the local IP address on the VTEP for VXLAN tunnel termination. First, find the physical switch name as recorded in OVSDB:

```
cumulus@switch:~$ sudo vtep-ctl list-ps
vtep7
```

Then set the tunnel source IP address of the VTEP. This is the datapath address of the VTEP, which is typically an address on a loopback interface on the switch that is reachable from the underlying layer 3 network:

```
cumulus@switch:~$ sudo vtep-ctl set Physical_Switch vtep7
tunnel_ips=172.16.20.157
```

After you generate the certificate, keep the terminal session active; you need to paste the certificate into NSX Manager when you configure the VTEP gateway.

Enable ovs-vtepd to Use the VLAN-aware Bridge

By default, in stand-alone mode, the ovs-vtep daemon creates traditional bridges for each VXLAN VTEP. To use the VLAN-aware bridge with the VTEPs, edit the `/usr/share/openvswitch/scripts/ovs-ctl-vtep` file and uncomment the `--enable-vlan-aware-mode` line:

```
# Start ovs-vtepd
set ovs-vtepd unix:"$DB_SOCK "
set "$@ " -vconsole:emer -vsyslog:err -vfile:info
#set "$@ " --enable-vlan-aware-mode
```

Then restart the OVSDB server and VTEPd:

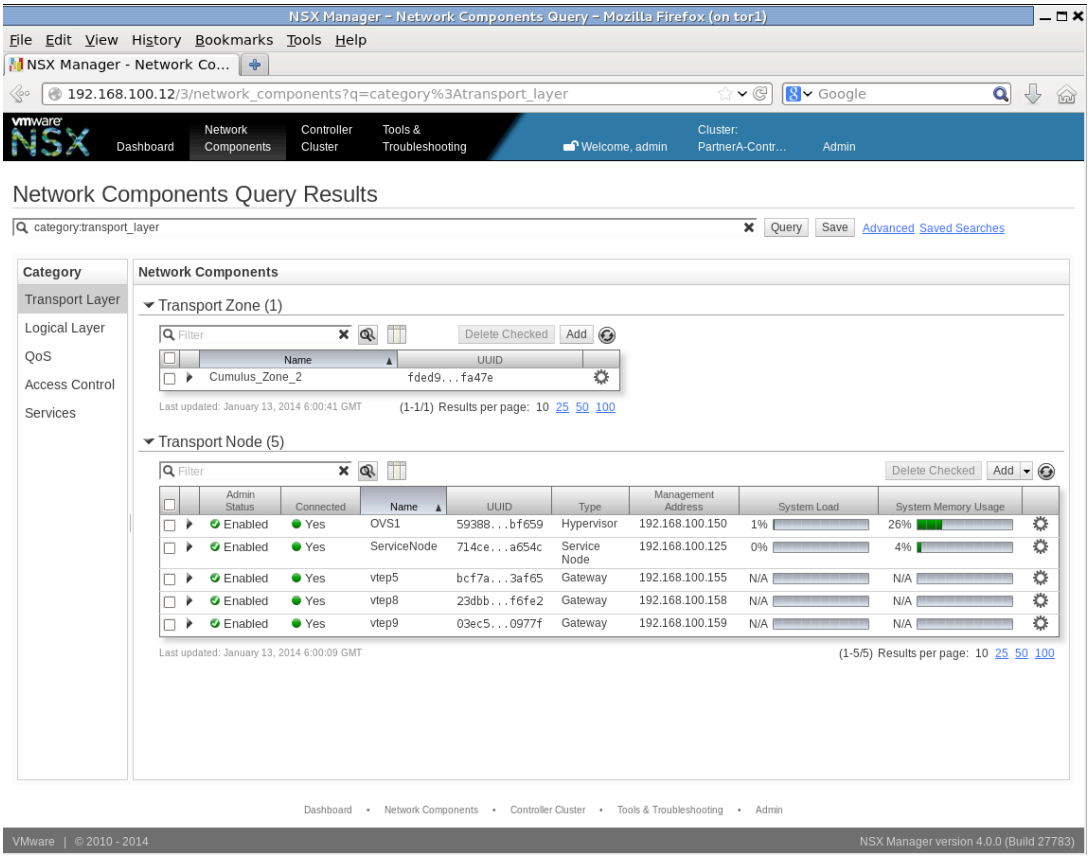
```
cumulus@switch:~$ sudo systemctl restart openvswitch-
vtep.service
```

Provision VMware NSX-MH

Configure the Switch as a VTEP Gateway

After you create a certificate, connect to NSX Manager in a browser to configure a Cumulus Linux switch as a VTEP gateway. In this example, the IP address of the NSX Manager is 192.168.100.12.

1. In NSX Manager, add a new gateway. Click the **Network Components** tab, then the **Transport Layer** category. Under **Transport Node**, click **Add**, then select **Manually Enter All Fields**. The Create Gateway wizard opens.



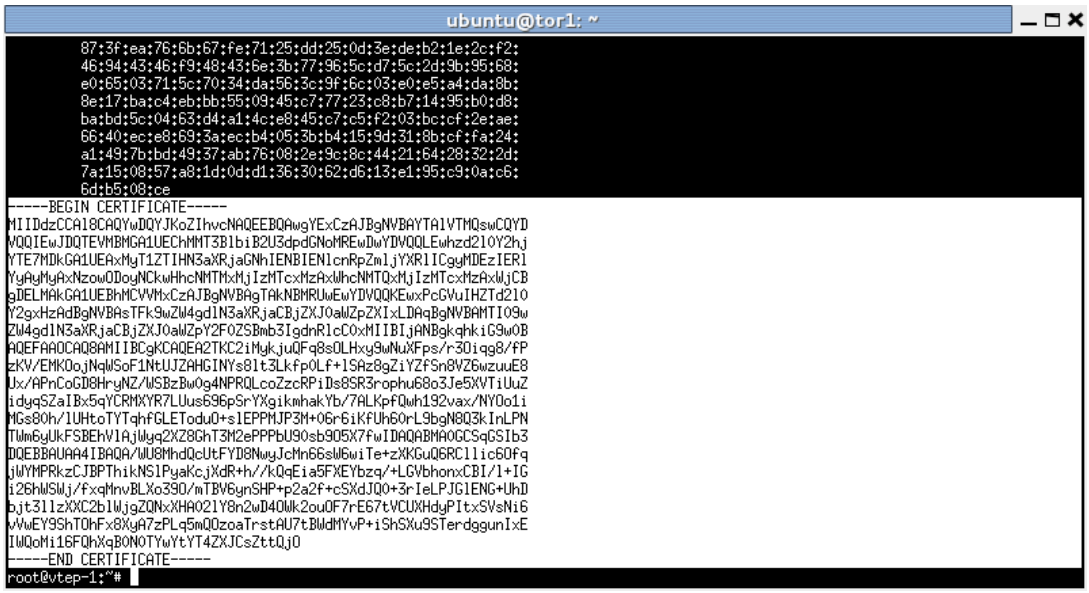
2. In the Create Gateway dialog, select *Gateway* for the **Transport Node Type**, then click **Next**.

3. In the **Display Name** field, provide a name for the gateway, then click **Next**.

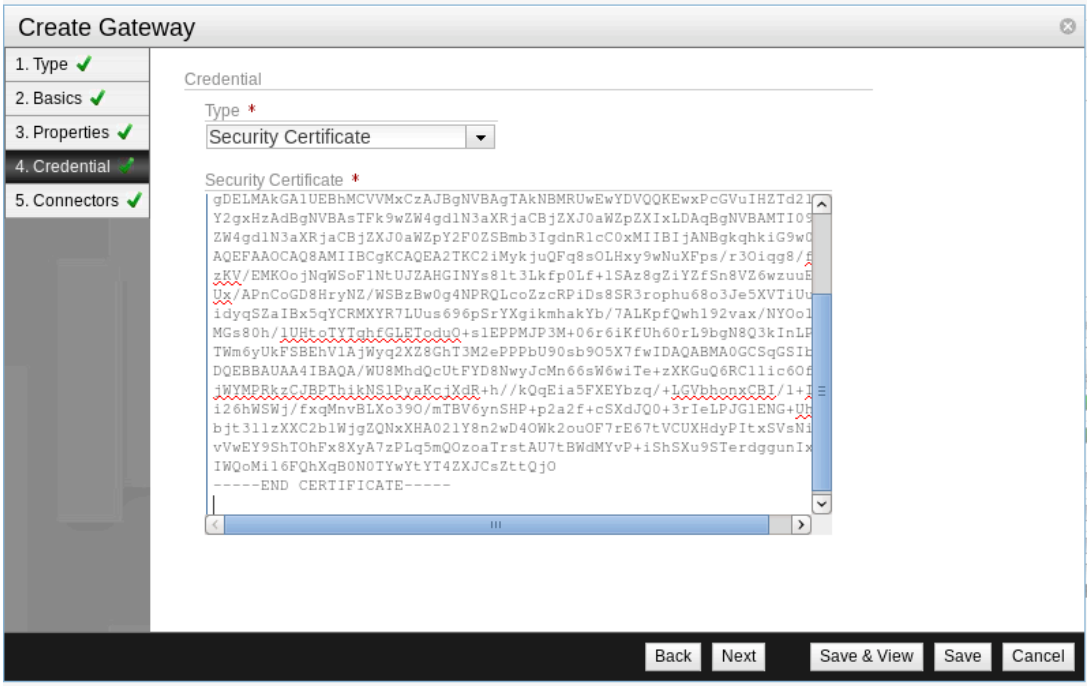
4. Enable the VTEP service. Select the **VTEP Enabled** checkbox, then click **Next**.

5. From the terminal session connected to the switch where you generated the certificate, copy the certificate and paste it into the **Security Certificate** text field. Copy only the bottom portion, including the `BEGIN`

`CERTIFICATE` and `END CERTIFICATE` lines. For example, copy all the highlighted text in the terminal:



Paste it into NSX Manager, then click **Next**:



6. In the Connectors dialog, click **Add Connector** to add a transport connector. This defines the tunnel endpoint that terminates the VXLAN tunnel and connects NSX to the physical gateway. You must choose a tunnel **Transport Type** of *VXLAN*. Choose an existing transport zone for the connector or click **Create** to create a new transport zone.

7. Define the IP address of the connector (the underlay IP address on the switch for tunnel termination).

8. Click **OK** to save the connector, then click **Save** to save the gateway.

After communication is established between the switch and the controller, a `controller.cacert` file downloads onto the switch.

Verify that the controller and switch handshake is successful. In a terminal

connected to the switch, run this command:

```
cumulus@switch:~$ sudo ovsdb-client dump -f list | grep -A 7
"Manager"
Manager table
_uuid               : 505f32af-9acb-4182-a315-022e405aa479
inactivity_probe    : 30000
is_connected        : true
max_backoff         : []
other_config        : {}
status              : {sec_since_connect="18223",
sec_since_disconnect="18225", state=ACTIVE}
target              : "ssl:192.168.100.17:6632"
```

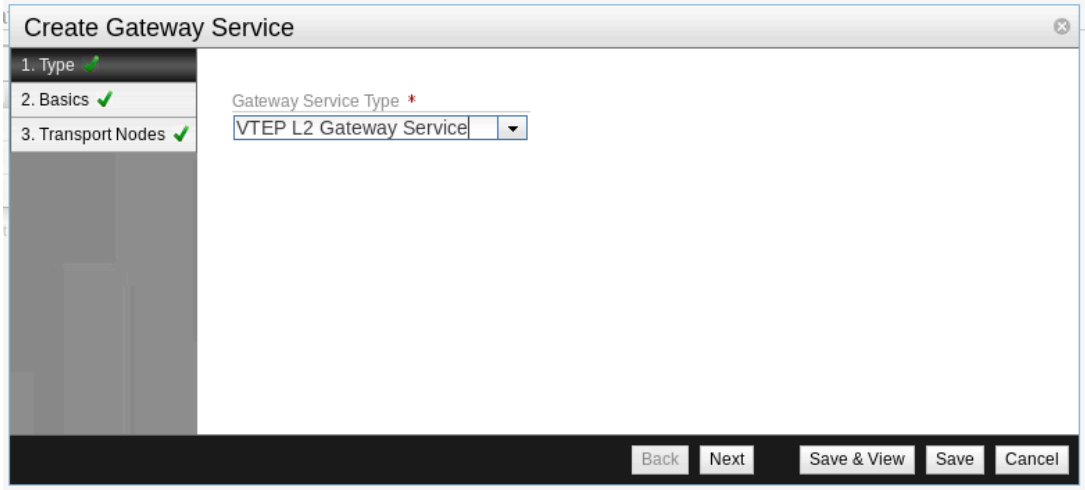
Configure the Transport and Logical Layers

Configure the Transport Layer

After you finish configuring the NSX-MH integration on the switch, configure the transport layer. For each host-facing switch port to be associated with a VXLAN instance, define a **Gateway Service** for the port.

1. In NSX Manager, add a new gateway service. Click the **Network Components** tab, then the **Services** category. Under **Gateway Service**, click **Add**. The Create Gateway Service wizard opens.

2. In the Create Gateway Service dialog, select *VTEP L2 Gateway Service* as the **Gateway Service Type**.



3. Provide a **Display Name** for the service to represent the VTEP in NSX.

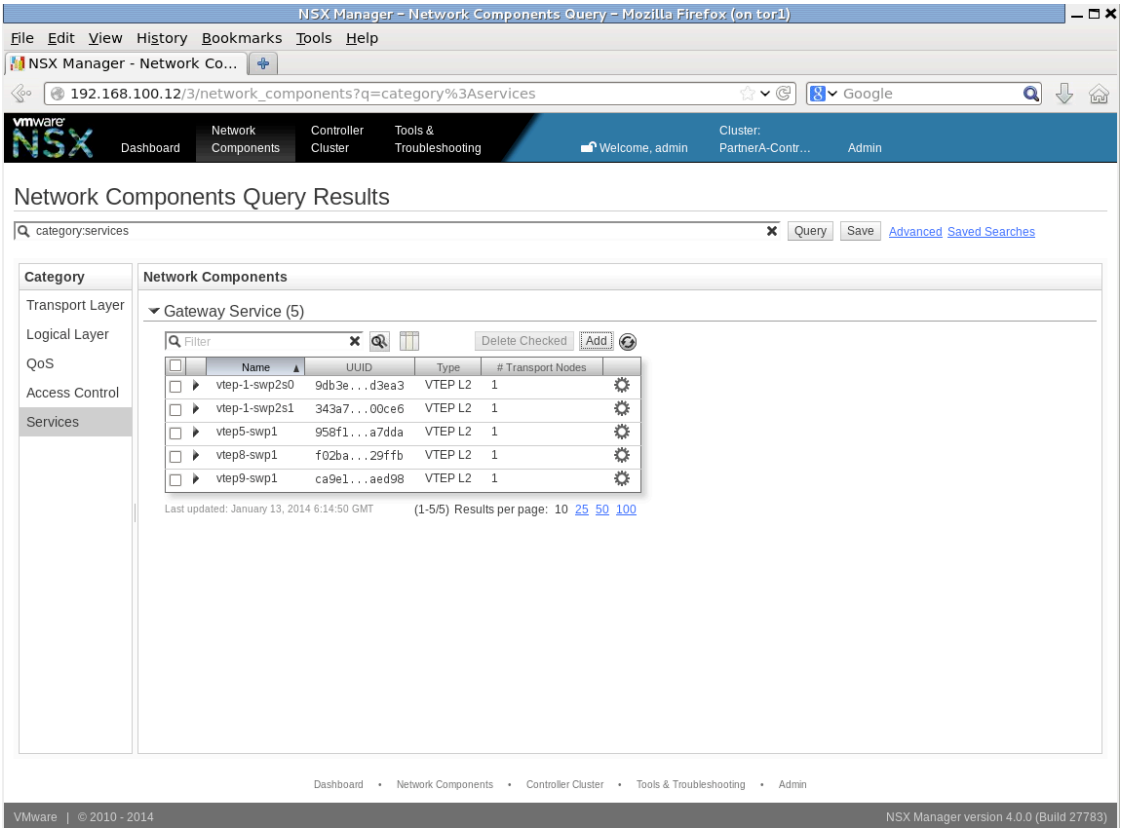
4. Click **Add Gateway** to associate the service with the gateway you created earlier.

5. In the **Transport Node** field, choose the name of the gateway you created earlier.

6. In the **Port ID** field, choose the physical port on the gateway (for example, swp10) that will connect to a logical layer 2 segment and carry data traffic.

7. Click **OK** to save this gateway in the service, then click **Save** to save the gateway service.

The gateway service displays as type *VTEP L2* in NSX.



Next, configure the logical layer on NSX.

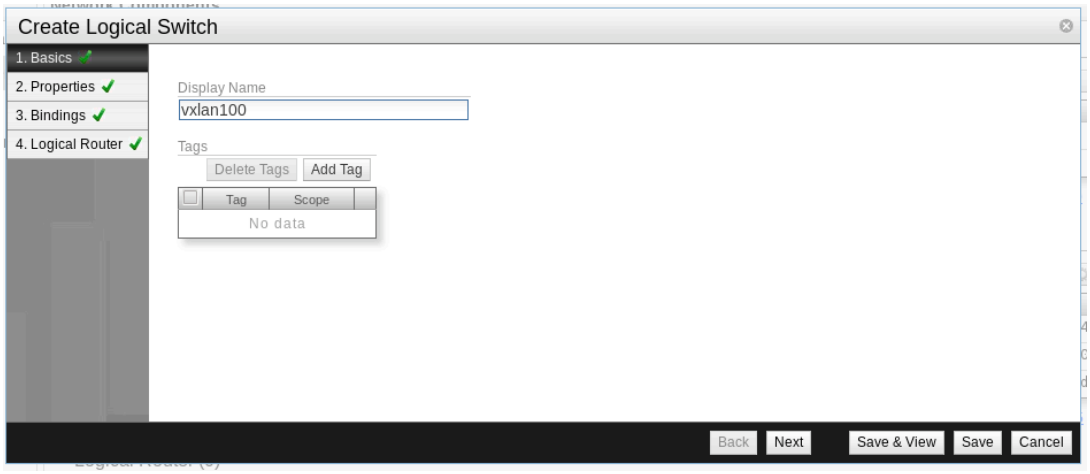
Configure the Logical Layer

To complete the integration with NSX, you need to configure the logical layer, which requires defining a logical switch (the VXLAN instance) and all the logical ports needed.

To define the logical switch:

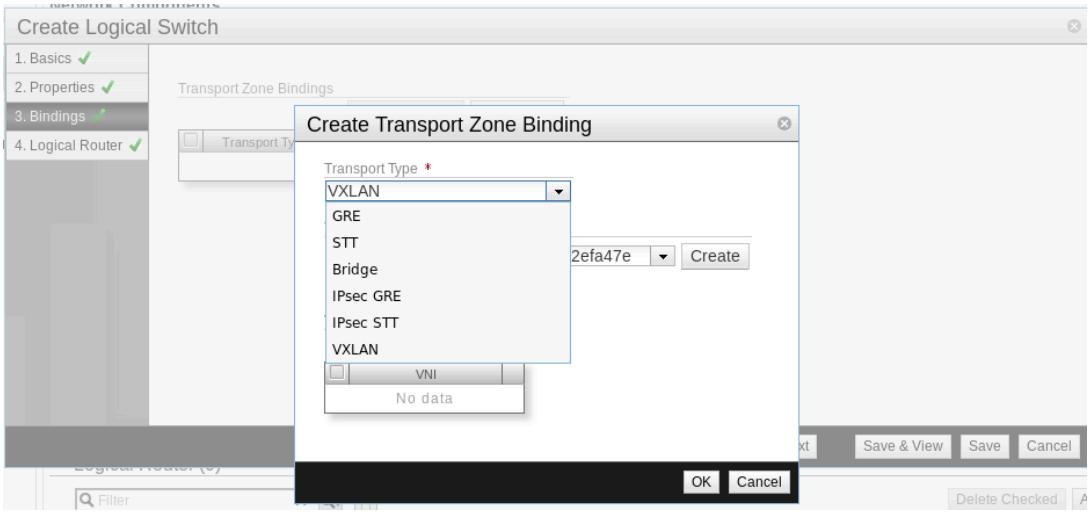
1. In NSX Manager, add a new logical switch. Click the **Network Components** tab, then the **Logical Layer** category. Under **Logical Switch**, click **Add**. The Create Logical Switch wizard opens.

2. In the **Display Name** field, enter a name for the logical switch, then click **Next**.



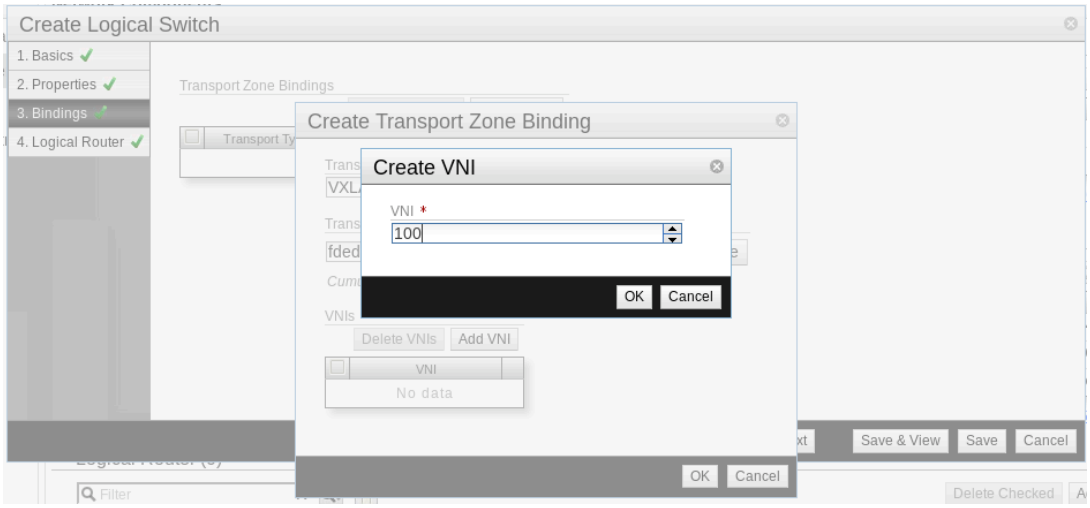
3. Under **Replication Mode**, select **Service Nodes**, then click **Next**.

4. Specify the transport zone bindings for the logical switch. Click **Add Binding**. The Create Transport Zone Binding dialog opens.



5. In the **Transport Type** list, select *VXLAN*, then click **OK** to add the

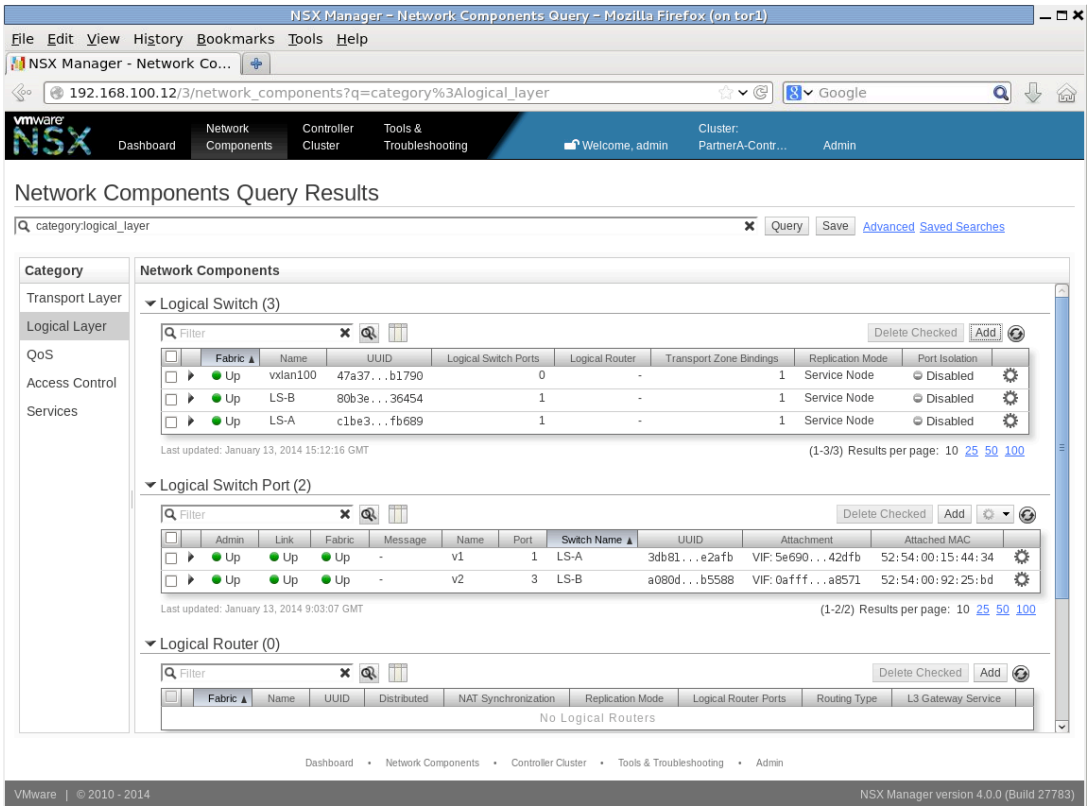
binding to the logical switch.



6. In the **VNI** field, assign the switch a VNI ID, then click **OK**.

> **NOTE**
>
> Do not use 0 or 16777215 as the VNI ID; these are reserved values under Cumulus Linux.

7. Click **Save** to save the logical switch configuration.

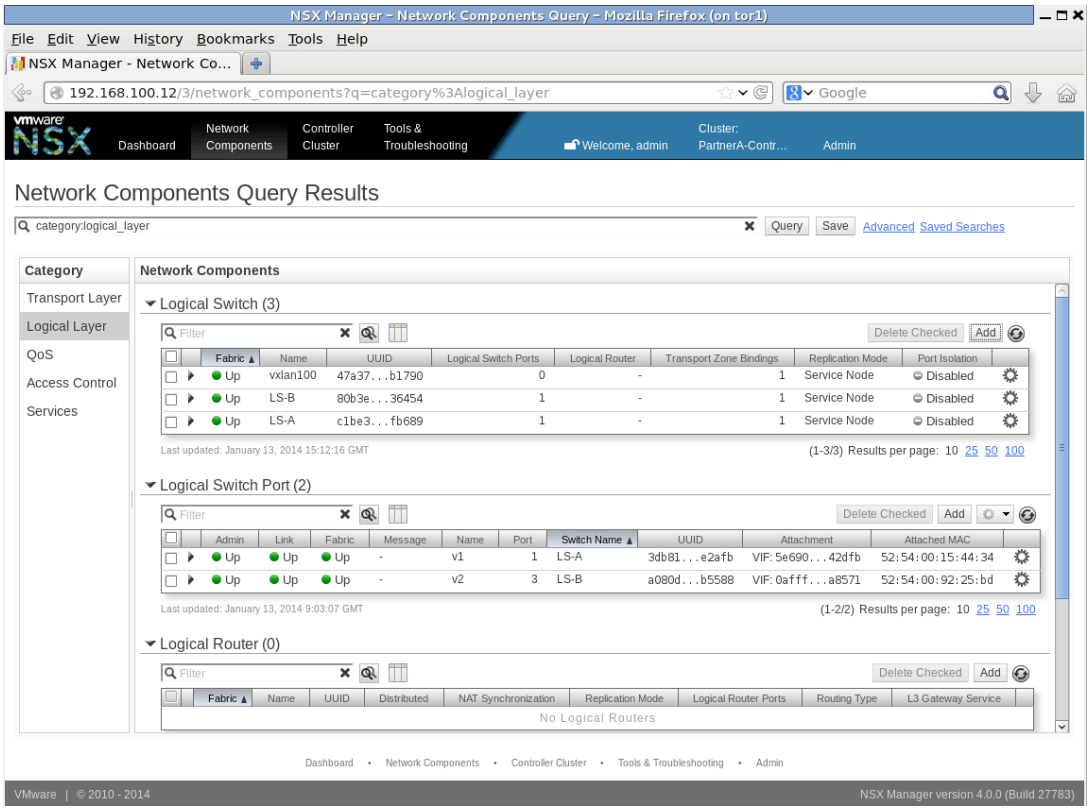


Define Logical Switch Ports

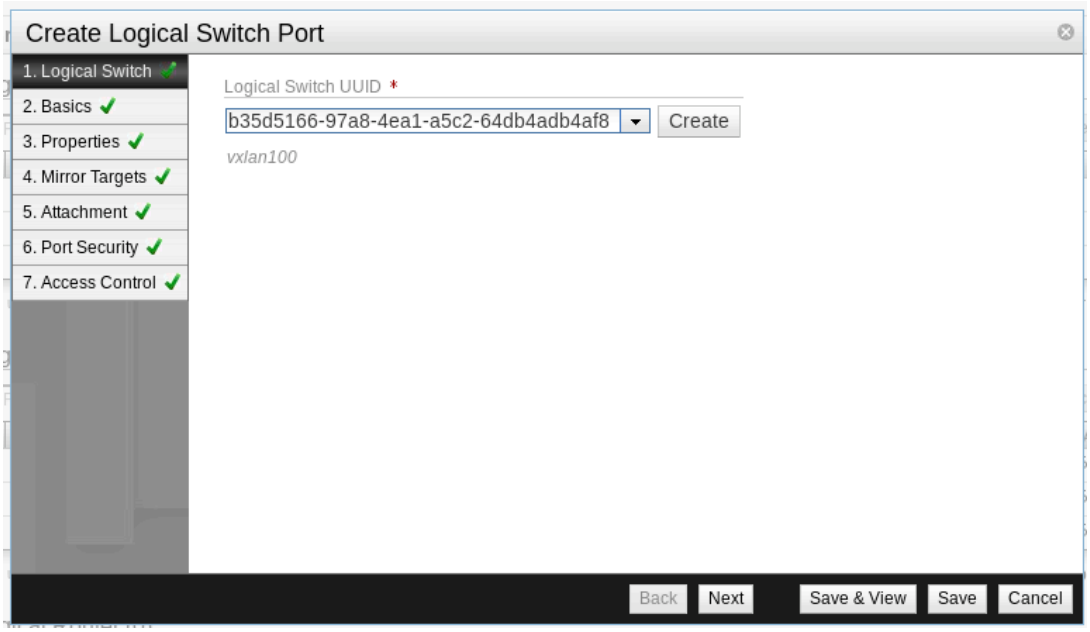
Logical switch ports can be virtual machine VIF interfaces from a registered OVS or a VTEP gateway service instance on this switch, as defined above in the Configuring the Transport Layer. You can define a VLAN binding for each VTEP gateway service associated with the particular logical switch.

To define the logical switch ports:

1. In NSX Manager, add a new logical switch port. Click the **Network Components** tab, then the **Logical Layer** category. Under **Logical Switch Port**, click **Add**. The Create Logical Switch Port wizard opens.



2. In the **Logical Switch UUID** list, select the logical switch you created above, then click **Create**.



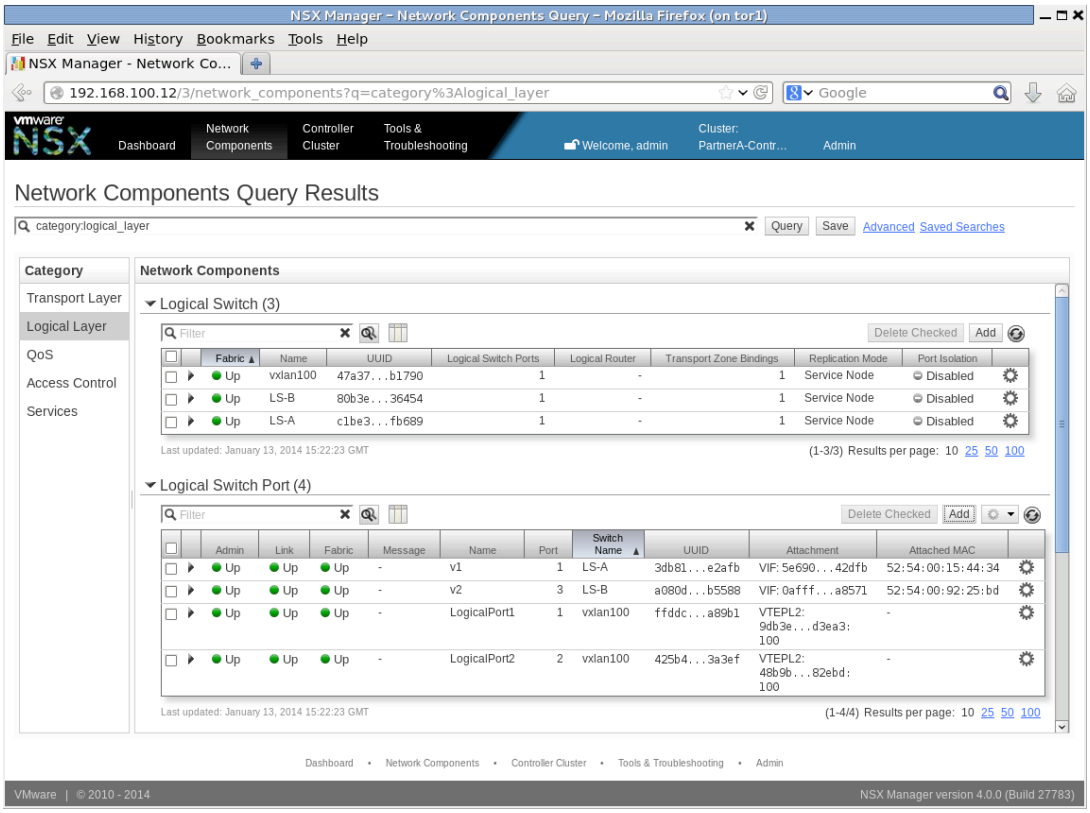
3. In the **Display Name** field, provide a name for the port that indicates it is the port that connects the gateway, then click **Next**.

4. In the **Attachment Type** list, select *VTEP L2 Gateway*.

5. In the **VTEP L2 Gateway Service UUID** list, choose the name of the gateway service you created earlier.

6. In the **VLAN** list, you can choose a VLAN if you want to connect only traffic on a specific VLAN of the physical network. Leave it blank to handle all traffic.

7. Click **Save** to save the logical switch port. Connectivity is established. Repeat this procedure for each logical switch port you want to define.



Verify the VXLAN Configuration

After configuration is complete, verify the VXLAN configuration in a terminal connected to the switch using these Cumulus Linux commands:

```
cumulus@switch1:~$ sudo ip -d link show vxln100
71: vxln100: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
noqueue master br-vxln100 state UNKNOWN mode DEFAULT
    link/ether d2:ca:78:bb:7c:9b brd ff:ff:ff:ff:ff:ff
    vxlan id 100 local 172.16.20.157 port 32768 61000
```

```
nolearning ageing 1800 svcnode 172.16.21.125
```

or

```
cumulus@switch1:~$ sudo bridge fdb show
52:54:00:ae:2a:e0 dev vxln100 dst 172.16.21.150 self permanent
d2:ca:78:bb:7c:9b dev vxln100 permanent
90:e2:ba:3f:ce:34 dev swp2s1.100
90:e2:ba:3f:ce:35 dev swp2s0.100
44:38:39:00:48:0e dev swp2s1.100 permanent
44:38:39:00:48:0d dev swp2s0.100 permanent
```

Use the `ovsdb-client dump` command to troubleshoot issues on the switch. This command verifies that the controller and switch handshake is successful (and works only for VXLANs integrated with NSX):

```
cumulus@switch:~$ sudo ovsdb-client dump -f list | grep -A 7
"Manager"
Manager table
_uuid               : 505f32af-9acb-4182-a315-022e405aa479
inactivity_probe    : 30000
```

```
is_connected        : true
max_backoff         : []
other_config        : {}
status              : {sec_since_connect="18223",
sec_since_disconnect="18225", state=ACTIVE}
target              : "ssl:192.168.100.17:6632"
```

OVSDB Server High Availability

WARNING

OVSDB server high availability is an early access feature.

Cumulus Linux supports integration with VMware NSX in both *standalonemode* and *OVSDB server high availability mode* (where the data plane is running in active-active mode). For information about VMware NSX in standalone mode and for a description of the components that work together to integrate VMware NSX and Cumulus Linux, see Integrating Hardware VTEPs with VMware NSX-MH or Integrating Hardware VTEPs with VMware NSX-V.

NOTE

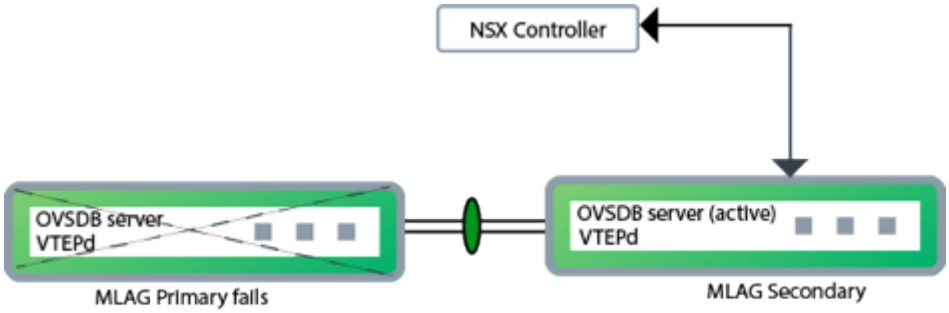
Cumulus Linux supports OVSDB service node replication only.

With OVSDB server high availability mode, you use two peer Cumulus Linux switches in an MLAG configuration. Both the MLAG primary and MLAG

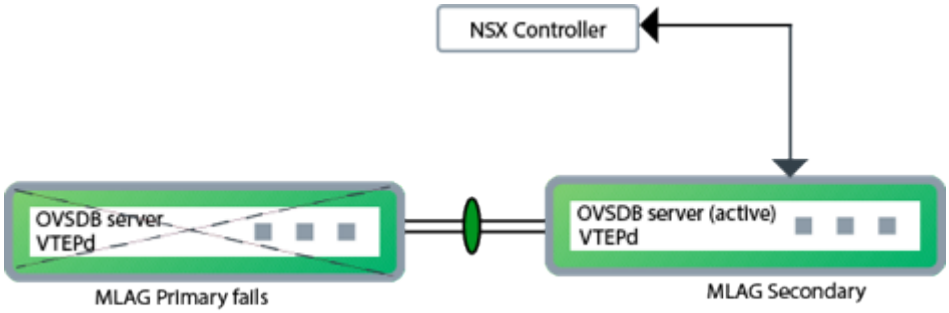
secondary switch contain OVSDB server and VTEPd. The OVSDB servers synchronize their databases with each other and always maintain the replicated state unless failover occurs; for example, the peer link bond breaks, a switch fails, or the OVSDB server goes down. Both of the VTEPd components talk to the active OVSDB server to read the configuration and then push the configuration to the kernel. Only the active OVSDB server communicates with the NSX controller, unless failover occurs and then the standby OVSDB server takes over automatically. Although the Cumulus switches are configured as an MLAG pair, the NSX controller sees them as a single system (the NSX controller is not aware that multiple switches exist).

The following examples show OVSDB server high availability mode.

Example 1: The OVSDB server on the MLAG primary switch is active. The OVSDB server on the MLAG secondary switch is the hot standby. Only the active OVSDB server communicates with the NSX controller.



Example 2: If failover occurs, the OVSDB server on the MLAG secondary switch becomes the active OVSDB server and communicates with the NSX controller.



When the OVSDB server on the MLAG primary switch starts responding again, it resynchronizes its database, becomes the active OVSDB server, and connects to the controller. At the same time, the OVSDB server on the MLAG secondary switch stops communicating with the NSX controller, synchronizes with the now active OVSDB server, and takes the standby role again.

> **NOTE**
>
> When you upgrade Cumulus Linux, both the `/usr/share/openvswitch/scripts/ovs-ctl-vtep` file and the database file `conf.db` are overwritten. Be sure to back up both files before upgrading.

Getting Started

Before you configure OVSDB server high availability, make sure you have **two switches running Cumulus Linux in an MLAG configuration**. Cumulus Linux includes OVSDB server (`ovsdb-server`) and VTEPd (`ovs-vtepd`), which support VLAN-aware bridges.

The following example configuration in the `/etc/network/interfaces` file shows the *minimum* MLAG configuration required (the MLAG peerlink configuration and the dual-connected bonds on the peer switches). The dual-connected bonds are identified in the NSX controller by their `clag-id` (single-connected bonds or ports are identified by their usual interface names prepended with the name of the particular switch to which they belong). When you create the Gateway Service for the dual-connected bonds (described in Configuring the Transport and Logical Layers, below), make sure to select the `clag-id` named interfaces instead of the underlying individual physical ports. All the logical network configurations are provisioned by the NSX controller.

```
auto peerlink-3
iface peerlink-3
    bond-slaves swp5 swp6
    bond-mode 802.3ad
    bond-min-links 1
    bond-lacp-rate 1
    mtu  9202
    alias Local Node/s leaf01 and Ports swp5 swp6 <==> Remote
Node/s leaf02 and Ports swp5 swp6

auto peerlink-3.4094
iface peerlink-3.4094
    address 10.0.0.24/32
```

```
    address 169.254.0.9/29
    mtu 9202
    alias clag and vxlan communication primary path
    clagd-priority 4096
    clagd-sys-mac 44:38:39:ff:ff:02
    clagd-peer-ip 169.254.0.10
    clagd-args --vm --debug 0x0
    # post-up sysctl -w net.ipv4.conf.peerlink-3/
4094.accept_local=1
    clagd-backup-ip 10.0.0.25

auto hostbond4
iface hostbond4
    bond-slaves swp7
    bond-mode 802.3ad
    bond-min-links 1
    bond-lacp-rate 1
    mtu  9152
    alias Local Node/s leaf01 and Ports swp7 <==> Remote  Node/
s hostd-01 and Ports swp1
    clag-id 1

auto hostbond5
iface hostbond5
```

```
    bond-slaves swp8
    bond-mode 802.3ad
    bond-min-links 1
    bond-lacp-rate 1
    mtu  9152
    alias Local Node/s leaf01 and Ports swp8 <==> Remote  Node/
s hostd-02 and Ports swp1
    clag-id 2
```

To configure OVSDB server high availability, you need to:

- Determine on which switch you want to run the active OVSDB server (the MLAG primary switch or the MLAG secondary switch).
- Configure the NSX integration on both switches.
- Configure the Transport and Logical Layers from the NSX Manager.
- Verify the VXLAN Configuration.

NOTE

The OVSDB server cannot select the loopback interface as the source IP address, causing top of rack registration to the controller to fail. To work around this issue, run the `net add bgp redistribute connected` command followed by the `net commit` command.

Configure the NSX Integration on the Switch

Before you start configuring the gateway service, the logical switches, and ports that comprise the VXLAN, you need to enable and start the `openvswitch-vtep` service, then run the configuration script **on both the MLAG primary and MLAG secondary switches**. Follow these steps:

Enable and start the `openvswitch-vtep` service:

```
cumulus@switch:~$ sudo systemctl enable openvswitch-vtep.service
cumulus@switch:~$ sudo systemctl start openvswitch-vtep.service
```

Run the configuration script provided with Cumulus Linux:

1. On the switch where you want to run the active OVSDB server, run the `vtep-bootstrap` command with these options:

   - `db_ha active` specifies that the OVSDB server on this switch is the *active* server.
   - `db_ha_vip` is any unused IP address in the subnet used by the peerlink control subinterface (4094 is typically used). This creates a /32 route that can be reached from either MLAG switch (`169.254.0.11:9998` in the example below).
   - `db_ha_repl_sv` specifies the IP address of the **active** OVSDB server

(`169.254.0.9:9999` in the example command below). The standby OVSDB server uses this IP address to synchronize the database.

- `controller_port` is the port used to communicate with the NSX controller.
- `controller_ip` is the IP address of the NSX controller (192.168.100.17 in the example command below).
- The ID for the VTEP (`vtep7` in the example command below).
- The datapath IP address of the VTEP (`172.16.20.157` in the example command below). This is the VXLAN anycast IP address.
- The IP address of the management interface on the switch (`192.168.100.157` in the example command below). This interface is used for control traffic.

```
cumulus@switch:~$ vtep-bootstrap  --db_ha active  --
db_ha_vip 169.254.0.11:9998  --db_ha_repl_sv
169.254.0.9:9999 --controller_ip 192.168.100.17 vtep7
172.16.20.157 192.168.100.157
Executed:
          create certificate on a switch, to be used for
authentication with controller
            ().
```

```
Executed:
          sign certificate
            (vtep7-req.pem   Tue Sep 11 21:11:27 UTC 2018
              fingerprint
a4cda030fe5e458c0d7ba44e22f52650f01bcd75).
Executed:
          define physical switch
           ().
Executed:
          define NSX controller IP address in OVSDB
            ().
Executed:
          define local tunnel IP address on the switch
            ().
Executed:
          define management IP address on the switch
            ().
Executed:
          restart a service
           ().
```

2. On the switch where you want to run the standby OVSDB server, run `vtep-bootstrap` command with the same options as above but replace `db_ha active` with `db_ha standby`:

```
cumulus@switch:~$ vtep-bootstrap  --db_ha standby  --
db_ha_vip 169.254.0.11:9998  --db_ha_repl_sv 169.254.0.9:9999
--controller_ip 192.168.100.17 vtep7 172.16.20.157
192.168.100.157
Executed:
        create certificate on a switch, to be used for
authentication with controller
          ().
Executed:
        sign certificate
          (vtep7-req.pem   Tue Sep 11 21:11:27 UTC 2018
            fingerprint
a4cda030fe5e458c0d7ba44e22f52650f01bcd75).
Executed:
        define physical switch
          ().
Executed:
        define NSX controller IP address in OVSDB
          ().
Executed:
        define local tunnel IP address on the switch
          ().
Executed:
        define management IP address on the switch
```

```
          ().
Executed:
        restart a service
          ().
```

3. From the switch running the active OVSDB server, copy the certificate files (`hostname-cert.pem` and `hostname-privkey.pem`) to the same location on the switch with the standby OVSDB server.

> **NOTE**
>
> The certificate and key pairs for authenticating with the NSX controller are generated automatically when you run the configuration script and are stored in the `/home/cumulus` directory. The same certificate must be used for both switches.

4. On the switch running the *active* OVSDB server and then the switch running the *standby* OVSDB server, run the following commands in the order shown to complete the configuration process:

```
cumulus@switch:~$ sudo systemctl restart openvswitch-
```

```
vtep.service
cumulus@switch:~$ sudo ifreload -a
cumulus@switch:~$ sudo systemctl restart networking.service
```

For information about the configuration script, read `man vtep-bootstrap` or run the command `vtep-bootstrap --help`.

Configure the Transport and Logical Layers

After you finish configuring the NSX integration on both the MLAG primary and MLAG secondary switch, you need to configure the transport and logical layers from the NSX Manager. Refer to Integrating Hardware VTEPs with VMware NSX-MH or Integrating Hardware VTEPs with VMware NSX-V.

Troubleshooting

After you configure OVSDB server high availability, you can check that configuration is successful.

To check the sync status on the *active* OVSDB server, run the following command:

```
cumulus@switch:~$ sudo ovs-appctl -t /var/run/openvswitch/ovsdb-
```

```
server.`pidof ovsdb-server`.ctl ovsdb-server/sync-status
state: active
```

To check the sync status on the *standby* OVSDB server, run the following command:

```
cumulus@switch:~$ sudo ovs-appctl -t /var/run/openvswitch/ovsdb-
server.`pidof ovsdb-server`.ctl ovsdb-server/sync-status
state: backup
replicating: tcp:169.254.0.9:9999
database: hardware_vtep
```

To check that the active OVSDB server is connected to the NSX controller, run the `ovsdb-client dump Manager` command:

```
cumulus@switch:~$ sudo ovsdb-client dump Manager
Manager table
_uuid                                inactivity_probe
is_connected max_backoff other_config
status                                 target
---------------------------------- ----------------
```

```
------------ ----------- ------------
------------------------------------ -------------------
e700ad21-8fd8-4f09-96dc-fa7cc6e498d8 30000
true         []          {}           {sec_since_connect="68 ",
state=ACTIVE} "ssl:54.0.0.2:6632"
```

To make sure the MLAG configuration is correct, run the `clagctl` command:

```
cumulus@switch:~$ sudo clagctl
```

The following example command output shows that MLAG *is* configured correctly on the active OVSDB server:

```
cumulus@switch:~$ sudo clagctl
The peer is alive
  Our Priority, ID, and Role: 4096 00:02:00:00:00:46 primary
  Peer Priority, ID, and Role: 8192 00:02:00:00:00:4e secondary
  Peer Interface and IP: peerlink-3.4094 169.254.0.10
  VxLAN Anycast IP: 36.0.0.1
  Backup IP: 27.0.0.22 (active)
  System MAC: 44:38:39:ff:ff:01
```

```
CLAG Interfaces
Our Interface Peer Interface CLAG Id Conflicts Proto-Down Reason
---------------- ---------------- ------- --------------------
-----------------
  vxln14567102 vxln14567102 - - -
  vxln14567103 vxln14567103 - - -
```

The following example command output shows that MLAG is *not* configured correctly on the active OVSDB server or that the peer is down:

```
cumulus@switch:~$ sudo clagctl
The peer is not alive
  Our Priority, ID, and Role: 4096 00:02:00:00:00:46 primary
  Peer Interface and IP: peerlink-3.4094 169.254.0.10
  VxLAN Anycast IP: 36.0.0.1
  Backup IP: 27.0.0.22 (inactive)
  System MAC: 44:38:39:ff:ff:01
CLAG Interfaces
Our Interface Peer Interface CLAG Id Conflicts Proto-Down Reason
---------------- ---------------- ------- --------------------
-----------------
  vxln14567102 - - - -
  vxln14567103 - - - -
```

To make sure that the BFD sessions are up and running, run the `ptmctl -b` command:

```
cumulus@switch:~$ sudo ptmctl -b


------------------------------------------------------------
port    peer       state  local     type       diag  vrf
------------------------------------------------------------
vxln0   54.0.0.4   Up     36.0.0.1  singlehop  N/A   N/A
vxln0   54.0.0.3   Up     36.0.0.1  singlehop  N/A   N/A
```

> **NOTE**
>
> When `switchd` and networking services are restarted, the BFD sessions might be interrupted. If you notice that the sessions are down, restart the `openvswitch-vtep`.service.

If you encounter interface or VXLAN bridge configuration issues after adding the hardware bindings, run the `ifreload -a` command to reload all network interfaces.

```
cumulus@switch:~$ sudo ifreload -a
```

If you still encounter issues with high availability after you restart `openvswitch-vtep.service`, run `ifreload -a`, and restart `networking.service`, reboot the switch running the *standby* OVSDB server.

```
cumulus@switch:~$ sudo reboot
```

Layer 3

This section describes layer 3 configuration. Read this section to understand routing protocols and learn how to configure routing on the Cumulus Linux switch.

Routing

Network routing is the process of selecting a path across one or more networks. When the switch receives a packet, it reads the packet headers to find out its intended destination. It then determines where to route the packet based on information in its routing tables, which can be static or dynamic.

Cumulus Linux supports both Static Routing, where you enter routes and specify the next hop manually and dynamic routing such as BGP, and OSP, where you configure a routing protocol on your switch and the routing protocol learns about other routers automatically.

For the number of route table entries supported per platform, see Supported Route Table Entries.

Static Routing

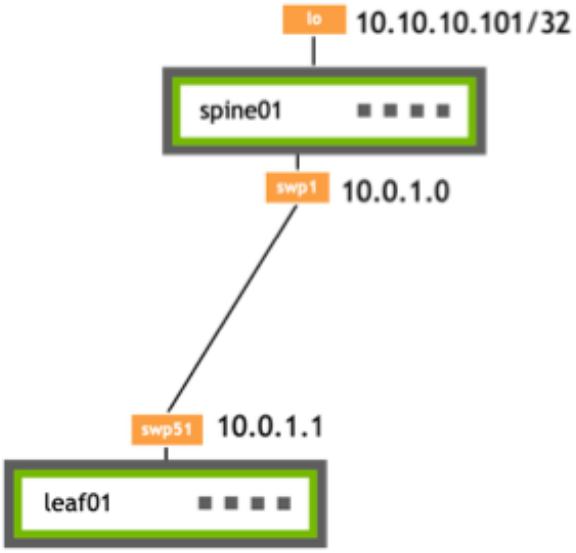
You can use static routing if you don't require the complexity of a dynamic routing protocol (such as BGP or OSPF), if you have routes that do not change frequently and for which the destination is only one or two paths away.

With static routing, you configure the switch manually to send traffic with a specific destination prefix to a specific next hop. When the switch receives a packet, it looks up the destination IP address in the routing table and forwards the packet accordingly.

Configure a Static Route

Static routes are added to the FRRouting routing table and then the kernel routing table.

The following example commands configure Cumulus Linux to send traffic with the destination prefix 10.10.10.101/32 out swp51 (10.0.1.1/31) to the next hop 10.0.1.0.



NCLU Commands | Linux and vtysh Commands

```
cumulus@leaf01:~$ net add interface swp51 ip address
10.0.1.1/31
cumulus@leaf01:~$ net add routing route 10.10.10.101/32
10.0.1.0
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The commands save the static route configuration in the `/etc/frr/frr.conf` file. For example:

```
...
ip route 10.10.10.101/32 10.0.1.0
...
```

The following example commands configure Cumulus Linux to send traffic with the destination prefix 10.10.10.61/32 out swp3 (10.0.0.32/31) to the next hop 10.0.0.33 in vrf BLUE.

NCLU Commands | Linux and vtysh Commands

```
cumulus@border01:~$ net add interface swp3 ip address
10.0.0.32/31
cumulus@border01:~$ net add interface swp51 vrf BLUE
cumulus@border01:~$ net add routing route 10.10.10.61/32
10.0.0.33 vrf BLUE
cumulus@border01:~$ net pending
cumulus@border01:~$ net commit
```

The commands save the static route configuration in the `/etc/frr/frr.conf` file. For example:

```
...
vrf BLUE
 ip route 10.10.10.61/32 10.0.0.33
...
```

To delete a static route:

NCLU Commands vtysh Commands

```
cumulus@leaf01:~$ net del routing route 10.10.10.101/32
10.0.1.0
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

TIP

When you use NCLU commands to delete routing configuration such as static routes, commit ten or fewer delete commands at a time to avoid commit failures.

To view static routes, run the NCLU `net show route static` command or the vtysh `show ip route` command. For example:

```
cumulus@leaf01:mgmt:~$ net show route static
RIB entry for static
====================
```

```
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r -
rejected route

S>* 10.10.10.101/32 [1/0] via 10.0.1.0, swp51, weight 1,
00:02:07
```

You can also create a static route by adding the route to a switch port configuration. For example:

NCLU Commands | Linux Commands

```
cumulus@leaf01:~$ net add interface swp51 ip address
10.0.1.1/31
cumulus@leaf01:~$ net add interface swp51 post-up ip route
add 10.10.10.101/32 via 10.0.1.0
cumulus@leaf01:~$ net add interface swp51 post-down ip
route del 10.10.10.101/32 via 10.0.1.0
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The `ip route` command allows you to manipulate the kernel routing table directly from the Linux shell. See `man ip(8)` for details. FRRouting monitors the kernel routing table changes and updates its own routing table accordingly.

Configure a Gateway or Default Route

On each switch, consider creating a *gateway* or *default route* for traffic destined outside the switch's subnet or local network. All such traffic passes through the gateway, which is a system on the same network that routes packets to their destination beyond the local network.

The following example configures the default route 0.0.0.0/0, which indicates any IP address can be sent to the gateway. The gateway is another switch with the IP address 10.0.1.0.

NCLU Commands | vtysh Commands

```
cumulus@leaf01:~$ net add routing route 0.0.0.0/0 10.0.1.0
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The NCLU and vtysh commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
!
ip route 0.0.0.0/0 10.0.1.0
!
...
```

> **NOTE**
>
> The default route created by the `gateway` parameter in ifupdown2 is not installed in FRR and cannot be redistributed into other routing protocols. See ifupdown2 and the gateway Parameter for more information.

Considerations

Deleting Routes through the Linux Shell

To avoid incorrect routing, **do not** use the Linux shell to delete static routes that you added with `vtysh` commands. Delete the routes with the `vtysh` commands.

IPv6 Default Route with a Source IP Address on eth0

If you install an IPv6 default route on eth0 with a source IP address, the

configuration either fails at reboot or the first time you run `ifup -dv eth0`, you see a warning and the route is not installed. (If you run `ifup -dv eth0` a second time, the route is installed successfully.)

```
cumulus@leaf01:~$ sudo ifup -dv eth0
warning: eth0: post-up cmd '/sbin/ip route add default via
2001:db8:5ca1:160::1 /
src 2001:db8:5ca1:160::45 dev eth0' failed (RTNETLINK answers:
Invalid argument)<<<<<<<<<<
```

To avoid this issue, you can do one of the following:

- Add a delay to the eth0 interface:

```
cumulus@leaf01:~$ net add interface eth0 ipv6 address
2001:db8:5ca1:160::45/64 post-up /bin/sleep 2s
cumulus@leaf01:~$ net add interface eth0 post-up /sbin/ip
route add default via 2001:db8:5ca1:160::1 src
2001:db8:5ca11:160::45 dev eth0
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

- Exclude the `src` parameter with the `ip route add` command:

```
cumulus@leaf01:~$ net add interface eth0 ipv6 address
2001:db8:5ca1:160::45/64 post-up
cumulus@leaf01:~$ net add interface eth0 post-up /sbin/ip
route add default via 2001:db8:5ca1:160::1 dev eth0
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

IPv4 and IPv6 Neighbor Cache Aging Timer

Cumulus Linux does not support different neighbor cache aging timer settings for IPv4 and IPv6.

The `net.ipv4.neigh.default.base_reachable_time_ms` and `net.ipv6.neigh.default.base_reachable_time_ms` settings in the `/etc/sysctl.d/neigh.conf` file must have the same value:

```
cumulus@leaf01:~$ sudo cat /etc/sysctl.d/neigh.conf
...
net.ipv4.neigh.default.base_reachable_time_ms=1080000
net.ipv6.neigh.default.base_reachable_time_ms=1080000
...
```

Related Information

- Linux IP - ip route command
- FRRouting docs - static route commands

Supported Route Table Entries

Cumulus Linux advertises the maximum number of route table entries supported on a given platform, including:

- Layer 3 IPv4 LPM (longest prefix match) entries that have a mask less than /32
- Layer 3 IPv6 LPM entries that have a mask of /64 or less
- Layer 3 IPv6 LPM entries that have a mask greater than /64
- Layer 3 IPv4 neighbor (or host) entries that are the next hops seen in `ip neighbor`
- Layer 3 IPv6 neighbor entries that are the next hops seen in `ip -6 neighbor`
- ECMP next hops, which are IP address entries in a router's routing table that specify the next closest or most optimal router in its routing path
- MAC addresses

In addition, Tomahawk, Trident II, Trident II+, and Trident3 switches are configured to manage route table entries using Algorithm Longest Prefix Match (ALPM). In ALPM mode, the hardware can store significantly more route entries.

To determine the current table sizes on a switch, use either the NCLU `net show system asic` command or `cl-resource-query`.

Forwarding Table Profiles

On Mellanox Spectrum and some Broadcom ASICs, you can configure the allocation of forwarding table resources and mechanisms. Cumulus Linux provides a number of generalized profiles for the platforms described below. These profiles work only with layer 2 and layer 3 unicast forwarding.

Choose the profile that best suits your network architecture and specify the profile name for the `forwarding_table.profile` variable in the `/etc/cumulus/datapath/traffic.conf` file; for example:

```
cumulus@switch:~$ cat /etc/cumulus/datapath/traffic.conf | grep
forwarding_table -B 4
# Manage shared forwarding table allocations
# Valid profiles -
# default, l2-heavy, v4-lpm-heavy, v6-lpm-heavy
#
forwarding_table.profile = default
```

After you specify a different profile, restart `switchd` for the change to take effect. You can see the forwarding table profile when you run `cl-resource-query`.

NOTE

- Broadcom ASICs other than Maverick, Tomahawk/ Tomahawk+, Trident II, Trident II+, and Trident3 support only the *default* profile.
- For Broadcom ASICs, the maximum number of IP multicast entries is 8k.

Supported Route Entries

The following tables list the number of MAC addresses, layer 3 neighbors, and LPM routes validated for each forwarding table profile for supported platforms. If you do not specify any profiles as described above, the switch uses the *default* values.

TIP

The values in the following tables reflect results from testing on supported platforms, which might differ from published manufacturer specifications.

Mellanox Spectrum Switches

Profile	MAC Addresses	L3 Neighbors	Longest Prefix Match (LPM)
default	40k	32k (IPv4) and 16k (IPv6)	64k (IPv4) and 28k (IPv6-long)
l2-heavy	88k	48k (IPv4) and 40k (IPv6)	8k (IPv4) and 8k (IPv6-long)
l2-heavy-1	180K	8k (IPv4) and 8k (IPv6)	8k (IPv4) and 8k (IPv6-long)
v4-lpm-heavy	8k	8k (IPv4) and 16k (IPv6)	80k (IPv4) and 16k (IPv6-long)
v4-lpm-heavy-1	8k	8k (IPv4) and 2k (IPv6)	176k (IPv4) and 2k (IPv6-long)
v6-lpm-heavy	40k	8k (IPv4) and 40k (IPv6)	8k (IPv4) and 32k (IPv6-long) and 32K (IPv6/64)
lpm-balanced	8k	8k (IPv4) and 8k (IPv6)	60k (IPv4) and 60k (IPv6-long)

Broadcom Tomahawk/Tomahawk+ Switches

Profile	MAC Addresses	L3 Neighbors	Longest Prefix Match (LPM)
default	40k	40k	64k (IPv4) or 8k (IPv6-long)
l2-heavy	72k	72k	8k (IPv4) or 2k (IPv6-long)
v4-lpm-heavy v6-lpm-heavy	8k	8k	128k (IPv4) or 20k (IPv6-long)

Broadcom Trident II/Trident II+/Trident3 Switches

Profile	MAC Addresses	L3 Neighbors	Longest Prefix Match (LPM)
default	32k	16k	128k (IPv4) or 20k (IPv6-long)
l2-heavy	160k	96k	8k (IPv4) or 2k (IPv6-long)
v4-lpm-heavy v6-lpm-heavy	32k	16k	128k (IPv4) or 20k

Profile	MAC Addresses	L3 Neighbors	Longest Prefix Match (LPM)
			(IPv6-long)

Broadcom Helix4 Switches

Helix4 switches do *not* have profiles.

MAC Addresses	L3 Neighbors	Longest Prefix Match (LPM)
24k	12k	7.8k (IPv4) or 2k (IPv6-long)

> **NOTE**
>
> For Broadcom switches, IPv4 and IPv6 entries are not carved in separate spaces so it is not possible to define explicit numbers in the L3 Neighbors column of the tables above. An IPv6 entry takes up twice the space of an IPv4 entry.

TCAM Resource Profiles for Spectrum Switches

On the Mellanox Spectrum ASIC, you can configure TCAM resource allocation, which is shared between IP multicast forwarding entries and

ACL tables. Cumulus Linux provides a number of general profiles for this platform. Choose the profile that best suits your network architecture and specify that profile name in the `tcam_resource.profile` variable in the `/usr/lib/python2.7/dist-packages/cumulus/__chip_config/mlx/datapath.conf` file; for example:

```
cumulus@switch:~$ cat /usr/lib/python2.7/dist-packages/
cumulus/__chip_config/mlx/datapath.conf | grep -B3
"tcam_resource"
#TCAM resource forwarding profile


    1. Valid profiles -
    2. default, ipmc-heavy, acl-heavy, ipmc-max
       tcam_resource.profile = default
```

After you specify a different profile, restart `switchd` for the change to take effect.

When nonatomic updates are enabled (`acl.non_atomic_update_mode` is set to `TRUE` in the `/etc/cumulus/switchd.conf` file), the maximum number of mroute and ACL entries for each profile are:

Profile	Mroute Entries	ACL Entries
default	1000	500 (IPv6) or 1000 (IPv4)

Profile	Mroute Entries	ACL Entries
ipmc-heavy	8500	1000 (IPv6) or 1500 (IPv4)
acl-heavy	450	2000 (IPv6) or 3500 (IPv4)
ipmc-max	13000	1000 (IPv6) or 2000 (IPv4)

When nonatomic updates are disabled (`acl.non_atomic_update_mode` is set to `FALSE` in the `/etc/cumulus/switchd.conf` file), the maximum number of mroute and ACL entries for each profile are:

Profile	Mroute Entries	ACL Entries
default	1000	250 (IPv6) or 500 (IPv4)
ipmc-heavy	8500	500 (IPv6) or 750 (IPv4)
acl-heavy	450	1000 (IPv6) or 1750 (IPv4)
ipmc-max	13000	500 (IPv6) or 1000 (IPv4)

Route Entry Takes Precedence Over Neighbor Entry

On Broadcom switches running Cumulus Linux 4.0 and later, when there is a /32 IPv4 or /128 IPv6 route and the same prefix is also a neighbor entry in

the linux kernel, the route entry takes precedence over the neighbor entry in the forwarding lookup. To change this behaviour, update the `route_preferred_over_neigh` variable to FALSE in the `/etc/cumulus/switchd.conf` file.

Route Filtering and Redistribution

Route filtering lets you exclude routes that are advertised or received from neighbors. You can use route filtering to manipulate traffic flows, reduce memory utilization, and improve security.

This section discusses the following route filtering methods:

- Prefix lists
- Route maps
- Route redistribution

Prefix Lists

Prefix lists are access lists for route advertisements that match routes instead of traffic. Prefix lists are typically used with route maps and other filtering methods. A prefix list can match the prefix (the network itself) and the prefix-length (the length of the subnet mask).

The following example commands configure a prefix list that permits all prefixes in the range 10.0.0.0/16 with a subnet mask less than or equal to /30. For networks 10.0.0.0/24, 10.10.10.0/24, and 10.0.0.10/32, only 10.0.0.0/24 is matched (10.10.10.0/24 has a different prefix and 10.0.0.10/32 has a greater subnet mask).

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add routing prefix-list ipv4
prefixlist1 permit 10.0.0.0/16 le 30
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and vtysh commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router ospf
 ospf router-id 10.10.10.1
 timers throttle spf 80 100 6000
 passive-interface vlan10
 passive-interface vlan20
ip prefix-list prefixlist1 permit 10.0.0.0/16 le 30
```

To use this prefix list in a route map, see Configuration-Examples below.

Route Maps

Route maps are routing policies that are considered before the router

examines the forwarding table. Each statement in a route map is assigned a sequence number, and contains a series of match and set statements. The route map is parsed from the lowest sequence number to the highest, and stops when a match is found.

Configure a Route Map

The following example commands configure a route map that sets the metric to 50 for interface swp51:

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add routing route-map routemap1
permit 10 match interface swp51
cumulus@switch:~$ net add routing route-map routemap1
permit 10 set metric 50
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and vtysh commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router ospf
 ospf router-id 10.10.10.1
```

```
 timers throttle spf 80 100 6000
 passive-interface vlan10
 passive-interface vlan20
route-map routemap1 permit 10
 match interface swp51
 set metric 50
```

Apply a Route Map

To apply the route map, you specify the routing protocol (bgp, ospf, or static) and the route map name.

The following example filters routes from Zebra into the Linux kernel. The commands apply the route map called routemap1 to BGP:

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add routing protocol bgp route-map
routemap1
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and vtysh commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router ospf
 ospf router-id 10.10.10.1
 timers throttle spf 80 100 6000
 passive-interface vlan10
 passive-interface vlan20
ip protocol bgp route-map routemap1
```

For BGP, you can also apply a route map on route updates from BGP to Zebra. All the applicable match operations are allowed, such as match on prefix, next hop, communities, and so on. Set operations for this attach-point are limited to metric and next hop only. Any operation of this feature does not affect BGPs internal RIB. Both IPv4 and IPv6 address families are supported. Route maps work on multi-paths; however, the metric setting is based on the best path only.

To apply a route map to filter route updates from BGP into Zebra, run the following command:

```
cumulus@switch:$ net add bgp table-map routemap2
```

> **NOTE**
>
> In NCLU, you can only set the community number in a route map. You cannot set other community options such as `no-export`, `no-advertise`, or `additive`.

Route Redistribution

Route redistribution allows a network to use a routing protocol to route traffic dynamically based on the information learned from a different routing protocol or from static routes. Route redistribution helps increase accessibility within networks.

To redistribute protocol routes, run the `net add <protocol> redistribute` command. The following example commands redistribute routing information from ospf routes into BGP:

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add bgp redistribute ospf
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

To redistribute all directly connected networks, use the `redistribute`

`connected` command. For example:

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add bgp redistribute connected
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

NOTE

For OSPF, redistribution loads the database unnecessarily with type-5 LSAs. Only use this method to generate real external prefixes (type-5 LSAs).

Configuration Examples

This section shows the `/etc/frr/frr.conf` file configuration for example route filters and redistribution.

The following example filters all routes that are not originated in the local AS:

```
...
```

```
router bgp 65101
 bgp router-id 10.10.10.1
 neighbor underlay interface remote-as external
 !
 address-family ipv4 unicast
  neighbor underlay route-map my-as out
 exit-address-family
!
bgp as-path access-list my-as permit ^$
!
route-map my-as permit 10
 match as-path my-as
!
route-map my-as deny 20
!
```

The following example sets communities based on prefix-lists:

```
...
router bgp 65101
 bgp router-id 10.10.10.1
 neighbor underlay interface remote-as external
 !
```

```
 address-family ipv6 unicast
  neighbor underlay activate
  neighbor underlay route-map MARK-PREFIXES out
 exit-address-family
!
ipv6 prefix-list LOW-PRIO seq 5 permit 2001:db8:dead::/56 le 64
ipv6 prefix-list MID-PRIO seq 5 permit 2001:db8:beef::/56 le 64
ipv6 prefix-list HI-PRIO seq 5 permit 2001:db8:cafe::/56 le 64
!
route-map MARK-PREFIXES permit 10
 match ipv6 address prefix-list LOW-PRIO
 set community 123:200
!
route-map MARK-PREFIXES permit 20
 match ipv6 address prefix-list MID-PRIO
 set community 123:500
!
route-map MARK-PREFIXES permit 30
 match ipv6 address prefix-list HI-PRIO
 set community 123:1000
!
```

The following example filters routes from being advertised to the peer:

```
router bgp 65101
 bgp router-id 10.10.10.1
 neighbor underlay interface remote-as external
 !
 address-family ipv4 unicast
  neighbor underlay route-map POLICY-OUT out
 exit-address-family
!
ip prefix-list BLOCK-RFC1918 seq 5 permit 10.0.0.0/8 le 24
ip prefix-list BLOCK-RFC1918 seq 10 permit 172.16.0.0/12 le 24
ip prefix-list BLOCK-RFC1918 seq 15 permit 192.168.0.0/16 le 24
ip prefix-list ADD-COMM-OUT seq 5 permit 100.64.0.0/10 le 24
ip prefix-list ADD-COMM-OUT seq 10 permit 192.0.2.0/24
!
route-map POLICY-OUT deny 10
 match ip address prefix-list BLOCK-RFC1918
!
route-map POLICY-OUT permit 20
 match ip address prefix-list ADD-COMM-OUT
 set community 123:1000
!
route-map POLICY-OUT permit 30
```

The following example sets mutual redistribution between OSPF and BGP

(filters by route tags):

```
...
router ospf
  redistribute bgp route-map BGP-INTO-OSPF
!
router bgp 65101
 bgp router-id 10.10.10.1
 neighbor underlay interface remote-as external
 !
 address-family ipv4 unicast
  redistribute ospf route-map OSPF-INTO-BGP
 exit-address-family
!
route-map OSPF-INTO-BGP deny 10
 match tag 4271
!
route-map OSPF-INTO-BGP permit 20
 set tag 2328
!
route-map BGP-INTO-OSPF deny 10
 match tag 2328
!
route-map BGP-INTO-OSPF permit 20
 set tag 4271
```

The following example filters and modifies redistributed routes:

```
...
router ospf
  redistribute bgp route-map EXTERNAL-2-1K
!
route-map EXTERNAL-2-1K permit 10
 set metric 1000
 set metric-type type-1
```

Policy-based Routing

Typical routing systems and protocols forward traffic based on the destination address in the packet, which is used to look up an entry in a routing table. However, sometimes the traffic on your network requires a more hands-on approach. You might need to forward a packet based on the source address, the packet size, or other information in the packet header.

Policy-based routing (PBR) lets you make routing decisions based on filters that change the routing behavior of specific traffic so that you can override the routing table and influence where the traffic goes. For example, you can use PBR to help you reach the best bandwidth utilization for business-critical applications, isolate traffic for inspection or analysis, or manually load balance outbound traffic.

Policy-based routing is applied to incoming packets. All packets received on a PBR-enabled interface pass through enhanced packet filters that determine rules and specify where to forward the packets.

NOTE

- You can create a *maximum* of 255 PBR match rules and 256 next hop groups (this is the ECMP limit).
- You can apply only one PBR policy per input interface.

- You can match on *source* and *destination* IP address, or match on Differentiated Services Code Point (DSCP) or Explicit Congestion Notification (ECN) values within a packet.
- PBR is not supported for GRE or VXLAN tunneling.
- PBR is not supported on management interfaces, such as eth0.
- A PBR rule cannot contain both IPv4 and IPv6 addresses.

Configure PBR

A PBR policy contains one or more policy maps. Each policy map:

- Is identified with a unique map name and sequence number. The sequence number is used to determine the relative order of the map within the policy.
- Contains a match source IP rule and (or) a match destination IP rule and a set rule, or a match DSCP or ECN rule and a set rule.
  - To match on a source and destination address, a policy map can contain both match source and match destination IP rules.
  - A set rule determines the PBR next hop for the policy. The set rule can contain a single next hop IP address or it can contain a next hop group. A next hop group has more than one next hop IP address so that you

can use multiple interfaces to forward traffic. To use ECMP, you configure a next hop group.

To use PBR in Cumulus linux, you define a PBR policy and apply it to the ingress interface (the interface must already have an IP address assigned). Traffic is matched against the match rules in sequential order and forwarded according to the set rule in the first match. Traffic that does not match any rule is passed onto the normal destination based routing mechanism.

> **NOTE**
>
> For Tomahawk and Tomahawk+ platforms, you must configure the switch to operate in non-atomic mode, which offers better scaling as all TCAM resources are used to actively impact traffic. Add the line `acl.non_atomic_update_mode = TRUE` to the `/etc/cumulus/switchd.conf` file.

To configure a PBR policy:

NCLU Commands | vtysh Commands

1. Configure the policy map.

   The example commands below configure a policy map called `map1` with sequence number 1, that matches on destination address 10.1.2.0/24 and source address 10.1.4.1/24.

   If the IP address in the rule is `0.0.0.0/0 or ::/0`, any IP address is a match. You cannot mix IPv4 and IPv6 addresses in a rule.

```
cumulus@switch:~$ net add pbr-map map1 seq 1 match dst-ip
10.1.2.0/24
cumulus@switch:~$ net add pbr-map map1 seq 1 match src-ip
10.1.4.1/24
```

   Instead of matching on IP address, you can match packets according to the DSCP or ECN field in the IP header. The DSCP value can be an integer between 0 and 63 or the DSCP codepoint name. The ECN value can be an integer between 0 and 3. The following example command configures a policy map called `map1` with sequence number 1 that matches on packets with the DSCP value 10:

```
cumulus@switch:~$ net add pbr-map map1 seq 1 match dscp 10
```

   The following example command configures a policy map called `map1` with sequence number 1 that matches on packets with the ECN value 2:

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
interface swp51
 pbr-policy map1
...
nexthop-group group1
 nexthop 192.168.0.21 swp1 nexthop-vrf rocket
 nexthop 192.168.0.22
...
pbr-map map1 seq 1
 match dst-ip 10.1.2.0/24
 match src-ip 10.1.4.1/24
 set nexthop nexthop-group group1
...
```

Review Your Configuration

Use the following commands to see the configured PBR policies.

To see the policies applied to all interfaces on the switch, run the NCLU `net show pbr interface` command or the `vtysh show pbr interface` command. For example:

```
cumulus@switch:~$ net show pbr interface
swp55s3(67) with pbr-policy map1
```

To see the policies applied to a specific interface on the switch, add the interface name at the end of the command; for example, `net show pbr interface swp51` (or `show pbr interface swp51` in `vtysh`).

To see information about all policies, including mapped table and rule numbers, run the NCLU `net show pbr map` command or the `vtysh` `show pbr map` command. If the rule is not set, you see a reason why.

```
cumulus@switch:~$ net show pbr map
 pbr-map map1 valid: yes
  Seq: 700 rule: 999 Installed: yes Reason: Valid
      SRC Match: 10.0.0.1/32
  nexthop 192.168.0.32
      Installed: yes Tableid: 10003
  Seq: 701 rule: 1000 Installed: yes Reason: Valid
      SRC Match: 90.70.0.1/32
  nexthop 192.168.0.32
      Installed: yes Tableid: 10004
```

To see information about a specific policy, what it matches, and with which

interface it is associated, add the map name at the end of the command; for example, `net show pbr map map1` (or `show pbr map map1` in `vtysh`).

To see information about all next hop groups, run the NCLU `net show pbr nexthop-group` command or the `vtysh` `show pbr nexthop-group` command.

```
cumulus@switch:~$ net show pbr nexthop-group
Nexthop-Group: map1701 Table: 10004 Valid: yes Installed: yes
Valid: yes nexthop 10.1.1.2
Nexthop-Group: map1700 Table: 10003 Valid: yes Installed: yes
Valid: yes nexthop 10.1.1.2
Nexthop-Group: group1 Table: 10000 Valid: yes Installed: yes
Valid: yes nexthop 192.168.10.0 bond1
Valid: yes nexthop 192.168.10.2
Valid: yes nexthop 192.168.10.3 vlan70
Nexthop-Group: group2 Table: 10001 Valid: yes Installed: yes
Valid: yes nexthop 192.168.8.1
Valid: yes nexthop 192.168.8.2
Valid: yes nexthop 192.168.8.3
```

To see information about a specific next hop group, add the group name at the end of the command; for example, `net show pbr nexthop-group group1` (or `show pbr nexthop-group group1` in `vtysh`).

> **NOTE**
>
> A new Linux routing table ID is used for each next hop and next hop group.

Modify Existing PBR Rules

When you want to change or extend an existing PBR rule, you must first delete the conditions in the rule, then add the rule back with the modification or addition.

▼ Modify an existing match/set condition

▼ Add a match condition to an existing rule

Delete PBR Rules and Policies

You can delete a PBR rule, a next hop group, or a policy. The following commands provide examples.

> **NOTE**
>
> Use caution when deleting PBR rules and next hop groups, as you might create an incorrect configuration for the PBR policy.

NCLU Commands vtysh Commands

The following examples show how to delete a PBR rule match:

```
cumulus@switch:~$ net del pbr-map map1 seq 1 match dst-ip
10.1.2.0/24
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The following examples show how to delete a next hop from a group:

```
cumulus@switch:~$ net del nexthop-group group1 nexthop
192.168.0.32 swp1 nexthop-vrf rocket
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The following examples show how to delete a next hop group:

```
cumulus@switch:~$ net del nexthop-group group1
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The following examples show how to delete a PBR policy so that the PBR interface is no longer receiving PBR traffic:

```
cumulus@switch:~$ net del interface swp3 pbr-policy map1
cumulus@switch:~$ net pending
```

NOTE

If a PBR rule has multiple conditions (for example, a source IP match and a destination IP match), but you only want to delete one condition, you have to delete all conditions first, then re-add the ones you want to keep.

The example below shows an existing configuration that has a source IP match and a destination IP match.

```
Seq: 6 rule: 305 Installed: yes Reason: Valid
   SRC Match: 10.1.4.1/24
   DST Match: 10.1.2.0/24
nexthop 192.168.0.21
   Installed: yes Tableid: 10011
```

The NCLU commands for the above configuration are:

```
net add pbr-map pbr-policy seq 6 match src-ip 10.1.4.1/24
net add pbr-map pbr-policy seq 6 match dst-ip 10.1.2.0/24
net add pbr-map pbr-policy seq 6 set nexthop 192.168.0.21
```

To remove the destination IP match, you must first delete all existing conditions defined under this sequence:

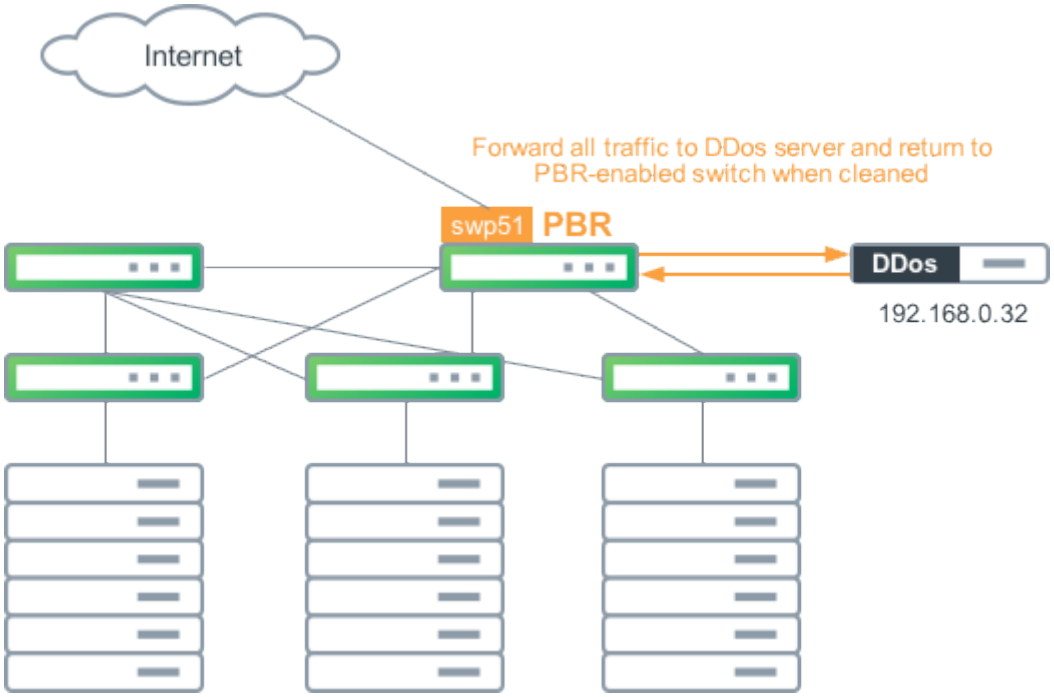
```
net del pbr-map pbr-policy seq 6 match src-ip 10.1.4.1/24
net del pbr-map pbr-policy seq 6 match dst-ip 10.1.2.0/24
net del pbr-map pbr-policy seq 6 set nexthop 192.168.0.21
net commit
```

Then, add back the conditions you want to keep:

```
net add pbr-map pbr-policy seq 6 match src-ip 10.1.4.1/24
net add pbr-map pbr-policy seq 6 set nexthop 192.168.0.21
net commit
```

Example Configuration

In the following example, the PBR-enabled switch has a PBR policy to route all traffic from the Internet to a server that performs anti-DDOS. The traffic returns to the PBR-enabled switch after being cleaned and is then passed onto the regular destination based routing mechanism.



The configuration for the example above is:

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add pbr-map map1 seq 1 match src-ip
0.0.0.0/0
cumulus@switch:~$ net add pbr-map map1 seq 1 set nexthop
192.168.0.32
cumulus@switch:~$ net add interface swp51 pbr-policy map1
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/`

`frr.conf` file. For example:

```
...
interface swp51
 pbr-policy map1
...
pbr-map map1 seq 1
 match src-ip 0.0.0.0/0
 set nexthop 192.168.0.32
...
```

Equal Cost Multipath Load Sharing - Hardware ECMP

Cumulus Linux supports hardware-based equal cost multipath (ECMP) load sharing. ECMP is enabled by default in Cumulus Linux. Load sharing occurs automatically for all routes with multiple next hops installed. ECMP load sharing supports both IPv4 and IPv6 routes.

Equal Cost Routing

ECMP operates only on equal cost routes in the Linux routing table.

In this example, the 10.1.1.0/24 route has two possible next hops that have been installed in the routing table:

```
cumulus@switch:~$ ip route show 10.1.1.0/24
10.1.1.0/24  proto zebra  metric 20
  nexthop via 192.168.1.1 dev swp1 weight 1 onlink
  nexthop via 192.168.2.1 dev swp2 weight 1 onlink
```

For routes to be considered equal they must:

- Originate from the same routing protocol. Routes from different sources are not considered equal. For example, a static route and an OSPF route are not considered for ECMP load sharing.

- Have equal cost. If two routes from the same protocol are unequal, only the best route is installed in the routing table.

> ⚠ **IMPORTANT**
>
> In Cumulus Linux, the BGP `maximum-paths` setting is enabled, so multiple routes are installed by default. See the ECMP section of the BGP chapter for more information.

ECMP Hashing

When multiple routes are installed in the routing table, a hash is used to determine which path a packet follows.

Cumulus Linux hashes on the following fields:

- IP protocol
- Ingress interface
- Source IPv4 or IPv6 address
- Destination IPv4 or IPv6 address

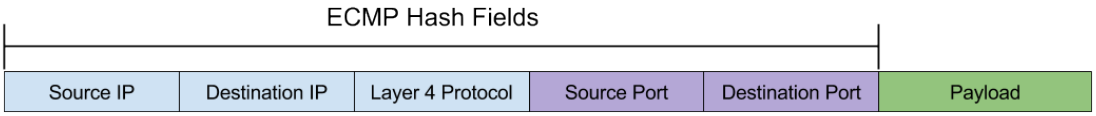
Further, on switches with Spectrum ASICs, Cumulus Linux hashes on these additional fields:

- Source MAC address

- Destination MAC address
- Ethertype
- VLAN ID

For TCP/UDP frames, Cumulus Linux also hashes on:

- Source port
- Destination port



To prevent out of order packets, ECMP hashing is done on a per-flow basis; all packets with the same source and destination IP addresses and the same source and destination ports always hash to the same next hop. ECMP hashing does not keep a record of flow states.

ECMP hashing does not keep a record of packets that have hashed to each next hop and does not guarantee that traffic sent to each next hop is equal.

Use cl-ecmpcalc to Determine the Hash Result

Because the hash is deterministic and always provides the same result for the same input, you can query the hardware and determine the hash result of a given input. This is useful when determining exactly which path a flow takes through a network.

On Cumulus Linux, use the `cl-ecmpcalc` command to determine a hardware hash result.

To use `cl-ecmpcalc`, all fields that are used in the hash must be provided. This includes ingress interface, layer 3 source IP, layer 3 destination IP, layer 4 source port, and layer 4 destination port.

```
cumulus@switch:~$ sudo cl-ecmpcalc -i swp1 -s 10.0.0.1 -d
10.0.0.1 -p tcp --sport 20000 --dport 80
ecmpcalc: will query hardware
swp3
```

If any field is omitted, `cl-ecmpcalc` fails.

```
cumulus@switch:~$ sudo cl-ecmpcalc -i swp1 -s 10.0.0.1 -d
10.0.0.1 -p tcp
ecmpcalc: will query hardware
usage: cl-ecmpcalc [-h] [-v] [-p PROTOCOL] [-s SRC] [--sport
SPORT] [-d DST]
                   [--dport DPORT] [--vid VID] [-i IN_INTERFACE]
                   [--sportid SPORTID] [--smodid SMODID] [-o
OUT_INTERFACE]
                   [--dportid DPORTID] [--dmodid DMODID] [--
hardware]
                   [--nohardware] [-hs HASHSEED]
                   [-hf HASHFIELDS [HASHFIELDS ...]]
                   [--hashfunction {crc16-ccitt,crc16-bisync}]
```

```
[-e EGRESS]
                    [-c MCOUNT]
cl-ecmpcalc: error: --sport and --dport required for TCP and
UDP frames
```

cl-ecmpcalc Limitations

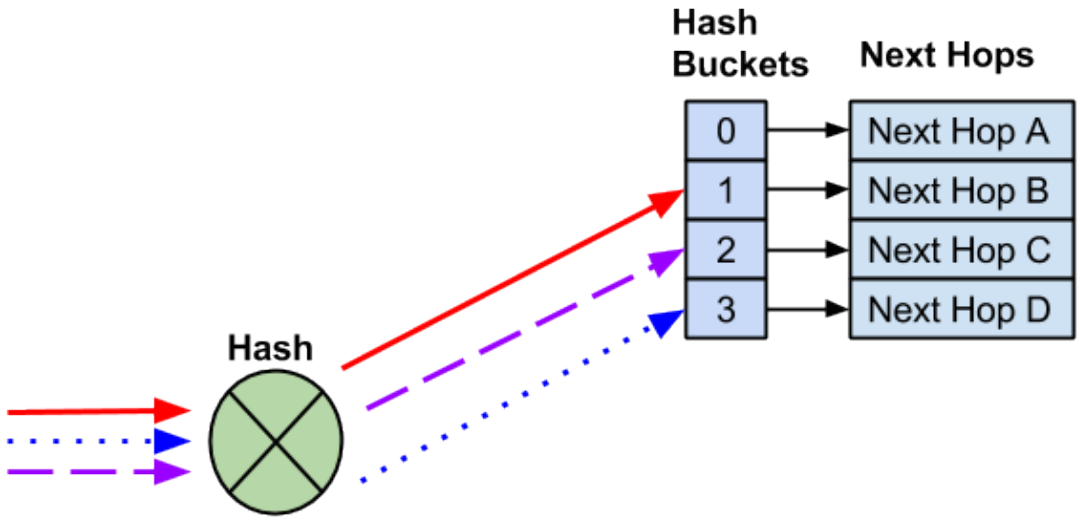
`cl-ecmpcalc` can only take input interfaces that can be converted to a single physical port in the port tab file, such as the physical switch ports (swp). Virtual interfaces like bridges, bonds, and subinterfaces are not supported.

`cl-ecmpcalc` is supported only on switches with the Mellanox Spectrum and the Broadcom Maverick, Tomahawk, Trident II, Trident II+ and Trident3 chipsets.

ECMP Hash Buckets

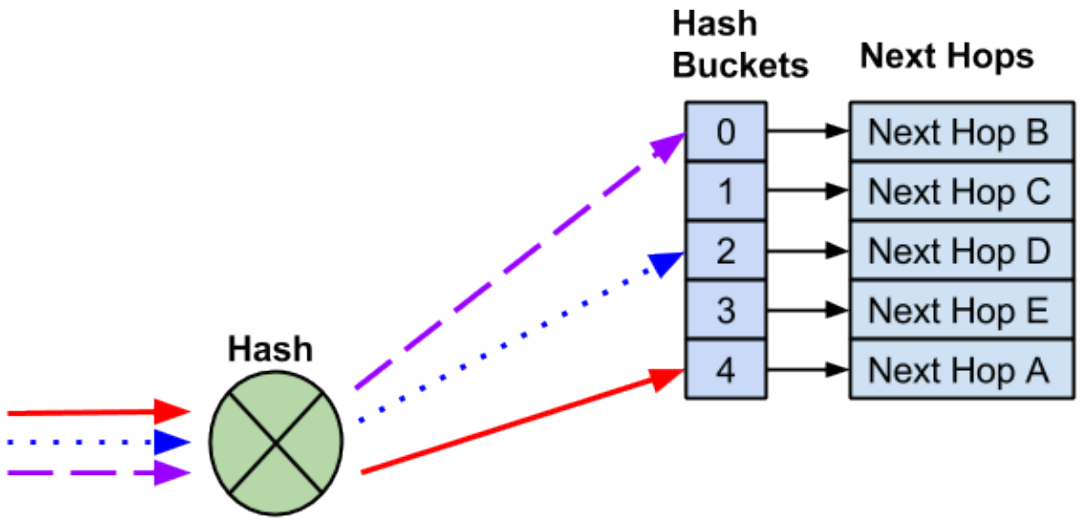
When multiple routes are installed in the routing table, each route is assigned to an ECMP *bucket*. When the ECMP hash is executed the result of the hash determines which bucket gets used.

In the following example, four next hops exist. Three different flows are hashed to different hash buckets. Each next hop is assigned to a unique hash bucket.



Add a Next Hop

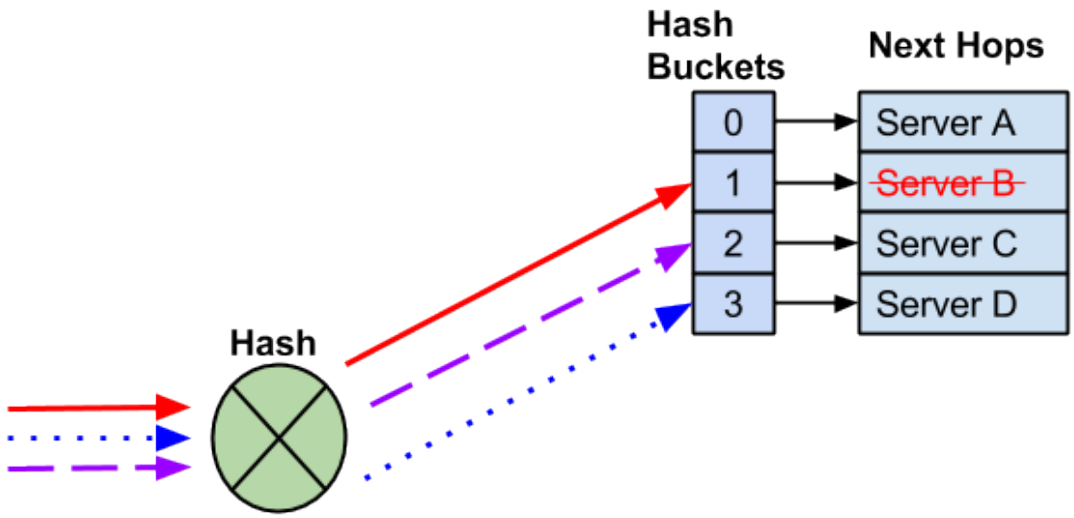
When a next hop is added, a new hash bucket is created. The assignment of next hops to hash buckets, as well as the hash result, might change when additional next hops are added.

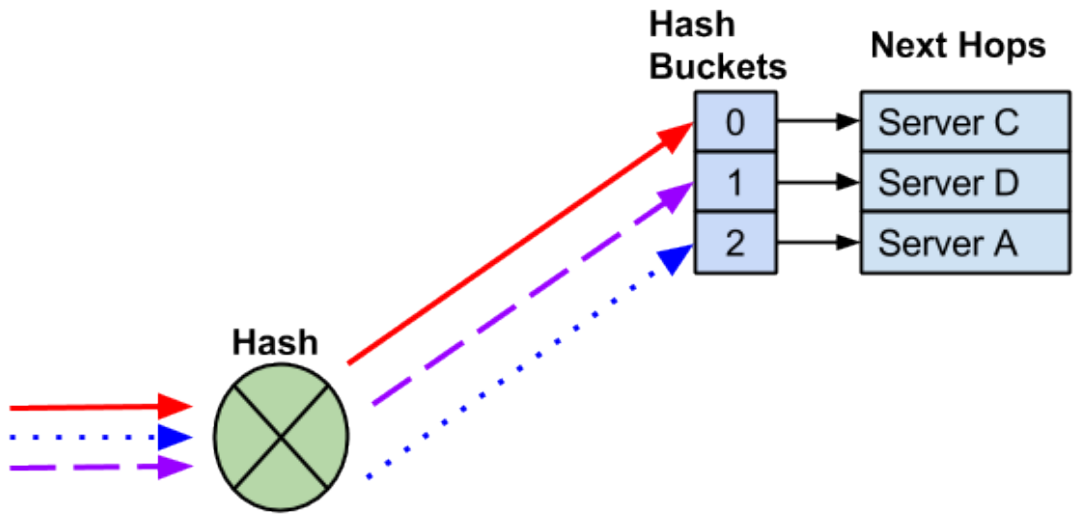


A new next hop is added and a new hash bucket is created. As a result, the hash and hash bucket assignment changes, causing the existing flows to be sent to different next hops.

Remove a Next Hop

When a next hop is removed, the remaining hash bucket assignments might change, again, potentially changing the next hop selected for an existing flow.





A next hop fails and the next hop and hash bucket are removed. The remaining next hops might be reassigned.

In most cases, the modification of hash buckets has no impact on traffic flows as traffic is being forwarded to a single end host. In deployments where multiple end hosts are using the same IP address (anycast), *resilient hashing* must be used.

Configure a Hash Seed to Avoid Hash Polarization

It is useful to have a unique hash seed for each switch. This helps avoid *hash polarization*, a type of network congestion that occurs when multiple data flows try to reach a switch using the same switch ports.

The hash seed is set by the `ecmp_hash_seed` parameter in the `/etc/cumulus/datapath/traffic.conf` file. It is an integer with a value from 0 to 4294967295. If you do not specify a value, `switchd` creates a randomly generated seed instead.

For example, to set the hash seed to *50*, run the following commands:

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add forwarding ecmp hash-seed 50
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

ECMP Custom Hashing

> **NOTE**
>
> Custom hashing is supported on Mellanox switches.

You can configure the set of fields used to hash upon during ECMP load balancing. For example, if you do not want to use source or destination port numbers in the hash calculation, you can disable the source port and destination port fields.

You can enable/disable the following fields:

- IP Protocol
- Source IP
- Destination IP
- Source port

- Destination port
- IPv6 flow label
- Ingress interface

You can also enable/disable these Inner header fields:

- Inner IP protocol
- Inner source IP
- Inner destination IP
- Inner source port
- Inner destination port
- Inner IPv6 flow label

To configure custom hashing, edit the `/etc/cumulus/datapath/traffic.conf` file:

1. To enable custom hashing, uncomment the `hash_config.enable = true` line.
2. To enable a field, set the field to `true`. To disable a field, set the field to `false`.
3. Run the `echo 1 > /cumulus/switchd/ctrl/hash_config_reload` command. This command does not cause any traffic interruptions.

The following shows an example `/etc/cumulus/datapath/traffic.conf` file:

```
cumulus@switch:~$ sudo nano /etc/cumulus/datapath/traffic.conf
```

```
...
# Uncomment to enable custom fields configured below
hash_config.enable = true

#hash Fields available ( assign true to enable)
#ip protocol
hash_config.ip_prot = true
#source ip
hash_config.sip = true
#destination ip
hash_config.dip = true
#source port
hash_config.sport = false
#destination port
hash_config.dport = false
#ipv6 flow label
hash_config.ip6_label = true
#ingress interface
hash_config.ing_intf = false

#inner fields for  IPv4-over-IPv6 and IPv6-over-IPv6
hash_config.inner_ip_prot = false
hash_config.inner_sip = false
hash_config.inner_dip = false
```

```
hash_config.inner_sport = false
hash_config.inner_dport = false
hash_config.inner_ip6_label = false
# Hash config end #
...
```

NOTE

Symmetric hashing is enabled by default on Mellanox switches. Make sure that the settings for the source IP (`hash_config.sip`) and destination IP (`hash_config.dip`) fields match, and that the settings for the source port (`hash_config.sport`) and destination port (`hash_config.dport`) fields match; otherwise symmetric hashing is disabled automatically. You can disable symmetric hashing manually in the `/etc/cumulus/datapath/traffic.conf` file by setting `symmetric_hash_enable = FALSE`.

Resilient Hashing

In Cumulus Linux, when a next hop fails or is removed from an ECMP pool, the hashing or hash bucket assignment can change. For deployments where there is a need for flows to always use the same next hop, like TCP anycast deployments, this can create session failures.

Resilient hashing is an alternate mechanism for managing ECMP groups. The ECMP hash performed with resilient hashing is exactly the same as the default hashing mode. Only the method in which next hops are assigned to hash buckets differs — they're assigned to buckets by hashing their header fields and using the resulting hash to index into the table of 2^n hash buckets. Since all packets in a given flow have the same header hash value, they all use the same flow bucket.

Resilient hashing supports both IPv4 and IPv6 routes.

Resilient hashing behaves slightly differently depending upon whether you are running Cumulus Linux on a switch with a Broadcom ASIC or Mellanox ASIC. The differences are described below.

- Resilient hashing prevents disruptions when next hops are removed. It does not prevent disruption when next hops are added.
- Resilient hashing is supported only on switches with the Broadcom Tomahawk, Trident II, Trident II+, and Trident3 as well as Mellanox Spectrum chipsets. You can run `net show system` to determine the chipset.

Resilient Hashing on Broadcom Switches

- When a next hop is removed, the assigned buckets are distributed to the remaining next hops.
- When a next hop is added, **some** buckets assigned to other next hops are migrated to the new next hop.
- The algorithm assigns buckets to next hops so as to make the number of buckets per next hop as close to equal as possible.

- The assignment of buckets to next hops is not changed in any other case. In particular, this assignment is not changed due to traffic loading or imbalance.
- Next hops are assigned to buckets randomly, to minimize the chance of systemic imbalance.

Resilient Hashing on Mellanox Switches

A Mellanox switch has two unique options for configuring resilient hashing, both of which you configure in the `/usr/lib/python2.7/dist-packages/cumulus/__chip_config/mlx/datapath.conf` file. The recommended values for these options depend largely on the desired outcome for a specific network implementation — the number and duration of flows, and the importance of keeping these flows pinned without interruption.

- `resilient_hash_active_timer`: A timer that protects TCP sessions from being disrupted while attempting to populate new next hops. You specify the number of seconds when at least one hash bucket consistently sees no traffic before Cumulus Linux rebalances the flows; the default is 120 seconds. If any one bucket is idle; that is, it sees no traffic for the defined period, the next new flow utilizes that bucket and flows to the new link. Thus, if the network is experiencing a large number of flows or very consistent or persistent flows, there may not be any buckets remaining idle for a consistent 120 second period, and the imbalance remains until that timer has been met. If a new link is brought up and added back to a group during this time, traffic does not get allocated to utilize it until a bucket qualifies as *empty*, meaning it has been idle for 120 seconds. This is when a rebalance can occur.

- `resilient_hash_max_unbalanced_timer`: You can force a rebalance every N seconds with this option. However, while this could correct the persistent imbalance that is expected with resilient hashing, this rebalance would result in the movement of all flows and thus a break in any TCP sessions that are active at that time.

Note that when you configure these options, a new next hop might not get populated for a long time.

The Mellanox Spectrum ASIC assigns packets to hash buckets and assigns hash buckets to next hops as follows. It also runs a background thread that monitors and may migrate buckets between next hops to rebalance the load.

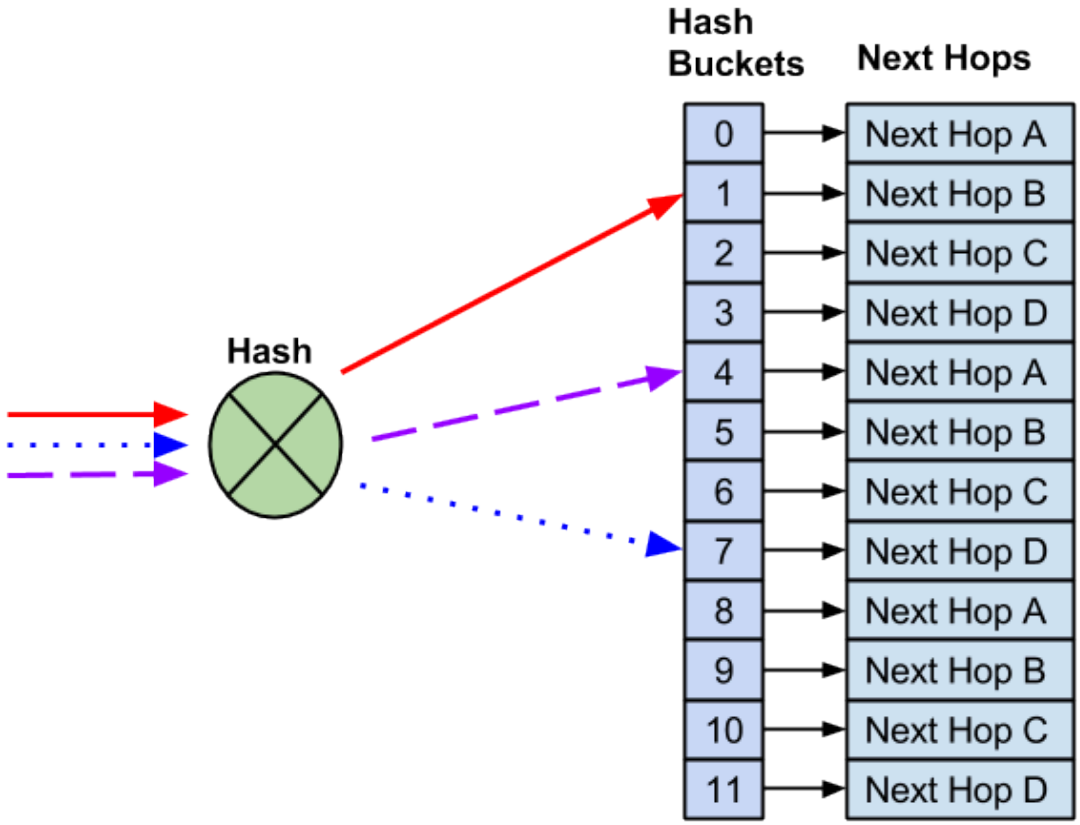
- When a next hop is removed, the assigned buckets are distributed to the remaining next hops.
- When a next hop is added, **no** buckets are assigned to the new next hop until the background thread rebalances the load.
- The load gets rebalanced when the active flow timer specified by the `resilient_hash_active_timer` setting expires if, and only if, there are inactive hash buckets available; the new next hop may remain unpopulated until the period set in `resilient_hash_active_timer` expires.
- When the `resilient_hash_max_unbalanced_timer` setting expires and the load is not balanced, the thread migrates any bucket(s) to different next hops to rebalance the load.

As a result, any flow may be migrated to any next hop, depending on flow activity and load balance conditions; over time, the flow may get pinned,

which is the default setting and behavior.

Resilient Hash Buckets

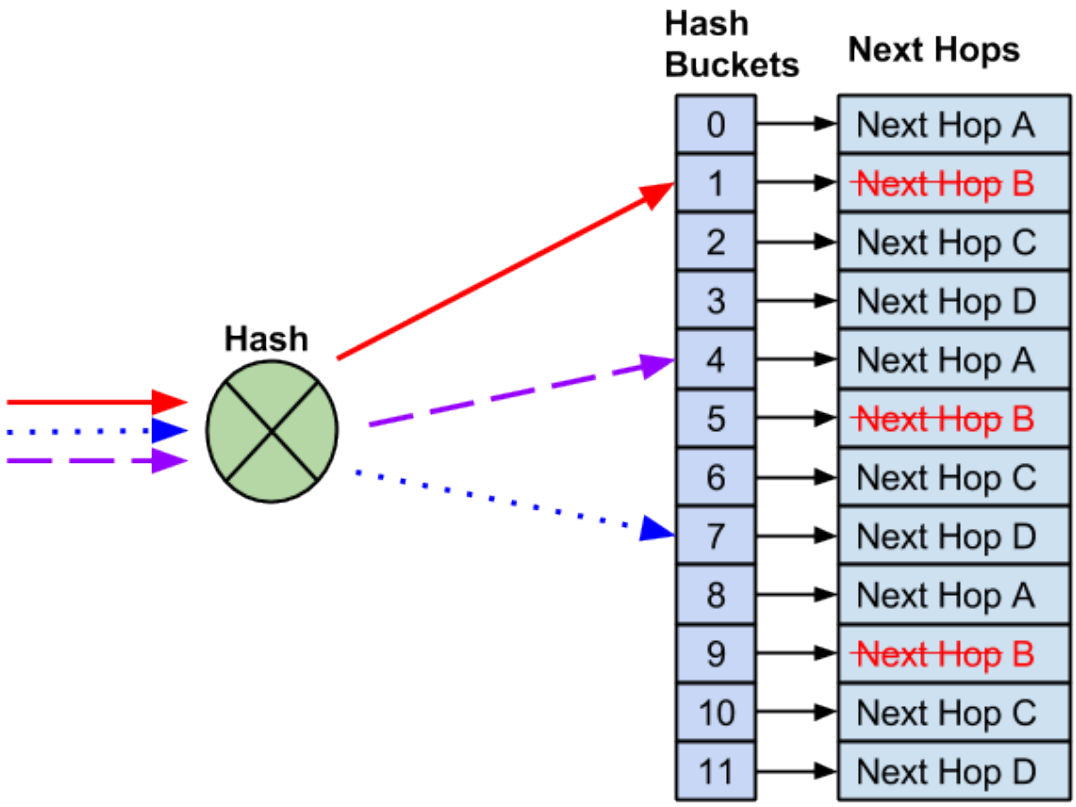
When resilient hashing is configured, a fixed number of buckets are defined. Next hops are then assigned in round robin fashion to each of those buckets. In this example, 12 buckets are created and four next hops are assigned.



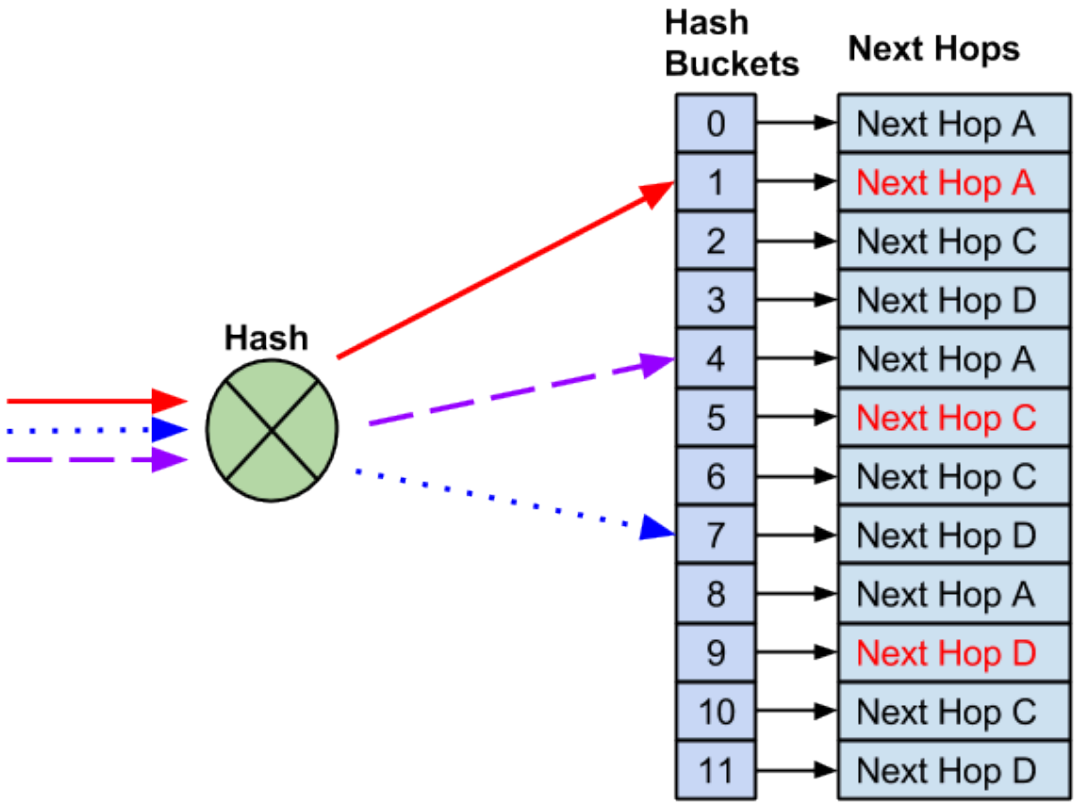
Remove Next Hops

Unlike default ECMP hashing, when a next hop needs to be removed, the

number of hash buckets does not change.



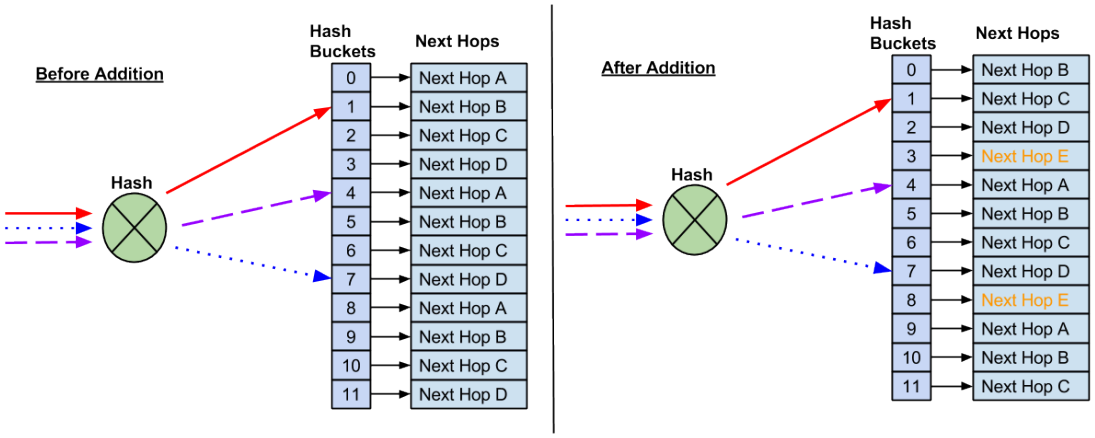
With 12 buckets assigned and four next hops, instead of reducing the number of buckets - which would impact flows to known good hosts - the remaining next hops replace the failed next hop.



After the failed next hop is removed, the remaining next hops are installed as replacements. This prevents impact to any flows that hash to working next hops.

Add Next Hops

Resilient hashing does not prevent possible impact to existing flows when new next hops are added. Due to the fact there are a fixed number of buckets, a new next hop requires reassigning next hops to buckets.



As a result, some flows may hash to new next hops, which can impact anycast deployments.

Configure Resilient Hashing

Resilient hashing is *not* enabled by default. When resilient hashing is enabled, 65,536 buckets are created to be shared among all ECMP groups. An ECMP group is a list of unique next hops that are referenced by multiple ECMP routes.

IMPORTANT

An ECMP route counts as a single route with multiple next hops. The following example is considered to be a single ECMP route:

```
cumulus@switch:~$ ip route show 10.1.1.0/24
```

```
10.1.1.0/24  proto zebra  metric 20
  nexthop via 192.168.1.1 dev swp1 weight 1 onlink
  nexthop via 192.168.2.1 dev swp2 weight 1 onlink
```

All ECMP routes must use the same number of buckets (the number of buckets cannot be configured per ECMP route).

The number of buckets can be configured as 64, 128, 256, 512 or 1024; the default is 128:

Number of Hash Buckets	Number of Supported ECMP Groups
64	1024
128	**512**
256	256
512	128
1024	64

> **NOTE**
>
> Mellanox switches with the Spectrum ASIC do not support 128 or 256 hash buckets. The default number of hash buckets is 64.

A larger number of ECMP buckets reduces the impact on adding new next hops to an ECMP route. However, the system supports fewer ECMP routes. If the maximum number of ECMP routes have been installed, new ECMP routes log an error and are not installed.

> **NOTE**
>
> Mellanox switches with the Spectrum ASIC allow for two custom options to allocate route and MAC address hardware resources depending on ECMP bucket size changes. More information about this is available in the Routing section on Mellanox Spectrum routing resources

To enable resilient hashing, edit `/etc/cumulus/datapath/traffic.conf`:

1. Enable resilient hashing:

```
# Enable resilient hashing
resilient_hash_enable = TRUE
```

2. **(Optional)** Edit the number of hash buckets:

```
# Resilient hashing flowset entries per ECMP group
# Valid values - 64, 128, 256, 512, 1024
resilient_hash_entries_ecmp = 256
```

3. Restart the `switchd` service:

```
cumulus@switch:~$ sudo systemctl restart switchd.service
```

WARNING

Restarting the `switchd` service causes all network ports to reset, interrupting network services, in addition to resetting the switch hardware configuration.

Considerations

IPv6 Route Replacement

When the next hop information for an IPv6 prefix changes (for example, when ECMP paths are added or deleted, or when the next hop IP address, interface, or tunnel changes), FRR deletes the existing route to that prefix from the kernel and then adds a new route with all the relevant new information. Because of this process, resilient hashing might not be maintained for IPv6 flows in certain situations.

To work around this issue, you can enable the IPv6 route replacement option.

IMPORTANT

Be aware that for certain configurations, the IPv6 route replacement option can lead to incorrect forwarding decisions and lost traffic. For example, it is possible for a destination to have next hops with a gateway value with the outbound interface or just the outbound interface itself, without a gateway address defined. If both types of next hops for the same destination exist, route replacement does not operate correctly; Cumulus Linux adds an additional route entry and next hop but does not delete the

previous route entry and next hop.

To enable the IPv6 route replacement option:

1. In the `/etc/frr/daemons` file, add the configuration option `--v6-rr-semantics` to the zebra daemon definition. For example:

```
cumulus@switch:~$ sudo nano /etc/frr/daemons
...
vtysh_enable=yes
zebra_options=" -M cumulus_mlag -M snmp -A 127.0.0.1 --v6-rr-
semantics -s 90000000"
bgpd_options=" -M snmp  -A 127.0.0.1"
ospfd_options=" -M snmp -A 127.0.0.1"
  ...
```

2. Restart FRR with this command:

```
cumulus@switch:~$ sudo systemctl restart frr.service
```

WARNING

Restarting FRR restarts all the routing protocol daemons that are enabled and running.

To verify that the IPv6 route replacement option is enabled, run the `systemctl status frr` command:

```
cumulus@switch:~$ systemctl status frr

● frr.service - FRRouting
  Loaded: loaded (/lib/systemd/system/frr.service; enabled;
vendor preset: enabled)
  Active: active (running) since Mon 2020-02-03 20:02:33 UTC;
3min 8s ago
    Docs: https://frrouting.readthedocs.io/en/latest/setup.html
  Process: 4675 ExecStart=/usr/lib/frr/frrinit.sh start
(code=exited, status=0/SUCCESS)
  Memory: 14.4M
  CGroup: /system.slice/frr.service
```

```
        ├─4685 /usr/lib/frr/watchfrr -d zebra bgpd staticd
        ├─4701 /usr/lib/frr/zebra -d -M snmp -A 127.0.0.1 --
v6-rr-semantics -s 90000000
        ├─4705 /usr/lib/frr/bgpd -d -M snmp -A 127.0.0.1
        └─4711 /usr/lib/frr/staticd -d -A 127.0.0.1
```

Unequal Cost Multipath with BGP Link Bandwidth

Unequal Cost Multipath (UCMP) is deployed in data center networks that rely on anycast routing to provide network-based load balancing. Cumulus Linux supports UCMP by using the BGP link bandwidth extended community to load balance traffic towards anycast services for IPv4 and IPv6 routes in a layer 3 deployment and for prefix (type-5) routes in an EVPN deployment.

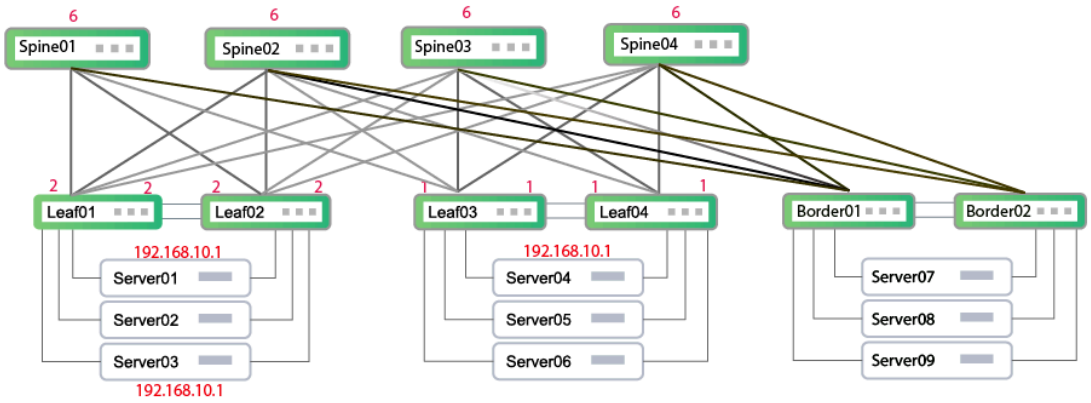
UCMP Routing

In ECMP, the route to a destination has multiple next hops and traffic is equally distributed across them. Flow-based hashing is used so that all traffic associated with a particular flow uses the same next hop and the same path across the network.

In UCMP, along with the ECMP flow-based hash, a weight is associated with each next hop and traffic is distributed across the next hops in proportion to their weight. Cumulus Linux relies on the BGP link bandwidth extended community to carry information about the anycast server distribution through the network; this is mapped to the weight of the corresponding next hop. This mapping is a factoring of a particular path's bandwidth value against the total bandwidth values of all possible paths, mapped to the range 1 to 100. There is no change to either the BGP best path selection algorithm or to the multipath computation algorithm that determines which

paths can be used for load balancing.

UCMP Example



The above example shows how traffic towards 192.168.10.1/32 is load balanced when you use UCMP routing:

- Leaf01 has two ECMP paths to 192.168.10.1/32 (via Server01 and Server03) whereas Leaf03 and Leaf04 have a single path to Server04.
- Leaf01, Leaf02, Leaf03, and Leaf04 are configured to generate BGP link bandwidth based on the number of BGP multipaths for a prefix.
- When announcing the prefix to the spines, Leaf01 and Leaf02 generate a link bandwidth of two while Leaf03 and Leaf04 generate a link bandwidth of one.
- Each spine advertises the 192.168.10.1/32 prefix to the border leafs with an accumulated bandwidth of 6. This combines the value of 2 from Leaf01, 2 from Leaf02, 1 from Leaf03 and 1 from Leaf04.

Now, each spine has four UCMP routes:

- via Leaf01 with weight 2
- via Leaf02 with weight 2
- via Leaf03 with weight 1
- via Leaf04 with weight 1

The border leafs also have four UCMP routes:

- via Spine01 with weight 6
- via Spine02 with weight 6
- via Spine03 with weight 6
- via Spine04 with weight 6

The border leafs balance traffic equally; all weights are equal to the spines. Only the spines have unequal load sharing based on the weight values.

Configure UCMP

Use the `set extcommunity bandwidth num-multipaths` command in a route map to set the extended community against all prefixes, or against a specific or set of prefixes using the match clause of the route map. Apply the route map at the first device to receive the prefix; against the BGP neighbor that generated this prefix.

The BGP link bandwidth extended community is encoded in bytes-per-second. To convert the number of ECMP paths, a reference bandwidth of 1024Kbps is used. For example, if there are four ECMP paths to an anycast IP, the encoded bandwidth in the extended community is 512,000. The actual value is not important, as long as all routers originating the link

bandwidth are converting the number of ECMP paths in the same way.

Cumulus Linux accepts the bandwidth extended community by default. No additional configuration is required on transit devices where UCMP routes are not being originated.

> **NOTE**
>
> - The bandwidth used in the extended community has no impact on or relation to port bandwidth.
> - You can only apply the route weight information on the outbound direction to a peer; you cannot apply route weight information on the inbound direction from peers advertising routes to the switch.

Set the BGP Link Bandwidth Extended Community Against All Prefixes

The following command examples show how you can set the BGP link bandwidth extended community against **all** prefixes.

NCLU Commands | vtysh Commands

```
cumulus@leaf01:~$ net add routing route-map ucmp-route-map
permit 10 set extcommunity bandwidth num-multipaths
cumulus@leaf01:~$ net add bgp neighbor 10.1.1.1 route-map
ucmp-route-map out
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
address-family ipv4 unicast
 neighbor 10.1.1.1 route-map ucmp-route-map out
!
route-map ucmp-route-map permit 10
 set extcommunity bandwidth num-multipaths
...
```

Set the BGP Link Bandwidth Extended Community Against Certain Prefixes

The following command examples show how you can set the BGP link

bandwidth extended community for anycast servers in the 192.168/16 IP address range.

NCLU Commands | vtysh Commands

```
cumulus@leaf01:~$ net add routing prefix-list ipv4 anycast-
ip permit 192.168.0.0/16 le 32
cumulus@leaf01:~$ net add routing route-map ucmp-route-map
permit 10 match ip address prefix-list anycast-ip
cumulus@leaf01:~$ net add routing route-map ucmp-route-map
permit 10 set extcommunity bandwidth num-multipaths
cumulus@leaf01:~$ net add bgp neighbor 10.1.1.1 route-map
ucmp-route-map out
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
address-family ipv4 unicast
 neighbor 10.1.1.1 route-map ucmp-route-map out
!
ip prefix-list anycast-ip permit 192.168.0.0/16 le 32
```

```
route-map ucmp-route-map permit 10
 match ip address prefix-list anycast-ip
 set extcommunity bandwidth num-multipaths
...
```

EVPN Configuration

For EVPN configuration, make sure that you activate the commands under the EVPN address family. The following shows an example EVPN configuration that sets the BGP link bandwidth extended community against **all** prefixes.

NCLU Commands | vtysh Commands

```
cumulus@leaf01:~$ net add routing route-map ucmp-route-map
permit 10 set extcommunity bandwidth num-multipaths
cumulus@leaf01:~$ net add bgp vrf turtle l2vpn evpn
advertise ipv4 unicast route-map ucmp-route-map
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
 address-family l2vpn evpn
  advertise ipv4 unicast route-map ucmp-route-map
 exit-address-family
!
ip prefix-list anycast-ip permit 192.168.0.0/16 le 32
route-map ucmp-route-map permit 10
 match ip address prefix-list anycast-ip
 set extcommunity bandwidth num-multipaths
...
```

Control UCMP on the Receiving Switch

To control UCMP on the receiving switch, you can:

- Set default values for UCMP routes.
- Disable the advertisement of all BGP extended communities on specific peerings.

Set Default Values for UCMP Routes

By default, if some of the multipaths do not have link bandwidth, Cumulus Linux ignores the bestpath bandwidth value in any of the multipaths and performs ECMP. However, you can set one of the following options instead:

- Ignore link bandwidth completely and perform ECMP.

- Skip paths without link bandwidth and perform UCMP among the others (if at least some paths have link bandwidth).
- Assign a low default weight (value 1) to paths that do not have link bandwidth.

Change this setting per BGP instance for both IPv4 and IPv6 unicast routes in the BGP instance. For EVPN, set the options on the tenant VRF.

Either run the NCLU `net add bestpath bandwidth ignore|skip-missing|default-weight-for-missing` command or the `vtysh` `bgp bestpath bandwidth ignore|skip-missing|default-weight-for-missing` command.

The following commands set link bandwidth processing to skip paths without link bandwidth and perform UCMP among the other paths:

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add bgp bestpath bandwidth skip-
missing
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
router bgp 65011
  bgp bestpath as-path multipath-relax
  neighbor LEAF peer-group
  neighbor LEAF remote-as external
  neighbor swp1 interface peer-group LEAF
  neighbor swp2 interface peer-group LEAF
  neighbor swp3 interface peer-group LEAF
  neighbor swp4 interface peer-group LEAF
  bgp bestpath bandwidth skip-missing
!
  address-family ipv4 unicast
    network 10.0.0.1/32
  exit-address-family
 ...
```

BGP Link Bandwidth Outside a Domain

The BGP link bandwidth extended community is automatically passed on with the prefix to eBGP peers. If you do not want to pass on the BGP link bandwidth extended community outside of a particular domain, you can disable the advertisement of all BGP extended communities on specific peerings.

> **NOTE**
>
> You cannot disable just the BGP link bandwidth extended community from being advertised to a neighbor; you either send all BGP extended communities, or none.

To disable all BGP extended communities on a peer or peer group (per address family), either run the NCLU `net del bgp neighbor <neighbor> send-community extended` command or the `vtysh` `no neighbor <neighbor> send-community extended` command:

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net del bgp neighbor 10.10.0.2 send-
community extended
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Troubleshooting

To show the extended community in a received or local route, run the NCLU `net show bgp` command or the `vtysh` `show bgp` command.

The following example shows that an IPv4 unicast route is received with the

BGP link bandwidth attribute from two peers. The link bandwidth extended community is encoded in bytes-per-second and shown in Mbps per second: `Extended Community: LB:65002:131072000 (1000.000 Mbps)` and `Extended Community: LB:65001:65536000 (500.000 Mbps)`.

```
cumulus@switch:~$ net show bgp ipv4 unicast 192.168.10.1/32
BGP routing table entry for 192.168.10.1/32
Paths: (2 available, best #2, table default)
  Advertised to non peer-group peers:
  l1(swp1) l2(swp2) l3(swp3) l4(swp4)
  65002
    fe80::202:ff:fe00:1b from l2(swp2) (10.0.0.2)
    (fe80::202:ff:fe00:1b) (used)
      Origin IGP, metric 0, valid, external, multipath,
bestpath-from-AS 65002
      Extended Community: LB:65002:131072000 (1000.000 Mbps)
      Last update: Thu Feb 20 18:34:16 2020

  65001
    fe80::202:ff:fe00:15 from l1(swp1) (110.0.0.1)
    (fe80::202:ff:fe00:15) (used)
      Origin IGP, metric 0, valid, external, multipath,
bestpath-from-AS 65001, best (Older Path)
      Extended Community: LB:65001:65536000 (500.000 Mbps)
      Last update: Thu Feb 20 18:22:34 2020
```

> **NOTE**
>
> The bandwidth value used by UCMP is only to determine the percentage of load to a given next hop and has no impact on actual link or flow bandwidth.

To show EVPN type-5 routes, run the NCLU `net show bgp l2vpn evpn route type prefix` command or the `vtysh show bgp l2vpn evpn route type prefix` command.

The bandwidth is displayed both in the way it is carried in the extended community (as bytes-per-second - unsigned 32 bits) as well as in Gbps, Mbps, or Kbps. For example:

```
cumulus@switch:~$ net show bgp l2vpn evpn route type prefix
BGP table version is 1, local router ID is 10.0.0.11
Status codes: s suppressed, d damped, h history, * valid, >
best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete
...
*> [5]:[0]:[32]:[192.168.10.1]
            10.0.0.5                              0 65100 65050
65200 i
```

```
            RT:65050:104001 LB:65050:134217728 (1.000 Gbps)
ET:8 Rmac:36:4f:15:ea:81:90
```

To see weights associated with next hops for a route with multiple paths, run the NCLU `net show route` command or the `vtysh` `show ip route` command. For example:

```
cumulus@switch:~$ net show route 192.168.10.1/32
Routing entry for 192.168.10.1/32
  Known via "bgp", distance 20, metric 0, best
  Last update 00:00:32 ago
  * fe80::202:ff:fe00:1b, via swp2, weight 66
  * fe80::202:ff:fe00:15, via swp1, weight 33
```

Considerations

UCMP with BGP link bandwidth is only available for BGP-learned routes.

Related Information

IETF draft - BGP Link Bandwidth Extended Community

Redistribute Neighbor

Redistribute neighbor provides a mechanism for IP subnets to span racks without forcing the end hosts to run a routing protocol.

The fundamental premise behind redistribute neighbor is to announce individual host /32 routes in the routed fabric. Other hosts on the fabric can then use this new path to access the hosts in the fabric. If multiple equal-cost paths (ECMP) are available, traffic can load balance across the available paths natively.

The challenge is to accurately compile and update this list of reachable hosts or neighbors. Luckily, existing commonly-deployed protocols are available to solve this problem. Hosts use ARP to resolve MAC addresses when sending to an IPv4 address. A host then builds an ARP cache table of known MAC addresses: IPv4 tuples as they receive or respond to ARP requests.

In the case of a leaf switch, where the default gateway is deployed for hosts within the rack, the ARP cache table contains a list of all hosts that have ARP'd for their default gateway. In many scenarios, this table contains all the layer 3 information that is needed. This is where redistribute neighbor comes in, as it is a mechanism of formatting and syncing this table into the routing protocol.

Redistribute neighbor is distributed as `python-rdnbrd`.

NOTE

The current implementation of redistribute neighbor:

- Supports IPv4 only.
- Does not support VRFs.
- Supports a maximum of 1024 interfaces. Using more than 1024 interfaces might crash the `rdnbrd` service.

Target Use Cases and Best Practices

Redistribute neighbor is typically used in these configurations:

- Virtualized clusters
- Hosts with service IP addresses that migrate between racks
- Hosts that are dual connected to two leaf nodes without using proprietary protocols such as MLAG
- Anycast services that need dynamic advertisement from multiple hosts

Follow these guidelines:

- You can connect a host to one or more leafs. Each leaf advertises the /32 it sees in its neighbor table.
- Make sure that a host-bound bridge/VLAN is local to each switch.
- Connect leaf switches with redistribute neighbor enabled directly to the hosts.

- Make sure that IP addressing is non-overlapping, as the host IP addresses are directly advertised into the routed fabric.
- Run redistribute neighbor on Linux-based hosts. NVIDIA has not actively tested other host operating systems.

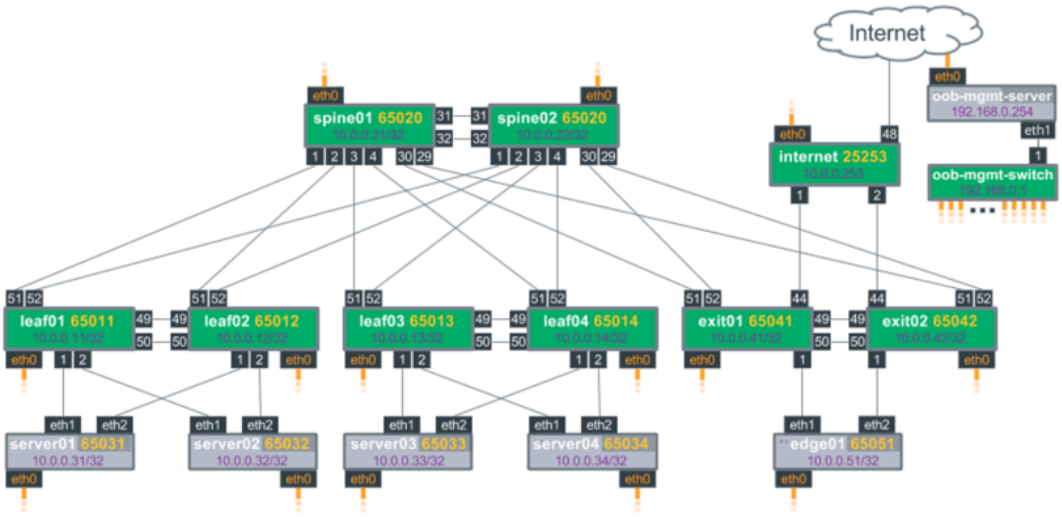
How It Works

Redistribute neighbor works as follows:

1. The leaf/ToR switches learn about connected hosts when the host sends an ARP request or ARP reply.
2. An entry for the host is added to the kernel neighbor table of each leaf switch.
3. The redistribute neighbor daemon, `rdnbrd`, monitors the kernel neighbor table and creates a /32 route for each neighbor entry. This /32 route is created in kernel table 10.
4. FRRouting is configured to import routes from kernel table 10.
5. A route-map controls which routes from table 10 are imported.
6. In FRRouting these routes are imported as *table* routes.
7. BGP, OSPF and so on, are then configured to redistribute the table 10 routes.

Example Configuration

The following example configuration is based on the reference topology created by NVIDIA. Other configurations are possible, based on the use cases outlined above. Here is a diagram of the topology:



Configure the Leafs

The following steps demonstrate how to configure leaf01, but you can follow the same steps on any of the leafs.

NCLU Commands | vtysh Commands

1. Configure the host facing ports using the same IP address on both host-facing interfaces as well as a /32 prefix. In this case, swp1 and swp2 are configured as they are the ports facing server01 and server02:

```
cumulus@leaf01:~$ net add loopback lo ip address
10.0.0.11/32
cumulus@leaf01:~$ net add interface swp1-2 ip address
10.0.0.11/32
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

2. Enable the daemon so it starts at bootup, then start the daemon:

```
cumulus@leaf01:~$ sudo systemctl enable rdnbrd.service
cumulus@leaf01:~$ sudo systemctl restart rdnbrd.service
```

3. Configure routing:

   a. Define a route-map that matches on the host-facing interfaces:

```
cumulus@leaf01:~$ net add routing route-map
REDIST_NEIGHBOR permit 10 match interface swp1
cumulus@leaf01:~$ net add routing route-map
REDIST_NEIGHBOR permit 20 match interface swp2
```

The NCLU and vtysh commands save the configuration in the `/etc/frr/frr.conf` file. The following example uses OSPF as the routing protocol:

```
frr defaults datacenter
ip import-table 10 route-map REDIST_NEIGHBOR
username cumulus nopassword
!
service integrated-vtysh-config
!
log syslog informational
!
router bgp 65001
 !
 address-family ipv4 unicast
  redistribute table 10
 exit-address-family
!
route-map REDIST_NEIGHBOR permit 10
 match interface swp1
!
route-map REDIST_NEIGHBOR permit 20
 match interface swp2
!
router ospf
 redistribute table 10
```

```
!
line vty
!
```

Configure the Hosts

There are a few possible host configurations that range in complexity. This document only covers the basic use case: dual-connected Linux hosts with static IP addresses assigned.

Configure a Dual-connected Host

Configure a host with the same /32 IP address on its loopback (lo) and uplinks (in this example, eth1 and eth2). This is done so both leaf switches advertise the same /32 regardless of the interface. Cumulus Linux relies on ECMP to load balance across the interfaces southbound, and an equal cost static route (see the configuration below) for load balancing northbound.

The loopback hosts the primary service IP address(es) and to which you can bind services.

Configure the loopback and physical interfaces. Referring back to the topology diagram, server01 is connected to leaf01 via eth1 and to leaf02 via eth2. You should note:

- The loopback IP is assigned to lo, eth1 and eth2.

- The post-up ARPing is used to force the host to ARP as soon as its interface comes up. This allows the leaf to learn about the host as soon as possible.
- The post-up `ip route replace` is used to install a default route via one or both leaf nodes if both swp1 and swp2 are up.

▼ Click to expand

Install ifplugd

Additionally, install and use `ifplugd`. `ifplugd` modifies the behavior of the Linux routing table when an interface undergoes a link transition (carrier up/down). The Linux kernel by default leaves routes up even when the physical interface is unavailable (NO-CARRIER).

After you install `ifplugd`, edit `/etc/default/ifplugd` as follows, where *eth1* and *eth2* are the interface names that your host uses to connect to the leaves.

```
user@server01:$ sudo nano /etc/default/ifplugd
INTERFACES="eth1 eth2"
HOTPLUG_INTERFACES=""
ARGS="-q -f -u10 -d10 -w -I"
SUSPEND_ACTION="stop"
```

For full instructions on installing `ifplugd` on Ubuntu, follow this guide.

Troubleshooting

How do I determine if rdnbrd (the redistribute neighbor daemon) is running?

Run the `systemctl status rdnbrd.service` command:

```
cumulus@leaf01:~$ systemctl status rdnbrd.service
* rdnbrd.service - Cumulus Linux Redistribute Neighbor Service
 Loaded: loaded (/lib/systemd/system/rdnbrd.service; enabled)
 Active: active (running) since Wed 2016-05-04 18:29:03 UTC; 1h
13min ago
 Main PID: 1501 (python)
 CGroup: /system.slice/rdnbrd.service
 `-1501 /usr/bin/python /usr/sbin/rdnbrd -d
```

How do I change the default configuration of rdnbrd?

Edit the `/etc/rdnbrd.conf` file, then run `systemctl restart rdnbrd.service`:

```
cumulus@leaf01:~$ sudo nano /etc/rdnbrd.conf
# syslog logging level CRITICAL, ERROR, WARNING, INFO, or DEBUG
loglevel = INFO
```

```
# TX an ARP request to known hosts every keepalive seconds
keepalive = 1

# If a host does not send an ARP reply for holdtime consider
the host down
holdtime = 3

# Install /32 routes for each host into this table
route_table = 10

# Uncomment to enable ARP debugs on specific interfaces.
# Note that ARP debugs can be very chatty.
# debug_arp = swp1 swp2 swp3 br1
# If we already know the MAC for a host, unicast the ARP
request. This is
# unusual for ARP (why ARP if you know the destination MAC) but
we will be
# using ARP as a keepalive mechanism and do not want to
broadcast so many ARPs
# if we do not have to. If a host cannot handle a unicasted ARP
request, set
#
# Unicasting ARP requests is common practice (in some
```

```
scenarios) for other
# networking operating systems so it is unlikely that you will
need to set
# this to False.
unicast_arp_requests = True
cumulus@leaf01:~$ sudo systemctl restart rdnbrd.service
```

What is table 10? Why was table 10 chosen?

The Linux kernel supports multiple routing tables and can utilize 0 through 255 as table IDs; however tables 0, 253, 254 and 255 are reserved, and 1 is usually the first one utilized. Therefore, `rdnbrd` only allows you to specify 2-252. Cumulus Linux uses table ID 10, however you can set the ID to any value between 2-252. You can see all the tables specified here:

```
cumulus@leaf01:~$ cat /etc/iproute2/rt_tables
#
# reserved values
#
255 local
254 main
253 default
0 unspec
```

```
#
# local
#
#1  inr.ruhep
```

For more information, refer to Linux route tables or you can read the Ubuntu man pages for ip route.

How do I determine that the /32 redistribute neighbor routes are being advertised to my neighbor?

For BGP, run the NCLU `net show bgp neighbor <interface> advertise-routes` command or the vtysh `show ip bgp neighbor swp51 advertised-routes` command. For example:

```
cumulus@leaf01:~$ net show bgp neighbor swp51 advertise-routes
BGP table version is 5, local router ID is 10.0.0.11
Status codes: s suppressed, d damped, h history, * valid, >
best, = multipath,
              i internal, r RIB-failure, S Stale, R Removed
Origin codes: i - IGP, e - EGP, ? - incomplete

    Network          Next Hop            Metric LocPrf Weight
```

```
Path
*> 10.0.0.11/32     0.0.0.0                  0         32768 i
*> 10.0.0.12/32     ::                                     0
65020 65012 i
*> 10.0.0.21/32     ::                                     0
65020 i
*> 10.0.0.22/32     ::                                     0
65020 i

Total number of prefixes 4
```

How do I verify that the kernel routing table is being correctly populated?

Use the following workflow to verify that the kernel routing table isbeing populated correctly and that routes are being correctly imported/ advertised:

1. Verify that ARP neighbor entries are being populated into the Kernel routing table 10.

```
cumulus@leaf01:~$ ip route show table 10
10.0.1.101 dev swp1 scope link
```

If these routes are not being generated, verify the following that the `rdnbrd` daemon is running and check the `/etc/rdnbrd.conf` file to verify the correct table number is used.

2. Verify that routes are being imported into FRRouting from the kernel routing table 10.

```
cumulus@leaf01:~$ sudo vtysh
leaf01# show ip route table
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, A - Babel, T - Table,
       > - selected route, * - FIB route
T[10]>* 10.0.1.101/32 [19/0] is directly connected, swp1,
01:25:29
```

Both the > and * should be present so that table 10 routes are installed as preferred into the routing table. If the routes are not being installed, verify the imported distance of the locally imported kernel routes with the `ip import 10 distance X` command (where X is **not** less than the administrative distance of the routing protocol). If the distance is too low, routes learned from the protocol might overwrite the locally imported routes. Also, verify that the routes are in the kernel routing table.

3. Confirm that routes are in the BGP/OSPF database and are being advertised.

```
leaf01# show ip bgp
```

Considerations

TCAM Route Scale

This feature adds each ARP entry as a /32 host route into the routing table of all switches within a summarization domain. Take care to keep the number of hosts minus fabric routes under the TCAM size of the switch. Review the Cumulus Networks datasheets for up to date scalability limits of your chosen hardware platforms. If in doubt, contact your support representative.

Possible Uneven Traffic Distribution

Linux uses *source* layer 3 addresses only to do load balancing on most older distributions.

Silent Hosts Never Receive Traffic

Freshly provisioned hosts that have never sent traffic may not ARP for their default gateways. The post-up ARPing in `/etc/network/interfaces` on the host should take care of this. If the host does not ARP, then `rdnbrd` on the leaf cannot learn about the host.

Unsupported with EVPN

Redistribute neighbor is unsupported when the BGP EVPN Address Family is enabled. Enabling both redistribute neighbor and EVPN will lead to unreachable IPv4 ARP and IPv6 neighbor entries.

Segment Routing

WARNING

Segment routing is an early access feature in Cumulus Linux and is supported only on switches with Spectrum ASICs.

Cumulus Linux supports *segment routing,* also known as source routing, where a source node can specify the path a packet should take (traffic engineering). In some more advanced cases, you can use segment routing to have offline multiprotocol label switching (MPLS) controllers program labels into the network for traffic engineering.

Cumulus Linux provides full label-based forwarding, relying on BGP for label exchange. However, Cumulus Linux does not provide LDP interoperability for MPLS and it does not support VRFs for tenant isolation.

Features

Segment routing is MPLS for the data plane **only**. In this EA release, Cumulus Linux does not impose the labels, the host does. The MTUs should be large enough to accommodate the MPLS shim header and label stack. Segment routing supports the following features:

- MPLS label edge router (LER) functionality for IPv4 and IPv6 routing with ECMP. An ingress LER first adds an MPLS label to an IP packet. An egress LER removes the outermost MPLS label (also called *popping* the label).
- MPLS label switch router (LSR) functionality with ECMP. The LSR receives a packet with a label and forwards it based on that label.
- FRRouting support for MPLS transit label switched paths (LSPs) and labeled routes (LER), both static routes and routes using BGP labeled-unicast (LU).
- FRR support for BGP/MPLS segment routing based on draft-ietf-idr-bgp-prefix-sid-06.

Configure Segment Routing

To configure the segment routing example above:

NCLU Commands | vtysh Commands

1. For each switch in the topology, add the label indexes:

```
cumulus@switch:~$ net add bgp network 10.1.1.1/32 label-
index 1
cumulus@switch:~$ net add bgp network 10.1.1.2/32 label-
index 2
cumulus@switch:~$ net add bgp network 10.1.1.3/32 label-
index 3
cumulus@switch:~$ net add bgp network 10.1.1.4/32 label-
index 4
cumulus@switch:~$ net add bgp network 10.1.1.5/32 label-
index 5
```

2. For each switch in the topology, define the *global-block* of labels to use for segment routing in FRR. The default global-block is 16000-23999. The example configuration uses global-block `100 200`. The *local label* is the MPLS label global-block plus the label-index.

```
cumulus@switch:~$ net add mpls label global-block 100 200
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and vtysh commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router bgp 65444
 bgp router-id 10.1.1.4
 no bgp default ipv4-unicast
 neighbor EBGP peer-group
 neighbor EBGP remote-as external
 neighbor swp1 interface peer-group EBGP
 neighbor swp2 interface peer-group EBGP
 !
 address-family ipv4 unicast
 network 10.1.1.1/32 label-index 1
 network 10.1.1.2/32 label-index 2
 network 10.1.1.3/32 label-index 3
 network 10.1.1.4/32 label-index 4
 network 10.1.1.5/32 label-index 5
exit-address-family
!
address-family ipv4 labeled-unicast
 neighbor EBGP activate
exit-address-family
!
mpls label global-block 100 200
```

```
...
```

View the Configuration

You can see the label-index when you show the BGP configuration on a router. Run the NCLU `net show configuration bgp` command or the vtysh `show running-config bgp` command. For example:

```
cumulus@r4:~$ net show configuration bgp
...
router bgp 65444
 bgp router-id 10.1.1.4


 address-family ipv4 unicast
 network 10.1.1.1/32 label-index 1
 network 10.1.1.2/32 label-index 2
 network 10.1.1.3/32 label-index 3
 network 10.1.1.4/32 label-index 4
 network 10.1.1.5/32 label-index 5
...
```

From another node in the network, run the NCLU `net show bgp <ip-address>` command or the vtysh `show ip bgp <ip-address>` command:

```
cumulus@r1:~$ net show bgp 10.1.1.4/32
BGP routing table entry for 10.1.1.4/32
Local label: 104
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to non peer-group peers:
  h1(swp1) r2(swp2) r4(swp3)
  400
    fe80::202:ff:fe00:c from r4(swp3) (10.1.1.4)
    (fe80::202:ff:fe00:c) (used)
      Origin IGP, metric 0, localpref 100, valid, external,
bestpath-from-AS 65444, best
      Remote label: 3
      Label Index: 4
      AddPath ID: RX 0, TX 14
      Last update: Tue Aug 15 13:57:45 2017
cumulus@r1:~$
```

To show the FRR MPLS table, run the NCLU `net show mpls table` command or the vtysh `show mpls table` command. You can see the FRR MPLS table in the output below, where r1 receives a packet with label 104. Its outbound label is 3, which appears as *implicit-null* below, so it pops then the payload is forwarded out of swp3, the interface to r4:

```
cumulus@r1:~$ net show mpls table

Inbound                                 Outbound
  Label     Type              Nexthop     Label
--------  -------  --------------------  --------
    102      BGP  fe80::202:ff:fe00:6         3
    103      BGP  fe80::202:ff:fe00:6       103
    104      BGP  fe80::202:ff:fe00:c         3
    105      BGP  fe80::202:ff:fe00:c       105
    106      BGP  fe80::202:ff:fe00:1         3
    107      BGP  fe80::202:ff:fe00:6       107

cumulus@r1:~$
cumulus@r1:~$ net show mpls table 104
Local label: 104 (installed)
 type: BGP remote label: implicit-null distance: 150
  via fe80::202:ff:fe00:c dev swp3 (installed)
cumulus@r1:~$
```

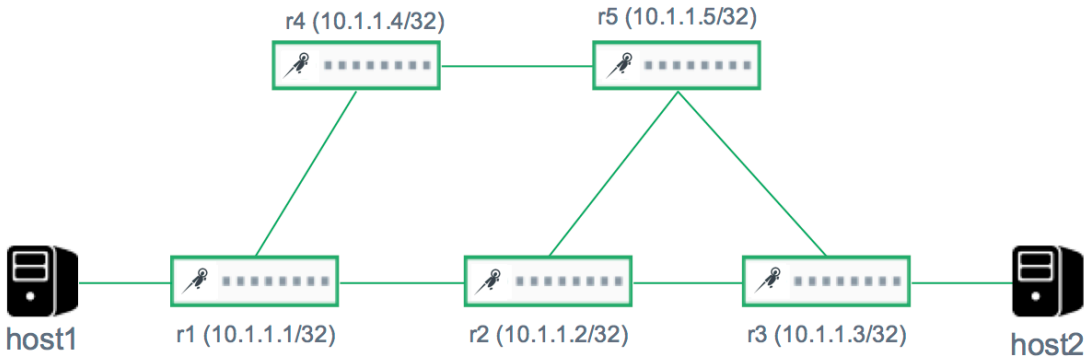
You can see the MPLS routing table that is installed in the kernel as well:

```
cumulus@r1:~$ ip -f mpls route show
102 via inet6 fe80::202:ff:fe00:6 dev swp2  proto zebra
```

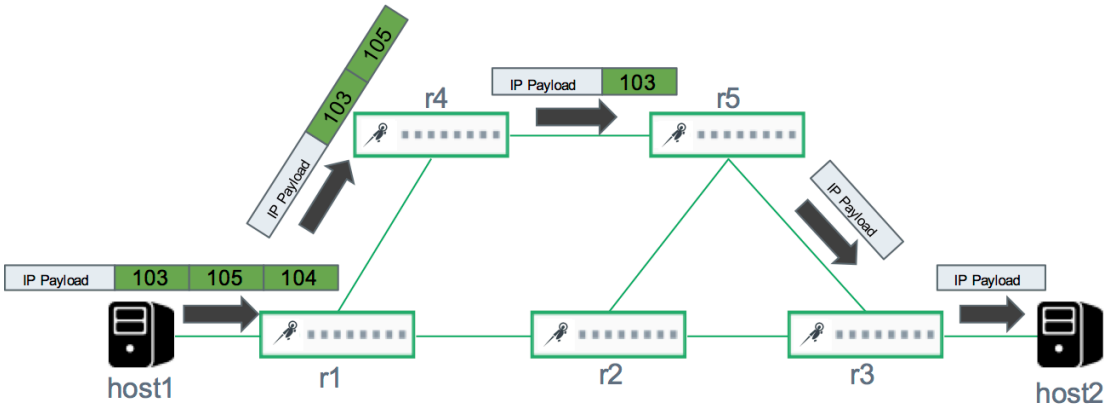
```
103 as to 103 via inet6 fe80::202:ff:fe00:6 dev swp2  proto
zebra
104 via inet6 fe80::202:ff:fe00:c dev swp3  proto zebra
105  proto zebra
    nexthop as to 105  via inet6 fe80::202:ff:fe00:6  dev swp2
    nexthop as to 105  via inet6 fe80::202:ff:fe00:c  dev swp3
106 via inet6 fe80::202:ff:fe00:1 dev swp1  proto zebra
107 as to 107 via inet6 fe80::202:ff:fe00:6 dev swp2  proto
zebra
cumulus@r1:~$
cumulus@r1:~$ ip -f mpls route show 104
104 via inet6 fe80::202:ff:fe00:c dev swp3  proto zebra
cumulus@r1:~$
```

Example Configuration

Consider the following topology. Typically, host1 sends traffic to host2 through r1, r2 and r3. However, you can use segment routing to route traffic through a specific path. In the examples below, HTTP traffic is routed from host1 to host2 via r1, r4, r5 then r3. In addition, FTP traffic is routed via r5 without worrying what path it takes to get there.

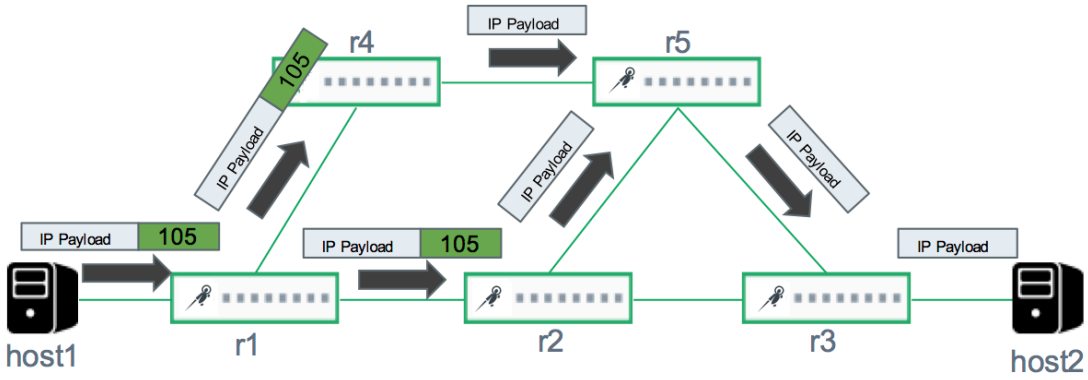


For HTTP traffic to be routed from host1 to host2 via r1, r4, r5 then r3, the MPLS controller tells host1 to push *label stack* 103,105,104 on all HTTP traffic destined for host2; 104 is the outside label and 103 is the inside label. Switch r1 sees label 104, then pops that outermost label and forwards the payload towards switch r4. Switch r4 sees label 105, then pops that label and forwards the payload towards switch r5. Switch r5 sees label 103, then pops that label and forwards the payload towards switch r3. Switch r3 sees just an IP packet, and routes it as usual.



For FTP traffic to be routed from host1 to host2 through r5, the MPLS controller tells host1 to push label stack 105 on all FTP traffic destined for host2. Switch r1 sees label 105, then uses ECMP using swap with label 105

and forwards the payload towards switches r4 and r2. Switches r2 and r4 see label 105, then they pop the label and forward the payload towards switches r5 and r3. Switches r5 and r3 both see just an IP packet and route it as usual.



Switches r1 through r5 announce their loopbacks (the 10.1.1.* addresses above) in BGP with a *label-index*.

The table below contains the configuration for all five nodes.

Node r1 | Node r2 | Node r3 | Node r4 | Node r5

/etc/network/interfaces

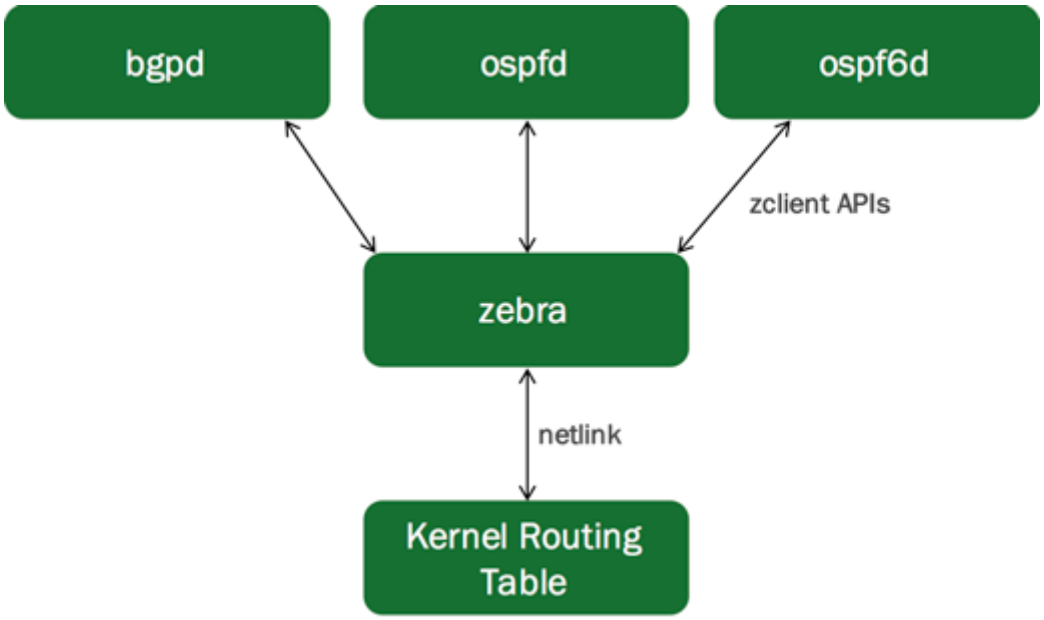
```
auto lo
iface lo inet loopback
    address 10.1.1.1/32

auto swp2
iface swp2

auto swp4
iface swp4

auto swp10
iface swp10
    address 192.168.11.1/24

auto vagrant
iface vagrant inet dhcp

auto eth0
iface eth0 inet dhcp
 vrf mgmt

auto mgmt
iface mgmt
  address 127.0.0.1/8
  address ::1/128
  vrf-table auto
```

FRRouting

Cumulus Linux uses FRRouting (FRR) to provide the routing protocols for dynamic routing and supports the following routing protocols:

- Open Shortest Path First (v2 and v3)
- Border Gateway Protocol - BGP

Architecture



The FRRouting suite consists of various protocol-specific daemons and a protocol-independent daemon called `zebra`. Each of the protocol-specific daemons are responsible for running the relevant protocol and building the routing table based on the information exchanged.

It is not uncommon to have more than one protocol daemon running at the same time. For example, at the edge of an enterprise, protocols internal to

an enterprise (called IGP for Interior Gateway Protocol) such as OSPF text or RIP run alongside the protocols that connect an enterprise to the rest of the world (called EGP or Exterior Gateway Protocol) such as BGP.

About zebra

`zebra` is the daemon that resolves the routes provided by multiple protocols (including the static routes you specify) and programs these routes in the Linux kernel via `netlink` (in Linux). The FRRouting documentation defines `zebra` as the IP routing manager for FRRouting that "provides kernel routing table updates, interface lookups, and redistribution of routes between different routing protocols."

Related Information

- FRRouting website
- FRRouting project on GitHub

Configure FRRouting

FRRouting does not start by default in Cumulus Linux. Before you run FRRouting, make sure you have enabled the relevant daemons that you intend to use (`bgpd`, `ospfd`, `ospf6d` or `pimd`) in the `/etc/frr/daemons` file.

> ⚠ **IMPORTANT**
>
> NVIDIA has not tested RIP, RIPv6, IS-IS and Babel.

The `zebra` daemon is enabled by default. You can enable the other daemons according to how you plan to route your network.

Before you start FRRouting, edit the `/etc/frr/daemons` file to enable each daemon you want to use. For example, to enable BGP, set `bgpd` to *yes*:

```
...
bgpd=yes
ospfd=no
ospf6d=no
ripd=no
ripngd=no
```

```
isisd=no
fabricd=no
pimd=no
ldpd=no
nhrpd=no
eigrpd=no
babeld=no
sharpd=no
pbrd=no
vrrpd=no
...
```

Enable and Start FRRouting

After you enable the appropriate daemons, enable and start the FRRouting service:

```
cumulus@switch:~$ sudo systemctl enable frr.service
cumulus@switch:~$ sudo systemctl start frr.service
```

TIP

All the routing protocol daemons (`bgpd`, `ospfd`, `ospf6d`, `ripd`, `ripngd`, `isisd` and `pimd`) are dependent on `zebra`. When you start FFRouting, `systemd` determines whether zebra is running; if zebra is not running, `systemd` starts `zebra`, then starts the dependent service, such as `bgpd`.

In general, if you restart a service, its dependent services are also restarted. For example, running `systemctl restart frr.service` restarts any of the routing protocol daemons that are enabled and running.

For more information on the `systemctl` command and changing the state of daemons, read Services and Daemons in Cumulus Linux.

Integrated Configurations

By default in Cumulus Linux, FRRouting saves all daemon configurations in a single integrated configuration file, `frr.conf`.

You can disable this mode by running the following command in the `vtysh` FRRouting CLI:

```
cumulus@switch:~$ sudo vtysh
switch# configure terminal
switch(config)# no service integrated-vtysh-config
```

To reenable integrated configuration file mode, run:

```
switch(config)# service integrated-vtysh-config
```

If you disable integrated configuration mode, FRRouting saves each daemon-specific configuration file in a separate file. At a minimum for a daemon to start, that daemon must be enabled and its daemon-specific configuration file must be present, even if that file is empty.

To save the current configuration:

```
switch# write memory
Building Configuration...
Integrated configuration saved to /etc/frr/frr.conf
[OK]
switch# exit
cumulus@switch:~$
```

> **NOTE**
>
> You can use `write file` instead of `write memory`.

When integrated configuration mode is disabled, the output looks like this:

```
switch# write memory
Building Configuration...
Configuration saved to /etc/frr/zebra.conf
Configuration saved to /etc/frr/bgpd.conf
[OK]
```

Restore the Default Configuration

If you need to restore the FRRouting configuration to the default running configuration, delete the `frr.conf` file and restart the `frr` service.

Back up `frr.conf` (or any configuration files you want to remove) before proceeding.

1. Confirm that `service integrated-vtysh-config` is enabled:

```
cumulus@switch:~$ net show configuration | grep integrated
service integrated-vtysh-config
```

2. Remove `/etc/frr/frr.conf`:

```
cumulus@switch:~$ sudo rm /etc/frr/frr.conf
```

If integrated configuration file mode is disabled, remove all the configuration files (such as `zebra.conf` or `ospf6d.conf`) instead of `frr.conf`.

3. Restart FRR with this command:

```
cumulus@switch:~$ sudo systemctl restart frr.service
```

WARNING

Restarting FRR restarts all the routing protocol daemons that are enabled and running.

Interface IP Addresses and VRFs

FRRouting inherits the IP addresses and any associated routing tables for the network interfaces from the `/etc/network/interfaces` file. This is the recommended way to define the addresses; do **not** create interfaces using FRRouting. For more information, see Configure IP Addresses and Virtual Routing and Forwarding - VRF.

FRRouting vtysh Modal CLI

FRRouting provides a command-line interface (CLI) called `vtysh` for configuring and displaying protocol state. To start the CLI, run the `sudo vtysh` command:

```
cumulus@switch:~$ sudo vtysh

Hello, this is FRRouting (version 0.99.23.1+cl3u2).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

switch#
```

`vtysh` provides a Cisco-like modal CLI and many of the commands are similar to Cisco IOS commands. There are different modes to the CLI and certain commands are only available within a specific mode. Configuration is available with the `configure terminal` command:

```
switch# configure terminal
switch(config)#
```

The prompt displays the current CLI mode. For example, when the interface-specific commands are invoked, the prompt changes to:

```
switch(config)# interface swp1
switch(config-if)#
```

When the routing protocol specific commands are invoked, the prompt changes to:

```
switch(config)# router ospf
switch(config-router)#
```

`?` displays the list of available top-level commands:

```
switch(config-if)# ?
  bandwidth    Set bandwidth informational parameter
  description  Interface specific description
  end          End current mode and change to enable mode
  exit         Exit current mode and down to previous mode
  ip           IP Information
  ipv6         IPv6 Information
  isis         IS-IS commands
  link-detect  Enable link detection on interface
  list         Print command list
  mpls-te      MPLS-TE specific commands
  multicast    Set multicast flag to interface
  no           Negate a command or set its defaults
  ptm-enable   Enable neighbor check with specified topology
  quit         Exit current mode and down to previous mode
  shutdown     Shutdown the selected interface
```

?-based completion is also available to see the parameters that a command takes:

```
switch(config-if)# bandwidth ?
<1-10000000>  Bandwidth in kilobits
switch(config-if)# ip ?
```

```
address  Set the IP address of an interface
irdp     Alter ICMP Router discovery preference this interface
ospf     OSPF interface commands
rip      Routing Information Protocol
router   IP router interface commands
```

To search for specific `vtysh` commands so that you can identify the correct syntax to use, run the `sudo vtysh -c 'find <term>'` command. For example, to show only commands that include `mlag`:

```
cumulus@leaf01:mgmt:~$ sudo vtysh -c 'find mlag'
  (view)  show ip pim [mlag] vrf all interface [detail|WORD]
[json]
  (view)  show ip pim [vrf NAME] interface [mlag] [detail|WORD]
[json]
  (view)  show ip pim [vrf NAME] mlag upstream [A.B.C.D
[A.B.C.D]] [json]
  (view)  show ip pim mlag summary [json]
  (view)  show ip pim vrf all mlag upstream [json]
  (view)  show zebra mlag
  (enable)  [no$no] debug zebra mlag
  (enable)  debug pim mlag
```

```
  (enable)  no debug pim mlag
  (enable)  test zebra mlag
<none$none|primary$primary|secondary$secondary>
  (enable)  show ip pim [mlag] vrf all interface [detail|WORD]
[json]
  (enable)  show ip pim [vrf NAME] interface [mlag]
[detail|WORD] [json]
  (enable)  show ip pim [vrf NAME] mlag upstream [A.B.C.D
[A.B.C.D]] [json]
  (enable)  show ip pim mlag summary [json]
  (enable)  show ip pim vrf all mlag upstream [json]
  (enable)  show zebra mlag
  (config)  [no$no] debug zebra mlag
  (config)  debug pim mlag
  (config)  ip pim mlag INTERFACE role [primary|secondary]
state [up|down] addr A.B.C.D
  (config)  no debug pim mlag
  (config)  no ip pim mlag
```

Displaying state can be done at any level, including the top level. For example, to see the routing table as seen by `zebra`:

```
switch# show ip route
```

```
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, T - Table,
       > - selected route, * - FIB route
B>* 0.0.0.0/0 [20/0] via fe80::4638:39ff:fe00:c, swp29, 00:11:57
  *                  via fe80::4638:39ff:fe00:52, swp30,
00:11:57
B>* 10.0.0.1/32 [20/0] via fe80::4638:39ff:fe00:c, swp29,
00:11:57
  *                    via fe80::4638:39ff:fe00:52, swp30,
00:11:57
B>* 10.0.0.11/32 [20/0] via fe80::4638:39ff:fe00:5b, swp1,
00:11:57
B>* 10.0.0.12/32 [20/0] via fe80::4638:39ff:fe00:2e, swp2,
00:11:58
B>* 10.0.0.13/32 [20/0] via fe80::4638:39ff:fe00:57, swp3,
00:11:59
B>* 10.0.0.14/32 [20/0] via fe80::4638:39ff:fe00:43, swp4,
00:11:59
C>* 10.0.0.21/32 is directly connected, lo
B>* 10.0.0.51/32 [20/0] via fe80::4638:39ff:fe00:c, swp29,
00:11:57
  *                     via fe80::4638:39ff:fe00:52, swp30,
00:11:57
B>* 172.16.1.0/24 [20/0] via fe80::4638:39ff:fe00:5b, swp1,
```

```
00:11:57
  *                     via fe80::4638:39ff:fe00:2e, swp2,
00:11:57
B>* 172.16.3.0/24 [20/0] via fe80::4638:39ff:fe00:57, swp3,
00:11:59
  *                     via fe80::4638:39ff:fe00:43, swp4,
00:11:59
```

To run the same command at a config level, prepend `do` to it:

```
switch(config-router)# do show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, T - Table,
       > - selected route, * - FIB route
B>* 0.0.0.0/0 [20/0] via fe80::4638:39ff:fe00:c, swp29, 00:05:17
  *                  via fe80::4638:39ff:fe00:52, swp30,
00:05:17
B>* 10.0.0.1/32 [20/0] via fe80::4638:39ff:fe00:c, swp29,
00:05:17
  *                    via fe80::4638:39ff:fe00:52, swp30,
00:05:17
B>* 10.0.0.11/32 [20/0] via fe80::4638:39ff:fe00:5b, swp1,
00:05:17
```

```
B>* 10.0.0.12/32 [20/0] via fe80::4638:39ff:fe00:2e, swp2,
00:05:18
B>* 10.0.0.13/32 [20/0] via fe80::4638:39ff:fe00:57, swp3,
00:05:18
B>* 10.0.0.14/32 [20/0] via fe80::4638:39ff:fe00:43, swp4,
00:05:18
C>* 10.0.0.21/32 is directly connected, lo
B>* 10.0.0.51/32 [20/0] via fe80::4638:39ff:fe00:c, swp29,
00:05:17
  *                    via fe80::4638:39ff:fe00:52, swp30,
00:05:17
B>* 172.16.1.0/24 [20/0] via fe80::4638:39ff:fe00:5b, swp1,
00:05:17
  *                     via fe80::4638:39ff:fe00:2e, swp2,
00:05:17
B>* 172.16.3.0/24 [20/0] via fe80::4638:39ff:fe00:57, swp3,
00:05:18
  *                     via fe80::4638:39ff:fe00:43, swp4,
00:05:18
```

To run single commands with `vtysh`, use the `-c` option of `vtysh`:

```
cumulus@switch:~$ sudo vtysh -c 'sh ip route'
```

```
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

K>* 0.0.0.0/0 via 192.168.0.2, eth0
C>* 192.0.2.11/24 is directly connected, swp1
C>* 192.0.2.12/24 is directly connected, swp2
B>* 203.0.113.30/24 [200/0] via 192.0.2.2, swp1, 11:05:10
B>* 203.0.113.31/24 [200/0] via 192.0.2.2, swp1, 11:05:10
B>* 203.0.113.32/24 [200/0] via 192.0.2.2, swp1, 11:05:10
C>* 127.0.0.0/8 is directly connected, lo
C>* 192.168.0.0/24 is directly connected, eth0
```

To run a command multiple levels down:

```
cumulus@switch:~$ sudo vtysh -c 'configure terminal' -c 'router
ospf' -c 'area 0.0.0.1 range 10.10.10.0/24'
```

Notice that the commands also take a partial command name (for example, `sh ip route`) as long as the partial command name is not aliased:

```
cumulus@switch:~$ sudo vtysh -c 'sh ip r'
% Ambiguous command.
```

To disable a command or feature in FRRouting, prepend the command with `no`. For example:

```
cumulus@switch:~$ sudo vtysh


switch# configure terminal
switch(config)# router ospf
switch(config-router)# no area 0.0.0.1 range 10.10.10.0/24
switch(config-router)# exit
switch(config)# exit
switch# write mem
switch# exit
cumulus@switch:~$
```

To view the current state of the configuration, run the `show running-config` command:

▼ Example command

> **NOTE**
>
> If you try to configure a routing protocol that has not been started, `vtysh` silently ignores those commands.

If you do not want to use a modal CLI to configure FRRouting, you can use a suite of Cumulus Linux-specific commands instead.

Reload the FRRouting Configuration

If you make a change to your routing configuration, you need to reload FRRouting so your changes take place. *FRRouting reload* enables you to apply only the modifications you make to your FRRouting configuration, synchronizing its running state with the configuration in `/etc/frr/frr.conf`. This is useful for optimizing FRRouting automation in your environment or to apply changes made at runtime.

> **NOTE**
>
> FRRouting reload only applies to an integrated service configuration, where your FRRouting configuration is stored in a single `frr.conf` file instead of one configuration file per FRRouting daemon (like `zebra` or `bgpd`).

To reload your FRRouting configuration after you modify `/etc/frr/frr.conf`, run:

```
cumulus@switch:~$ sudo systemctl reload frr.service
```

Examine the running configuration and verify that it matches the configuration in `/etc/frr/frr.conf`:

```
cumulus@switch:~$ net show configuration
```

If the running configuration is not what you expect, submit a support request and supply the following information:

- The current running configuration (run `net show configuration` and output the contents to a file)
- The contents of `/etc/frr/frr.conf`
- The contents of `/var/log/frr/frr-reload.log`

FRR Logging

By default, Cumulus Linux configures FFR with syslog severity level 6 (informational). Log output is written to the `/var/log/frr/frr.log` file.

> **NOTE**
>
> To write debug messages to the log file, you must run the `log syslog debug` command to configure FRR with syslog severity 7 (debug); otherwise, when you issue a debug command such as, `debug bgp neighbor-events`, no output is sent to `/var/log/frr/frr.log`. However, when you manually define a log target with the `log file /var/log/frr/debug.log` command, FRR automatically defaults to severity 7 (debug) logging and the output is logged to `/var/log/frr/debug.log`.

Considerations

Obfuscated Passwords

In FRRouting, Cumulus Linux stores obfuscated passwords for BGP and OSPF (ISIS, OSPF area, and BGP neighbor passwords). All passwords in configuration files and those displayed in `show` commands are obfuscated. The obfuscation algorithm protects passwords from casual viewing. The system can retrieve the original password when needed.

Duplicate Hostnames

If you change the hostname, either with NCLU or with the `hostname` command in `vtysh`, the switch can have two hostnames in the FRR

configuration. For example:

```
Spine01# configure terminal
Spine01(config)# hostname Spine01-1
Spine01-1(config)# do sh run
Building configuration...
Current configuration:
!
frr version 7.0+cl4u3
frr defaults datacenter
hostname Spine01
hostname Spine01-1
...
```

NOTE

Accidentally configuring the same numbered BGP neighbor using both the `neighbor x.x.x.x` and `neighbor swp# interface` commands results in two neighbor entries being present for the same IP address in the configuration and operationally. To correct this issue, update the configuration and restart the FRR service.

Related Information

- FRR BGP documentation

- FRR IPv6 support
- FRR Zebra documentation

Comparing NCLU and vtysh Commands

Using NCLU is the recommended way to configure routing in Cumulus Linux; however, you can use the `vtysh` modal CLI.

The following table shows the FRRouting commands and the equivalent Cumulus Linux NCLU commands.

Action	NCLU Commands	FRRouting Commands
Display the routing table	`cumulus@switch:~$ net show route`	`switch# show ip route`
Create a new neighbor	`cumulus@switch:~$ net add bgp autonomous-system 65002` `cumulus@switch:~$ net add bgp neighbor`	`switch(config)# router bgp 65002` `switch(config-router)# neighbor 14.0.0.22`

Action	NCLU Commands	FRRouting Commands
	`14.0.0.22`	
Redistribute routing information from static route entries into RIP tables	`cumulus@switch:~$ net add bgp redistribute static`	`switch(config)# router bgp 65002` `switch(config-router)# redistribute static`
Define a static route	`cumulus@switch:~$ net add routing route 155.1.2.20/24 bridge 45`	`switch(config)# ip route 155.1.2.20/24 bridge 45`

Action	NCLU Commands	FRRouting Commands
Configure an IPv6 address	`cumulus@switch:~$ net add interface swp3 ipv6 address 3002:2123:1234:1abc 64`	`switch(config)# int swp3 switch(config-if)# ipv6 address 3002:2123:1234:1abc::21/64`
Enable topology checking (PTM)	`cumulus@switch:~$ net add routing ptm-enable`	`switch(config)# ptm-enable`
Configure MTU in IPv6 network discovery for an interface	`cumulus@switch:~$ sudo cl-ra interface swp3 set mtu 9000`	`switch(config)# int swp3 switch(config-if)# ipv6 nd mtu 9000`

Action	NCLU Commands	FRRouting Commands
Set the OSPF interface priority	`cumulus@switch:~$ net add interface swp3 ospf6 priority 120`	`switch(config)# int swp3` `switch(config-if)# ip ospf6 priority 120`
Configure timing for OSPF SPF calculations	`cumulus@switch:~$ net add ospf6 timers throttle spf 40 50 60`	`switch(config)# router ospf6` `switch(config-ospf6)# timer throttle spf 40 50 60`
Configure the OSPF Hello packet interval in number of seconds for an interface	`cumulus@switch:~$ net add interface swp4 ospf6`	`switch(config)# int swp4` `switch(config-if)# ipv6`

Action	NCLU Commands	FRRouting Commands
	`hello-interval 60`	`ospf6 hello-interval 60`
Display BGP information	`cumulus@switch:~$ net show bgp summary`	`switch# show ip bgp summary`
Display OSPF debugging status	`cumulus@switch:~$ net show debugs`	`switch# show debugging ospf`
Show information about the interfaces on the switch	`cumulus@switch:~$ net show interface`	`switch# show interface`

Action	NCLU Commands	FRRouting Commands
		To quickly check important information, such as IP address, VRF, and operational status, in easy to read tabular format: `switch# show interface brief`

Border Gateway Protocol - BGP

BGP is the routing protocol that runs the Internet. It manages how packets get routed from network to network by exchanging routing and reachability information.

BGP is an increasingly popular protocol for use in the data center as it lends itself well to the rich interconnections in a Clos topology. RFC 7938 provides further details about using BGP in the data center.

How does BGP Work?

BGP directs packets between autonomous systems (AS), which are a set of routers under a common administration. Each router maintains a routing table that controls how packets are forwarded. The BGP process on the router generates information in the routing table based on information coming from other routers and from information in the BGP routing information base (RIB). The RIB is a database that stores routes and continually updates the routing table as changes occur.

Autonomous System

Because BGP was originally designed to peer between independently managed enterprises and service providers, each such enterprise is treated as an autonomous system responsible for a set of network addresses. Each such autonomous system is given a unique number called an *autonomous*

system number (ASN). ASNs are handed out by a central authority (ICANN); however, ASNs between 64512 and 65535 are reserved for private use. Using BGP within the data center relies on either using this number space or using the single ASN you own.

The ASN is central to how BGP builds a forwarding topology. A BGP route advertisement carries with it not only the ASN of the originator, but also the list of ASNs that this route advertisement passes through. When forwarding a route advertisement, a BGP speaker adds itself to this list. This list of ASNs is called the *AS path*. BGP uses the AS path to detect and avoid loops.

ASNs were originally 16-bit numbers, but were later modified to be 32-bit. FRRouting supports both 16-bit and 32-bit ASNs, but many implementations still run with 16-bit ASNs.

> **NOTE**
>
> In a VRF-lite deployment (where multiple independent routing tables work simultaneously on the same switch), Cumulus Linux supports multiple ASNs. Multiple ASNs are not supported in deployments that use EVPN or VRF route leaking.

Auto BGP

In a two-tier leaf and spine environment, you can use *auto BGP* to generate 32-bit ASNs automatically so that you don't have to think about which

numbers to allocate. Auto BGP helps build optimal ASN configurations in your data center to avoid suboptimal routing and path hunting, which occurs when you assign the wrong spine ASNs. Auto BGP makes no changes to standard BGP behavior or configuration.

Auto BGP assigns private ASNs in the range 4200000000 through 4294967294. This is the private space defined in RFC 6996. Each leaf is assigned a random and unique value in the range 4200000001 through 4294967294. Each spine is assigned 4200000000; the first number in the range. For information about configuring auto BGP, refer to Basic BGP Configuration.

NOTE

- Use auto BGP in new deployments to avoid conflicting ASNs in an existing configuration.
- It is not necessary to use auto BGP across all switches in your configuration. For example, you can use auto BGP to configure one switch but allocate ASNs manually to other switches.
- Auto BGP is intended for use in two-tier spine and leaf networks. Using auto BGP in three-tier networks with superspines might result in incorrect ASN assignments.
- The `leaf` keyword generates the ASN based on a hash of the switch MAC address. The ASN assigned might change

after a switch replacement.

- You can configure auto BGP with NCLU only.

eBGP and iBGP

When BGP is used to peer between autonomous systems, the peering is referred to as *external BGP* or eBGP. When BGP is used within an autonomous system, the peering used is referred to as *internal BGP* or iBGP. eBGP peers have different ASNs while iBGP peers have the same ASN.

The heart of the protocol is the same when used as eBGP or iBGP but there is a key difference in the protocol behavior between eBGP and iBGP. To prevent loops, an iBGP speaker does not forward routing information learned from one iBGP peer to another iBGP peer. eBGP prevents loops using the `AS_Path` attribute.

All iBGP speakers need to be peered with each other in a full mesh. In a large network, this requirement can quickly become unscalable. The most popular method to scale iBGP networks is to introduce a route reflector.

BGP Path Selection

BGP is a path-vector routing algorithm that does not rely on a single routing metric to determine the lowest cost route, unlike interior gateway

protocols (IGPs) like OSPF.

The BGP path selection algorithm looks at multiple factors to determine exactly which path is best. BGP multipath is enabled by default in Cumulus Linux so that multiple equal cost routes can be installed in the routing table but only a single route is advertised to BGP peers.

The order of the BGP algorithm process is as follows:

- **Highest Weight**: Weight is a value from 0 to 65535. Weight is not carried in a BGP update but is used locally to influence the best path selection. Locally generated routes have a weight of 32768.
- **Highest Local Preference**: Local preference is exchanged between iBGP neighbors only. Routes received from eBGP peers are assigned a local preference of 0. Whereas weight is used to make route selections without sending additional information to peers, local preference can be used to influence routing to iBGP peers.
- **Locally Originated Routes**: Any route that the local switch is responsible for placing into BGP is selected as best. This includes static routes, aggregate routes and redistributed routes.
- **Shortest AS Path**: The path received with the fewest number of ASN hops is selected.
- **Origin Check**: Preference is given to routes with an IGP origin (routes placed into BGP with a `network` statement) over incomplete origins (routes places into BGP through redistribution). The EGP origin attribute is no longer used.

- **Lowest MED**: The Multi-Exit Discriminator or MED is sent to eBGP peers to indicate a preference on how traffic enters an AS. A MED received from an eBGP peer is exchanged with iBGP peers but is reset to a value of 0 before advertising a prefix to another AS.
- **eBGP Routes**: A route received from an eBGP peer is preferred over a route learned from an iBGP peer.
- **Lowest IGP Cost to the Next Hop**: The route with the lowest IGP metric to reach the BGP next hop.
- **iBGP ECMP over eBGP ECMP**: If BGP multipath is configured, prefer equal iBGP routes over equal eBGP routes, unless as-path multipath-relax is also configured.
- **Oldest Route**: Preference is given to the oldest route in the BGP table.
- **Lowest Router ID**: Preference is given to the route received from the peer with the lowest Router ID attribute. If the route is received from a route reflector, the `ORIGINATOR_ID` attribute is used for comparison.
- **Shortest Route Reflector Cluster List**: If a route passes through multiple route reflectors, prefer the route with the shortest route reflector cluster list.
- **Highest Peer IP Address**: Preference is given to the route received from the peer with the highest IP address.

Cumulus Linux provides the reason it selects one path over another in

NCLU `net show bgp` and vtysh `show ip bgp` command output for a specific prefix.

When BGP multipath is in use, if multiple paths are equal, BGP still selects a single best path to advertise to peers. This path is indicated as best with the reason, although multiple paths might be installed into the routing table.

BGP Unnumbered

Historically, peers connect over IPv4 and TCP port 179, and after they establish a session, they exchange prefixes. When a BGP peer advertises an IPv4 prefix, it must include an IPv4 next hop address, which is usually the address of the advertising router. This requires that each BGP peer has an IPv4 address, which in a large network can consume a lot of address space, requiring a separate IP address for each peer-facing interface.

The BGP unnumbered standard, specified in RFC 5549, uses *extended next hop encoding* (ENHE) and no longer requires an IPv4 prefix to be advertised along with an IPv4 next hop. This means that you can set up BGP peering between your Cumulus Linux switches and exchange IPv4 prefixes without having to configure an IPv4 address on each switch; the interfaces that BGP uses are unnumbered.

The next hop address for each prefix is an IPv6 link-local address, which is assigned automatically to each interface. Using the IPv6 link-local address as a next hop instead of an IPv4 unicast address, BGP unnumbered saves you from having to configure IPv4 addresses on each interface.

When you use BGP unnumbered, BGP learns the prefixes, calculates the routes and installs them in IPv4 AFI to IPv6 AFI format. ENHE in Cumulus Linux does not install routes into the kernel in IPv4 prefix to IPv6 next hop format. For link-local peerings enabled by dynamically learning the other end's link-local address using IPv6 neighbor discovery router advertisements, an IPv6 next hop is converted into an IPv4 link-local address and a static neighbor entry is installed for this IPv4 link-local address with the MAC address derived from the link-local address of the other end.

NOTE

- If an IPv4 /30 or /31 IP address is assigned to the interface, IPv4 peering is used over IPv6 link-local peering.
- BGP unnumbered only works with two switches at a time, as it is designed to work with point-to-point links.
- The IPv6 implementation on the peering device uses the MAC address as the interface ID when assigning the IPv6 link-local address, as suggested by RFC 4291.
- Every router or end host must have an IPv4 address to complete a `traceroute` of IPv4 addresses. In this case, the IPv4 address used is that of the loopback device. Even if extended next-hop encoding (ENHE) is not used in the data center, link addresses are not typically advertised because

they take up valuable FIB resources and also expose an additional attack vector for intruders to use to either break in or engage in DDOS attacks. Assigning an IP address to the loopback device is essential.

Related Information

- BGP in the Data Center by Dinesh G. Dutt - a complete guide to Border Gateway Protocol for the modern data center
- Bidirectional forwarding detection (BFD) and BGP
- Wikipedia entry for BGP (includes list of useful RFCs)
- FRR BGP documentation
- IETF draft discussing BGP use within data centers
- RFC 1657, Definitions of Managed Objects for the Fourth Version of the Border Gateway Protocol (BGP-4) using SMIv2
- RFC 1997, BGP Communities Attribute
- RFC 2385, Protection of BGP Sessions via the TCP MD5 Signature Option
- RFC 2439, BGP Route Flap Damping
- RFC 2545, Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing
- RFC 2918, Route Refresh Capability for BGP-4
- RFC 4271, A Border Gateway Protocol 4 (BGP-4)
- RFC 4760, Multiprotocol Extensions for BGP-4

- RFC 5004, Avoid BGP Best Path Transitions from One External to Another
- RFC 5065, Autonomous System Confederations for BGP
- RFC 5291, Outbound Route Filtering Capability for BGP-4
- RFC 5492, Capabilities Advertisement with BGP-4
- RFC 5549, Advertising IPv4 Network Layer Reachability Information with an IPv6 Next Hop
- RFC 6793, BGP Support for Four-Octet Autonomous System (AS) Number Space
- RFC 7911, Advertisement of Multiple Paths in BGP
- draft-walton-bgp-hostname-capability-02, Fully Qualified Domain Name Capability for BGP

Basic BGP Configuration

This section describes how to configure BGP using either BGP numbered or BGP unnumbered. With BGP *unnumbered*, you can set up BGP peering between your Cumulus Linux switches and exchange IPv4 prefixes without having to configure an IPv4 address on each switch.

> **NOTE**
>
> BGP *unnumbered* simplifies configuration and is recommended for data center deployments.

BGP Numbered

To configure BGP numbered on a BGP node, you need to:

- Assign an ASN to identify this BGP node. In a two-tier leaf and spine configuration, you can use auto BGP, where Cumulus Linux assigns an ASN automatically.
- Assign a router ID, which is a 32-bit value and is typically the address of the loopback interface on the switch.
- Specify where to distribute routing information by providing the IP address and ASN of the neighbor.
  - For BGP numbered, this is the IP address of the interface between the two peers; the interface must be a layer 3 access port.
  - The ASN can be a number, or `internal` for a neighbor in the same AS

or `external` for a neighbor in a different AS.

- Specify which prefixes to originate from this BGP node.

NCLU Commands | vtysh Commands

leaf01 | spine01

1. Identify the BGP node by assigning an ASN.

   - To assign an ASN manually:

     ```
     cumulus@leaf01:~$ net add bgp autonomous-system
     65101
     ```

   - To use auto BGP to assign an ASN automatically on the leaf:

     ```
     cumulus@leaf01:~$ net add bgp auto leaf
     ```

     The auto BGP `leaf` keyword is only used to configure the ASN. The configuration files and `net show` commands display the AS number.

2. Assign the router ID.

   ```
   cumulus@leaf01:~$ net add bgp router-id 10.10.10.1
   ```

3. Specify the BGP neighbor to which you want to distribute routing information.

   ```
   cumulus@leaf01:~$ net add bgp neighbor 10.0.1.0
   ```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

leaf01 spine01

```
cumulus@leaf01:~$  sudo cat /etc/frr/frr.conf
...
router bgp 65101
 bgp router-id 10.10.10.1
 neighbor 10.0.1.0 remote-as external
 !
 address-family ipv4 unicast
  network 10.10.10.1/32
  network 10.1.10.0/24
 exit-address-family
...
```

NOTE

When using auto BGP, there are no references to `leaf` or `spine` in the configurations. Auto BGP determines the ASN for the system and configures it using standard vtysh commands.

BGP Unnumbered

The following example commands show a basic BGP unnumbered configuration for two switches, leaf01 and spine01, which are eBGP peers.

The only difference between a BGP unnumbered configuration and the BGP numbered configuration shown above is that the BGP neighbor is specified as an interface (insead of an IP address). The interface between the two peers does **not** need to have an IP address configured on each side.

NCLU Commands vtysh Commands

leaf01 spine01

```
cumulus@leaf01:~$ net add bgp autonomous-system 65101
cumulus@leaf01:~$ net add bgp router-id 10.10.10.1
cumulus@leaf01:~$ net add bgp neighbor swp51 remote-as
external
cumulus@leaf01:~$ net add bgp ipv4 unicast network
10.10.10.1/32
cumulus@leaf01:~$ net add bgp ipv4 unicast network
10.1.10.0/24
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

For BGP to advertise IPv6 prefixes, you need to run an additional command to activate the BGP neighbor under the IPv6 address family. The IPv4 address family is enabled by default and the `activate` command is not required for IPv4 route exchange.

```
cumulus@leaf01:~$ net add bgp autonomous-system 65101
cumulus@leaf01:~$ net add bgp router-id 10.10.10.1
cumulus@leaf01:~$ net add bgp neighbor swp51 remote-as
external
cumulus@leaf01:~$ net add bgp ipv6 unicast neighbor
swp51 activate
cumulus@leaf01:~$ net add bgp ipv6 unicast network
2001:db8::1/128
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The NCLU and vtysh commands save the configuration in the `/etc/frr/frr.conf` file. For example:

leaf01 spine01

```
cumulus@leaf01:~$  sudo cat /etc/frr/frr.conf
...
router bgp 65101
 bgp router-id 10.10.10.1
 neighbor swp51 interface
 neighbor swp51 remote-as external
 !
 address-family ipv4 unicast
  network 10.10.10.1/32
  network 10.1.10.0/24
 exit-address-family
...
```

Optional BGP Configuration

This section describes optional configuration. The steps provided in this section assume that you already configured basic BGP as described in Basic BGP Configuration.

Peer Groups

Instead of specifying properties of each individual peer, you can define one or more peer groups and associate all the attributes common to that peer session to a peer group. A peer needs to be attached to a peer group only once, when it then inherits all address families activated for that peer group.

> **NOTE**
>
> If the peer you want to add to a group already exists in the BGP configuration, delete it first, than add it to the peer group.

The following example commands create a peer group called SPINE that includes two external peers.

NCLU Commands vtysh Commands

```
cumulus@leaf01:~$ net add bgp neighbor SPINE peer-group
cumulus@leaf01:~$ net add bgp neighbor SPINE remote-as
external
cumulus@leaf01:~$ net add bgp neighbor 10.0.1.0 peer-group
SPINE
cumulus@leaf01:~$ net add bgp neighbor 10.0.1.12 peer-group
SPINE
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

For an unnumbered configuration, you can use a single command to configure a neighbor and attach it to a peer group.

NCLU Commands vtysh Commands

```
cumulus@leaf01:~$ net add bgp neighbor swp51 interface peer-
group SPINE
```

BGP Dynamic Neighbors

BGP dynamic neighbor provides BGP peering to a group of remote neighbors within a specified range of IPv4 or IPv6 addresses for a BGP peer

group. You can configure each range as a subnet IP address.

You configure dynamic neighbors using the `bgp listen range <ip-address> peer-group <group>` command. After you configure the dynamic neighbors, a BGP speaker can listen for, and form peer relationships with, any neighbor that is in the IP address range and is mapped to a peer group.

The following example commands create the peer group SPINE and configure BGP peering to remote neighbors within the address range 10.0.1.0/31.

NCLU Commands | vtysh Commands

```
cumulus@leaf01:~$ net add bgp neighbor SPINE peer-group
cumulus@leaf01:~$ net add bgp neighbor SPINE remote-as
external
cumulus@leaf01:~$ net add bgp listen range 10.0.1.0/24 peer-
group SPINE
cumulus@leaf01:~$ net add bgp listen limit 5
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The `net add bgp listen limit` command limits the number of dynamic peers. The default value is *100*.

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/`

`frr.conf` file. For example:

```
router bgp 65101
  neighbor SPINE peer-group
  neighbor SPINE remote-as external
  bgp listen limit 5
  bgp listen range 10.0.1.0/24 peer-group SPINE
```

eBGP Multihop

The eBGP multihop option lets you use BGP to exchange routes with an external peer that is more than one hop away.

To establish a connection between two eBGP peers that are not directly connected:

NCLU Commands | vtysh Commands

```
cumulus@leaf01:~$ net add bgp neighbor 10.10.10.101 remote-
as external
cumulus@leaf01:~$ net add bgp neighbor 10.10.10.101 ebgp-
multihop
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

BGP TTL Security Hop Count

You can use the TTL security hop count option to prevent attacks against eBGP, such as denial of service (DoS) attacks. By default, BGP messages are sent to eBGP neighbors with an IP time-to-live (TTL) of 1, which requires the peer to be directly connected, otherwise, the packets expire along the way. (You can adjust the TTL with the eBGP multihop option.) An attacker can easily adjust the TTL of packets so that they appear to be originating from a peer that is directly connected.

The BGP TTL security hops option inverts the direction in which the TTL is counted. Instead of accepting only packets with a TTL set to 1, only BGP messages with a TTL greater than or equal to 255 minus the specified hop count are accepted.

When TTL security is in use, eBGP multihop is no longer needed.

The following command example sets the TTL security hop count value to 200:

NCLU Commands | vtysh Commands

```
cumulus@leaf01:~$ net add bgp neighbor swp51 ttl-security
hops 200
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router bgp 65101
  ...
  neighbor swp51 ttl-security hops 200
...
```

NOTE

- When you configure `ttl-security hops` on a peer group instead of a specific neighbor, FRR does not add it to either the running configuration or to the `/etc/frr/frr.conf` file. To work around this issue, add `ttl-security hops` to individual neighbors instead of the peer group.
- Enabling `ttl-security hops` does not program the hardware with relevant information. Frames are forwarded to the CPU and are dropped. Use the `net add acl` command to explicitly add the relevant entry to hardware. For more information about ACLs, see Netfilter - ACLs.

MD5-enabled BGP Neighbors

You can authenticate your BGP peer connection to prevent interference with your routing tables.

To enable MD5 authentication for BGP peers, set the same password on each peer.

The following example commands set the password *mypassword* on BGP peers leaf01 and spine01:

NCLU Commands | vtysh Commands

leaf01 | spine01

```
cumulus@leaf01:~$ net add bgp neighbor swp51 password
mypassword
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

You can confirm the configuration with the NCLU command `net show bgp neighbor <neighbor>` or with the `vtysh` command `show ip bgp neighbor <neighbor>`.

▼ net show bgp neighbor <neighbor> example

> **NOTE**
>
> The MD5 password configured against a BGP listen-range peer group (used to accept and create dynamic BGP neighbors) is not enforced; connections are accepted from peers that do not specify a password.

Remove Private ASNs

If you use private ASNs in the data center, any routes you send out to the internet contain your private ASNs. You can remove all the private ASNs from routes to a specific neighbor.

The following example command removes private ASNs from routes sent to the neighbor on swp51 (an unnumbered interface):

```
cumulus@switch:~$ net add bgp neighbor swp51 remove-private-AS
```

You can replace the private ASNs with your public ASN with the following command:

```
cumulus@switch:~$ net add bgp neighbor swp51 remove-private-AS
```

```
replace-AS
```

ECMP

BGP supports equal-cost multipathing (ECMP). If a BGP node hears a certain prefix from multiple peers, it has all the information necessary to program the routing table and forward traffic for that prefix through all of these peers. BGP typically choses one best path for each prefix and installs that route in the forwarding table.

In Cumulus Linux, the *BGP multipath* option is enabled by default with the maximum number of paths set to 64 so that the switch can install multiple equal-cost BGP paths to the forwarding table and load balance traffic across multiple links. You can change the number of paths allowed, according to your needs.

The example commands change the maximum number of paths to 120. You can set a value between 1 and 256. 1 disables the BGP multipath option.

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add bgp maximum-paths 120
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `address-family` stanza of the `/etc/frr/frr.conf` file. For example:

```
...
!
address-family ipv4 unicast
 network 10.1.10.0/24
 network 10.10.10.1/32
 maximum-paths 120
exit-address-family
...
```

When *BGP multipath* is enabled, only BGP routes from the same AS are load balanced. If the routes go across several different AS neighbors, even if the AS path length is the same, they are not load balanced. To be able to load balance between multiple paths received from different AS neighbors, you need to set the `bestpath as-path multipath-relax` option.

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add bgp bestpath as-path multipath-
relax
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router bgp 65101
  bgp router-id 10.0.0.1
  bgp bestpath as-path multipath-relax
...
```

> **NOTE**
>
> When you disable the *bestpath as-path multipath-relax* option, EVPN type-5 routes do not use the updated configuration. Type-5 routes continue to use all available ECMP paths in the underlay fabric, regardless of ASN.

Advertise IPv4 Prefixes with IPv6 Next Hops

RFC 5549 defines the method used for BGP to advertise IPv4 prefixes with IPv6 next hops. The RFC does not make a distinction between whether the IPv6 peering and next hop values should be global unicast addresses (GUA) or link-local addresses. Cumulus Linux supports advertising IPv4 prefixes with IPv6 global unicast and link-local next hop addresses, with either *unnumbered* or *numbered* BGP.

When BGP peering uses IPv6 global addresses and IPv4 prefixes are being advertised and installed, IPv6 route advertisements are used to derive the MAC address of the peer so that FRR can create an IPv4 route with a link-local IPv4 next hop address (defined by RFC 3927). This is required to install the route into the kernel. These route advertisement settings are configured automatically when FRR receives an update from a BGP peer using IPv6 global addresses that contain an IPv4 prefix with an IPv6 next hop, and the enhanced-next hop capability has been negotiated.

To enable advertisement of IPv4 prefixes with IPv6 next hops over global IPv6 peerings, add the `extended-nexthop` capability to the global IPv6 neighbor statements on each end of the BGP sessions.

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add bgp neighbor
2001:db8:0002::0a00:0002 capability extended-nexthop
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
```

```
router bgp 65101
  ...
  neighbor 2001:db8:0002::0a00:0002 capability extended-nexthop
...
```

Ensure that the IPv6 peers are activated under the IPv4 unicast address family; otherwise, all peers are activated in the IPv4 unicast address family by default. If `no bgp default ipv4-unicast` is configured, you need to explicitly activate the IPv6 neighbor under the IPv4 unicast address family as shown below:

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add bgp neighbor
2001:db8:0002::0a00:0002 capability extended-nexthop
cumulus@switch:~$ net add bgp ipv4 unicast neighbor
2001:db8:0002::0a00:0002 activate
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router bgp 65101
router-id 10.10.10.1
no bgp default ipv4-unicast
neighbor 2001:db8:0002::0a00:0002 remote-as external
neighbor 2001:db8:0002::0a00:0002 capability extended-nexthop
!
address-family ipv4 unicast
  neighbor 2001:db8:0002::0a00:0002 activate
exit-address-family
...
```

Neighbor Maximum Prefixes

To protect against an internal network connectivity disruption caused by BGP, you can control how many route announcements (prefixes) can be received from a BGP neighbor.

The following example commands set the maximum number of prefixes allowed from the BGP neighbor on swp51 to 3000:

```
cumulus@leaf01:~$ sudo vtysh
```

```
leaf01# configure terminal
leaf01(config)# router bgp 65001
leaf01(config-router)# neighbor swp51 maximum-prefix 3000
leaf01(config-router)# end
leaf01# write memory
leaf01# exit
cumulus@leaf01:~$
```

Aggregate Addresses

To minimize the size of the routing table and save bandwidth, you can aggregate a range of networks in your routing table into a single prefix.

The following example command aggregates a range of addresses, such as 10.1.1.0/24, 10.1.2.0/24, 10.1.3.0/24 into the single prefix 10.1.0.0/16.

```
cumulus@switch:~$ net add bgp aggregate-address 10.1.0.0/16
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The `summary-only` option ensures that longer-prefixes inside the aggregate address are suppressed before sending BGP updates:

```
cumulus@switch:~$ net add bgp aggregate-address 10.1.0.0/16
summary-only
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

BGP add-path

Cumulus Linux supports both BGP add-path RX and BGP add-path TX.

BGP add-path RX

BGP add-path RX allows BGP to receive multiple paths for the same prefix. A path identifier is used so that additional paths do not override previously advertised paths. BGP add-path RX is enabled by default; no additional configuration is required.

To view the existing capabilities, run the NCLU command `net show bgp neighbor` or the `vtysh` command `show ip bgp neighbors`. The existing capabilities are listed in the subsection *Add Path*, below *Neighbor capabilities.*

The following example output shows that additional BGP paths can be sent and received and that the BGP neighbor on swp51 supports both.

```
cumulus@leaf01:~$ net show bgp neighbor
BGP neighbor on swp51: fe80::7c41:fff:fe93:b711, remote AS
65199, local AS 65101, external link
Hostname: spine01
  BGP version 4, remote router ID 10.10.10.101, local router ID
10.10.10.1
  BGP state = Established, up for 1d12h39m
  Last read 00:00:03, Last write 00:00:01
  Hold time is 9, keepalive interval is 3 seconds
  Neighbor capabilities:
    4 Byte AS: advertised and received
    AddPath:
      IPv4 Unicast: RX advertised IPv4 Unicast and received
    Extended nexthop: advertised and received
      Address families by peer:
                  IPv4 Unicast
    Route refresh: advertised and received(old & new)
    Address Family IPv4 Unicast: advertised and received
    Hostname Capability: advertised (name: leaf01,domain name:
n/a) received (name: spine01,domain name: n/a)
    Graceful Restart Capability: advertised and received
...
```

To view the current additional paths, run the NCLU command `net show bgp`

`<prefix>` or the `vtysh` command `show ip bgp <prefix>`. The example output shows an additional path that has been added by the TX node for receiving. Each path has a unique AddPath ID.

```
cumulus@leaf01:mgmt:~$ net show bgp 10.10.10.9
BGP routing table entry for 10.10.10.9/32
Paths: (2 available, best #1, table Default-IP-Routing-Table)
  Advertised to non peer-group peers:
  spine01(swp51) spine02(swp52)
  65020 65012
    fe80::4638:39ff:fe00:5c from spine01(swp51) (10.10.10.12)
    (fe80::4638:39ff:fe00:5c) (used)
      Origin incomplete, localpref 100, valid, external,
multipath, bestpath-from-AS 65020, best (Older Path)
      AddPath ID: RX 0, TX 6
      Last update: Wed Nov 16 22:47:00 2016
  65020 65012
    fe80::4638:39ff:fe00:2b from spine02(swp52) (10.10.10.12)
    (fe80::4638:39ff:fe00:2b) (used)
      Origin incomplete, localpref 100, valid, external,
multipath
      AddPath ID: RX 0, TX 3
      Last update: Fri Oct  2 03:56:33 2020
```

BGP add-path TX

BGP add-path TX enables BGP to advertise more than just the best path for a prefix. Cumulus Linux includes two options:

- `addpath-tx-all-paths` advertises all known paths to a neighbor
- `addpath-tx-bestpath-per-AS` advertises only the best path learned from each AS to a neighbor

The following example commands configure leaf01 to advertise the best path learned from each AS to the BGP neighbor on swp50:

NCLU Commands | vtysh Commands

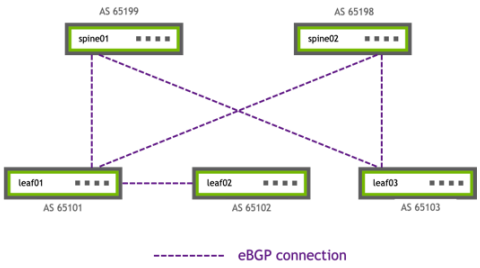
```
cumulus@leaf01:~$ net add bgp autonomous-system 65101
cumulus@leaf01:~$ net add bgp neighbor swp50 addpath-tx-
bestpath-per-AS
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The following example commands configure leaf01 to advertise all paths learned from each AS to the BGP neighbor on swp50:

NCLU Commands vtysh Commands

```
cumulus@leaf01:~$ net add bgp autonomous-system 65101
cumulus@leaf01:~$ net add bgp neighbor swp50 addpath-tx-all-
paths
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The following example configuration shows how BGP add-path TX is used to advertise the best path learned from each AS.

	In this configuration: • Every leaf and every spine has a different ASN • eBGP is configured between: ◦ leaf01 and spine01, spine02 ◦ leaf03 and spine01, spine02 ◦ leaf01 and leaf02 (leaf02 only has a single peer, which is leaf01) • leaf01 is configured to advertise the best path learned from each AS to BGP neighbor leaf02 • leaf03 generates a loopback IP address (10.10.10.3/32) into BGP with a network statement

When you run the `net show bgp 10.10.10.3/32` command on leaf02, the command output shows the leaf03 loopback IP address and that two BGP paths are learned, both from leaf01:

```
cumulus@leaf02:mgmt:~$ net show bgp 10.10.10.3/32
BGP routing table entry for 10.10.10.3/32
Paths: (2 available, best #2, table default)
       Advertised to non peer-group peers:
       leaf01(swp50)
  65101 65199 65103
    fe80::4638:39ff:fe00:13 from leaf01(swp50) (10.10.10.1)
    (fe80::4638:39ff:fe00:13) (used)
      Origin IGP, valid, external
      AddPath ID: RX 4, TX-All 0 TX-Best-Per-AS 0
      Last update: Thu Oct 15 18:31:46 2020
  65101 65198 65103
    fe80::4638:39ff:fe00:13 from leaf01(swp50) (10.10.10.1)
    (fe80::4638:39ff:fe00:13) (used)
      Origin IGP, valid, external, bestpath-from-AS 65101, best
(Nothing left to compare)
      AddPath ID: RX 3, TX-All 0 TX-Best-Per-AS 0
      Last update: Thu Oct 15 18:31:46 2020
```

BGP Timers

BGP includes several timers that you can configure.

Keepalive Interval and Hold Time

By default, BGP exchanges periodic keepalive messages to measure and ensure that a peer is still alive and functioning. If a keepalive or update message is not received from the peer within the hold time, the peer is declared down and all routes received by this peer are withdrawn from the local BGP table. By default, the keepalive interval is set to 3 seconds and the hold time is set to 9 seconds. To decrease CPU load, especially in the presence of a lot of neighbors, you can increase the values of these timers or disable the exchange of keepalives entirely. When manually configuring new values, the keepalive interval can be less than or equal to one third of the hold time, but cannot be less than 1 second. Setting the keepalive and hold time values to 0 disables the exchange of keepalives.

The following example commands set the keepalive interval to 10 seconds and the hold time to 30 seconds.

NCLU Commands | vtysh Commands

```
cumulus@leaf01:~$ net add bgp neighbor swp51 timers 10 30
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router bgp 65101
  ...
  neighbor swp51 timers 10 30
...
```

Reconnect Interval

By default, the BGP process attempts to connect to a peer after a failure (or on startup) every 10 seconds. You can change this value to suit your needs.

The following example commands set the reconnect value to 30 seconds:

NCLU Commands | vtysh Commands

```
cumulus@leaf01:~$ net add bgp neighbor swp51 timers connect
30
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router bgp 65101
  ...
  neighbor swp51 timers connect 30
...
```

Advertisement Interval

After making a new best path decision for a prefix, BGP can optionally insert a delay before advertising the new results to a peer. This delay is used to rate limit the amount of changes advertised to downstream peers and lowers processing requirements by slowing down convergence. By default, this interval is set to 0 seconds for both eBGP and iBGP sessions, which allows for very fast convergence. For more information about the advertisement interval, see this IETF draft.

The following example commands set the advertisement interval to 5 seconds:

NCLU Commands vtysh Commands

```
cumulus@leaf01:~$ net add bgp neighbor swp51 advertisement-
interval 5
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

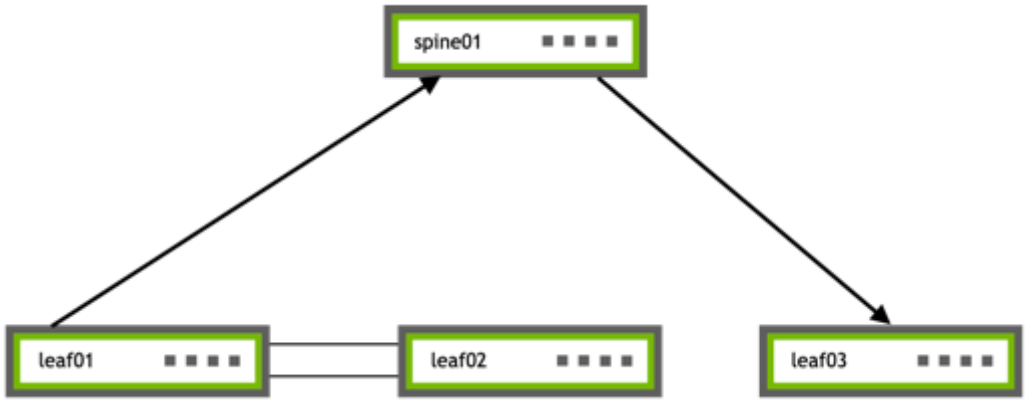
```
...
router bgp 65101
  ...
  neighbor swp51 advertisement-interval 5
...
```

Route Reflectors

iBGP rules state that a route learned from an iBGP peer can not be sent to another iBGP peer. In a data center spine and leaf network using iBGP, this prevents a spine from sending a route learned from a leaf to any other leaf. As a workaround, BGP introduced the concept of a *route reflector* that selectively ignores this rule so that when an iBGP speaker is configured as a

route reflector, it *can* send iBGP learned routes to other iBGP peers.

In the following example, spine01 is acting as a route reflector. The leaf switches, leaf01, leaf02 and leaf03 are *route reflector clients*. Any route that spine01 learns from a route reflector client is sent to other route reflector clients.



To configure the BGP node as a route reflector for a BGP peer, set the neighbor `route-reflector-client` option. The following example sets spine01 shown in the illustration above to be a route reflector for leaf01 (on swp1), which is a route reflector client. No configuration is required on the client.

NCLU Commands vtysh Commands

```
cumulus@spine01:~$ net add bgp neighbor swp1 route-
reflector-client
cumulus@spine01:~$ net pending
cumulus@spine01:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router bgp 65199
 bgp router-id 10.10.10.101
 neighbor swp51 remote-as external
 !
 address-family ipv4 unicast
  network 10.10.10.101/32
  neighbor swp51 route-reflector-client
 exit-address-family
...
```

> ⚠ **IMPORTANT**
>
> When configuring BGP for IPv6, you must run the `route-reflector-client` command **after** the `activate` command; otherwise, the `route-reflector-client` command is ignored.

Administrative Distance

Cumulus Linux uses the administrative distance to choose which routing protocol to use when two different protocols provide route information for the same destination. The smaller the distance, the more reliable the protocol. For example, if the switch receives a route from OSPF with an administrative distance of 110 and the same route from BGP with an administrative distance of 100, the switch chooses BGP.

Set the administrative distance with vtysh commands.

The following example commands set the administrative distance for routes from 10.10.10.101 to 100:

```
cumulus@spine01:~$ sudo vtysh
```

```
spine01# configure terminal
spine01(config)# router bgp 65101
spine01(config-router)# distance 100 10.10.10.101/32
spine01(config-router)# end
spine01# write memory
spine01# exit
cumulus@spine01:~$
```

The following example commands set the administrative distance for routes external to the AS to 150, routes internal to the AS to 110, and local routes to 100:

```
cumulus@spine01:~$ sudo vtysh

spine01# configure terminal
spine01(config)# router bgp 65101
spine01(config-router)# distance bgp 150 110 100
spine01(config-router)# end
spine01# write memory
spine01# exit
cumulus@spine01:~$
```

Graceful BGP Shutdown

To reduce packet loss during planned maintenance of a router or link, you can configure graceful BGP shutdown, which forces traffic to route around the BGP node:

NCLU Commands | vtysh Commands

To enable graceful shutdown:

```
cumulus@leaf01:~$ net add bgp graceful-shutdown
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

To disable graceful shutdown:

```
cumulus@leaf01:~$ net del bgp graceful-shutdown
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

When configured, the `graceful-shutdown` community is added to all paths from eBGP peers and the `local-pref` for that route is set to `0`. To see the configuration, run the NCLU command `net show bgp <route>` or the `vtysh`

command `show ip bgp <route>`. For example:

```
cumulus@switch:~$ net show bgp 10.10.10.0/24
BGP routing table entry for 10.10.10.0/24
Paths: (2 available, best #1, table Default-IP-Routing-Table)
  Advertised to non peer-group peers:
  bottom0(10.10.10.2)
  30 20
    10.10.10.2 (metric 10) from top1(10.10.10.2) (10.10.10.2)
      Origin IGP, localpref 100, valid, internal, bestpath-from-
AS 30, best
      Community: 99:1
      AddPath ID: RX 0, TX 52
      Last update: Mon Sep 18 17:01:18 2017

  20
    10.10.10.3 from bottom0(10.10.10.32) (10.10.10.10)
      Origin IGP, metric 0, localpref 0, valid, external,
bestpath-from-AS 20
      Community: 99:1 graceful-shutdown
      AddPath ID: RX 0, TX 2
      Last update: Mon Sep 18 17:01:18 2017
```

Enable Read-only Mode

As BGP peers are established and updates are received, prefixes might be

installed in the RIB and advertised to BGP peers even though the information from all peers is not yet received and processed. Depending on the timing of the updates, prefixes might be installed and propagated through BGP, and then immediately withdrawn and replaced with new routing information. Read-only mode minimizes this BGP route churn in both the local RIB and with BGP peers.

Enable read-only mode to reduce CPU and network usage when restarting the BGP process. Because intermediate best paths are possible for the same prefix as peers get established and start receiving updates at different times, read-only mode is particularly useful in topologies where BGP learns a prefix from many peers and the network has a high number of prefixes.

> **NOTE**
>
> While in read-only mode, BGP does not run best-path or generate any updates to its peers.

To enable read-only mode, you set the `max-delay` timer and, optionally, the `establish-wait` timer. Read-only mode begins as soon as the first peer reaches its established state and the `max-delay` timer starts, and continues until either of the following two conditions are met:

- All the configured peers (except the shutdown peers) have sent an explicit EOR (End-Of-RIB) or an implicit EOR. The first keep-alive after

BGP reaches the established state is considered an implicit EOR. If you specify the `establish-wait` option, BGP only considers peers that have reached the established state from the moment the `max-delay` timer starts until the `establish-wait` period ends. The minimum set of established peers for which EOR is expected are the peers that are established during the `establish-wait` window, not necessarily all the configured neighbors.

- The timer reaches the configured `max-delay`.

The default value for `max-delay` is 0, which disables read-only mode. The `update delay` and `establish wait` can be any value between 0 and 3600 seconds. The `establish-wait` setting is optional; however, if specified, it must be shorter than the `max-delay`.

The following example commands enable read-only mode by setting the `max-delay` timer to 300 seconds and the `establish-wait` timer to 90 seconds.

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add bgp update-delay 300 90
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

To show information about the state of the update delay, run the NCLU

command `net show bgp summary` or the `vtysh` command `show ip bgp summary`.

BGP Community Lists

You can use *community lists* to define a BGP community to tag one or more routes. You can then use the communities to apply a route policy on either egress or ingress.

The BGP community list can be either *standard* or *expanded.* The standard BGP community list is a pair of values (such as *100:100*) that can be tagged on a specific prefix and advertised to other neighbors or applied on route ingress. Or, it can be one of four BGP default communities:

- *internet*: a BGP community that matches all routes
- *local-AS*: a BGP community that restricts routes to your confederation's sub-AS
- *no-advertise*: a BGP community that is not advertised to anyone
- *no-export*: a BGP community that is not advertised to the eBGP peer

An expanded BGP community list takes a regular expression of communities and matches the listed communities.

When the neighbor receives the prefix, it examines the community value and takes action accordingly, such as permitting or denying the community member in the routing policy.

Here is an example of a standard community list filter:

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add routing community-list standard
COMMUNITY1 permit 100:100
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

You can apply the community list to a route map to define the routing policy:

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add bgp table-map ROUTE-MAP1
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Related Information

- RFC 4360, BGP Extended Communities Attribute
- RFC 4456, BGP Route Reflection - An Alternative to Full Mesh Internal BGP (iBGP)

Troubleshooting

Use the following commands to troubleshoot BGP.

Basic Troubleshooting Commands

The following example commands run on a BGP unnumbered configuration and show IPv6 next hops or the interface name for any IPv4 prefix.

To show a summary of the BGP configuration on the switch, run the NCLU `net show bgp summary` command or the vtysh `show ip bgp summary` command. For example:

```
cumulus@switch:~$ net show bgp summary
how bgp ipv4 unicast summary
============================
BGP router identifier 10.10.10.1, local AS number 65101 vrf-id 0
BGP table version 88
RIB entries 25, using 4800 bytes of memory
Peers 5, using 106 KiB of memory
Peer groups 1, using 64 bytes of memory

Neighbor              V         AS   MsgRcvd   MsgSent
TblVer  InQ OutQ  Up/Down State/PfxRcd
spine01(swp51)        4      65199     31122     31194
```

```
0    0    0 1d01h44m            7
spine02(swp52)        4      65199     31060     31151
0    0    0 01:47:13            7
spine03(swp53)        4      65199     31150     31207
0    0    0 01:48:31            7
spine04(swp54)        4      65199     31042     31098
0    0    0 01:46:57            7
leaf02(peerlink.4094) 4      65101     30919     30913
0    0    0 01:47:43           12

Total number of neighbors 5



show bgp ipv6 unicast summary
=============================
% No BGP neighbors found
```

TIP

To determine if the sessions above are iBGP or eBGP sessions, look at the ASNs.

To view the routing table as defined by BGP, run the NCLU `net show bgp ipv4 unicast` command or the vtysh `show ip bgp` command. For example:

```
cumulus@leaf01:~$ net show bgp ipv4 unicast
GP table version is 88, local router ID is 10.10.10.1, vrf id 0
Default local pref 100, local AS 65101
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
* i10.0.1.1/32      peerlink.4094            0    100      0 ?
*>                  0.0.0.0                  0         32768 ?
*= 10.0.1.2/32      swp54                                   0
65199 65102 ?
*=                  swp52                                   0
65199 65102 ?
* i                 peerlink.4094                 100      0
65199 65102 ?
*=                  swp53                                   0
65199 65102 ?
*>                  swp51                                   0
```

```
65199 65102 ?
*= 10.0.1.254/32    swp54                                  0
65199 65132 ?
*=                  swp52                                  0
65199 65132 ?
* i                 peerlink.4094                 100      0
65199 65132 ?
*=                  swp53                                  0
65199 65132 ?
*>                  swp51                                  0
65199 65132 ?
*> 10.10.10.1/32    0.0.0.0                  0         32768 ?
*>i10.10.10.2/32    peerlink.4094            0    100      0 ?
*= 10.10.10.3/32    swp54                                  0
65199 65102 ?
*=                  swp52                                  0
65199 65102 ?
* i                 peerlink.4094                 100      0
65199 65102 ?
*=                  swp53                                  0
65199 65102 ?
*>                  swp51                                  0
65199 65102 ?
...
```

```
Displayed  13 routes and 42 total paths
```

To show a more detailed breakdown of a specific neighbor, run the NCLU `net show bgp neighbor <neighbor>` command or the vtysh `show ip bgp neighbor <neighbor>` command:

```
cumulus@switch:~$ net show bgp neighbor swp51
GP neighbor on swp51: fe80::7c41:fff:fe93:b711, remote AS
65199, local AS 65101, external link
Hostname: spine01
 Member of peer-group underlay for session parameters
  BGP version 4, remote router ID 10.10.10.101, local router ID
10.10.10.1
  BGP state = Established, up for 1d01h47m
  Last read 00:00:00, Last write 00:00:00
  Hold time is 9, keepalive interval is 3 seconds
  Neighbor capabilities:
    4 Byte AS: advertised and received
    AddPath:
      IPv4 Unicast: RX advertised IPv4 Unicast and received
    Extended nexthop: advertised and received
      Address families by peer:
```

```
                  IPv4 Unicast
    Route refresh: advertised and received(old & new)
    Address Family IPv4 Unicast: advertised and received
    Hostname Capability: advertised (name: leaf01,domain name:
n/a) received (name: spine01,domain name: n/a)
    Graceful Restart Capability: advertised
  Graceful restart information:
    Local GR Mode: Helper*
    Remote GR Mode: Disable
    R bit: False
    Timers:
      Configured Restart Time(sec): 120
      Received Restart Time(sec): 0
  Message statistics:
    Inq depth is 0
    Outq depth is 0
                         Sent       Rcvd
    Opens:                  2          1
    Notifications:          0          0
    Updates:              309        237
    Keepalives:         30942      30943
    Route Refresh:          0          0
    Capability:             0          0
    Total:              31253      31181
```

```
  Minimum time between advertisement runs is 0 seconds


 For address family: IPv4 Unicast
  underlay peer-group member
  Update group 2, subgroup 2
  Packet Queue length 0
  Community attribute sent to this neighbor(all)
  7 accepted prefixes


  Connections established 1; dropped 0
  Last reset 1d01h47m,  No AFI/SAFI activated for peer
Local host: fe80::2294:15ff:fe02:7bbf, Local port: 179
Foreign host: fe80::7c41:fff:fe93:b711, Foreign port: 45548
Nexthop: 10.10.10.1
Nexthop global: fe80::2294:15ff:fe02:7bbf
Nexthop local: fe80::2294:15ff:fe02:7bbf
BGP connection: shared network
BGP Connect Retry Timer in Seconds: 10
Read thread: on  Write thread: on  FD used: 30
```

To see details of a specific route, such as from where it is received and where it is sent, run the NCLU `net show bgp <route>` command or the vtysh `show ip bgp <route>` command.

```
cumulus@switch:~$ net show bgp 10.10.10.3/32
GP routing table entry for 10.10.10.3/32
Paths: (5 available, best #5, table default)
  Advertised to non peer-group peers:
  spine01(swp51) spine02(swp52) spine03(swp53) spine04(swp54)
leaf02(peerlink.4094)
  65199 65102
    fe80::8e24:2bff:fe79:7d46 from spine04(swp54) (10.10.10.104)
    (fe80::8e24:2bff:fe79:7d46) (used)
      Origin incomplete, valid, external, multipath
      Last update: Wed Oct  7 13:13:13 2020
  65199 65102
    fe80::841:43ff:fe27:caf from spine02(swp52) (10.10.10.102)
    (fe80::841:43ff:fe27:caf) (used)
      Origin incomplete, valid, external, multipath
      Last update: Wed Oct  7 13:13:14 2020
  65199 65102
    fe80::90b1:7aff:fe00:3121 from leaf02(peerlink.4094)
(10.10.10.2)
      Origin incomplete, localpref 100, valid, internal
      Last update: Wed Oct  7 13:13:08 2020
  65199 65102
    fe80::48e7:fbff:fee9:5bcf from spine03(swp53) (10.10.10.103)
    (fe80::48e7:fbff:fee9:5bcf) (used)
```

```
      Origin incomplete, valid, external, multipath
      Last update: Wed Oct  7 13:13:13 2020
  65199 65102
    fe80::7c41:fff:fe93:b711 from spine01(swp51) (10.10.10.101)
    (fe80::7c41:fff:fe93:b711) (used)
      Origin incomplete, valid, external, multipath, bestpath-
from-AS 65199, best (Older Path)
      Last update: Wed Oct  7 13:13:13 2020
```

Troubleshoot BGP Unnumbered

To verify that FRR learned the neighboring link-local IPv6 address through the IPv6 neighbor discovery router advertisements on a given interface, run the NCLU `net show interface <interface>` command or the vtysh `show interface <interface>` command.

If `ipv6 nd suppress-ra` is not enabled on both ends of the interface, `Neighbor address(s):` has the other end's link-local address (the address that BGP uses when BGP is enabled on that interface).

> **NOTE**
>
> IPv6 route advertisements (RAs) are automatically enabled on an

interface with IPv6 addresses. The `no ipv6 nd suppress-ra` command is not needed for BGP unnumbered.

```
cumulus@switch:~$ net show interface swp51
    Name   MAC                Speed  MTU   Mode
--  -----  -----------------  -----  ----  -------
UP  swp51  10:d8:68:d4:a6:81  1G     9216  Default


Alias
-----
leaf to spine


cl-netstat counters
-------------------
RX_OK  RX_ERR  RX_DRP  RX_OVR  TX_OK  TX_ERR  TX_DRP  TX_OVR
-----  ------  ------  ------  -----  ------  ------  ------
 1874       0       0       0   1252       0       0       0


LLDP Details
------------
LocalPort  RemotePort(RemoteHost)
---------  ----------------------
```

```
swp51      swp1(spine01)


Routing
-------
  Interface swp51 is up, line protocol is up
  Link ups:       0    last: (never)
  Link downs:     0    last: (never)
  PTM status: disabled
  vrf: default
  OS Description: leaf to spine
  index 8 metric 0 mtu 9216 speed 1000
  flags: <UP,BROADCAST,RUNNING,MULTICAST>
  Type: Ethernet
  HWaddr: 10:d8:68:d4:a6:81
  inet6 fe80::12d8:68ff:fed4:a681/64
  Interface Type Other
  protodown: off
  ND advertised reachable time is 0 milliseconds
  ND advertised retransmit interval is 0 milliseconds
  ND advertised hop-count limit is 64 hops
  ND router advertisements sent: 217 rcvd: 216
  ND router advertisements are sent every 10 seconds
  ND router advertisements lifetime tracks ra-interval
  ND router advertisement default router preference is medium
```

```
  Hosts use stateless autoconfig for addresses.
  Neighbor address(s):
  inet6 fe80::f208:5fff:fe12:cc8c/128
```

Troubleshoot IPv4 Prefixes Learned with IPv6 Next Hops

To show IPv4 prefixes learned with IPv6 next hops, run the following commands.

The following examples show an IPv4 prefix learned from a BGP peer over an IPv6 session using IPv6 global addresses, but where the next hop installed by BGP is a link-local IPv6 address. This occurs when the session is directly between peers and both link-local and global IPv6 addresses are included as next hops in the BGP update for the prefix. If both global and link-local next hops exist, BGP prefers the link-local address for route installation.

```
cumulus@spine01:mgmt:~$ net show bgp ipv4 unicast summary
BGP router identifier 10.10.10.101, local AS number 65199 vrf-
id 0
BGP table version 3
RIB entries 3, using 576 bytes of memory
```

```
Peers 1, using 21 KiB of memory

Neighbor                    V      AS   MsgRcvd   MsgSent
TblVer  InQ OutQ  Up/Down State/PfxRcd
leaf01(2001:db8:2::a00:1) 4     65101       22        22
0    0    0  00:01:00           0

Total number of neighbors 1
```

```
cumulus@spine01:mgmt:~$ net show bgp ipv4 unicast
BGP table version is 3, local router ID is 10.10.10.101, vrf id
0
Default local pref 100, local AS 65199
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
   10.10.10.101/32   fe80::a00:27ff:fea6:b9fe      0     0
32768 i
```

```
Displayed  1 routes and 1 total paths
```

```
cumulus@spine01:~$ net show bgp ipv4 unicast 10.10.10.101/32
BGP routing table entry for 10.10.10.101/32
Paths: (1 available, best #1, table default)
  Advertised to non peer-group peers:
  Leaf01(2001:db8:0002::0a00:1)
  3
    2001:db8:0002::0a00:1 from Leaf01(2001:db8:0002::0a00:1)
(10.10.10.101)
    (fe80::a00:27ff:fea6:b9fe) (used)
      Origin IGP, metric 0, valid, external, bestpath-from-AS
3, best (First path received)
      AddPath ID: RX 0, TX 3
      Last update: Mon Oct 22 08:09:22 2018
```

The example output below shows the results of installing the route in the FRR RIB as well as the kernel FIB. Note that the next hop used for installation in the FRR RIB is the link-local IPv6 address, but then it is converted into an IPv4 link-local address as required for installation into the kernel FIB.

```
cumulus@spine01:~$ net show route 10.10.10.101/32
RIB entry for 10.10.10.101/32
===========================
Routing entry for 10.10.10.101/32
  Known via "bgp", distance 20, metric 0, best
  Last update 2d17h05m ago
  * fe80::a00:27ff:fea6:b9fe, via swp1

FIB entry for 10.10.10.101/32
===========================
10.10.10.101/32 via 10.0.1.0 dev swp1 proto bgp metric 20 onlink
```

If an IPv4 prefix is learned with only an IPv6 global next hop address (for example, when the route is learned through a route reflector), the command output shows the IPv6 global address as the next hop value and shows that it is learned recursively through the link-local address of the route reflector. When a global IPv6 address is used as a next hop for route installation in the FRR RIB, it is still converted into an IPv4 link-local address for installation into the kernel.

```
cumulus@leaf01:~$ net show bgp ipv4 unicast summary
BGP router identifier 10.10.10.1, local AS number 65101 vrf-id 0
BGP table version 1
```

```
RIB entries 1, using 152 bytes of memory
Peers 1, using 19 KiB of memory

Neighbor             V AS MsgRcvd  MsgSent  TblVer  InQ  OutQ
Up/Down  State/PfxRcd
Spine01(2001:db8:0002::0a00:2) 4 1   74       68         0
0     0     00:00:45      1

Total number of neighbors 1
```

```
cumulus@leaf01:~$ net show bgp ipv4 unicast
  BGP table version is 1, local router ID is 10.10.10.1
  Status codes: s suppressed, d damped, h history, * valid, >
best, = multipath,
                i internal, r RIB-failure, S Stale, R Removed
  Origin codes: i - IGP, e - EGP, ? - incomplete

Network          Next Hop    Metric LocPrf Weight Path
*>i10.1.10.0/24 2001:2:2::4       0    100      0    i

Displayed 1 routes and 1 total paths
```

```
cumulus@leaf01:~$ net show bgp ipv4 unicast 10.10.10.101/32
BGP routing table entry for 10.10.10.101/32
Paths: (1 available, best #1, table default)
  Not advertised to any peer
  Local
  2001:2:2::4 from Spine01(2001:1:1::1) (10.10.10.104)
    Origin IGP, metric 0, localpref 100, valid, internal,
bestpath-from-AS Local, best (First path received)
    Originator: 10.0.0.14, Cluster list: 10.10.10.111
    AddPath ID: RX 0, TX 5
    Last update: Mon Oct 22 14:25:30 2018
```

```
cumulus@leaf01:~$ net show route 10.10.10.1/32
RIB entry for 10.10.10.1/32
===========================
Routing entry for 10.10.10.1/32
  Known via "bgp", distance 200, metric 0, best
  Last update 00:01:13 ago
  2001:2:2::4 (recursive)
  * fe80::a00:27ff:fe5a:84ae, via swp1

FIB entry for 10.10.10.1/32
===========================
```

```
10.10.10.1/32 via 10.0.1.1 dev swp1 proto bgp metric 20 onlink
```

To have only IPv6 global addresses used for route installation into the FRR RIB, you must add an additional route map to the neighbor or peer group statement in the appropriate address family. When the route map command `set ipv6 next-hop prefer-global` is applied to a neighbor, if both a link-local and global IPv6 address are in the BGP update for a prefix, the IPv6 global address is preferred for route installation.

With this additional configuration, the output in the FRR RIB changes in the direct neighbor case as shown below:

```
router bgp 65101
  bgp router-id 10.10.10.1
  neighbor 2001:db8:2::a00:1 remote-as internal
  neighbor 2001:db8:2::a00:1 capability extended-nexthop
  !
  address-family ipv4 unicast
  neighbor 2001:db8:2::a00:1 route-map GLOBAL in
  exit-address-family
!
route-map GLOBAL permit 20
  set ipv6 next-hop prefer-global
```

```
!
```

The resulting FRR RIB output is as follows:

```
cumulus@leaf01:~$ net show route
Codes: K - kernel route, C - connected, S - static, R - RIP,
    O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
    T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
    F - PBR,
    > - selected route, * - FIB route

B 0.0.0.0/0 [200/0] via 2001:2:2::4, swp2, 00:01:00
K 0.0.0.0/0 [0/0] via 10.0.2.2, eth0, 1d02h29m
C>* 10.0.0.9/32 is directly connected, lo, 5d18h32m
C>* 10.0.2.0/24 is directly connected, eth0, 03:51:31
B>* 172.16.4.0/24 [200/0] via 2001:2:2::4, swp2, 00:01:00ß
C>* 172.16.10.0/24 is directly connected, swp3, 5d18h32m
```

When the route is learned through a route reflector, it appears like this:

```
router bgp 65101
```

```
  bgp router-id 10.10.10.1
  neighbor 2001:db8:2::a00:2 remote-as internal
  neighbor 2001:db8:2::a00:2 capability extended-nexthop
  !
  address-family ipv6 unicast
  neighbor 2001:db8:2::a00:2 activate
  neighbor 2001:db8:2::a00:2 route-map GLOBAL in
  exit-address-family
!
route-map GLOBAL permit 10
  set ipv6 next-hop prefer-global
```

```
cumulus@leaf01:~$ net show route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR,
       > - selected route, * - FIB route

B   0.0.0.0/0 [200/0] via 2001:2:2::4, 00:00:01
K   0.0.0.0/0 [0/0] via 10.0.2.2, eth0, 3d00h26m
C>* 10.0.0.8/32 is directly connected, lo, 3d00h26m
C>* 10.0.2.0/24 is directly connected, eth0, 03:39:18
```

```
C>* 172.16.3.0/24 is directly connected, swp2, 3d00h26m
B>  172.16.4.0/24 [200/0] via 2001:2:2::4 (recursive), 00:00:01
  *                       via 2001:1:1::1, swp1, 00:00:01
C>* 172.16.10.0/24 is directly connected, swp3, 3d00h26m
```

Check BGP Timer Settings

To check BGP timers, such as the BGP keepalive interval, hold time, and advertisement interval, run the NCLU `net show bgp neighbor <peer>` command or the vtysh `show ip bgp neighbor <peer>` command. For example:

```
cumulus@leaf01:~$ net show bgp neighbor swp51
GP neighbor on swp51: fe80::f208:5fff:fe12:cc8c, remote AS
65199, local AS 65101, external link
Hostname: spine01
 Member of peer-group underlay for session parameters
  BGP version 4, remote router ID 10.10.10.101, local router ID
10.10.10.1
  BGP state = Established, up for 06:50:58
  Last read 00:00:03, Last write 00:00:03
  Hold time is 9, keepalive interval is 3 seconds
  Neighbor capabilities:
```

```
    4 Byte AS: advertised and received
    AddPath:
      IPv4 Unicast: RX advertised IPv4 Unicast and received
    Extended nexthop: advertised and received
      Address families by peer:
                  IPv4 Unicast
    Route refresh: advertised and received(old & new)
    Address Family IPv4 Unicast: advertised and received
    Hostname Capability: advertised (name: leaf01,domain name:
n/a) received (name: spine01,domain name: n/a)
    Graceful Restart Capability: advertised and received
      Remote Restart timer is 120 seconds
      Address families by peer:
        none
  Graceful restart information:
    End-of-RIB send: IPv4 Unicast
    End-of-RIB received: IPv4 Unicast
    Local GR Mode: Helper*
    Remote GR Mode: Helper
    R bit: True
    Timers:
      Configured Restart Time(sec): 120
      Received Restart Time(sec): 120
    IPv4 Unicast:
```

```
      F bit: False
      End-of-RIB sent: Yes
      End-of-RIB sent after update: No
      End-of-RIB received: Yes
      Timers:
        Configured Stale Path Time(sec): 360
  Message statistics:
    Inq depth is 0
    Outq depth is 0
                         Sent       Rcvd
    Opens:                  2          1
    Notifications:          0          0
    Updates:               54         59
    Keepalives:          8219       8219
    Route Refresh:          0          0
    Capability:             0          0
    Total:               8275       8279
  Minimum time between advertisement runs is 0 seconds
```

Neighbor State Change Log

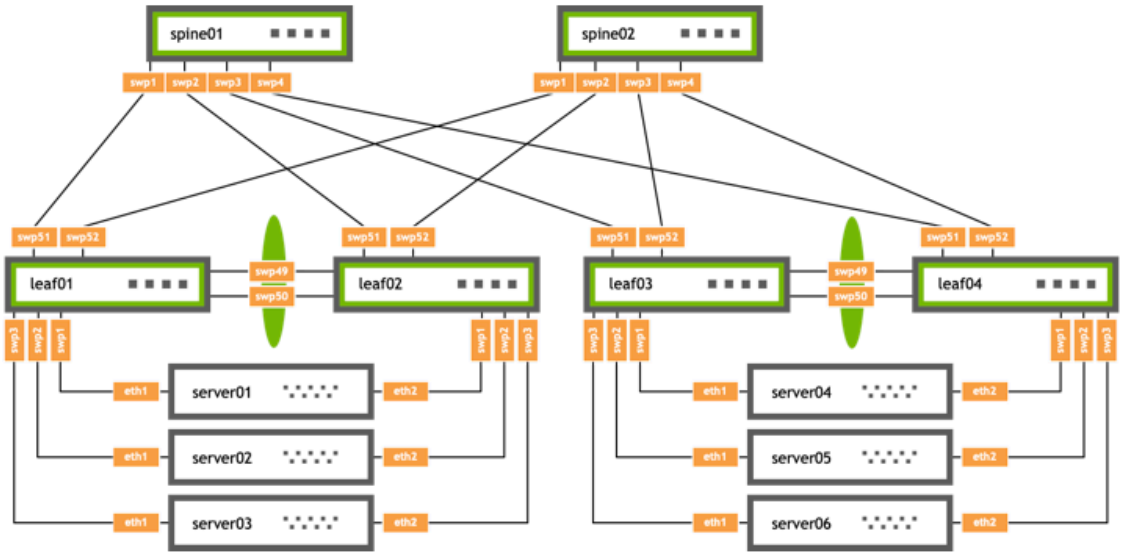
Cumulus Linux records the changes that a neighbor goes through in `syslog` and in the `/var/log/frr/frr.log` file. For example:

```
020-10-05T15:51:32.621773-07:00 leaf01 bgpd[10104]:
%NOTIFICATION: sent to neighbor peerlink.4094 6/7 (Cease/
Connection collision resolution) 0 bytes
2020-10-05T15:51:32.623023-07:00 leaf01 bgpd[10104]:
%ADJCHANGE: neighbor peerlink.4094(leaf02) in vrf default Up
2020-10-05T15:51:32.623156-07:00 leaf01 bgpd[10104]:
%NOTIFICATION: sent to neighbor peerlink.4094 6/7 (Cease/
Connection collision resolution) 0 bytes
2020-10-05T15:51:32.623496-07:00 leaf01 bgpd[10104]:
%ADJCHANGE: neighbor peerlink.4094(leaf02) in vrf default Down
No AFI/SAFI activated for peer
2020-10-05T15:51:33.040332-07:00 leaf01 bgpd[10104]: [EC
33554454] swp53 [Error] bgp_read_packet error: Connection reset
by peer
2020-10-05T15:51:33.279468-07:00 leaf01 bgpd[10104]: [EC
33554454] swp52 [Error] bgp_read_packet error: Connection reset
by peer
2020-10-05T15:51:33.339487-07:00 leaf01 bgpd[10104]:
%ADJCHANGE: neighbor swp54(spine04) in vrf default Up
2020-10-05T15:51:33.340893-07:00 leaf01 bgpd[10104]:
%ADJCHANGE: neighbor swp53(spine03) in vrf default Up
2020-10-05T15:51:33.341648-07:00 leaf01 bgpd[10104]:
%ADJCHANGE: neighbor swp52(spine02) in vrf default Up
2020-10-05T15:51:33.342369-07:00 leaf01 bgpd[10104]:
```

```
%ADJCHANGE: neighbor swp51(spine01) in vrf default Up

2020-10-05T15:51:33.627958-07:00 leaf01 bgpd[10104]:

%ADJCHANGE: neighbor peerlink.4094(leaf02) in vrf default Up
```

Configuration Example

This section shows a BGP configuration example based on the reference topology. The example configures BGP *unnumbered* on all leafs and spines and uses the peer group *underlay*. MLAG is configured on leaf01 and leaf02, and on leaf03 and leaf04.



/etc/network/interfaces

leaf01 leaf02 leaf03 leaf04 spine01 spine02

```
cumulus@leaf01:~$ cat /etc/network/interfaces
auto lo
iface lo inet loopback
    address 10.10.10.1/32

auto mgmt
iface mgmt
    vrf-table auto
    address 127.0.0.1/8
    address ::1/128

auto eth0
iface eth0 inet dhcp
    vrf mgmt

auto bridge
iface bridge
    bridge-ports peerlink bond1 bond2 bond3
    bridge-vids 10 20 30
    bridge-vlan-aware yes

auto vlan10
iface vlan10
    address 10.1.10.2/24
    vlan-raw-device bridge
    vlan-id 10

auto vlan20
iface vlan20
```

/etc/frr/frr.conf

leaf01 leaf02 leaf03 leaf04 spine01 spine02

```
cumulus@leaf01:~$ cat /etc/frr/frr.conf
...
log syslog informational
!
router bgp 65101
 bgp router-id 10.10.10.1
 bgp bestpath as-path multipath-relax
 neighbor underlay peer-group
 neighbor underlay remote-as external
 neighbor peerlink.4094 interface remote-as internal
 neighbor swp51 interface peer-group underlay
 neighbor swp52 interface peer-group underlay
 !
 address-family ipv4 unicast
  redistribute connected
 exit-address-family
!
line vty
```

Open Shortest Path First - OSPF

Open Shortest Path First (OSPF) is a link-state routing protocol used between routers to exchange information about routes and the cost to reach an intended destination. OSPF routers exchange information about their links, prefixes, and associated cost with Link State Advertisements (LSAs). This topology information is used to build a topology database. Each router within an area has an identical database and calculates its own routing table using the Shortest Path First (SPF) algorithm. The SPF algorithm is used any time there are changes to routing information in the network. OSPF uses the concept of areas to try and limit the size of the topology database on different routers. The routers that exist in more than one area are called Area Border Routers (ABRs) and they simplify the information contained within LSAs when advertising LSAs from one area to another. ABRs are the only routers in OSPF that are allowed to implement route filtering or route summarization.

Cumulus Linux supports:

- Open Shortest Path First v2 - OSPFv2 for IPv4.
- Open Shortest Path First v3 - OSPFv3 for IPv6.

Open Shortest Path First v2 - OSPFv2

This topic describes OSPFv2, which is a link-state routing protocol for IPv4. For IPv6 commands, refer to Open Shortest Path First v3 - OSPFv3.

Basic OSPFv2 Configuration

You can configure OSPF using either numbered interfaces or unnumbered interfaces. Both methods are described below.

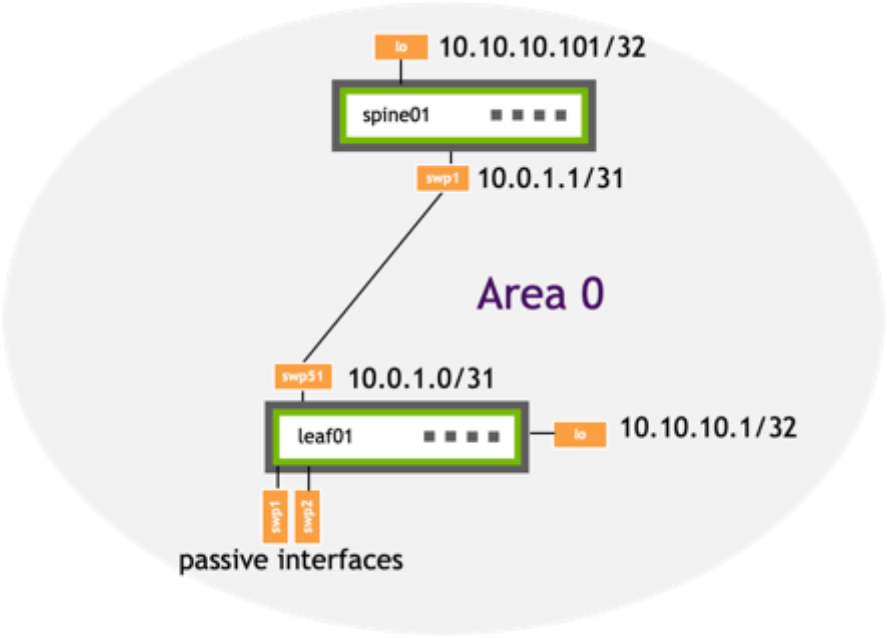
OSPFv2 Numbered

To configure OSPF using numbered interfaces, you specify the router ID, IP subnet prefix, and area address. All the interfaces on the switch with an IP address that matches the network subnet are put into the specified area. OSPF attempts to discover other OSPF routers on those interfaces. All matching interface network addresses are added to a Type-1 Router LSA and advertised to discovered neighbors for proper reachability.

If you do not want to bring up an OSPF adjacency on certain interfaces, but want to advertise those networks in the OSPF database, you can configure the interfaces as *passive interfaces*. A passive interface creates a database entry but does not send or recieve OSPF hello packets. For example, in a data center topology, the host-facing interfaces do not need to run OSPF, however, the corresponding IP addresses still need to be advertised to neighbors.

Network statements can be as inclusive or generic as necessary to cover the interface networks.

The following example commands configure OSPF numbered on leaf01 and spine01.



leaf01	spine01
• The loopback address is 10.10.10.1/32 • The IP address on swp51 is 10.0.1.0/31 • The router ID is 10.10.10.1 • All the interfaces on the switch with an IP address that matches subnet 10.10.10.1/32 and swp51 with IP address	• The loopback address is 10.10.10.101/32 • The IP address on swp1 is 10.0.1.1/31 • The router ID is 10.10.10.101 • All interfaces on the switch with an IP address that matches subnet 10.10.10.101/32 and swp1 with IP address

leaf01	spine01
10.0.1.0/31 are in area 0 • swp1 and swp2 are passive interfaces	10.0.1.1/31 are in area 0.

ⓘ NOTE

The configuration below uses the `network` command to configure the IP subnet prefix with an area address per network (`net add ospf network 10.0.1.0/31 area 0`). Alternatively, you can configure OSPF per interface with the `net add interface` command (`net add interface swp1 ospf area 0`). However, you *cannot* use both methods in the same configuration.

NCLU Commands | Linux and vtysh Commands

leaf01 | spine01

```
cumulus@leaf01:~$ net add loopback lo ip address
10.10.10.1/32
cumulus@leaf01:~$ net add interface swp51 ip address
10.0.1.0/31
cumulus@leaf01:~$ net add ospf router-id 10.10.10.1
cumulus@leaf01:~$ net add ospf network 10.10.10.1/32
area 0
cumulus@leaf01:~$ net add ospf network 10.0.1.0/31
area 0
cumulus@leaf01:~$ net add ospf passive-interface swp1
cumulus@leaf01:~$ net add ospf passive-interface swp2
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

You can use the `net add ospf passive-interface default` command to set all interfaces as *passive* and the `net del ospf passive-interface <interface>` command to selectively bring up protocol adjacency only on certain interfaces:

```
cumulus@leaf01:~$ net add ospf passive-interface
default
cumulus@leaf01:~$ net del ospf passive-interface swp51
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

leaf01 spine01

```
...
router ospf
 ospf router-id 10.10.10.1
 network 10.10.10.1/32 area 0
 network 10.0.1.0/31 area 0
 passive-interface swp1
 passive-interface swp2
...
```

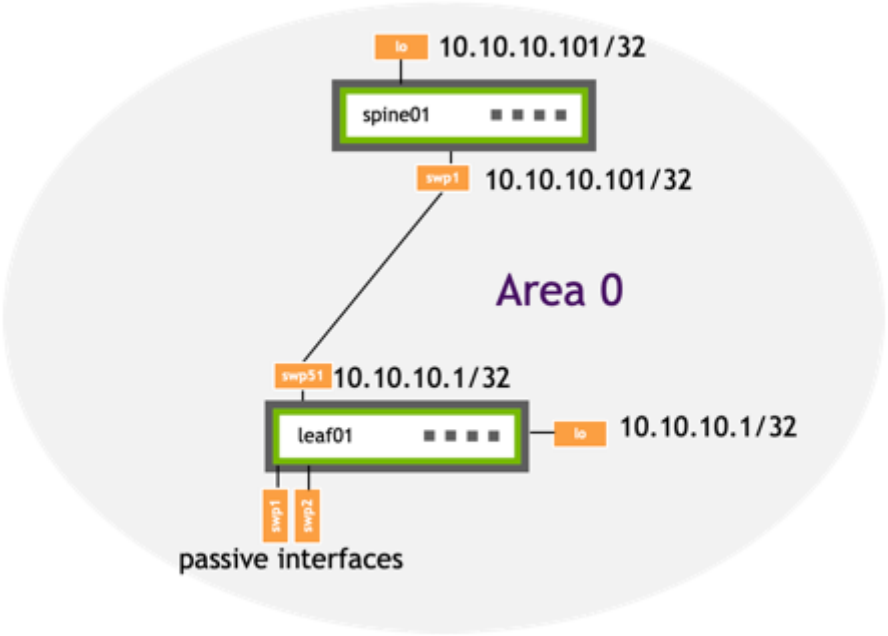
OSPFv2 Unnumbered

Unnumbered interfaces are interfaces without unique IP addresses; multiple interfaces share the same IP address. In OSPFv2, unnumbered interfaces reduce the need for unique IP addresses on leaf and spine interfaces and simplify the OSPF database, reducing the memory footprint and improving SPF convergence times.

To configure an unnumbered interface, take the IP address of loopback interface (called the *anchor*) and use that as the IP address of the unnumbered interface.

> **NOTE**
>
> OSPF unnumbered is supported with point-to-point interfaces only and does *not* support network statements.

The following example commands configure OSPF unnumbered on leaf01 and spine01.



leaf01	spine01
• The loopback address is 10.10.10.1/32 • The IP address of the	• The loopback address is 10.10.10.101/32 • The IP address of the

leaf01	spine01
unnumbered interface (swp51) is 10.10.10.1/32 • The router ID is 10.10.10.1 • OSPF is enabled on the loopback interface and on swp51 in area 0 • swp1 and swp2 are passive interfaces • swp51 is a point-to-point interface (point-to-point is required for unnumbered interfaces)	unnumbered interface (swp1) is 10.10.10.101/32 • The router ID is 10.10.10.101 • OSPF is enabled on the loopback interface and on swp1 in area 0 • swp1 is a point-to-point interface (point-to-point is required for unnumbered interfaces)

NCLU Commands | Linux and vtysh Commands

leaf01 | spine01

Configure the unnumbered interface:

```
cumulus@leaf01:~$ net add loopback lo ip address
10.10.10.1/32
cumulus@leaf01:~$ net add interface swp51 ip address
10.10.10.1/32
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

Configure OSPF:

```
cumulus@leaf01:~$ net add ospf router-id 10.10.10.1
cumulus@leaf01:~$ net add loopback lo ospf area 0
cumulus@leaf01:~$ net add interface swp51 ospf area 0
cumulus@leaf01:~$ net add ospf passive-interface swp1
cumulus@leaf01:~$ net add ospf passive-interface swp2
cumulus@leaf01:~$ net add interface swp51 ospf network
point-to-point
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

You can use the `net add ospf passive-interface default` command to set all interfaces as *passive* and the `net del ospf passive-interface <interface>` command to selectively bring up protocol adjacency only on certain interfaces:

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

leaf01 spine01

```
...
interface lo
 ip ospf area 0
interface swp51
 ip ospf area 0
 ip ospf network point-to-point
router ospf
 ospf router-id 10.10.10.1
 passive-interface swp1,swp2
...
```

Optional OSPFv2 Configuration

This section describes optional configuration. The steps provided in this section assume that you already configured basic OSPFv2 as described in Basic OSPF Configuration, above.

Interface Parameters

You can define the following OSPF parameters per interface:

- Network type (point-to-point or broadcast). Broadcast is the default

setting. Configure the interface as point-to-point unless you intend to use the Ethernet media as a LAN with multiple connected routers. Point-to-point provides a simplified adjacency state machine; there is no need for DR/BDR election and *LSA reflection*. See RFC5309 for a more information.

> **NOTE**
>
> Point-to-point is required for OSPFv2 unnumbered.

- Hello interval. The number of seconds between hello packets sent on the interface. The default is 10 seconds.
- Dead interval. The number of seconds before neighbors declare the router down after they stop hearing hello packets. The default is 40 seconds.
- Priority in becoming the OSPF Designated Router (DR) on a broadcast interface. The default is priority 1.

The following command example sets the network type to point-to-point.

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add interface swp51 ospf network
point-to-point
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example

```
...
interface swp51
 ip ospf network point-to-point
...
```

The following command example sets the hello interval to 5 seconds and the dead interval to 60 seconds. The hello interval and dead inteval can be any value between 1 and 65535 seconds.

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add interface swp51 ospf hello-
interval 5
cumulus@switch:~$ net add interface swp51 ospf dead-
interval 60
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example

```
...
interface swp51
 ip ospf hello-interval 5
 ip ospf dead-interval 60
...
```

The following command example sets the priority to 5 for swp51. The priority can be any value between 0 to 255 (0 configures the interface to never become the OSPF Designated Router (DR) on a broadcast interface).

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add interface swp51 ospf priority 5
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example

```
...
interface swp51
 ip ospf priority 5
...
```

To see the currently configured OSPF interface parameter values, run the NCLU `net show ospf interface` command or the vtysh `show ip ospf interface` command.

SPF Timer Defaults

OSPF uses the following default timers to prevent consecutive SPFs from overburdening the CPU:

- 0 milliseconds from the initial event until SPF runs

- 50 milliseconds between consecutive SPF runs (the number doubles with each SPF, until it reaches the maximum time between SPF runs)
- 5000 milliseconds maximum between SPFs

The following example commands change the number of milliseconds from the initial event until SPF runs to 80, the number of milliseconds between consecutive SPF runs to 100, and the maximum number of milliseconds between SPFs to 6000.

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add ospf timers throttle spf 80 100
6000
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router ospf
 ospf router-id 10.10.10.1
 passive-interface swp1
 passive-interface swp2
```

```
 network 10.10.10.1/32 area 0
 timers throttle spf 80 100 6000
...
```

To see the configured SPF timer values, run the NCLU `net show ospf` command or the vtysh `show ip ospf` command.

MD5 Authentication

To configure MD5 authentication on the switch, you need to create a key and a key ID, then enable MD5 authentication. The *key ID* must be a value between 1 and 255 that represents the key used to create the message digest. This value must be consistent across all routers on a link. The *key* must be a value with an upper range of 16 characters (longer strings are truncated) that represents the actual message digest.

The following example commands create key ID 1 with the key `thisisthekey` and enable MD5 authentication on swp51 on leaf01 and on swp1 on spine01.

NCLU Commands vtysh Commands

leaf01 spine01

```
cumulus@leaf01:~$ net add interface swp51 ospf message-
digest-key 1 md5 thisisthekey
cumulus@leaf01:~$ net add interface swp51 ospf
authentication message-digest
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

leaf01 spine01

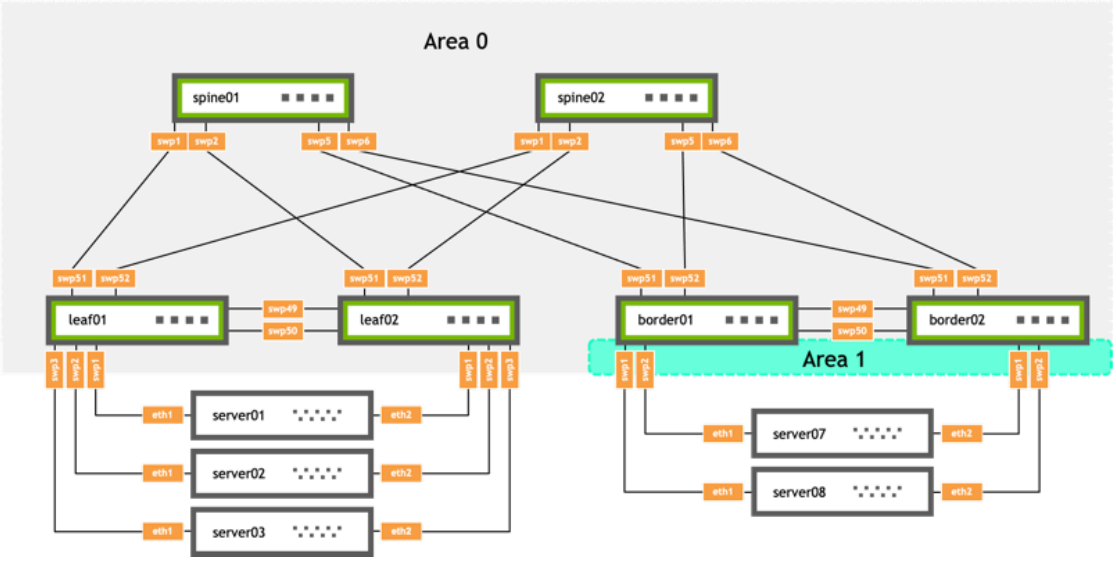
```
...
interface swp51
 ip ospf authentication message-digest
 ip ospf message-digest-key 1 md5 thisisthekey
 ...
```

> **NOTE**
>
> To remove existing MD5 authentication hashes, run the NCLU `net del` command (`net del interface swp51 ospf message-digest-key 1 md5 thisisthekey`) or the vtysh `no ip ospf` command (`no ip ospf message-digest-key 1 md5 thisisthekey`).

Summarization and Prefix Range

By default, an area border router (ABR) creates a summary (type-3) LSA for each route in an area and advertises it in adjacent areas. Prefix range configuration optimizes this behavior by creating and advertising one summary LSA for multiple routes. OSPF only allows for route summarization between areas on a ABR. This is done with the area range command.

The following example shows a topology divided into area 0 and area 1. border01 and border02 are *area border routers* (ABRs) that have links to multiple areas and perform a set of specialized tasks, such as SPF computation per area and summarization of routes across areas.



On border01:

- swp1 is in area 1 and is configured with IP addresses 10.0.0.24/31, 172.16.1.1/32, 172.16.1.2/32, and 172.16.1.3/32
- swp51 is in area 0 and is configured with IP address 10.0.1.9/31

These commands create a summary route for all the routes in the range 172.16.1.0/24 in area 0:

```
cumulus@leaf01:~$ sudo vtysh


leaf01# configure terminal
leaf01(config)# router ospf
leaf01(config-router)# area 0 range 172.16.1.0/24
leaf01(config-router)# end
leaf01# write memory
```

```
leaf01# exit
cumulus@leaf01:~$
```

The `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file.

For example:

```
cumulus@border01:mgmt:~$ sudo cat /etc/frr/frr.conf
...
interface lo
 ip ospf area 0
interface swp1
 ip ospf area 1
interface swp2
 ip ospf area 1
interface swp51
 ip ospf area 0
interface swp52
 ip ospf area 0
router ospf
 ospf router-id 10.10.10.63
 area 0 range 172.16.1.0/24
```

Stub Areas

External routes are the routes redistributed into OSPF from another protocol. They have an AS-wide flooding scope. In many cases, external link states make up a large percentage of the link-state database (LSDB). Stub *areas* reduce the LSDB size by not flooding AS-external LSAs.

All routers must agree that an area is a stub, otherwise they will not become OSPF neighbors.

To configure a stub area:

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add ospf area 1 stub
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router ospf
 router-id 10.10.10.63
```

```
 area 1 stub
...
```

Stub areas still receive information about networks that belong to other areas of the same OSPF domain. If summarization is not configured (or is not comprehensive), the information can be overwhelming for the nodes. *Totally stubby areas* address this issue. Routers in totally stubby areas keep information about routing within their area in their LSDB.

To configure a totally stubby area:

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add ospf area 1 stub no-summary
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router ospf
```

```
 router-id 10.10.10.63
 area 1 stub no-summary
...
```

Here is a brief summary of the area type differences:

Type	Behavior
Normal non-zero area	LSA types 1, 2, 3, 4 area-scoped, type 5 externals, inter-area routes summarized
Stub area	LSA types 1, 2, 3, 4 area-scoped, no type 5 externals, inter-area routes summarized
Totally stubby area	LSA types 1, 2 area-scoped, default summary, no type 3, 4, 5 LSA types allowed

Auto-cost Reference Bandwidth

When you set the *auto-cost reference bandwidth,* Cumulus Linux dynamically calculates the OSPF interface cost to support higher speed links. The default value is *100000* for 100Gbps link speed. The cost of interfaces with link speeds lower than 100Gbps is higher.

TIP

To avoid routing loops, set the bandwidth to a consistent value across all OSPF routers.

The following example commands configure the auto-cost reference bandwidth for 90Gbps link speed:

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add ospf auto-cost reference-
bandwidth 90000
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router ospf
```

```
 router-id 10.10.10.1
 auto-cost reference-bandwidth 90000
...
```

Administrative Distance

Cumulus Linux uses the administrative distance to choose which routing protocol to use when two different protocols provide route information for the same destination. The smaller the distance, the more reliable the protocol. For example, if the switch receives a route from OSPF with an administrative distance of 110 and the same route from BGP with an administrative distance of 100, the switch chooses BGP.

Cumulus Linux provides several commands to change the distance for OSPF routes. The default value is 110.

The following example commands set the distance for an entire group of routes:

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add ospf distance 254
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The following example commands change the OSPF administrative distance to 150 for internal routes and 220 for external routes:

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add ospf distance ospf intra-area 150
inter-area 150 external 220
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The following example commands change the OSPF administrative distance to 150 for internal routes to a subnet or network inside the same area as the router:

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add ospf distance ospf intra-area 150
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The following example commands change the OSPF administrative distance to 150 for internal routes to a subnet in an area of which the router is *not* a part:

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add ospf distance ospf inter-area 150
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration to the `/etc/frr/frr.conf` file. For example:

```
...
router ospf
  ospf router-id 10.10.10.1
  distance ospf intra-area 150 inter-area 150 external 220
...
```

Topology Changes and OSPF Reconvergence

When you remove a router or OSPF interface, LSA updates trigger throughout the network to inform all routers of the topology change. When the LSA is received and SPF is run, it does a routing update. This can cause short-duration outages while the network detects the failure and updates the OSPF database.

If the outage is planned (during a maintenance window), you can configure

the OSPF router with an OSPF max-metric to notify its neighbors not to use it as part of the OSPF topology. While the network converges, all traffic forwarded to the max-metric router is still forwarded. After the network is fully updated, the max-metric router no longer receives any traffic and can be safely modified. To remove a single interface, you can configure the OSPF cost for that specific interface.

For failure events, traffic might be lost during reconvergence (until SPF on all nodes computes an alternative path around the failed link or node to each of the destinations).

To configure the max-metric (for all interfaces):

```
cumulus@switch:~$ sudo vtysh
switch# configure terminal
switch(config)# router ospf
switch(config-router)# max-metric router-lsa administrative
switch(config-router)# end
switch# write memory
switch# exit
cumulus@switch:~$
```

To configure the cost (for a specific interface):

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add interface swp51 ospf cost 65535
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Troubleshooting

Cumulus Linux provides several OSPF troubleshooting commands:

To...	NCLU Command	vtysh Command
Show neighbor states	`net show ospf neighbor`	`show ip ospf neighbor`
Verify that the LSDB is synchronized across all routers in the network	`net show ospf database`	`show ip ospf database`
Determine why an OSPF route is not being forwarded correctly	`net show route ospf`	`show ip route ospf`
Show OSPF interfaces	`net show ospf interface`	`show ip ospf interface`
Show information about the OSPF	`net show ospf`	`show ip ospf`

To...	NCLU Command	vtysh Command
process		

The following example shows the `net show ospf neighbor` command output:

```
cumulus@leaf01:mgmt:~$ net show ospf neighbor
Neighbor ID     Pri State           Dead Time Address
Interface                         RXmtL RqstL DBsmL
10.10.10.101      1 Full/Backup       30.307s 10.0.1.1
swp51:10.0.1.0                        0     0     0
```

The following example shows the `net show route ospf` command output:

```
cumulus@leaf01:mgmt:~$ net show route ospf
RIB entry for ospf
==================
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r -
rejected route
```

```
O   10.0.1.0/31 [110/100] is directly connected, swp51, weight
1, 00:02:37
O   10.10.10.1/32 [110/0] is directly connected, lo, weight 1,
00:02:37
O>* 10.10.10.101/32 [110/100] via 10.0.1.1, swp51, weight 1,
00:00:57
```

To capture OSPF packets, run the `sudo tcpdump -v -i swp1 ip proto ospf` command.

For a list all of the OSPF debug options, refer to Debugging OSPF.

Related Information

- FRR OSPFv2
- Perlman, Radia (1999); *Interconnections: Bridges, Routers, Switches, and Internetworking Protocols (2 ed.)*; Addison-Wesley
- Moy, John T.; *OSPF: Anatomy of an Internet Routing Protocol*; Addison-Wesley
- RFC 2328 OSPFv2
- RFC 3101 OSPFv2 Not-So-Stubby Area (NSSA)

Open Shortest Path First v3 - OSPFv3

OSPFv3 is a revised version of OSPFv2 and supports the IPv6 address family.

> **NOTE**
>
> IETF has defined extensions to OSPFv3 to support multiple address families (both IPv6 and IPv4). FRR does not currently support multiple address families.

Basic OSPFv3 Configuration

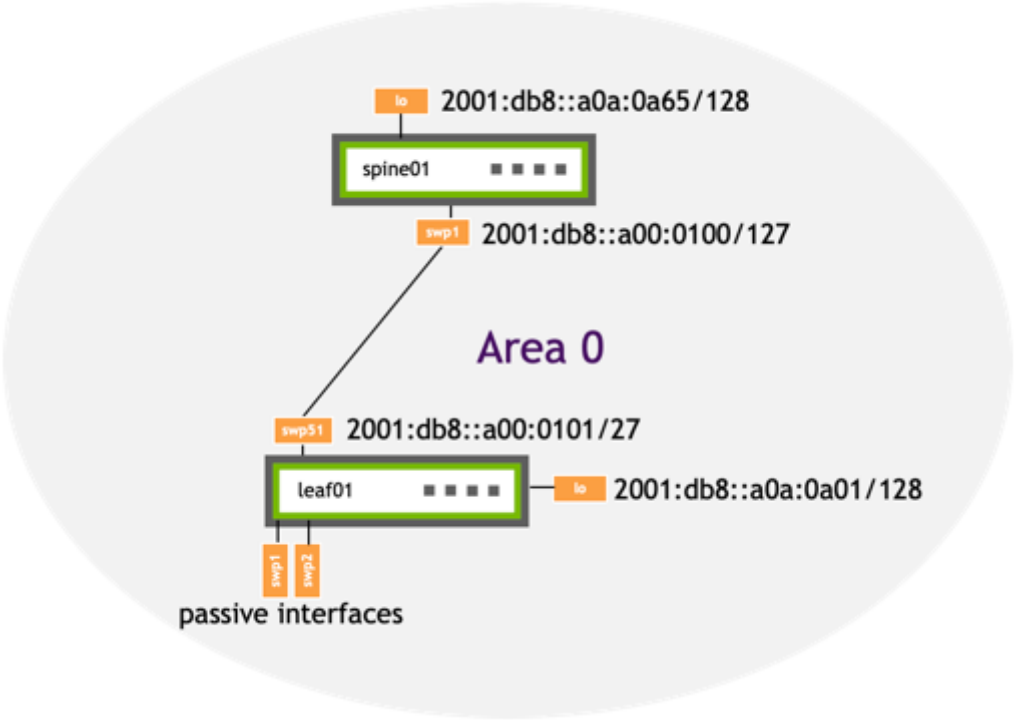
You can configure OSPFv3 using either numbered interfaces or unnumbered interfaces. Both methods are described below.

OSPFv3 Numbered

To configure OSPF using numbered interfaces, you specify the router ID, IP subnet prefix, and area address. All the interfaces on the switch with an IP address that matches the network subnet are put into the specified area. OSPF attempts to discover other OSPF routers on those interfaces. All matching interface network addresses are added to a Type-1 Router LSA and advertised to discovered neighbors for proper reachability.

If you do not want to bring up an OSPF adjacency on certain interfaces, but want to advertise those networks in the OSPF database, you can configure the interfaces as *passive interfaces*. A passive interface creates a database entry but does not send or recieve OSPF hello packets. For example, in a data center topology, the host-facing interfaces do not need to run OSPF, however, the corresponding IP addresses still need to be advertised to neighbors.

The following example commands configure OSPF numbered on leaf01 and spine01.



leaf01	spine01
• The loopback address is 2001:db8::a0a:0a01/128 • The IP address on swp51 is 2001:db8::a00:0101/127 • The router ID is 10.10.10.1 • All the interfaces on the switch with an IP address that matches subnet 2001:db8::a0a:0a01/128 and swp51 with IP address 2001:db8::a00:0101/127 are in area 0.0.0.0 • swp1 and swp2 are passive interfaces	• The loopback address is 2001:db8::a0a:0a65/128 • The IP address on swp1 is 22001:db8::a00:0100/127 • The router ID is 10.10.10.101 • All interfaces on the switch with an IP address that matches subnet 2001:db8::a0a:0a65/128 and swp1 with IP address 2001:db8::a00:0100/127 are in area 0.0.0.0.

NCLU Commands | Linux and vtysh Commands

leaf01 | spine01

```
cumulus@leaf01:~$ net add loopback lo ip address
2001:db8::a0a:0a01/128
cumulus@leaf01:~$ net add interface swp51 ip address
2001:db8::a00:0101/127
cumulus@leaf01:~$ net add ospf6 router-id 10.10.10.1
cumulus@leaf01:~$ net add ospf6 interface lo area
0.0.0.0
cumulus@leaf01:~$ net add ospf6 interface swp51 area
0.0.0.0
cumulus@leaf01:~$ net add interface swp1 ospf6 passive
cumulus@leaf01:~$ net add interface swp2 ospf6 passive
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

leaf01 spine01

```
...
router ospf6
 ospf6 router-id 10.10.10.1
 interface lo area 0.0.0.0
 interface swp51 area 0.0.0.0
interface swp1
 ipv6 ospf6 passive
interface swp2
 ipv6 ospf6 passive
...
```

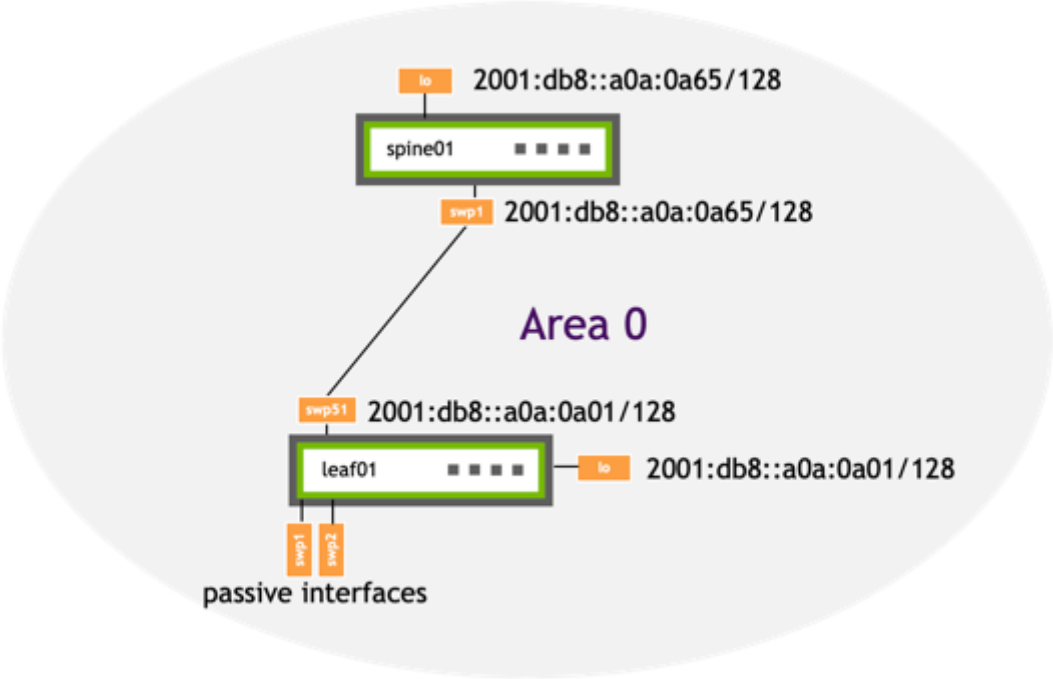
OSPFv3 Unnumbered

Unnumbered interfaces are interfaces without unique IP addresses; multiple interfaces share the same IP address.

To configure an unnumbered interface, take the IP address of another interface (called the *anchor*) and use that as the IP address of the unnumbered interface. The anchor is typically the loopback interface on the switch.

> **NOTE**
>
> OSPFv3 Unnumbered is supported with point-to-point interfaces only.

The following example commands configure OSPFv3 unnumbered on leaf01 and spine01.



leaf01	spine01
• The loopback address is 2001:db8::a0a:0a01/128	• The loopback address is 2001:db8::a0a:0a65/128

leaf01	spine01
• The router ID is 10.10.10.1 • OSPF is enabled on the loopback interface and on swp51 in area 0.0.0.0 • swp1 and swp2 are passive interfaces • swp51 is a point-to-point interface (point-to-point is required for unnumbered interfaces)	• The router ID is 10.10.10.101 • OSPF is enabled on the loopback interface and on swp1 in area 0.0.0.0 • swp1 is a point-to-point interface (point-to-point is required for unnumbered interfaces)

NCLU Commands | Linux and vtysh Commands

leaf01 | spine01

```
cumulus@leaf01:~$ net add loopback lo ip address
2001:db8::a0a:0a01/128
cumulus@leaf01:~$ net add interface swp51 ip address
2001:db8::a0a:0a01/128
cumulus@leaf01:~$ net add ospf6 router-id 10.10.10.1
cumulus@leaf01:~$ net add ospf6 interface lo area
0.0.0.0
cumulus@leaf01:~$ net add ospf6 interface swp51 area
0.0.0.0
cumulus@leaf01:~$ net add interface swp1 ospf6 passive
cumulus@leaf01:~$ net add interface swp2 ospf6 passive
cumulus@leaf01:~$ net add interface swp51 ospf6
network point-to-point
cumulus@leaf01:~$ net pending
cumulus@leaf01:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

leaf01 spine01

```
...
router ospf6
 ospf6 router-id 10.10.10.1
 interface lo area 0.0.0.0
 interface swp51 area 0.0.0.0
interface swp1
 ipv6 ospf6 passive
interface swp2
 ipv6 ospf6 passive
interface swp51
 ipv6 ospf6 network point-to-point
...
```

Optional OSPFv3 Configuration

This section describes optional configuration. The steps provided in this section assume that you already configured basic OSPFv3 as described in Basic OSPF Configuration, above.

Interface Parameters

You can define the following OSPF parameters per interface:

- Network type (point-to-point or broadcast). Broadcast is the default

setting. Configure the interface as point-to-point unless you intend to use the Ethernet media as a LAN with multiple connected routers. Point-to-point provides a simplified adjacency state machine; there is no need for DR/BDR election and *LSA reflection*. See RFC5309 for a more information.

> **NOTE**
>
> Point-to-point is required for OSPFv3 unnumbered.

- Hello interval. The number of seconds between hello packets sent on the interface. The default is 10 seconds.
- Dead interval. Then number of seconds before neighbors declare the router down after they stop hearing hello packets. The default is 40 seconds.
- Priority in becoming the OSPF Designated Router (DR) on a broadcast interface. The default is priority 1.
- Advertise prefix list. The prefix list defines the outbound route filter.
- Cost. The cost determines the shortest paths to the destination.

The following command example sets the network type to point-to-point on swp51.

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add interface swp51 ospf6 network
point-to-point
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
interface swp51
 ipv6 ospf6 network point-to-point
...
```

The following command example sets the hello interval to 5 seconds, the dead interval to 60 seconds, and the priority to 5 for swp51. The hello interval and dead inteval can be any value between 1 and 65535 seconds. The priority can be any value between 0 to 255 (0 configures the interface to never become the OSPF Designated Router (DR) on a broadcast interface).

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add interface swp51 ospf6 hello-
interval 5
cumulus@switch:~$ net add interface swp51 ospf6 dead-
interval 60
cumulus@switch:~$ net add interface swp51 ospf6 priority 5
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
interface swp51
 ipv6 ospf6 hello-interval 5
 ipv6 ospf6 dead-interval 60
 ipv6 ospf6 priority 5
...
```

The following example command configures interface swp51 with the IPv6 advertise prefix list named `myfilter`:

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add interface swp51 ospf6 advertise
prefix-list myfilter
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
interface swp51
  ipv6 ospf6 advertise prefix-list myfilter
...
```

The following example command configures the cost for swp51.

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add interface swp51 ospf6 cost 1
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
interface swp51
  ipv6 ospf6 cost 1
...
```

To show the currently configured OSPF interface parameter values, run the NCLU `net show ospf6 interface` command or the vtysh `show ipv6 ospf6 interface` command.

SPF Timer Defaults

OSPF3 uses the following default timers to prevent consecutive SPFs from overburdening the CPU:

- 0 milliseconds from the initial event until SPF runs
- 50 milliseconds between consecutive SPF runs (the number doubles with each SPF, until it reaches the maximum time between SPF runs)
- 5000 milliseconds maximum between SPFs

The following example commands change the number of milliseconds from the initial event until SPF runs to 80, the number of milliseconds between consecutive SPF runs to 100, and the maximum number of milliseconds between SPFs to 6000.

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add ospf6 timers throttle spf 80 100
6000
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router ospf6
 ospf router-id 10.10.10.1
 passive-interface swp1
 passive-interface swp2
 network swp51 area 0.0.0.0
 timers throttle spf 80 100 6000
...
```

To see the configured SPF timer values, run the NCLU `net show ospf6` command or the vtysh `show ipv6 ospf6` command.

Configure the OSPFv3 Area

You can use different areas to control routing. You can:

- Limit an OSPFv3 area from reaching another area.
- Manage the size of the routing table by creating a summary route for all the routes in a particular address range.

The following section provides command examples.

NCLU Commands vtysh Commands

The following example command removes the `3:3::/64` route from the routing table. Without a route in the table, any destinations in that network are not reachable.

```
cumulus@switch:~$ net add ospf6 area 0.0.0.0 range 3:3::/64
not-advertise
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The following example command creates a summary route for all the routes in the range 2001::/64:

```
cumulus@switch:~$ net add ospf6 area 0.0.0.0 range
2001::/64 advertise
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

You can also configure the cost for a summary route, which is used to determine the shortest paths to the destination. The value for cost must be between 0 and 16777215.

```
cumulus@switch:~$ net add ospf6 area 0.0.0.0 range
2001::/64 cost 160
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router ospf6
  ospf6 router-id 10.10.10.1
  area 0.0.0.0 range 3:3::/64 not-advertise
  area 0.0.0.0 range 2001::/64 advertise
  area 0.0.0.0 range 2001::/64 cost 160
...
```

Stub Areas

External routes are the routes redistributed into OSPF from another protocol. They have an AS-wide flooding scope. In many cases, external link states make up a large percentage of the link-state database (LSDB). Stub *areas* reduce the LSDB size by not flooding AS-external LSAs.

All routers must agree that an area is a stub, otherwise they will not become OSPF neighbors.

To configure a stub area:

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add ospf6 area 0.0.0.1 stub
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router ospf6
 ospf6 router-id 10.10.10.63
 area 0.0.0.1 stub
...
```

Stub areas still receive information about networks that belong to other areas of the same OSPF domain. If summarization is not configured (or is not comprehensive), the information can be overwhelming for the nodes. *Totally stubby areas* address this issue. Routers in totally stubby areas keep information about routing within their area in their LSDB.

To configure a totally stubby area:

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add ospf6 area 0.0.0.1 stub no-summary
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router ospf6
 ospf6 router-id 10.10.10.63
 area 0.0.0.1 stub no-summary
...
```

Here is a brief summary of the area type differences:

Type	Behavior
Normal non-zero area	LSA types 1, 2, 3, 4 area-scoped, type 5 externals, inter-area routes summarized
Stub area	LSA types 1, 2, 3, 4 area-scoped, no type 5 externals, inter-area

Type	Behavior
	routes summarized
Totally stubby area	LSA types 1, 2 area-scoped, default summary, no type 3, 4, 5 LSA types allowed

Auto-cost Reference Bandwidth

When you set the *auto-cost reference bandwidth,* Cumulus Linux dynamically calculates the OSPF interface cost to support higher speed links. The default value is *100000* for 100Gbps link speed. The cost of interfaces with link speeds lower than 100Gbps is higher.

> **TIP**
>
> To avoid routing loops, set the bandwidth to a consistent value across all OSPF routers.

The following example commands configure the auto-cost reference bandwidth for 90Gbps link speed:

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add ospf6 auto-cost reference-
bandwidth 90000
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router ospf6
 ospf6 router-id 10.10.10.1
 interface lo area 0.0.0.0
 interface swp51 area 0.0.0.0
 auto-cost reference-bandwidth 90000
...
```

Administrative Distance

Cumulus Linux uses the administrative distance to choose which routing protocol to use when two different protocols provide route information for the same destination. The smaller the distance, the more reliable the protocol. For example, if the switch receives a route from OSPFv3 with an

administrative distance of 110 and the same route from BGP with an administrative distance of 100, the switch chooses BGP.

Cumulus Linux provides several commands to change the administrative distance for OSPF routes. The default value is 110.

NCLU Commands vtysh Commands

This example command sets the distance for an entire group of routes:

```
cumulus@switch:~$ net add ospf6 distance 254
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

This example command changes the OSPF administrative distance to 150 for internal routes and 220 for external routes:

```
cumulus@switch:~$ net add ospf6 distance ospf6 intra-area
150 inter-area 150 external 220
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

This example command changes the OSPF administrative distance to 150 for internal routes to a subnet or network inside the same area as the router:

```
cumulus@switch:~$ net add ospf6 distance ospf6 intra-area
150
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

This example command changes the OSPF administrative distance to 150 for internal routes to a subnet in an area of which the router is *not* a part:

The NCLU and `vtysh` commands save the configuration to the `/etc/frr/frr.conf` file. For example:

```
...
router ospf6
 ospf6 router-id 10.10.10.1
 interface lo area 0.0.0.0
 distance ospf6 intra-area 150 inter-area 150 external 220
...
```

Troubleshooting

Cumulus Linux provides several OSPFv3 troubleshooting commands:

To...	NCLU Command	vtysh Command
Show neighbor states	`net show ospf6 neighbor`	`show ip ospf6 neighbor`
Verify that the LSDB is synchronized across all routers in the network	`net show ospf6 database`	`show ip ospf6 database`
Determine why an OSPF route is not being forwarded correctly	`net show route ospf6`	`show ip route ospf6`
Show OSPF	`net show ospf6`	`show ip ospf6`

To...	NCLU Command	vtysh Command
interfaces	`interface`	`interface`
Help visualize the network view	`net show ospf6 spf tree`	`show ip ospf6 spf tree`
Show information about the OSPFv3 process	`net show ospf6`	`show ip ospf6`

The following example shows the `net show ospf6 neighbor` command output:

```
cumulus@leaf01:mgmt:~$ net show ospf6 neighbor
Neighbor ID     Pri    DeadTime    State/IfState
Duration I/F[State]
10.10.10.101      1    00:00:34     Full/BDR
00:02:58 swp51[DR]
```

The following example shows the `net show route ospf6` command output:

```
cumulus@leaf01:mgmt:~$ net show route ospf6
RIB entry for ospf6
===================
Codes: K - kernel route, C - connected, S - static, R - RIPng,
```

```
       O - OSPFv3, I - IS-IS, B - BGP, N - NHRP, T - Table,
       v - VNC, V - VNC-Direct, A - Babel, D - SHARP, F - PBR,
       f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r -
rejected route

O   2001:db8::a00:100/127 [110/100] is directly connected,
swp51, weight 1, 00:00:20
O   2001:db8::a0a:a01/128 [110/10] is directly connected, lo,
weight 1, 00:01:40
O>* 2001:db8::a0a:a65/128 [110/110] via fe80::4638:39ff:fe00:2,
swp51, weight 1, 00:00:15
```

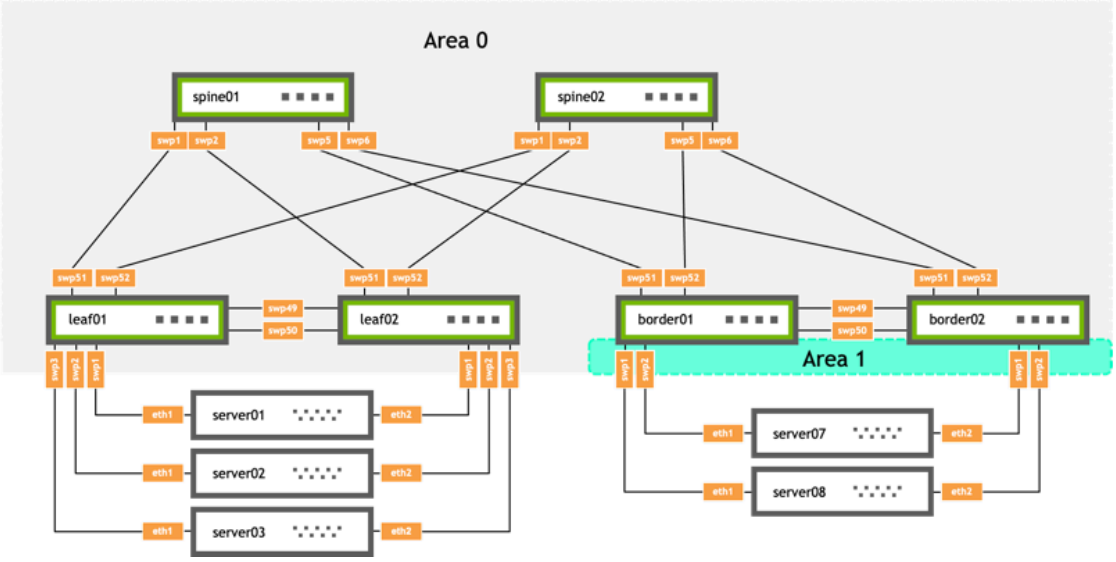
To capture OSPF packets, run the `sudo tcpdump -v -i swp1 ip proto ospf6` command.

Related Information

- FRR OSPFv3
- RFC 2740 OSPFv3 OSPF for IPv6

OSPF Configuration Example

This section shows an OSPF configuration example based on the reference topology.



In the example configuration:

- OSPFv2 *unnumbered* is configured on all leafs and spines
- MLAG is configured on leaf01 and leaf02, and on border01 and border02
- leaf01, leaf02, spine01, and spine02 are in area 0
- border01 and border02 are ABRs and are in area 0 (lo, swp51, swp52) and area 1 (swp1, swp2)

/etc/network/interfaces

leaf01 leaf02 spine01 spine02 border01 border02

```
cumulus@leaf01:~$ sudo cat /etc/network/interfaces
auto lo
iface lo inet loopback
    address 10.10.10.1/32

auto mgmt
iface mgmt
    vrf-table auto
    address 127.0.0.1/8
    address ::1/128

auto eth0
iface eth0 inet dhcp
    vrf mgmt

auto bridge
iface bridge
    bridge-ports peerlink bond1 bond2 bond3
    bridge-vids 10 20 30
    bridge-vlan-aware yes

auto vlan10
iface vlan10
    address 10.1.10.2/24
    vlan-raw-device bridge
    vlan-id 10

auto vlan20
iface vlan20
```

/etc/frr/frr.conf

leaf01 leaf02 spine01 spine02 border01 border02

```
cumulus@leaf01:~$ sudo cat /etc/frr/frr.conf
...
log syslog informational
!
interface lo
 ip ospf area 0
!
interface vlan10
  ip ospf area 0
!
interface vlan20
  ip ospf area 0
!
interface vlan30
  ip ospf area 0
!
interface swp51
 ip ospf area 0
 ip ospf network point-to-point
 ip ospf hello-interval 5
 ip ospf dead-interval 60
!
interface swp52
 ip ospf area 0
 ip ospf network point-to-point
 ip ospf hello-interval 5
 ip ospf dead-interval 60
!
router ospf
```

VRFs

This section discusses:

- Virtual Routing and Forwarding (VRF)
- Management VRF

Virtual Routing and Forwarding - VRF

Cumulus Linux provides *virtual routing and forwarding* (VRF) to allow for the presence of multiple independent routing tables working simultaneously on the same router or switch. This permits multiple network paths without the need for multiple switches. Think of this feature as VLAN for layer 3, but unlike VLANs, there is no field in the IP header carrying it. Other implementations call this feature *VRF-Lite*.

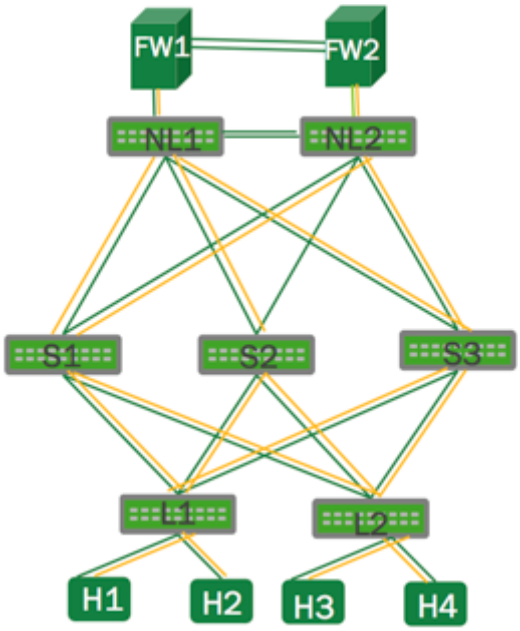
The primary use cases for VRF in a data center are similar to VLANs at layer 2: using common physical infrastructure to carry multiple isolated traffic streams for multi-tenant environments, where these streams are allowed to cross over only at configured boundary points, typically firewalls or IDS. You can also use it to burst traffic from private clouds to enterprise networks where the burst point is at layer 3. Or you can use it in an OpenStack deployment.

VRF is fully supported in the Linux kernel, so it has the following characteristics:

- The VRF is presented as a layer 3 master network device with its own associated routing table.
- The layer 3 interfaces (VLAN interfaces, bonds, switch virtual interfaces/SVIs) associated with the VRF are enslaved to that VRF; IP rules direct FIB (forwarding information base) lookups to the routing table for the VRF device.

- The VRF device can have its own IP address, known as a *VRF-local loopback*.
- Applications can use existing interfaces to operate in a VRF context by binding sockets to the VRF device or passing the `ifindex` using `cmsg`. By default, applications on the switch run against the default VRF. Services started by `systemd` run in the default VRF unless the VRF instance is used. When management VRF is enabled, logins to the switch default to the management VRF. This is a convenience so that you do not have to specify management VRF for each command. Management VRF is enabled by default in Cumulus Linux.
- Listen sockets used by services are VRF-global by default unless the application is configured to use a more limited scope (see services in the management VRF). Connected sockets (like TCP) are then bound to the VRF domain in which the connection originates. The kernel provides a sysctl that allows a single instance to accept connections over all VRFs. For TCP, connected sockets are bound to the VRF on which the first packet is received. This sysctl is enabled for Cumulus Linux.
- Connected and local routes are placed in appropriate VRF tables.
- Neighbor entries continue to be per-interface, and you can view all entries associated with the VRF device.
- A VRF does not map to its own network namespace; however, you can nest VRFs in a network namespace.
- You can use existing Linux tools, such as `tcpdump`, to interact with a VRF.

Cumulus Linux supports up to 255 VRFs on a switch.



Configure VRF

Each routing table is called a *VRF table*, and has its own table ID.

To configure VRF, you associate each subset of interfaces to a VRF routing table and configure an instance of the routing protocol (BGP or OSPFv2) for each routing table.Configuring a VRF is similar to configuring other network interfaces. Keep in mind the following:

- A VRF table can have an IP address, which is a loopback interface for the VRF.
- Associated rules are added automatically.
- You can also add a default route to avoid skipping across tables when the kernel forwards the packet.
- Names for VRF tables can be a maximum of 15 characters. However, you **cannot** use the name *mgmt*, as this name can **only** be used for the

management VRF.

To configure a VRF, run the following commands:

NCLU Commands | Linux Commands

The following example commands configure a VRF called rocket with a table ID that is automatically assigned:

```
cumulus@switch:~$ net add vrf rocket vrf-table auto
cumulus@switch:~$ net add interface swp1 vrf rocket
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Specify a Table ID

Instead of having Cumulus Linux assign a table ID for the VRF table, you can specify your own table ID in the configuration. The table ID to name mapping is saved in `/etc/iproute2/rt_tables.d/` for name-based references. Instead of using the `auto` option as shown above, specify the table ID. For example:

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add vrf rocket vrf-table 1016
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

> **NOTE**
>
> The table ID **must** be in the range of 1001 to 1255, which is reserved in Cumulus Linux for VRF table IDs.

Bring a VRF Up After You Run ifdown

If you take down a VRF using `ifdown`, you need to run one of the following commands to bring the VRF back up:

- `ifup --with-depends <vrf-name>`
- `ifreload -a`

For example:

```
cumulus@switch:~$ sudo ifdown rocket
```

```
cumulus@switch:~$ sudo ifup --with-depends rocket
```

Use the vrf Command

Run the `vrf` command to show information about VRF tables not available in other Linux commands, such as `iproute`.

To show a list of VRF tables, run the `vrf list` command:

```
cumulus@switch:~$ vrf list


VRF              Table
---------------- -----
rocket            1016
```

To show a list of processes and PIDs associated with a specific VRF table, run the `ip vrf pids <vrf-name>` command. For example:

```
cumulus@switch:~$ ip vrf pids rocket


VRF: rocket
-----------------------
```

```
dhclient          2508
sshd              2659
bash              2681
su                2702
bash              2720
vrf               2829
```

To determine which VRF table is associated with a particular PID, run the `ip vrf identify <pid>` command. For example:

```
cumulus@switch:~$ ip vrf identify 2829
rocket
```

IPv4 and IPv6 Commands in a VRF Context

You can execute non-VRF-specific Linux commands and perform other tasks against a given VRF table. This typically applies to single-use commands started from a login shell, as they affect only AF_INET and AF_INET6 sockets opened by the command that gets executed; it has no impact on netlink sockets, associated with the `ip` command.

To execute such a command against a VRF table, run `ip vrf exec <vrf-name> <command>`. For example, to SSH from the switch to a device accessible through VRF *rocket*:

```
cumulus@switch:~$ sudo ip vrf exec rocket ssh user@host
```

Services in VRFs

For services that need to run against a specific VRF, Cumulus Linux uses `systemd` instances, where the instance is the VRF. In general, you start a service within a VRF with the `systemctl start <service>@<vrf-name>` command. For example, to run the NTP service in the turtle VRF:

```
cumulus@switch:~$ sudo systemctl start ntp@turtle
```

In most cases, the instance running in the default VRF needs to be stopped before a VRF instance can start. This is because the instance running in the default VRF owns the port across all VRFs (it is VRF global). Cumulus Linux stops `systemd`-based services when the VRF is deleted and starts them when the VRF is created (when you restart networking or run an `ifdown`/`ifup` sequence). Refer to the management VRF chapter for details.

The following services work with VRF instances:

- `chef-client`
- `collectd`
- `dhcpd`
- `dhcrelay`

- `hsflowd`
- `netq-agent`
- `ntp`
- `puppet`
- `snmptrapd`
- `ssh`
- `zabbix-agent`

> **NOTE**
>
> There are cases where `systemd` instances do not work; you must use a service-specific configuration option instead. For example, to configure `rsyslogd` to send messages to remote systems over a VRF:
>
> ```
> action(type="omfwd" Target="hostname or ip here"
> Device="mgmt" Port=514
> Protocol="udp")
> ```

VRF Route Leaking

The most common use case for VRF is to use multiple independent routing and forwarding tables; however, there are situations where destinations in one VRF must be reachable (leaked) from another VRF. For example, to

make a service (such as a firewall) available to multiple VRFs or to enable routing to external networks (or the Internet) for multiple VRFs, where the external network itself is reachable through a specific VRF.

- An interface is always assigned to only one VRF; any packets received on that interface are routed using the associated VRF routing table.
- Route leaking is not allowed for overlapping addresses.
- Route leaking is supported for both IPv4 and IPv6 routes.
- Route leaking is supported with EVPN in a symmetric routing configuration only.
- VRF route leaking is not supported between the tenant VRF and the default VRF with onlink next hops (BGP unnumbered).
- The NCLU command to configure route leaking fails if the VRF is named `red` (lowercase letters only). This is not a problem if the VRF is named `RED` (uppercase letters) or has a name other than red. To work around this issue, rename the VRF or run the `vtysh` command instead. This is a known limitation in `network-docopt`.

NOTE

VRF route leaking uses BGP to replicate the leaked routes across VRFs. However, Cumulus Linux 4.2.0 and earlier cannot replicate the host routes for neighbors local to a switch where the leak is configured. To discover all directly connected neighbors in the source VRF of a leaked route, enable the `vrf_route_leak_enable_dynamic` option in the `/etc/cumulus/`

`switchd.conf` file. These routes are then replicated into the target or destination VRF as specified in the leaked route.
The `vrf_route_leak_enable_dynamic` option makes certain inter-VRF traffic ASIC accelerated. Enable this option if you are experiencing slow performance.
In Cumulus Linux 4.2.1 and later, the `vrf_route_leak_enable_dynamic` option is enabled by default.

Configure Route Leaking

For route leaking, a destination VRF is interested in the routes of a source VRF. As routes come and go in the source VRF, they are dynamically leaked to the destination VRF through BGP. If the routes in the source VRF are learned through BGP, no additional configuration is necessary. If the routes in the source VRF are learned through OSPF, or if they are statically configured or directly-connected networks have to be reached, the routes need to be first *redistributed* into BGP (in the source VRF) for them to be leaked.

You can also use route leaking to reach remote destinations as well as directly connected destinations in another VRF. Multiple VRFs can import routes from a single source VRF and a VRF can import routes from multiple source VRFs. This is typically used when a single VRF provides connectivity to external networks or a shared service for many other VRFs. The routes

that are leaked dynamically across VRFs can be controlled using a route-map.

Because route leaking happens through BGP, the underlying mechanism relies on the BGP constructs of the Route Distinguisher (RD) and Route Targets (RTs). However, you do not need to configure these parameters; they are automatically derived when you enable route leaking between a pair of VRFs.

When you use route leaking:

- You cannot reach the loopback address of a VRF (the address assigned to the VRF device) from another VRF.
- When using route leaking, you must use the `redistribute` command in BGP to leak non-BGP routes (connected or static routes); you cannot use the `network` command.
- Routes in the management VRF with the next-hop as eth0 or the management interface are not leaked.
- Routes learned with iBGP or multi-hop eBGP in a VRF can be leaked even if their next hops become unreachable. Therefore, route leaking for BGP-learned routes is recommended only when they are learned through single-hop eBGP.
- You cannot configure VRF instances of BGP in multiple autonomous systems (AS) or an AS that is not the same as the global AS.
- Do not use the default VRF as a shared service VRF. Create another VRF for shared services.
- An EVPN symmetric routing configuration on a Mellanox switch with a Spectrum ASIC or a Broadcom switch has certain limitations when

leaking routes between the default VRF and non-default VRFs. The default VRF has underlay routes (routes to VTEP addresses) that cannot be leaked to any tenant VRFs. If you need to leak routes between the default VRF and a non-default VRF, you must filter out routes to the VTEP addresses to prevent leaking these routes. Use caution with such a configuration. Run common services in a separate VRF (service VRF) instead of the default VRF to simplify configuration and avoid using route-maps for filtering.

In the following example commands, routes in the BGP routing table of VRF `rocket` are dynamically leaked into VRF `turtle`.

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add bgp vrf turtle ipv4 unicast
import vrf rocket
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router bgp 65001 vrf turtle
```

```
 !
 address-family ipv4 unicast
  import vrf rocket
...
```

Exclude Certain Prefixes

You can exclude certain prefixes from being imported. The prefixes must be configured in a route map.

The following example configures a route map to match the source protocol BGP and imports the routes from VRF turtle to VRF rocket. For the imported routes, the community is set to 11:11 in VRF rocket.

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add bgp vrf rocket ipv4 unicast
import vrf turtle
cumulus@switch:~$ net add routing route-map turtle-to-
rocket-IPV4 permit 10
cumulus@switch:~$ net add routing route-map turtle-to-
rocket-IPV4 permit 10 match source-protocol bgp
cumulus@switch:~$ net add routing route-map turtle-to-
rocket-IPV4 permit 10 set community 11:11
cumulus@switch:~$ net add bgp vrf rocket ipv4 unicast
import vrf route-map turtle-to-rocket-IPV4
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Verify Route Leaking Configuration

To check the status of VRF route leaking, run the NCLU `net show bgp vrf <vrf-name> ipv4|ipv6 unicast route-leak` command or the `vtysh show ip bgp vrf <vrf-name> ipv4|ipv6 unicast route-leak` command. For example:

```
cumulus@switch:~$ net show bgp vrf turtle ipv4 unicast route-
```

```
leak
This VRF is importing IPv4 Unicast routes from the following
VRFs:
  rocket
Import RT(s): 0.0.0.0:3
This VRF is exporting IPv4 Unicast routes to the following VRFs:
  rocket
RD: 10.1.1.1:2
Export RT: 10.1.1.1:2
```

- To view the BGP routing table, run the NCLU `net show bgp vrf <vrf-name> ipv4|ipv6 unicast` command or the `vtysh` `show ip bgp vrf <vrf-name> ipv4|ipv6 unicast` command.
- To view the FRRouting IP routing table, use the NCLU `net show route vrf <vrf-name>` command or the `vtysh` `show ip route vrf <vrf-name>` command. These commands show all routes, including routes leaked from other VRFs.

The following example commands show all routes in VRF `turtle`, including routes leaked from VRF `rocket`:

```
cumulus@switch:~$ net show route vrf turtle
Codes: K - kernel route, C - connected, S - static, R - RIP,
```

```
       O - OSPF, I - IS-IS, B - BGP, P - PIM, E - EIGRP, N -
NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR,
       > - selected route, * - FIB route

VRF turtle:
K * 0.0.0.0/0 [255/8192] unreachable (ICMP unreachable),
6d07h01m
C>* 10.1.1.1/32 is directly connected, turtle, 6d07h01m
B>* 10.0.100.1/32 [200/0] is directly connected, rocket(vrf
rocket), 6d05h10m
B>* 10.0.200.0/24 [20/0] via 10.10.2.2, swp1.11, 5d05h10m
B>* 10.0.300.0/24 [200/0] via 10.20.2.2, swp1.21(vrf rocket),
5d05h10m
C>* 10.10.2.0/30 is directly connected, swp1.11, 6d07h01m
C>* 10.10.3.0/30 is directly connected, swp2.11, 6d07h01m
C>* 10.10.4.0/30 is directly connected, swp3.11, 6d07h01m
B>* 10.20.2.0/30 [200/0] is directly connected, swp1.21(vrf
rocket), 6d05h10m
```

Delete Route Leaking Configuration

To remove route leaking configuration, run the following commands. These commands ensure that all leaked routes are removed and routes are no

longer leaked from the specified source VRF.

The following example commands delete leaked routes from VRF `rocket` to VRF `turtle`:

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net del bgp vrf turtle ipv4 unicast
import vrf rocket
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

NOTE

Do not use the kernel commands; they are no longer supported and might cause issues when used with VRF route leaking in FRR.

FRRouting Operation in a VRF

In Cumulus Linux, BGP, OSPFv2 and static routing (IPv4 and IPv6) are supported within a VRF context. Various FRRouting routing constructs, such as routing tables, nexthops, router-id, and related processing are also VRF-aware.

FRRouting learns of VRFs provisioned on the system as well as interface

attachment to a VRF through notifications from the kernel.

You can assign switch ports to each VRF table with an interface-level configuration, and BGP instances can be assigned to the table with a BGP router-level command.

Because BGP is VRF-aware, per-VRF neighbors, both iBGP and eBGP, as well as numbered and unnumbered interfaces are supported. Non-interface-based VRF neighbors are bound to the VRF, which is how you can have overlapping address spaces in different VRFs. Each VRF can have its own parameters, such as address families and redistribution. Incoming connections rely on the Linux kernel for VRF-global sockets. BGP neighbors can be tracked using BFD, both for single and multiple hops. You can configure multiple BGP instances, associating each with a VRF.

A VRF-aware OSPFv2 configuration also supports numbered and unnumbered interfaces. Supported layer 3 interfaces include SVIs, sub-interfaces and physical interfaces. The VRF supports types 1 through 5 (ABR/ASBR - external LSAs) and types 9 through 11 (opaque LSAs) link state advertisements, redistributing other routing protocols, connected and static routes, and route maps. As with BGP, you can track OSPF neighbors with BFD.

NOTE

Cumulus Linux does not support multiple VRFs in multi-instance

OSPF.

VRFs are provisioned using NCLU. VRFs can be pre-provisioned in FRRouting too, but they become active only when configured with NCLU.

- You pre-provision a VRF in FRRouting by running the command `vrf vrf-name`.
- You can pre-provision a BGP instance corresponding to a VRF with the NCLU `net add bgp vrf <vrf-name> autonomous-system <value>` command. Under this context, all existing BGP parameters can be configured: neighbors, peer-groups, address-family configuration, redistribution, and so on.
- An OSPFv2 instance can be configured using the NCLU `net add ospf vrf <vrf-name>` command; as with BGP, you can configure all OSPFv2 parameters.
- You can provision static routes (IPv4 and IPv6) in a VRF by specifying the VRF along with the static route configuration. For example, `ip route prefix dev vrf <vrf-name>`. The VRF has to exist for this configuration to be accepted - either already defined through `/etc/network/interfaces` or pre-provisioned in FRRouting.

Example VRF Configuration in BGP

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add bgp vrf vrf1012 autonomous-system
64900
cumulus@switch:~$ net add bgp vrf vrf1012 router-id 6.0.2.7
cumulus@switch:~$ net add bgp vrf vrf1012 neighbor ISL peer-
group
cumulus@switch:~$ net add bgp vrf vrf1012 neighbor ISLv6
peer-group
cumulus@switch:~$ net add bgp vrf vrf1012 neighbor swp1.2
interface v6only peer-group ISLv6
cumulus@switch:~$ net add bgp vrf vrf1012 neighbor swp1.2
remote-as external
cumulus@switch:~$ net add bgp vrf vrf1012 neighbor swp3.2
interface v6only peer-group ISLv6
cumulus@switch:~$ net add bgp vrf vrf1012 neighbor swp3.2
remote-as external
cumulus@switch:~$ net add bgp vrf vrf1012 neighbor
169.254.2.18 remote-as external
cumulus@switch:~$ net add bgp vrf vrf1012 neighbor
169.254.2.18 peer-group ISL
cumulus@switch:~$ net add bgp vrf vrf1012 ipv4 unicast
network 20.7.2.0/24
cumulus@switch:~$ net add bgp vrf vrf1012 ipv4 unicast
neighbor ISL activate
cumulus@switch:~$ net add bgp vrf vrf1012 neighbor ISL
route-map ALLOW_BR2 out
cumulus@switch:~$ net add bgp vrf vrf1012 ipv6 unicast
network 2003:7:2::/125
cumulus@switch:~$ net add bgp vrf vrf1012 ipv6 unicast
neighbor ISLv6 activate
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router bgp 64900 vrf vrf1012
  bgp router-id 6.0.2.7
  no bgp default ipv4-unicast
  neighbor ISL peer-group
  neighbor ISLv6 peer-group
  neighbor swp1.2 interface v6only peer-group ISLv6
  neighbor swp1.2 remote-as external
  neighbor swp3.2 interface v6only peer-group ISLv6
  neighbor swp3.2 remote-as external
  neighbor 169.254.2.18 remote-as external
  neighbor 169.254.2.18 peer-group ISL
  !
  address-family ipv4 unicast
    network 20.7.2.0/24
    neighbor ISL activate
    neighbor ISL route-map ALLOW_BR2 out
  exit-address-family
  !
  address-family ipv6 unicast
    network 2003:7:2::/125
    neighbor ISLv6 activate
```

```
    neighbor ISLv6 route-map ALLOW_BR2_v6 out
  exit-address-family
...
```

Example VRF Configuration in OSPF

NCLU Commands | vtysh Commands

```
cumulus@switch:~$ net add ospf vrf vrf1
cumulus@switch:~$ net add ospf vrf vrf1 router-id 4.4.4.4
cumulus@switch:~$ net add ospf vrf vrf1 log-adjacency-
changes detail
cumulus@switch:~$ net add ospf vrf vrf1 network 10.0.0.0/24
area 0.0.0.1
cumulus@switch:~$ net add ospf vrf vrf1 network 9.9.0.0/16
area 0.0.0.0
cumulus@switch:~$ net add ospf vrf vrf1 redistribute
connected
cumulus@switch:~$ net add ospf vrf vrf1 redistribute bgp
cumulus@switch:~$ net add interface swp1 ospf network point-
to-point
cumulus@switch:~$ net add interface swp2 ospf network point-
to-point
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
interface swp1
  ip address 192.0.2.1/32
  ip ospf network point-to-point
!
interface swp2
  ip address 192.0.2.1/32
  ip ospf network point-to-point
!
router ospf vrf vrf1
  ospf router-id 4.4.4.4
  log-adjacency-changes detail
  redistribute connected
  redistribute bgp
  network 9.9.0.0/16 area 0.0.0.0
  network 10.0.0.0/24 area 0.0.0.1
...
```

BGP Unnumbered Interfaces with VRF

BGP unnumbered interface configurations are supported with VRF. In BGP unnumbered, there are no addresses on any interface. However, debugging tools like `traceroute` need at least a single IP address per node as the node's source IP address. Typically, this address is assigned to the loopback device. With VRF, you need a loopback device for each VRF table since

VRF is based on interfaces, not IP addresses. While Linux does not support multiple loopback devices, it does support the concept of a dummy interface, which is used to achieve the same goal.

An IP address can be associated with the VRF device, which will then act as the dummy (loopback-like) interface for that VRF.

The BGP unnumbered configuration is the same as for a non-VRF, applied under the VRF context.

To configure BGP unnumbered:

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add vrf vrf1 vrf-table auto
cumulus@switch:~$ net add vrf vrf1 ip address 6.1.0.6/32
cumulus@switch:~$ net add vrf vrf1 ipv6 address 2001:6:1::6/
128
cumulus@switch:~$ net add interface swp1 link speed 10000
cumulus@switch:~$ net add interface swp1 link autoneg off
cumulus@switch:~$ net add interface swp1 vrf vrf1
cumulus@switch:~$ net add vlan 101 ip address 20.1.6.1/24
cumulus@switch:~$ net add vlan 101 ipv6 address
2001:20:1:6::1/80
cumulus@switch:~$ net add bridge bridge ports vlan101
```

Here is the FRRouting BGP configuration:

```
cumulus@switch:~$ net add bgp vrf vrf1 autonomous-system
65001
cumulus@switch:~$ net add bgp vrf vrf1 bestpath as-path
multipath-relax
cumulus@switch:~$ net add bgp vrf vrf1 bestpath compare-
routerid
cumulus@switch:~$ net add bgp vrf vrf1 neighbor LEAF peer-
group
cumulus@switch:~$ net add bgp vrf vrf1 neighbor LEAF remote-
as external
cumulus@switch:~$ net add bgp vrf vrf1 neighbor LEAF
capability extended-nexthop
cumulus@switch:~$ net add bgp vrf vrf1 neighbor swp1.101
interface peer-group LEAF
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router bgp 65001 vrf vrf1
 no bgp default ipv4-unicast
 bgp bestpath as-path multipath-relax
 bgp bestpath compare-routerid
 neighbor LEAF peer-group
 neighbor LEAF remote-as external
 neighbor LEAF capability extended-nexthop
 neighbor swp1.101 interface peer-group LEAF
 neighbor swp2.101 interface peer-group LEAF
 !
 address-family ipv4 unicast
  redistribute connected
  neighbor LEAF activate
  exit-address-family
  !
 address-family ipv6 unicast
  redistribute connected
  neighbor LEAF activate
 exit-address-family
...
```

DHCP with VRF

Because you can use VRF to bind IPv4 and IPv6 sockets to non-default VRF tables, you can start DHCP servers and relays in any non-default VRF table using the `dhcpd` and `dhcrelay` services. These services must be managed by `systemd` to run in a VRF context. In addition, the services must be listed in the `/etc/vrf/systemd.conf` file. By default, this file already lists these two services, as well as others like `ntp`. You can add more services as needed, such as `dhcpd6` and `dhcrelay6` for IPv6.

If you edit `/etc/vrf/systemd.conf`, run `sudo systemctl daemon-reload` to generate the `systemd` instance files for the newly added services. Then you can start the service in the VRF using `systemctl start <service>@<vrf-name>.service`, where `<service>` is the name of the service (such as `dhcpd` or `dhcrelay`) and `<vrf-name>` is the name of the VRF.

For example, to start the `dhcrelay` service after you configure a VRF named *turtle*, run:

```
cumulus@switch:~$ sudo systemctl start dhcrelay@turtle.service
```

To enable the service at boot time, you must also enable the service:

```
cumulus@switch:~$ sudo systemctl enable dhcrelay@turtle.service
```

In addition, you need to create a separate default file in `/etc/default` for every instance of a DHCP server and/or relay in a non-default VRF; this is where you set the server and relay options. To run multiple instances of any of these services, you need a separate file for each instance. The files must be named as follows:

- `isc-dhcp-server-<vrf-name>`
- `isc-dhcp-server6-<vrf-name>`
- `isc-dhcp-relay-<vrf-name>`
- `isc-dhcp-relay6-<vrf-name>`

See the example configuration below for more details.

NOTE

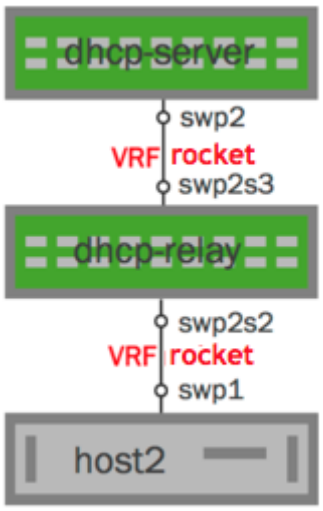
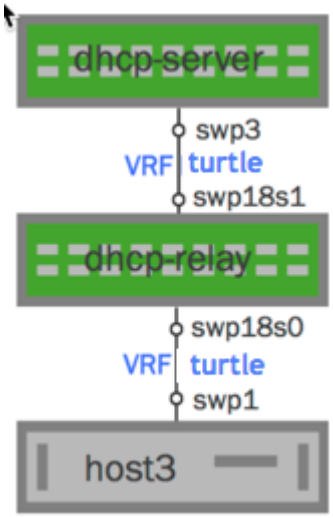
- Cumulus Linux does **not** support DHCP server and relay across VRFs; the server and host cannot be in different VRF tables. In addition, the server and relay cannot be in different VRF tables.
- Typically, a service running in the default VRF owns a port

across all VRFs. If the VRF local instance is preferred, first disable and stop the global instance.

- VRF is a layer 3 routing feature; only run programs that use AF_INET and AF_INET6 sockets in a VRF. VRF context does not affect any other aspects of the operation of a program.
- This method only works with `systemd`-based services.

Example Configuration

In the following example, there is one IPv4 network with a VRF named *rocket* and one IPv6 network with a VRF named *turtle*.

IPv4 DHCP Server/relay network	IPv6 DHCP Server/relay network
	

Configure each DHCP server and relay as follows:

DHCP Server | DHCP6 Server | DHCP Relay | DHCP6 Relay

1. Create the file `isc-dhcp-server-rocket` in `/etc/default/`. Here is sample content:

```
# Defaults for isc-dhcp-server initscript
# sourced by /etc/init.d/isc-dhcp-server
# installed at /etc/default/isc-dhcp-server by the
maintainer scripts
#
# This is a POSIX shell fragment
#
# Path to dhcpd's config file (default: /etc/dhcp/
dhcpd.conf).
DHCPD_CONF="-cf /etc/dhcp/dhcpd-rocket.conf"
# Path to dhcpd's PID file (default: /var/run/dhcpd.pid).
DHCPD_PID="-pf /var/run/dhcpd-rocket.pid"
# Additional options to start dhcpd with.
# Don't use options -cf or -pf here; use DHCPD_CONF/
DHCPD_PID instead
#OPTIONS=""
# On what interfaces should the DHCP server (dhcpd) serve
DHCP requests?
# Separate multiple interfaces with spaces, e.g. "eth0
eth1".
INTERFACES="swp2"
```

2. Enable the DHCP server:

Use ping or traceroute on a VRF

You can run `ping` or `traceroute` on a VRF from the default VRF.

To ping a VRF from the default VRF, run the `ping -I <vrf-name>` command. For example:

```
cumulus@switch:~$ ping -I turtle
```

To run `traceroute` on a VRF from the default VRF, run the `traceroute -i <vrf-name>` command. For example:

```
cumulus@switch:~$ sudo traceroute -i turtle
```

Troubleshooting

You can use NCLU, `vtysh`, or Linux show commands commands to troubleshoot VRFs.

NCLU Commands | vtysh Commands | Linux Commands

To show the routes in a VRF, run the `net show route vrf <vrf-name>` command. For example:

```
cumulus@switch:~$ net show route vrf rocket
RIB entry for rocket
=================
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, T - Table,
       > - selected route, * - FIB route

C>* 169.254.2.8/30 is directly connected, swp1.2
C>* 169.254.2.12/30 is directly connected, swp2.2
C>* 169.254.2.16/30 is directly connected, swp3.2
```

To show the BGP summary for a VRF, run the `net show bgp vrf <vrf-name> summary` command. For example:

```
cumulus@switch:~$ net show bgp vrf rocket summary
BGP router identifier 6.0.2.7, local AS number 64900 vrf-id
14
BGP table version 0
RIB entries 1, using 120 bytes of memory
Peers 6, using 97 KiB of memory
Peer groups 2, using 112 bytes of memory

Neighbor        V  AS   MsgRcvd MsgSent   TblVer  InQ OutQ
Up/Down  State/PfxRcd
```

Considerations

- Switches using the Hurricane2 ASIC (such as the Penguin Computing Arctica 4804IP) do not support VRFs.
- Table selection is based on the incoming interface only; currently, packet attributes or output-interface-based selection are not available.
- Setting the router ID outside of BGP using the `router-id` option causes all BGP instances to get the same router ID. If you want each BGP instance to have its own router ID, specify the `router-id` under the BGP instance using `bgp router-id`. If both are specified, the one under the BGP instance overrides the one provided outside BGP.
- You cannot configure EVPN address families within a VRF.
- When you take down a VRF using `ifdown`, Cumulus Linux removes all routes associated with that VRF from FRR but it does *not* remove the routes from the kernel.
- The NCLU command to remove a BGP neighbor (`net del bgp neighbor <interface> remote-as <asn>`) does not remove the BGP neighbor statement in the `/etc/network/interfaces` file when the BGP unnumbered interface belongs to a VRF. However, if the interface belongs to the default VRF, the BGP neighbor statement is removed.

Management VRF

> **NOTE**
>
> In Cumulus Linux 4.0 and later, management VRF is enabled by default. This is a change from earlier Cumulus Linux releases, where management VRF is *disabled* by default. Be sure to update any configuration scripts, if necessary.

Management VRF is a subset of Virtual Routing and Forwarding - VRF (virtual routing tables and forwarding) and provides a separation between the out-of-band management network and the in-band data plane network. For all VRFs, the *main* routing table is the default table for all of the data plane switch ports. With management VRF, a second table, *mgmt*, is used for routing through the Ethernet ports of the switch. The *mgmt* name is special cased to identify the management VRF from a data plane VRF. FIB rules are installed for DNS servers because this is the typical deployment case.

Cumulus Linux only supports eth0 (or eth1, depending on the switch platform) for *out-of-band management*. The Ethernet ports are software-only ports that are not hardware accelerated by `switchd`. VLAN subinterfaces, bonds, bridges, and the front panel switch ports are not supported as OOB management interfaces.

> **NOTE**
>
> In band management of Cumulus Linux is possible using loopbacks and SVIs (switch virtual interfaces).

Management VRF is enabled by default in Cumulus Linux so logins to the switch are set into the management VRF context. IPv4 and IPv6 networking applications (for example, Ansible, Chef, and `apt-get`) run by an administrator communicate out the management network by default. This default context does not impact services run through `systemd` and the `systemctl` command, and does not impact commands examining the state of the switch, such as the `ip` command to list links, neighbors, or routes.

> **TIP**
>
> The management VRF configurations in this chapter contain a localhost loopback IPv4 address of 127.0.0.1/8 and IPv6 address of ::1/128. Management VRF must have an IPv6 address as well as an IPv4 address to work correctly. Adding the loopback address to the layer 3 domain of the management VRF prevents issues with applications that expect the loopback IP address to exist in the VRF, such as NTP.

To disable management VRF, either run the NCLU `net del vrf mgmt` command or remove the `auto mgmt` and `auto eth0` stanzas from the `/etc/network/interfaces` file, then reboot the switch:

Bring Up the Management VRF

If you take down the management VRF using `ifdown`, to bring it back up you need to do one of two things:

- Run the `ifup --with-depends mgmt` command
- Run `ifreload -a` command

The following command example brings down the management VRF, then brings it back up with the `ifup --with-depends mgmt` command:

```
cumulus@switch:~$ sudo ifdown mgmt
cumulus@switch:~$ sudo ifup --with-depends mgmt
```

> **NOTE**
>
> Running `ifreload -a` disconnects the session for any interface configured as *auto*.

Run Services within the Management VRF

At installation, the only two enabled services that run in the management VRF are NTP (`ntp@mgmt.service`) and netqd (`netqd@mgmt`). However, you can run a variety of services within the management VRF instead of the default VRF. When you run a `systemd` service inside the management VRF, that service runs **only** on eth0. You cannot configure the same service to run successfully in both the management VRF and the default VRF; you must stop and disable the normal service with `systemctl`.

You must disable the following services in the default VRF if you want to run them in the management VRF:

- chef-client
- collectd
- dhcpd
- dhcrelay
- hsflowd
- netq-agent
- netq-notifier
- puppet

- snmpd
- snmptrapd
- ssh
- zabbix-agent

You can configure certain services (such as `snmpd`) to use multiple routing tables, some in the management VRF, some in the default or additional VRFs. The kernel provides a `sysctl` that allows a single instance to accept connections over all VRFs.

> **NOTE**
>
> For TCP, connected sockets are bound to the VRF on which the first packet is received.

The following steps show how to enable the SNMP service to run in the management VRF. You can enable any of the services listed above, except for `dhcrelay` (see DHCP Relays).

1. If SNMP is running, stop the service:

```
cumulus@switch:~$ sudo systemctl stop snmpd.service
```

2. Disable SNMP from starting automatically in the default VRF:

```
cumulus@switch:~$ sudo systemctl disable snmpd.service
```

3. Start SNMP in the management VRF:

```
cumulus@switch:~$ sudo systemctl start snmpd@mgmt.service
```

4. Enable `snmpd@mgmt` so that it starts when the switch boots:

```
cumulus@switch:~$ sudo systemctl enable snmpd@mgmt.service
```

5. Verify that the SNMP service is running in the management VRF:

```
cumulus@switch:~$ ps aux | grep snmpd
snmp      3083  0.1  1.9  35916 13292 ?        Ss   21:07
0:00 /usr/sbin/snmpd -y -LS 0-4 d -Lf /dev/null -u snmp -g
snmp -I -smux -p /run/snmpd.pid -f
cumulus   3225  0.0  0.1   6076   884 pts/0    S+   21:07
0:00 grep snmpd
```

Run the following command to show the process IDs associated with the

management VRF:

```
cumulus@switch:~$ ip vrf pids mgmt
1149  ntpd
 1159  login
 1227  bash
16178  vi
  948  dhclient
20934  sshd
20975  bash
21343  sshd
21384  bash
21477  ip
```

Run the following command to show the VRF association of the specified process:

```
cumulus@switch:~$ ip vrf identify 2055
mgmt
```

Run `ip vrf help` for additional `ip vfr` commands.

> **NOTE**
>
> You might see a warning, similar to the one below from `systemctl` for any management VRF service. You can ignore this warning. This is a problem in `systemd` in Debian 10 (buster).
>
> Warning: The unit file, source configuration file or drop-ins of ntp@mgmt.service changed on disk. Run 'systemctl daemon-reload' to reload unit

Enable Polling with snmpd in a Management VRF

When you enable `snmpd` to run in the management VRF, you need to specify that VRF so that `snmpd` listens on eth0 in the management VRF; you can also configure `snmpd` to listen on other ports. In Cumulus Linux, SNMP configuration is VRF aware so `snmpd` can bind to multiple IP addresses each configured with a particular VRF (routing table). The `snmpd` daemon responds to polling requests on the interfaces of the VRF on which the request comes in. For information about configuring SNMP version 1, 2c, and 3 Traps and (v3) Inform messages, refer to Simple Network Management Protocol - SNMP.

> **NOTE**

The message `Duplicate IPv4 address detected, some interfaces may not be visible in IP-MIB` displays after starting `snmpd` in the management VRF. This is because the IP-MIB assumes that the same IP address cannot be used twice on the same device; the IP-MIB is not VRF aware. This message is a warning that the SNMP IP-MIB detects overlapping IP addresses on the system; it does *not* indicate a problem and is non-impacting to the operation of the switch.

ping or traceroute on the Management VRF

By default, when you issue a `ping` or `traceroute`, the packet is sent to the dataplane network (the main routing table). To use `ping` or `traceroute` on the management network, use `ping -I mgmt` or `traceroute -i mgmt`. To select a source address within the management VRF, use the `-s` flag for `traceroute`.

```
cumulus@switch:~$ ping -I mgmt <destination-ip>
```

Or:

```
cumulus@switch:~$ sudo traceroute -i mgmt -s <source-ip>
<destination-ip>
```

For additional information on using `ping` and `traceroute`, see Network Troubleshooting.

Run Services as a Non-root User

To run services in the management VRF as a non-root user, you need to create a custom service based on the original service file. The following example commands configure the SSH service to run in the management VRF as a non-root user.

1. Run the following command to create a custom service file in the `/etc/systemd/system` direcotry.

```
cumulus@switch:~$ sudo -E systemctl edit --full ssh.service
```

2. If a *User* directive exists under *[Service]*, comment it out.

```
cumulus@switch:~$ sudo nano /etc/systemd/system/ssh.service
...
```

```
[Service]
#User=username
ExecStart=/usr/local/bin/ssh agent -data-dir=/tmp/ssh
-bind=192.168.0.11
...
```

3. Modify the *ExecStart* line to `/usr/bin/ip vrf exec mgmt /sbin/runuser -u USER -- ssh`:

```
...
[Service]
#User=username
ExecStart=/usr/bin/ip vrf exec mgmt /sbin/runuser -u cumulus
-- ssh
...
```

OSPF and BGP

FRRouting is VRF-aware and sends packets based on the switch port routing table. This includes BGP peering via loopback interfaces. BGP does routing lookups in the default table. However, depending on how your routes are redistributed, you might want to perform the following modification.

Management VRF uses the mgmt table, including local routes. It does not affect how the routes are redistributed when using routing protocols such as OSPF and BGP.

To redistribute the routes in your network, use the `redistribute connected` command under BGP or OSPF. This enables the directly-connected network out of eth0 to be advertised to its neighbor.

> **NOTE**
>
> This also creates a route on the neighbor device to the management network through the data plane, which might not be desired.

Consider lways using route maps to control the advertised networks redistributed by the `redistribute connected` command. For example, you can specify a route map to redistribute routes in this way (for both BGP and OSPF):

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add routing route-map REDISTRIBUTE-
CONNECTED deny 100 match interface eth0
cumulus@switch:~$ net add routing route-map REDISTRIBUTE-
CONNECTED permit 1000
```

The NCLU and vtysh commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
<routing-protocol>
redistribute connected route-map REDISTRIBUTE-CONNECTED

route-map REDISTRIBUTE-CONNECTED deny 100
match interface eth0
!
route-map REDISTRIBUTE-CONNECTED permit 1000
...
```

SSH within a Management VRF Context

If you SSH to the switch through a switch port, SSH works as expected. If

you need to SSH from the device out of a switch port, use the `ip vrf exec default ssh <switch-port-ip-address>` command. For example:

```
cumulus@switch:~$ sudo ip vrf exec default ssh 10.23.23.2
10.3.3.3
```

View the Routing Tables

NCLU Commands | Linux Commands

The `ip route show` command shows the switch port (*main*) table. You can see the dataplane routing table with the `net show route vrf main` command.

To show information for eth0 (the management routing table), run the `net show route vrf mgmt` command:

```
cumulus@switch:~$ net show route vrf mgmt
default via 192.168.0.1 dev eth0
```

```
cumulus@switch:~$ net show route
default via 10.23.23.3 dev swp17  proto zebra  metric 20
10.3.3.3 via 10.23.23.3 dev swp17
10.23.23.0/24 dev swp17  proto kernel  scope link  src
10.23.23.2
192.168.0.0/24 dev eth0  proto kernel  scope link  src
192.168.0.11
```

If you run the `ip route get` command to return information about a single route, the command resolves over the *mgmt* table by default. To obtain information about the route in the switching silicon, run this command:

```
cumulus@switch:~$ net show route <ip-address>
```

To show the route for any VRF, run the `net show route vrf <vrf-name>`

mgmt Interface Class

In `ifupdown2`, interface classes are used to create a user-defined grouping for interfaces. The special class *mgmt* is available to separate the management interfaces of the switch from the data interfaces. This allows you to manage the data interfaces by default using `ifupdown2` commands. Performing operations on the *mgmt* interfaces requires specifying the `--allow-mgmt` option, which prevents inadvertent outages on the management interfaces. Cumulus Linux by default brings up all interfaces in both the *auto* (default) class and the *mgmt* interface class when the switch boots.

WARNING

The management VRF interface class is not supported if you are configuring Cumulus Linux using NCLU.

You configure the management interface in the `/etc/network/interfaces` file. In the example below, the management interface eth0 and the management VRF stanzas are added to the *mgmt* interface class:

```
...
```

```
auto lo
iface lo inet loopback

allow-mgmt eth0
iface eth0 inet dhcp
    vrf mgmt

allow-mgmt mgmt
iface mgmt
    address 127.0.0.1/8
    address ::1/128
    vrf-table auto
...
```

When you run `ifupdown2` commands against the interfaces in the mgmt class, include `--allow=mgmt` with the commands. For example, to see which interfaces are in the mgmt interface class, run:

```
cumulus@switch:~$ ifquery l --allow=mgmt
eth0
mgmt
```

To reload the configurations for interfaces in the mgmt class, run:

```
cumulus@switch:~$ sudo ifreload --allow=mgmt
```

You can still bring the management interface up and down using `ifup eth0` and `ifdown eth0`.

Management VRF and DNS

Cumulus Linux supports both DHCP and static DNS entries over management VRF through IP FIB rules. These rules are added to direct lookups to the DNS addresses out of the management VRF.

For DNS to use the management VRF, the static DNS entries must reference the management VRF in the `/etc/resolv.conf` file. You cannot specify the same DNS server address twice to associate it with different VRFs.

For example, to specify DNS servers and associate some of them with the management VRF, run the following commands:

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add dns nameserver ipv4 192.0.2.1
cumulus@switch:~$ net add dns nameserver ipv4 198.51.100.31
vrf mgmt
cumulus@switch:~$ net add dns nameserver ipv4 203.0.113.13
vrf mgmt
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

NOTE

- Because DNS lookups are forced out of the management interface using FIB rules, this might affect data plane ports if you use overlapping addresses. For example, when the DNS server IP address is learned over the management VRF, a FIB rule is created for that IP address. When DHCP relay is configured for the same IP address, a DHCP discover packet received on the front panel port is forwarded out of the management interface (eth0) even though a route is present out the front-panel port.
- If you do not specify a DNS server and you lose in band

connectivity, DNS does not work through the management VRF. Cumulus Linux does not assume all DNS servers are reachable through the management VRF.

Protocol Independent Multicast - PIM

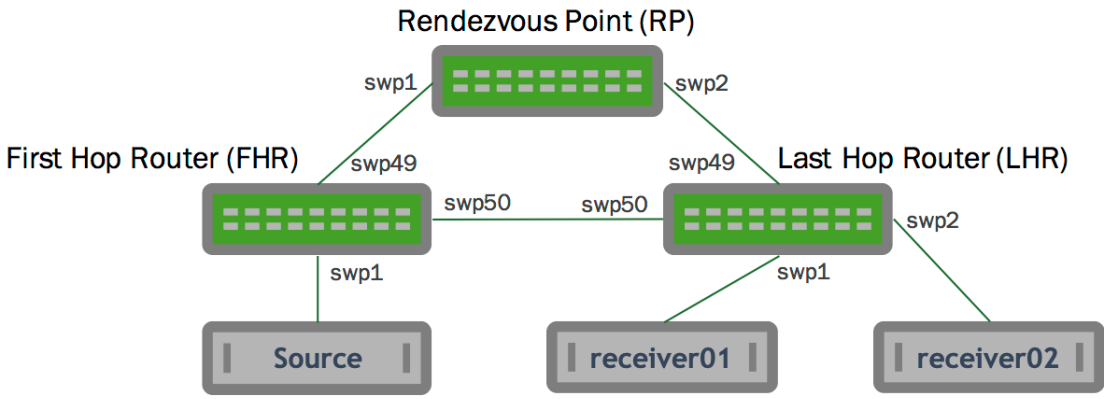
Protocol Independent Multicast (PIM) is a multicast control plane protocol that advertises multicast sources and receivers over a routed layer 3 network. Layer 3 multicast relies on PIM to advertise information about multicast capable routers, and the location of multicast senders and receivers. For this reason, multicast cannot be sent through a routed network without PIM.

PIM has two modes of operation: Sparse Mode (PIM-SM) and Dense Mode (PIM-DM).

> **NOTE**
>
> Cumulus Linux supports only PIM Sparse Mode.

Example PIM Topology

The following illustration shows a PIM configuration. The table below the illustration describes the network elements.



Network Element	Description
First Hop Router (FHR)	The router attached to the source. The FHR is responsible for the PIM register process.
Last Hop Router (LHR)	The last router in the path, attached to an interested multicast receiver. There is a single LHR for each network subnet with an interested receiver, however multicast groups can have multiple LHRs throughout the network.
Rendezvous Point (RP)	Allows for the discovery of multicast sources and multicast receivers. The RP is responsible for sending PIM Register Stop messages to FHRs. The PIM RP address must be globally routable.

Network Element	Description

	WARNING
	• `zebra` does not resolve the next hop for the RP through the default route. To prevent multicast forwarding from failing, either provide a specific route to the RP or specify the following command to be able to resolve the next hop for the RP through the default route:
	`cumulus@switch:~$ sudo vtysh`
	`switch#`

Network Element	Description
	``` configure terminal switch(config)# ip nht resolve-via-default switch(config)# exit switch# write memory ``` • Do not use a spine switch as an RP. If you are running BGP on a spine switch and it is **not** configured for allow-as in origin, BGP does not accept routes learned through

Network Element	Description
	other spines that do not originate on the spine itself. The RP must route to a multicast source. During a single failure scenario, this is not possible if the RP is on the spine. This also applies to Multicast Source Discovery Protocol (MSDP).
PIM Shared Tree (RP Tree) or (*,G) Tree	The multicast tree rooted at the RP. When receivers want to join a multicast group, join messages are sent along the shared tree towards the RP.
PIM Shortest Path Tree (SPT) or (S,G) Tree	The multicast tree rooted at the multicast source for a given

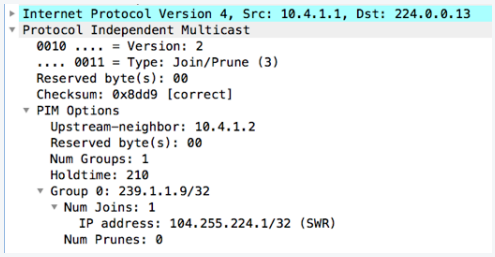
Network Element	Description
	group. Each multicast source has a unique SPT. The SPT can match the RP Tree, but this is not a requirement. The SPT represents the most efficient way to send multicast traffic from a source to the interested receivers.
Outgoing Interface (OIF)	Indicates the interface on which a PIM or multicast packet is to be sent out. OIFs are the interfaces towards the multicast receivers.
Incoming Interface (IIF)	Indicates the interface on which a multicast packet is received. An IIF can be the interface towards the source or towards the RP.
Reverse Path Forwarding Interface (RPF Interface)	The path used to reach the RP or source. There must be a valid PIM neighbor to determine the RPF unless directly connected to source.
Multicast Route (mroute)	Indicates the multicast source and multicast group as well as associated OIFs, IIFs, and RPF information.
Star-G mroute (*,G)	Represents the RP Tree. The * is a wildcard indicating any

Network Element	Description
	multicast source. The G is the multicast group. An example (*,G) is (*, 239.1.2.9).
S-G mroute (S,G)	This is the mroute representing the source entry. The S is the multicast source IP. The G is the multicast group. An example (S,G) is (10.1.1.1, 239.1.2.9).

PIM Messages

PIM Message	Description
PIM Hello	Announce the presence of a multicast router on a segment. PIM hellos are sent every 30 seconds by default. For example: `22.1.2.2 > 224.0.0.13 PIMv2, length 34 Hello, cksum 0xfdbb (correct) Hold Time Option (1), length 2, Value: 1m45s 0x0000: 0069 LAN Prune Delay Option (2), length 4, Value: T-bit=0, LAN delay`

PIM Message	Description
	500ms, Override interval 2500ms 0x0000: 01f4 09c4 DR Priority Option (19), length 4, Value: 1 0x0000: 0000 0001 Generation ID Option (20), length 4, Value 0x2459b190 0x0000: 2459 b190
PIM Join/Prune (J/P)	Indicate the groups that a multicast router wants to receive or no longer receive. Often PIM join/prune messages are described as distinct message types, but are actually a single PIM message with a list of groups to join and a second list of groups to leave. PIM J/P messages can be to join or prune from the SPT or RP trees (also called (*,G) joins or (S,G) joins). **Note**: PIM join/prune messages are sent to PIM neighbors on individual interfaces. Join/prune

PIM Message	Description
	messages are never unicast.  This PIM join/prune is for group 239.1.1.9, with 1 join and 0 prunes for the group. Join/prunes for multiple groups can exist in a single packet. The following shows an S,G Prune example: `21:49:59.470885 IP (tos 0x0, ttl 255, id 138, offset 0, flags [none], proto PIM (103), length 54) 22.1.2.2 > 224.0.0.13: PIMv2, length 34 Join / Prune, cksum 0xb9e5 (correct), upstream-neighbor: 22.1.2.1 1 group(s), holdtime: 3m30s`

PIM Message	Description
	group #1: 225.1.0.0, joined sources: 0, pruned sources: 1 pruned source #1: 33.1.1.1(S)
PIM Register	Unicast packets sent from an FHR destined to the RP to advertise a multicast group. The FHR fully encapsulates the original multicast packet in PIM register messages. The RP is responsible for decapsulating the PIM register message and forwarding it along the (*,G) tree towards the receivers.
PIM Null Register	A special type of PIM register message where the Null-Register flag is set within the packet. Null register messages are used for an FHR to signal to an RP that a source is still sending multicast traffic. Unlike normal PIM register messages, null register messages do not encapsulate the original data packet.
PIM Register Stop	Sent by an RP to the FHR to

PIM Message	Description
	indicate that PIM register messages must no longer be sent. For example: `21:37:00.419379 IP (tos 0x0, ttl 255, id 24, offset 0, flags [none], proto PIM (103), length 38) 100.1.2.1 > 33.1.1.10: PIMv2, length 18 Register Stop, cksum 0xd8db (correct) group=225.1.0.0 source=33.1.1.1`
IGMP Membership Report (IGMP Join)	Sent by multicast receivers to tell multicast routers of their interest in a specific multicast group. IGMP join messages trigger PIM *,G joins. IGMP version 2 queries are sent to the all hosts multicast address, 224.0.0.1. IGMP version 2 reports (joins) are sent to the group's multicast address. IGMP version 3 messages are sent to an IGMP v3 specific multicast address, 224.0.0.22.

PIM Message	Description
IGMP Leave	Tell a multicast router that a multicast receiver no longer wants the multicast group. IGMP leave messages trigger PIM *,G prunes.

PIM Neighbors

When PIM is configured on an interface, `PIM Hello` messages are sent to the link local multicast group 224.0.0.13. Any other router configured with PIM on the segment that hears the PIM Hello messages builds a PIM neighbor with the sending device.

> **NOTE**
>
> PIM neighbors are stateless. No confirmation of neighbor relationship is exchanged between PIM endpoints.

Configure PIM

To configure PIM, run the following commands:

NCLU Commands vtysh Commands

1. Configure the PIM interfaces:

```
cumulus@switch:~$ net add interface swp1 pim
```

> **NOTE**
>
> You must enable PIM on all interfaces facing multicast sources or multicast receivers, as well as on the interface where the RP address is configured.

> **NOTE**
>
> In Cumulus Linux 4.0 and later the *sm* keyword is no longer required. In Cumulus Linux releases 3.7 and earlier, the correct command is `net add interface swp1 pim sm`.

2. Enable IGMP on all interfaces with hosts attached. IGMP version 3 is the default. Only specify the version if you exclusively want to use IGMP version 2. SSM requires the use of IGMP version 3.

```
cumulus@switch:~$ net add interface swp1 igmp
```

> **NOTE**

PIM Sparse Mode (PIM-SM)

PIM Sparse Mode (PIM-SM) is a *pull* multicast distribution method; multicast traffic is only sent through the network if receivers explicitly ask for it. When a receiver *pulls* multicast traffic, the network must be periodically notified that the receiver wants to continue the multicast stream.

> **NOTE**
>
> This behavior is in contrast to PIM Dense Mode (PIM-DM), where traffic is flooded, and the network must be periodically notified that the receiver wants to stop receiving the multicast stream.

PIM-SM has three configuration options:

- Any-source Mulitcast (ASM) is the traditional, and most commonly deployed PIM implementation. ASM relies on rendezvous points to connect multicast senders and receivers that then dynamically determine the shortest path through the network between source and receiver, to efficiently send multicast traffic.
- Bidirectional PIM (BiDir) forwards all traffic through the multicast rendezvous point (RP) instead of tracking multicast source IPs, allowing for greater scale while resulting in inefficient forwarding of network traffic.
- Source Specific Multicast (SSM) requires multicast receivers to know exactly from which source they want to receive multicast traffic instead

of relying on multicast rendezvous points. SSM requires the use of IGMPv3 on the multicast clients.

> **NOTE**
>
> Cumulus Linux only supports ASM and SSM. PIM BiDir is not currently supported.

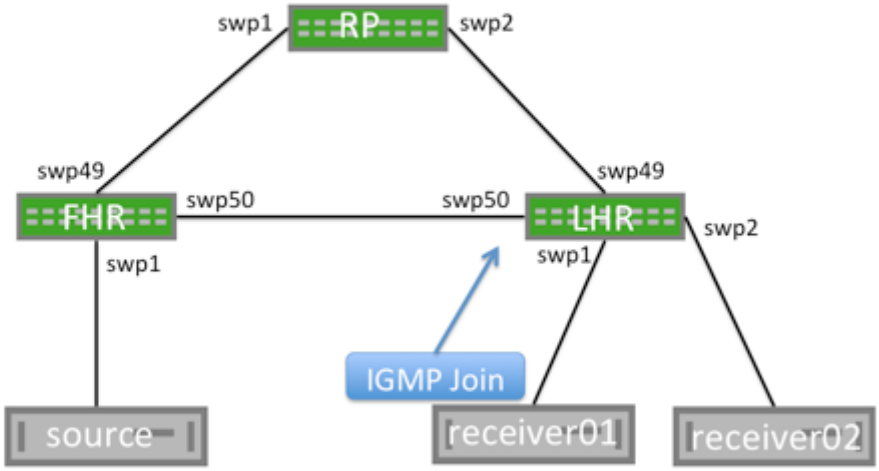
For additional information, see RFC 7761 - Protocol Independent Multicast - Sparse Mode.

Any-source Multicast Routing (ASM)

Multicast routing behaves differently depending on whether the source is sending before receivers request the multicast stream, or if a receiver tries to join a stream before there are any sources.

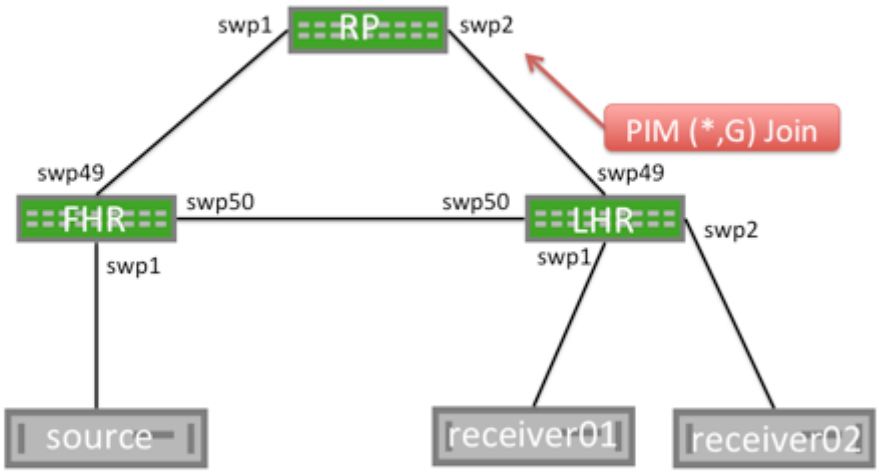
Receiver Joins First

When a receiver joins a group, an IGMP membership join message is sent to the IGMPv3 multicast group, 224.0.0.22. The PIM multicast router for the segment that is listening to the IGMPv3 group receives the IGMP membership join message and becomes an LHR for this group.



This creates a (*,G) mroute with an OIF of the interface on which the IGMP Membership Report is received and an IIF of the RPF interface for the RP.

The LHR generates a PIM (*,G) join message and sends it from the interface towards the RP. Each multicast router between the LHR and the RP builds a (*,G) mroute with the OIF being the interface on which the PIM join message is received and an Incoming Interface of the reverse path forwarding interface for the RP.



NOTE

When the RP receives the (*,G) Join message, it does not send any additional PIM join messages. The RP maintains a (*,G) state as long as the receiver wants to receive the multicast group.

NOTE

Unlike multicast receivers, multicast sources do not send IGMP (or

PIM) messages to the FHR. A multicast source begins sending, and the FHR receives the traffic and builds both a (*,G) and an (S,G) mroute. The FHR then begins the PIM register process.

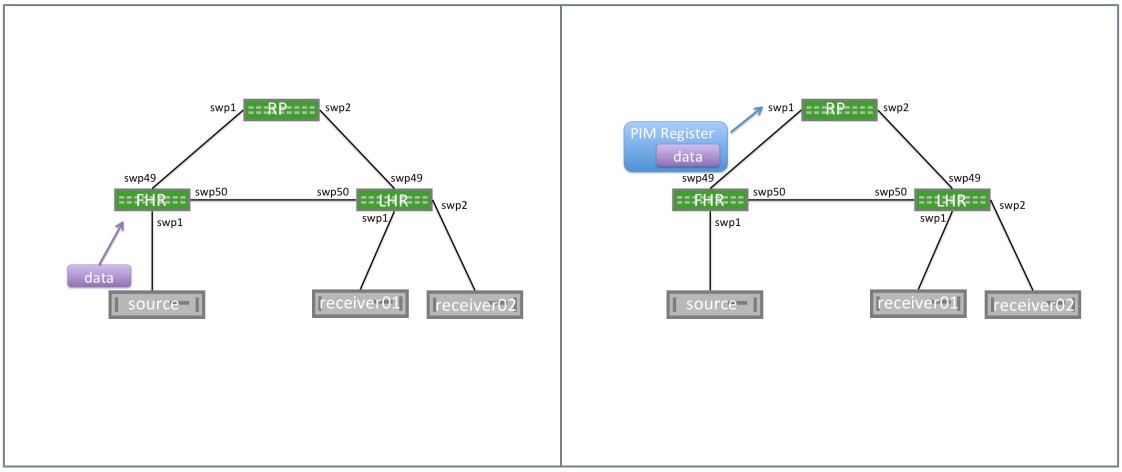
PIM REGISTER PROCESS

When a first hop router (FHR) receives a multicast data packet from a source, the FHR does not know if there are any interested multicast receivers in the network. The FHR encapsulates the data packet in a unicast PIM register message. This packet is sourced from the FHR and destined to the RP address. The RP builds an (S,G) mroute, decapsulates the multicast packet, and forwards it along the (*,G) tree.

As the unencapsulated multicast packet travels down the (*,G) tree towards the interested receivers, at the same time, the RP sends a PIM (S,G) join towards the FHR. This builds an (S,G) state on each multicast router between the RP and FHR.

When the FHR receives a PIM (S,G) join, it continues encapsulating and sending PIM register messages, but also makes a copy of the packet and sends it along the (S,G) mroute.

The RP then receives the multicast packet along the (S,G) tree and sends a PIM register stop to the FHR to end the register process.



PIM SPT SWITCHOVER

When the LHR receives the first multicast packet, it sends a PIM (S,G) join towards the FHR to efficiently forward traffic through the network. This builds the shortest path tree (SPT), or the tree that is the shortest path to the source. When the traffic arrives over the SPT, a PIM (S,G) RPT prune is sent up the shared tree towards the RP. This removes multicast traffic from the shared tree; multicast data is only sent over the SPT.

You can configure SPT switchover on a per-group basis, allowing for some groups to never switch to a shortest path tree; this is also called *SPT infinity*. The LHR now sends both (*,G) joins and (S,G) RPT prune messages towards the RP.

To configure a group to never follow the SPT, create the necessary prefix-lists, then configure SPT switchover for the *spt-range* prefix-list:

```
cumulus@switch:~$ sudo vtysh
switch# configure terminal
switch(config)# ip prefix-list spt-range permit 235.0.0.0/8 ge
32
switch(config)# ip prefix-list spt-range permit 238.0.0.0/8 ge
32
switch(config)# ip pim spt-switchover infinity prefix-list spt-
range
switch(config)# end
switch# exit
cumulus@switch:~$
```

To view the configured prefix-list, run the `vtysh show ip mroute` command or the NCLU `net show mroute` command. The following command shows that *235.0.0.0* is configured for SPT switchover, identified by *pimreg*.

```
switch# show ip mroute
Source          Group           Proto  Input      Output
TTL  Uptime
*               235.0.0.0       IGMP   swp31s0    pimreg
1    00:03:3
                                IGMP              br1
1    00:03:38
```

```
*               238.0.0.0       IGMP    swp31s0     br1
1     00:02:08
```

Sender Starts Before Receivers Join

A multicast sender can send multicast data without any additional IGMP or PIM signaling. When the FHR receives the multicast traffic, it encapsulates it and sends a PIM register to the rendezvous point (RP).

When the RP receives the PIM register, it builds an (S,G) mroute; however, there is no (*,G) mroute and no interested receivers.

The RP drops the PIM register message and immediately sends a PIM register stop message to the FHR.

Receiving a PIM register stop without any associated PIM joins leaves the FHR without any outgoing interfaces. The FHR drops this multicast traffic until a PIM join is received.

> **NOTE**
>
> PIM register messages are sourced from the interface that receives the multicast traffic and are destined to the RP address. The PIM register is not sourced from the interface towards the RP.

PIM Null-Register

To notify the RP that multicast traffic is still flowing when the RP has no receiver, or if the RP is not on the SPT tree, the FHR periodically sends PIM null register messages. The FHR sends a PIM register with the Null-Register flag set, but without any data. This special PIM register notifies the RP that a multicast source is still sending, in case any new receivers come online.

After receiving a PIM Null-Register, the RP immediately sends a PIM register stop to acknowledge the reception of the PIM null register message.

Source Specific Multicast Mode (SSM)

The source-specific multicast method uses prefix lists to configure a receiver to only allow traffic to a multicast address from a single source. This removes the need for an RP, as the source must be known before traffic can be accepted. There is no additional PIM configuration required to enable SSM beyond enabling PIM and IGMPv3 on the relevant interfaces.

Receiver Joins First

When a receiver sends an IGMPv3 Join with the source defined the LHR builds an S,G entry and sends a PIM S,G join to the PIM neighbor closest to the source, according to the routing table.

The full path between LHR and FHR contains an S,G state, although no multicast traffic is flowing. Periodic IGMPv3 joins between the receiver and LHR, as well as PIM S,G joins between PIM neighbors, maintain this state until the receiver leaves.

When the sender begins, traffic immediately flows over the pre-built SPT from the sender to the receiver.

Sender Starts Before Receivers Join

In SSM when a sender begins sending, the FHR does not have any existing mroutes. The traffic is dropped and nothing further happens until a receiver joins. SSM does no rely on an RP; there is no PIM Register process.

Differences between Source Specific Multicast and Any Source Multicast

SSM differs from ASM multicast in the following ways:

- An RP is not configured or used. SSM does not require an RP since receivers always know the addresses of the senders.
- There is no *,G PIM Join message. The multicast sender is always known so the PIM Join messages used in SSM are always S,G Join messages.
- There is no Shared Tree or *,G tree. The PIM join message is always sent towards the source, building the SPT along the way. There is no shared tree or *,G state.
- IGMPv3 is required. ASM allows for receivers to specify only the group they want to join without knowledge of the sender. This can be done in both IGMPv2 and IGMPv3. Only IGMPv3 supports requesting a specific source for a multicast group (the sending an S,G IGMP join).
- No PIM Register process or SPT Switchover. Without a shared tree or RP, there is no need for the PIM register process. S,G joins are sent directly towards the FHR.

PIM Active-Active with MLAG

For a multicast sender or receiver to be supported over a dual-attached MLAG bond, you must configure `pim active-active`.

To configure PIM active-active with MLAG, run the following commands:

NCLU Commands vtysh Commands

1. On the VLAN interface where multicast sources or receivers exist, configure `pim active-active` and `igmp`. For example:

```
cumulus@switch:~$ net add vlan 12 pim active-active
cumulus@switch:~$ net add vlan 12 igmp
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

> **NOTE**
> Enabling PIM active-active automatically enables PIM on that interface.

2. Confirm PIM active-active is configured with the `net show pim mlag summary` command:

```
cumulus@leaf01:mgmt:~$ net show pim mlag summary
MLAG daemon connection: up
MLAG peer state: up
Zebra peer state: up
MLAG role: PRIMARY
Local VTEP IP: 0.0.0.0
Anycast VTEP IP: 0.0.0.0
Peerlink: peerlink.4094
Session flaps: mlagd: 0 mlag-peer: 0 zebra-peer: 0
Message Statistics:
```

Multicast Sender

When a multicast sender is attached to an MLAG bond, the sender hashes the outbound multicast traffic over a single member of the bond. Traffic is received on one of the MLAG enabled switches. Regardless of which switch receives the traffic, it is forwarded over the MLAG peer link to the other MLAG-enabled switch, because the peerlink is always considered a multicast router port and will always receive the multicast stream.

> **NOTE**
>
> Traffic from multicast sources attached to an MLAG bond is always sent over the MLAG peerlink. Be sure to size the peerlink appropriately to accommodate this traffic.

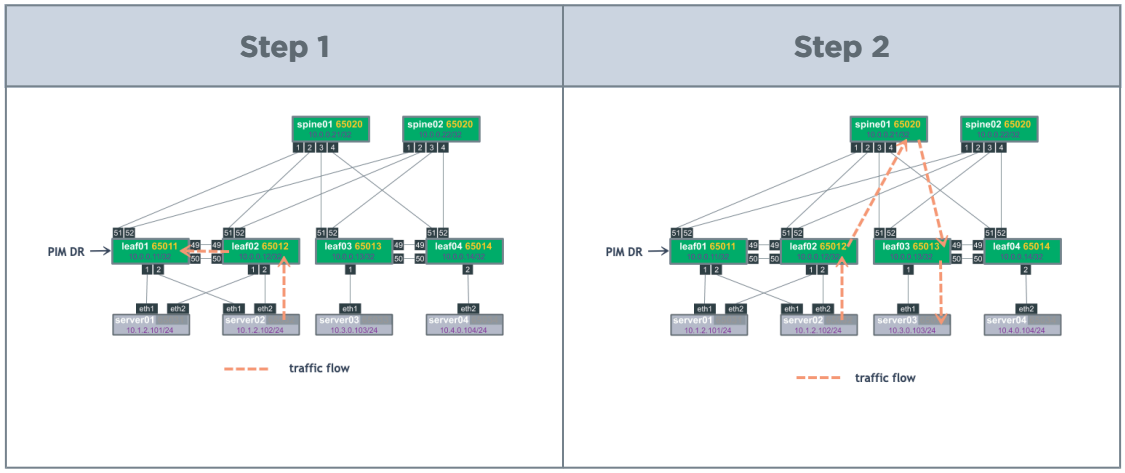
The PIM DR for the VLAN where the source resides is responsible for sending the PIM register towards the RP. The PIM DR is the PIM speaker with the highest IP address on the segment. After the PIM register process is complete and traffic is flowing along the Shortest Path Tree (SPT), either MLAG switch will forward traffic towards the receivers.

Examples are provided below that show the flow of traffic between server02 and server03:

- **Step 1**: server02 sends traffic to leaf02. leaf02 forwards traffic to leaf01 because the peerlink is a multicast router port. leaf01 also receives a PIM

register from leaf02. leaf02 syncs the *,G table from leaf01 as an MLAG active-active peer.

- **Step 2**: leaf02 has the *,G route indicating that traffic is to be forwarded toward spine01. Either leaf02 or leaf01 sends this traffic directly based on which MLAG switch receives it from the attached source. In this case, leaf02 receives the traffic on the MLAG bond and forwards it directly upstream.



To show the PIM DR, run the NCLU `net show pim interface` command or the `vtysh` `show ip pim interface` command. The following example shows that in Vlan12 the DR is 10.1.2.12.

```
cumulus@leaf01:mgmt:~$ net show pim interface
Interface         State          Address  PIM Nbrs
PIM DR  FHR IfChannels
lo                up        10.0.0.11         0
```

```
local    0          0
pimreg               up          0.0.0.0         0
local    0          0
swp51                up        10.0.0.11         1
10.0.0.21    0          4
swp52                up        10.0.0.11         1
10.0.0.22    0          0
vlan12               up        10.1.2.11         1
10.1.2.12    0          2
```

PIM joins sent towards the source can be ECMP load shared by upstream PIM neighbors (spine01 and spine02 in the example above). Either MLAG member can receive the PIM join and forward traffic, regardless of DR status.

Multicast Receiver

A dual-attached multicast receiver sends an IGMP join on the attached VLAN. The specific interface that is used is determined based on the host. The IGMP join is received on one of the MLAG switches, and the IGMP join is added to the IGMP Join table and layer 2 MDB table. The layer 2 MDB table, like the unicast MAC address table, is synced via MLAG control messages over the peerlink. This allows both MLAG switches to program IGMP and MDB table forwarding information.

Both switches send *,G PIM Join messages towards the RP. If the source is

already sending, both MLAG switches receive the multicast stream.

> **NOTE**
>
> Traditionally, the PIM DR is the only node to send the PIM *,G Join, but to provide resiliency in case of failure, both MLAG switches send PIM *,G Joins towards the RP to receive the multicast stream.

To prevent duplicate multicast packets, a Designated Forward (DF) is elected. The DF is the `primary` member of the MLAG pair. As a result, the MLAG secondary puts the VLAN in the Outgoing Interface List (OIL), preventing duplicate multicast traffic.

Additional PIM Features

Custom SSM multicast group ranges

PIM considers `232.0.0.0/8` the default SSM range. You can change the SSM range by defining a prefix-list and attaching it to the `ssm-range` command. You can change the default SSM group or add additional group ranges to be treated as SSM groups.

> **NOTE**

If you use the `ssm-range` command, **all** SSM ranges must be in the prefix-list, including `232.0.0.0/8`.

NCLU Commands vtysh Commands

Create a prefix-list with the `permit` keyword to match address ranges that should be treated as SSM groups and `deny` keyword for those ranges which should not be treated as SSM enabled ranges.

```
cumulus@switch:~$ net add routing prefix-list ipv4 my-
custom-ssm-range seq 5 permit 232.0.0.0/8 ge 32
cumulus@switch:~$ net add routing prefix-list ipv4 my-
custom-ssm-range seq 10 permit 238.0.0.0/8 ge 32
```

Apply the custom prefix-list as an `ssm-range`

```
cumulus@switch:~$ net add pim ssm prefix-list my-custom-ssm-
range
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

To view the configured prefix-lists, run the `net show ip prefix-list` command:

```
cumulus@switch:~$ net show ip prefix-list my-custom-ssm-
range
ZEBRA: ip prefix-list my-custom-ssm-range: 1 entries
   seq 5 permit 232.0.0.0/8 ge 32
PIM: ip prefix-list my-custom-ssm-range: 1 entries
   seq 10 permit 238.0.0.0/8 ge 32
```

PIM and ECMP

PIM uses the RPF procedure to choose an upstream interface to build a forwarding state. If you configure equal-cost multipaths (ECMP), PIM chooses the RPF based on the ECMP hash algorithm.

NCLU Commands vtysh Commands

Run the `net add pim ecmp` command to enable PIM to use all the available nexthops for the installation of mroutes. For example, if you have four-way ECMP, PIM spreads the S,G and *,G mroutes across the four different paths.

```
cumulus@switch:~$ net add pim ecmp
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Run the `ip pim ecmp rebalance` command to recalculate all stream paths in the event of a loss of path over one of the ECMP paths. Without this command, only the streams that are using the path that is lost are moved to alternate ECMP paths. Rebalance does not affect existing groups.

```
cumulus@switch:~$ net add pim ecmp rebalance
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

WARNING

The rebalance command might cause some packet loss.

To show which nexthop is selected for a specific source/group, run the `show ip pim nexthop` command from the `vtysh` shell:

```
cumulus@switch:~$ sudo vtysh
switch# show ip pim nexthop
Number of registered addresses: 3
Address         Interface      Nexthop
-----------------------------------------
6.0.0.9         swp31s0        169.254.0.9
6.0.0.9         swp31s1        169.254.0.25
6.0.0.11        lo             0.0.0.0
6.0.0.10        swp31s0        169.254.0.9
6.0.0.10        swp31s1        169.254.0.25
```

IP Multicast Boundaries

Multicast boundaries enable you to limit the distribution of multicast traffic by setting boundaries with the goal of pushing multicast to a subset of the network.

With such boundaries in place, any incoming IGMP or PIM joins are dropped or accepted based upon the prefix-list specified. The boundary is implemented by applying an IP multicast boundary OIL (outgoing interface list) on an interface.

To configure the boundary, first create a prefix-list as described above, then run the following commands to configure the IP multicast boundary:

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add interface swp1 multicast boundary
oil <prefix-list>
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Multicast Source Discovery Protocol (MSDP)

You can use the Multicast Source Discovery Protocol (MSDP) to connect multiple PIM-SM multicast domains together, using the PIM-SM RPs. By configuring any cast RPs with the same IP address on multiple multicast switches (primarily on the loopback interface), the PIM-SM limitation of only one RP per multicast group is relaxed. This allows for an increase in both failover and load-balancing throughout.

When an RP discovers a new source (typically a PIM-SM register message), a source-active (SA) message is sent to each MSDP peer. The peer then determines if any receivers are interested.

NOTE

Cumulus Linux MSDP support is primarily for anycast-RP configuration, rather than multiple multicast domains. You must

configure each MSDP peer in a full mesh, as SA messages are not received and reforwarded.

NOTE

Cumulus Linux currently only supports one MSDP mesh group.

The following steps demonstrate how to configure a Cumulus switch to use the MSDP:

NCLU Commands | vtysh Commands

1. Add an anycast IP address to the loopback interface for each RP in the domain:

```
cumulus@rp01:~$ net add loopback lo ip address 10.1.1.1/32
cumulus@rp01:~$ net add loopback lo ip address 10.1.1.100/
32
```

2. On every multicast switch, configure the group to RP mapping using the anycast address:

```
cumulus@switch:$ net add pim rp 10.1.1.100 224.0.0.0/4
cumulus@switch:$ net pending
cumulus@switch:$ net commit
```

3. Configure the MSDP mesh group for all active RPs (the following example uses 3 RPs):

> **NOTE**
>
> The mesh group must include all RPs in the domain as members, with a unique address as the source. This configuration results in MSDP peerings between all RPs.

```
cumulus@rp01:$ net add msdp mesh-group cumulus member
100.1.1.2
```

PIM in a VRF

VRFs divide the routing table on a per-tenant basis, ultimately providing for separate layer 3 networks over a single layer 3 infrastructure. With a VRF, each tenant has its own virtualized layer 3 network, so IP addresses can overlap between tenants.

PIM in a VRF enables PIM trees and multicast data traffic to run inside a layer 3 virtualized network, with a separate tree per domain or tenant. Each VRF has its own multicast tree with its own RP(s), sources, and so on. Therefore, you can have one tenant per corporate division, client, or product; for example.

VRFs on different switches typically connect or are peered over subinterfaces, where each subinterface is in its own VRF, provided MP-BGP VPN is not enabled or supported.

To configure PIM in a VRF, run the following commands.

NCLU Commands vtysh Commands

First, add the VRFs and associate them with switch ports:

```
cumulus@switch:~$ net add vrf blue
cumulus@switch:~$ net add vrf purple
cumulus@switch:~$ net add interface swp1 vrf blue
cumulus@switch:~$ net add interface swp2 vrf purple
```

Then add the PIM configuration to FRR, review and commit the changes:

```
cumulus@switch:~$ net add interface swp1 pim sm
cumulus@switch:~$ net add interface swp2 pim sm
cumulus@switch:~$ net add bgp vrf blue auto 65001
cumulus@switch:~$ net add bgp vrf purple auto 65000
cumulus@switch:~$ net add bgp vrf blue router-id 10.1.1.1
cumulus@switch:~$ net add bgp vrf purple router-id 10.1.1.2
cumulus@switch:~$ net add bgp vrf blue neighbor swp1
interface remote-as external
cumulus@switch:~$ net add bgp vrf purple neighbor swp2
interface remote-as external
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

To show VRF information, run the NCLU `net show mroute vrf <vrf-name>` command or the `vtysh` `show ip mroute vrf <vrf-name>` command:

```
cumulus@fhr:~$ net show mroute vrf blue
Source          Group           Proto  Input      Output
TTL  Uptime
11.1.0.1        239.1.1.1       IGMP   swp32s0    swp32s1
1    00:01:13
                                IGMP               br0.200
1    00:01:13
*               239.1.1.2       IGMP   mars       pimreg1001
1    00:01:13
                                IGMP               swp32s1
1    00:01:12
                                IGMP               br0.200
1    00:01:13
```

BFD for PIM Neighbors

You can use bidirectional forward detection (BFD) for PIM neighbors to quickly detect link failures. When you configure an interface, include the `pim bfd` option. For example:

NCLU Commands vtysh Commands

```
cumulus@switch:~$ net add interface swp31s3 pim bfd
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Verify PIM

The following outputs are based on the Cumulus Reference Topology with `cldemo-pim`.

NCLU Commands vtysh Commands

Source Starts First

On the FHR, an mroute is built, but the upstream state is *Prune*. The FHR flag is set on the interface receiving multicast. Run the NCLU `net show` commands to review detailed output for the FHR. For example:

```
cumulus@fhr:~$ net show mroute
Source          Group           Proto  Input
Output     TTL  Uptime
172.16.5.105    239.1.1.1       none   br0
none       0    --:--:--
!
cumulus@fhr:~$ net show pim upstream
Iif Source Group State Uptime JoinTimer RSTimer KATimer
RefCnt
br0 172.16.5.105 239.1.1.1 Prune 00:07:40 --:--:-- 00:00:36
00:02:50 1
!
cumulus@fhr:~$ net show pim upstream-join-desired
Interface Source          Group           LostAssert Joins
PimInclude JoinDesired EvalJD
!
cumulus@fhr:~$ net show pim interface
Interface  State          Address  PIM Nbrs           PIM
DR  FHR
br0           up       172.16.5.1         0
local    1
swp51         up         10.1.0.17        1
local    0
```

Troubleshooting

FHR Stuck in Registering Process

When a multicast source starts, the FHR sends unicast PIM register messages from the RPF interface towards the source. After the PIM register is received by the RP, a `PIM register stop` message is sent from the RP to the FHR to end the register process. If an issue occurs with this communication, the FHR becomes stuck in the registering process, which can result in high CPU, as PIM register packets are generated by the FHR CPU and sent to the RP CPU.

To assess this issue:

Review the FHR. The output interface of `pimreg` can be seen here. If this does not change to an interface within a few seconds, the FHR is likely stuck.

```
cumulus@fhr:~$ net show mroute
Source          Group           Proto  Input      Output
TTL  Uptime
172.16.5.105    239.2.2.3       PIM    br0        pimreg
1    00:03:59
```

To troubleshoot the issue:

1. Validate that the FHR can reach the RP. If the RP and FHR can not communicate, the registration process fails:

```
cumulus@fhr:~$ ping 10.0.0.21 -I br0
PING 10.0.0.21 (10.0.0.21) from 172.16.5.1 br0: 56(84) bytes
of data.
^C
--- 10.0.0.21 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time
3000ms
```

2. On the RP, use `tcpdump` to see if the PIM register packets are arriving:

```
cumulus@rp01:~$ sudo tcpdump -i swp30
tcpdump: verbose output suppressed, use -v or -vv for full
protocol decode
listening on swp30, link-type EN10MB (Ethernet), capture size
262144 bytes
23:33:17.524982 IP 172.16.5.1 > 10.0.0.21: PIMv2, Register,
length 66
```

3. If PIM registration packets are being received, verify that they are seen by PIM by issuing `debug pim packets` from within FRRouting:

```
cumulus@fhr:~$ sudo vtysh -c "debug pim packets"
PIM Packet debugging is on

cumulus@rp01:~$ sudo tail /var/log/frr/frr.log
2016/10/19 23:46:51 PIM: Recv PIM REGISTER packet from
172.16.5.1 to 10.0.0.21 on swp30: ttl=255 pim_version=2
pim_msg_size=64 checksum=a681
```

4. Repeat the process on the FHR to see if PIM register stop messages are being received on the FHR and passed to the PIM process:

```
cumulus@fhr:~$ sudo tcpdump -i swp51
23:58:59.841625 IP 172.16.5.1 > 10.0.0.21: PIMv2, Register,
length 28
23:58:59.842466 IP 10.0.0.21 > 172.16.5.1: PIMv2, Register
Stop, length 18

cumulus@fhr:~$ sudo vtysh -c "debug pim packets"
PIM Packet debugging is on

cumulus@fhr:~$ sudo tail -f /var/log/frr/frr.log
2016/10/19 23:59:38 PIM: Recv PIM REGSTOP packet from
10.0.0.21 to 172.16.5.1 on swp51: ttl=255 pim_version=2
```

```
pim_msg_size=18 checksum=5a39
```

No *,G Is Built on LHR

The most common reason for a *,G to not be built on an LHR is for if both PIM **and** IGMP are not enabled on an interface facing a receiver.

```
lhr# show run
!
interface br0
 ip igmp
 ip ospf area 0.0.0.0
 ip pim sm
```

To troubleshoot this issue, if both PIM and IGMP are enabled, ensure that IGMPv3 joins are being sent by the receiver:

```
cumulus@lhr:~$ sudo tcpdump -i br0 igmp
tcpdump: verbose output suppressed, use -v or -vv for full
protocol decode
listening on br0, link-type EN10MB (Ethernet), capture size
262144 bytes
```

```
00:03:55.789744 IP 172.16.1.101 > igmp.mcast.net: igmp v3
report, 1 group record(s)
```

No mroute Created on FHR

To troubleshoot this issue:

1. Verify that multicast traffic is being received:

```
cumulus@fhr:~$ sudo tcpdump -i br0
tcpdump: verbose output suppressed, use -v or -vv for full
protocol decode
listening on br0, link-type EN10MB (Ethernet), capture size
262144 bytes
00:11:52.944745 IP 172.16.5.105.51570 > 239.2.2.9.1000: UDP,
length 9
```

2. Verify that PIM is configured on the interface facing the source:

```
fhr# show run
!
interface br0
```

```
ip ospf area 0.0.0.0
ip pim sm
```

3. If PIM is configured, verify that the RPF interface for the source matches the interface on which the multicast traffic is received:

```
fhr# show ip rpf 172.16.5.105
Routing entry for 172.16.5.0/24 using Multicast RIB
Known via "connected", distance 0, metric 0, best
* directly connected, br0
```

4. Verify that an RP is configured for the multicast group:

```
fhr# show ip pim rp-info
RP address       group/prefix-list   OIF         I am RP
10.0.0.21        224.0.0.0/4         swp51       no
```

No S,G on RP for an Active Group

An RP does not build an mroute when there are no active receivers for a multicast group, even though the mroute was created on the FHR.

```
cumulus@rp01:~$ net show mroute
Source          Group           Proto  Input      Output
TTL  Uptime
spine01#

cumulus@rp01:~$ net show mroute
Source          Group           Proto  Input      Output
TTL  Uptime
172.16.5.105    239.2.2.9       none   br0        none
0    --:--:--
```

This is expected behavior. You can see the active source on the RP with either the NCLU `net show pim upstream` command or the `vtysh` `show ip pim upstream` command:

```
cumulus@rp01:~$ net show pim upstream
Iif       Source          Group           State       Uptime
JoinTimer RSTimer   KATimer   RefCnt
swp30     172.16.5.105    239.2.2.9       Prune       00:08:03
--:--:--  --:--:--  00:02:20       1
```

No mroute Entry Present in Hardware

Use the `cl-resource-query` command to verify that the hardware IP multicast entry is the maximum value:

```
cumulus@switch:~$ cl-resource-query  | grep Mcast
  Total Mcast Routes:          450,   0% of maximum value    450
```

> **NOTE**
>
> You can also run the NCLU command equivalent:`net show system asic | grep Mcast`.

For Spectrum chipsets, refer to TCAM Resource Profiles for Spectrum Switches.

Verify MSDP Session State

To verify the state of MSDP sessions, run either the NCLU `net show msdp mesh-group` command or the `vtysh show ip msdp mesh-group` command:

```
cumulus@switch:~$ net show msdp mesh-group
Mesh group : pod1
```

```
  Source : 100.1.1.1
  Member                 State
  100.1.1.2        established
  100.1.1.3        established
cumulus@switch:~$
cumulus@switch:~$ net show msdp peer
Peer                      Local        State    Uptime   SaCnt
100.1.1.2             100.1.1.1  established  00:07:21       0
100.1.1.3             100.1.1.1  established  00:07:21       0
```

View the Active Sources

To review the active sources learned locally (through PIM registers) and from MSDP peers, run either the NCLU `net show msdp sa` command or the `vtysh` `show ip msdp sa` command:

```
cumulus@switch:~$ net show msdp sa
Source                    Group               RP  Local
SPT    Uptime
44.1.11.2             239.1.1.1        100.1.1.1      n    n
00:00:40
44.1.11.2             239.1.1.2        100.1.1.1      n    n
00:00:25
```

Example Configurations

▼ Complete Multicast Network Configuration Example

Considerations

- Cumulus Linux only supports *PIM sparse mode* (PIM-SM), *any-source multicast* (PIM-SM ASM), and *source-specific multicast* (SSM). *Dense mode* and *bidirectional multicast* are not supported.
- Non-native forwarding (register decapsulation) is not supported. Initial packet loss is expected while the PIM *,G tree is built from the rendezvous point to the FHR to trigger native forwarding.
- Cumulus Linux does not currently build an S,G mroute when forwarding over an *,G tree.

GRE Tunneling

WARNING

GRE Tunneling is an early access feature.

Generic Routing Encapsulation (GRE) is a tunneling protocol that encapsulates network layer protocols inside virtual point-to-point links over an Internet Protocol network. The two endpoints are identified by the tunnel source and tunnel destination addresses at each endpoint.

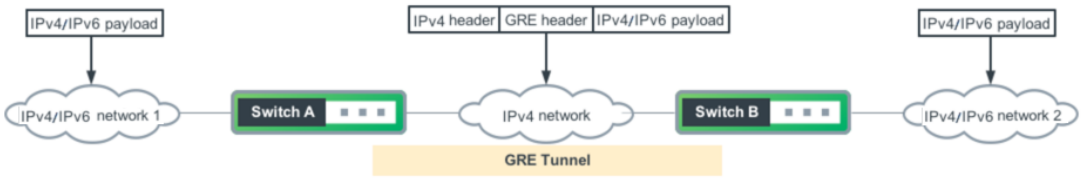
GRE packets travel directly between the two endpoints through a virtual tunnel. As a packet comes across other routers, there is no interaction with its payload; the routers only parse the outer IP packet. When the packet reaches the endpoint of the GRE tunnel, the outer packet is de-encapsulated, the payload is parsed, then forwarded to its ultimate destination.

GRE uses multiple protocols over a single-protocol backbone and is less demanding than some of the alternative solutions, such as VPN. You can use GRE to transport protocols that the underlying network does not support, work around networks with limited hops, connect non-contiguous subnets, and allow VPNs across wide area networks.

NOTE

- GRE tunneling is supported for switches with Spectrum ASICs only.
- Only static routes are supported as a destination for the tunnel interface.
- IPv6 endpoints are not supported.

The following example shows two sites that use IPv4 addresses. Using GRE tunneling, the two end points can encapsulate an IPv4 or IPv6 payload inside an IPv4 packet. The packet is routed based on the destination in the outer IPv4 header.



Configure GRE Tunneling

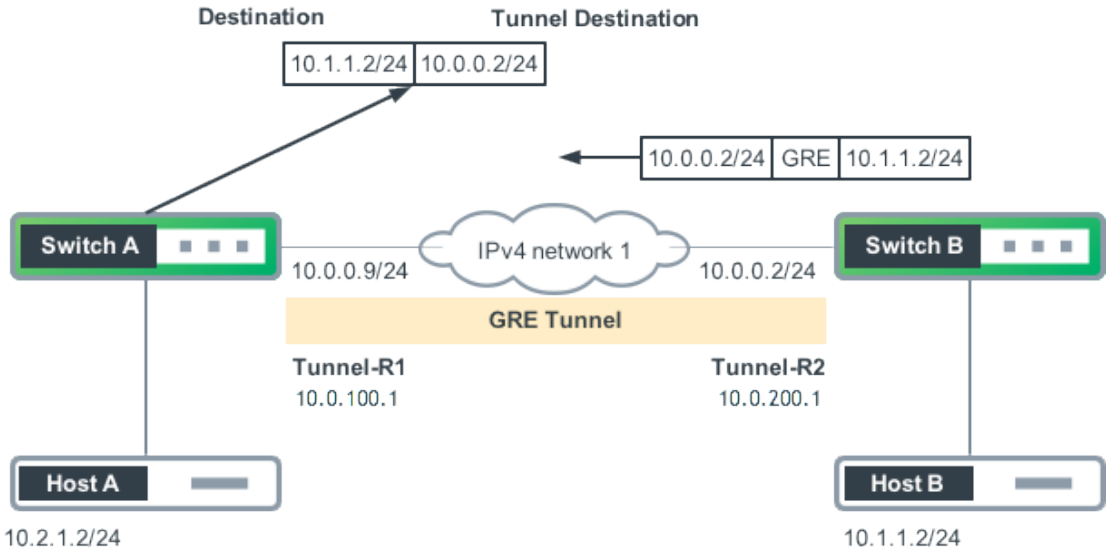
To configure GRE tunneling, you create a GRE tunnel interface with routes for tunneling on both endpoints as follows:

1. Create a tunnel interface by specifying an interface name, the tunnel

mode as `gre`, the source (local) and destination (remote) underlay IP address, and the `ttl` (optional).

2. Bring the GRE tunnel interface up.
3. Assign an IP address to the tunnel interface.
4. Add route entries to encapsulate the packets using the tunnel interface.

The following configuration example shows the commands used to set up a bidirectional GRE tunnel between two endpoints: `Tunnel-R1` and `Tunnel-R2`. The local tunnel endpoint for `Tunnel-R1` is 10.0.0.9 and the remote endpoint is 10.0.0.2. The local tunnel endpoint for `Tunnel-R2` is 10.0.0.2 and the remote endpoint is 10.0.0.9.



Tunnel-R1 commands:

```
cumulus@switch:~$ sudo ip tunnel add Tunnel-R2 mode gre remote
```

```
10.0.0.2 local 10.0.0.9 ttl 255
cumulus@switch:~$ sudo ip link set Tunnel-R2 up
cumulus@switch:~$ sudo ip addr add 10.0.100.1 dev Tunnel-R2
cumulus@switch:~$ sudo ip route add 10.0.100.0/24 dev Tunnel-R2
```

Tunnel-R2 commands:

```
cumulus@switch:~$ sudo ip Tunnel add Tunnel-R1 mode gre remote
10.0.0.9 local 10.0.0.2 ttl 255
cumulus@switch:~$ sudo ip link set Tunnel-R1 up
cumulus@switch:~$ sudo ip addr add 10.0.200.1 dev Tunnel-R1
cumulus@switch:~$ sudo ip route add 10.0.200.0/24 dev Tunnel-R1
```

To apply the GRE tunnel configuration automatically at reboot, instead of running the commands from the command line (as above), you can add the following commands directly in the `/etc/network/interfaces` file.

```
cumulus@switch:~$ sudo nano /etc/network/interfaces
...
# Tunnel-R1 configuration
auto swp1 #underlay interface for tunnel
```

```
iface swp1
    link-speed 10000
    link-duplex full
    link-autoneg off
    address 10.0.0.9/24

auto Tunnel-R2
iface Tunnel-R2
    tunnel-mode gre
    tunnel-endpoint 10.0.0.2
    tunnel-local 10.0.0.9
    tunnel-ttl 255
    address 10.0.100.1
    up ip route add 10.0.100.0/24 dev Tunnel-R2

# Tunnel-R2 configuration
auto swp1 #underlay interface for tunnel
iface swp1
    link-speed 10000
    link-duplex full
    link-autoneg off
    address 10.0.0.2/24

auto Tunnel-R1
```

```
iface Tunnel-R1
    tunnel-mode gre
    tunnel-endpoint 10.0.0.9
    tunnel-local 10.0.0.2
    tunnel-ttl 255
    address 10.0.200.1
    up ip route add 10.0.200.0/24 dev Tunnel-R1
```

Verify GRE Tunnel Settings

To check GRE tunnel settings, run the `ip tunnel show` command or the `ifquery --check` command. For example:

```
cumulus@switch:~$ ip tunnel show
gre0: gre/ip remote any local any ttl inherit nopmtudisc
Tunnel-R1: gre/ip remote 10.0.0.2 local 10.0.0.9 ttl 255
```

```
cumulus@switch:~$ ifquery --check Tunnel-R1
auto Tunnel-R1
iface Tunnel-R1
[pass]
```

```
        up ip route add 10.0.200.0/24 dev Tunnel-
R1                  []
        tunnel-ttl 255
[pass]
        tunnel-endpoint 10.0.0.9
[pass]
        tunnel-local 10.0.0.2
[pass]
        tunnel-mode gre
[pass]
        address 10.0.200.1/32
[pass]
```

Delete a GRE Tunnel Interface

To delete a GRE tunnel, remove the tunnel interface, and remove the routes configured with the tunnel interface, run the `ip tunnel del` command. For example:

```
cumulus@switch:~$ sudo ip tunnel del Tunnel-R2 mode gre remote
10.0.0.2 local 10.0.0.9 ttl 255
```

> **NOTE**
>
> You can delete a GRE tunnel directly from the `/etc/network/interfaces` file instead of using the `ip tunnel del` command. Make sure you run the `ifreload -a` command after you update the interfaces file.
>
> This action is disruptive as the tunnel is removed, then recreated with the new settings.

Change GRE Tunnel Settings

Use the `ip tunnel change` command to make changes to the GRE tunnel settings. The following example changes the remote underlay IP address from the original setting to 11.0.0.4:

```
cumulus@switch:~$ sudo ip tunnel change Tunnel-R2 mode gre
local 10.0.0.2 remote 10.0.0.4
```

> **NOTE**
>
> You can make changes to GRE tunnel settings directly in the `/etc/network/interfaces` file instead of using the `ip tunnel change`

command. Make sure you run the `ifreload -a` command after you update the interfaces file.

Network Address Translation - NAT

Network Address Translation (NAT) enables your network to use one set of IP addresses for internal traffic and a second set of addresses for external traffic.

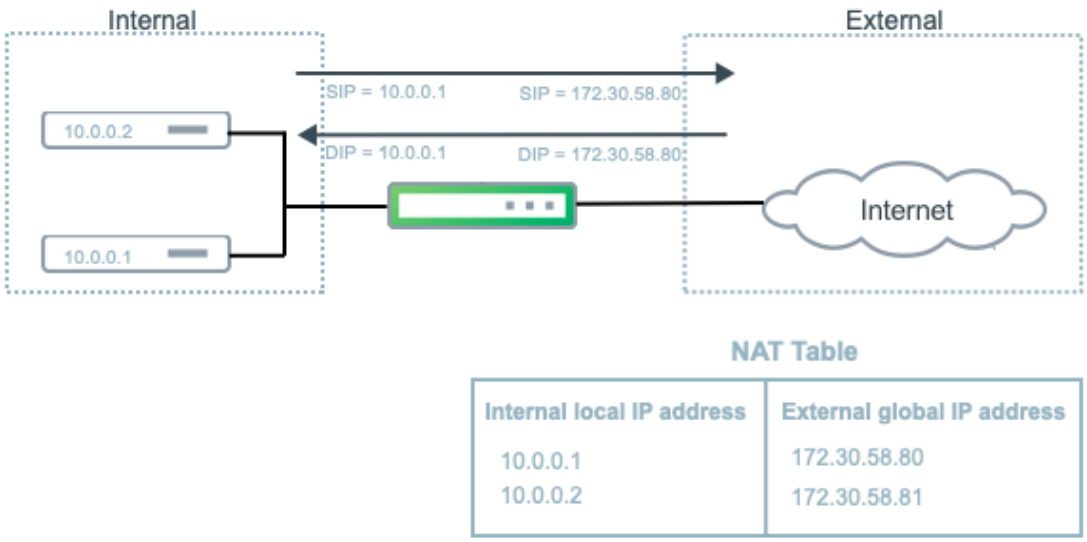
NAT was designed to overcome addressing problems due to the explosive growth of the Internet. In addition to preventing the depletion of IPv4 addresses, NAT enables you to use the private address space internally and still have a way to access the Internet.

Cumulus Linux supports both static NAT and dynamic NAT. Static NAT provides a permanent mapping between one private IP address and a single public address. Dynamic NAT maps private IP addresses to public addresses; these public IP addresses come from a pool. The translations are created as needed dynamically, so that a large number of private addresses can share a smaller pool of public addresses.

Static and dynamic NAT both support:

- Basic NAT, which only translates the IP address in the packet: the source IP address in the outbound direction and the destination IP address in the inbound direction.
- Port Address Translation (PAT), which translates both the IP address and layer 4 port: the source IP address and port in the outbound direction and the destination IP address and port in the inbound direction.

The following illustration shows a basic NAT configuration.



NAT Table

Internal local IP address	External global IP address
10.0.0.1	172.30.58.80
10.0.0.2	172.30.58.81

NOTE

- NAT is supported on physical interfaces and bond interfaces and only in the default VRF.
- IPv6 to IPv4 translation is not supported.
- Multicast traffic is not supported.
- NAT is *not* supported in an EVPN configuration.
- NAT is supported on Broadcom Trident3 X7 and Mellanox Spectrum-2 switches only.

Static NAT

Static NAT provides a one-to-one mapping between a private IP address inside your network and a public IP address. For example, if you have a web server with the private IP address 10.0.0.10 and you want a remote host to be able to make a request to the web server using the IP address 172.30.58.80, you must configure a static NAT mapping between the two IP addresses.

Static NAT entries do not time out from the translation table.

Enable Static NAT

To enable static NAT, edit the `/etc/cumulus/switchd.conf` file and uncomment the `nat.static_enable = TRUE` option:

```
cumulus@switch:~$ sudo nano /etc/cumulus/switchd.conf
...
# NAT configuration
# Enables NAT
nat.static_enable = TRUE
...
```

Then restart `switchd`.

```
cumulus@switch:~$ sudo systemctl restart switchd.service
```

WARNING

Restarting the `switchd` service causes all network ports to reset, interrupting network services, in addition to resetting the switch hardware configuration.

NOTE

Other options in the NAT configuration section of the `switchd.conf` file, such as `nat.age_poll_interval` and `nat.table_size` are dynamic NAT configuration options and are not supported with static NAT.

Configure Static NAT

For static **NAT**, create a rule that matches a source or destination IP address and translates the IP address to a public IP address.

For static **PAT**, create a rule that matches a source or destination IP address

together with the layer 4 port and translates the IP address and port to a public IP address and port.

For Mellanox Spectrum-2 switches, you can include the outgoing or incoming interface.

To create rules, you can use either NCLU or `cl-acltool`.

NCLU Commands | cl-acltool Commands

Use the following NCLU commands:

NAT

```
net add nat static snat|dnat <protocol> <ip-address> [out-
interface|in-interface <interface>] translate <ip-address>
```

PAT

```
net add nat static snat|dnat <protocol> <ip-address> <port>
[out-interface|in-interface <interface>] translate <ip-
address> <port>
```

Where:

- `snat` is the source NAT
- `dnat` is the destination NAT
- `protocol` is TCP, ICMP, or UDP. The protocol is required.
- `out-interface` is the outbound interface for `snat` (Mellanox Spectrum-2 switches only)
- `in-interface` is the inbound interface for `dnat` (Mellanox Spectrum-2 switches only)

Command Examples

The following rule matches TCP packets with source IP address 10.0.0.1 and translates the IP address to 172.20.58.80:

Dynamic NAT

Dynamic NAT maps private IP addresses and ports to a public IP address and port range or a public IP address range and port range. IP addresses are assigned from a pool of addresses dynamically. When entries are released after a period of inactivity, new incoming connections are dynamically mapped to the freed up addresses and ports.

Enable Dynamic NAT

To enable dynamic NAT, edit the `/etc/cumulus/switchd.conf` file and uncomment the `nat.dynamic_enable = TRUE` option:

```
cumulus@switch:~$ sudo nano /etc/cumulus/switchd.conf
...
# NAT configuration
# Enables NAT
nat.dynamic_enable = TRUE
...
```

Then restart `switchd`.

```
cumulus@switch:~$ sudo systemctl restart switchd.service
```

WARNING

Restarting the `switchd` service causes all network ports to reset, interrupting network services, in addition to resetting the switch hardware configuration.

NOTE

For dynamic NAT to work on switches with the Broadcom Trident3 ASIC, you must also enable static NAT. Uncomment the `nat.static_enable = TRUE` option in addition to the `nat.dynamic_enable = TRUE` option.

Optional Dynamic NAT Settings

The `/etc/cumulus/switchd.conf` file includes the following configuration options for dynamic NAT. Only change these options if dynamic NAT is enabled.

Option	Description
nat.age_poll_interval	The period of inactivity before `switchd` releases a NAT entry

Option	Description
	from the translation table. The default value is 5 minutes. The minimum value is 1 minute. The maximum value is 24 hours.
nat.table_size	The maximum number of dynamic `snat` and `dnat` entries in the translation table. The default value is 1024. Trident3 switches support a maximum of 1024 entries. Mellanox Spectrum-2 switches support a maximum of 8192 entries.
nat.config_table_size	The maximum number of rules allowed (NCLU or cl-acltool). The default value is 64. The minimum value is 64. The maximum value is 1024.

After you change any of the dynamic NAT configuration options, restart `switchd`.

```
cumulus@switch:~$ sudo systemctl restart switchd.service
```

WARNING

Restarting the `switchd` service causes all network ports to reset, interrupting network services, in addition to resetting the switch hardware configuration.

Configure Dynamic NAT

For dynamic **NAT**, create a rule that matches a IP address in CIDR notation and translates the address to a public IP address or IP address range.

For dynamic **PAT**, create a rule that matches an IP address in CIDR notation and translates the address to a public IP address and port range or an IP address range and port range. You can also match on an IP address in CIDR notation and port.

For Mellanox Spectrum-2 switches, you can include the outgoing or incoming interface in the rule. See the examples below.

NCLU Commands | cl-acltool Commands

Use the following NCLU commands:

NAT

```
net add nat dynamic snat|dnat <protocol> source-ip
<ipv4-address/prefixlen>|destination-ip <ip-address/
prefixlen> out-interface|in-interface translate
<ipv4-address>|<ip-address-range>
```

PAT

```
net add nat dynamic snat|dnat <protocol> source-ip
<ipv4-address/prefixlen>|destination-ip <ipv4-address/
prefixlen> source-port <port>|destination-port <port> out-
interface|in-interface translate <ipv4-address> <port-
range>|<ipv4-address-range> <port-range>
```

Where:

- `snat` is the source NAT
- `dnat` is the destination NAT
- `protocol` is TCP, ICMP, or UDP. The protocol is required.
- `out-interface` is the outbound interface for `snat` (Mellanox Spectrum-2 switches only)
- `in-interface` is the inbound interface for `dnat` (Mellanox Spectrum-2 switches only)

Example Commands

Show Configured NAT Rules

To see the NAT rules configured on the switch, run the `sudo iptables -t nat -v -L` or the `sudo cl-acltool -L ip -v` command. For example:

```
cumulus@switch:~$ sudo iptables -t nat -v -L -n
...
Chain POSTROUTING (policy ACCEPT 27 packets, 3249 bytes)
 pkts bytes target   prot opt in   out   source      destination
    0     0 SNAT     tcp --  any  any   10.0.0.1
anywhere     to:172.30.58.80
```

Show Conntrack Flows

To see the currently active connection tracking (conntrack) flows, run the `sudo cat /proc/net/nf_conntrack` command. The hardware offloaded flows contain `[OFFLOAD]` in the output.

```
cumulus@switch:~$ sudo cat /proc/net/nf_conntrack
ipv4     2 udp      17 src=172.30.10.5 dst=10.0.0.2 sport=5001
dport=5000 src=10.0.0.2 dst=10.1.0.10 sport=6000 dport=1026
[OFFLOAD] mark=0 zone=0 use=2
```

Bidirectional Forwarding Detection - BFD

Bidirectional Forwarding Detection (BFD) provides low overhead and rapid detection of failures in the paths between two network devices. It provides a unified mechanism for link detection over all media and protocol layers. Use BFD to detect failures for IPv4 and IPv6 single or multihop paths between any two network devices, including unidirectional path failure detection.

> **NOTE**
>
> Cumulus Linux does not support demand mode in BFD.

BFD Multihop Routed Paths

BFD multihop sessions are built over arbitrary paths between two systems, which results in some complexity that does not exist for single hop sessions. Here are some best practices for using multihop paths:

- To avoid **spoofing** with multihop paths, configure the maximum hop count (`max_hop_cnt`*)* for each peer, which limits the number of hops for a BFD session. All BFD packets exceeding the maximum hop count are dropped.
- Because multihop BFD sessions can take arbitrary paths, **demultiplex** the

initial BFD packet based on the source/destination IP address pair. Use FRRouting, which monitors connectivity to the peer, to determine the source/destination IP address pairs.

Cumulus Linux supports multihop BFD sessions for both IPv4 and IPv6 peers.

Configure BFD

You can configure BFD by either using FRRouting (with NCLU or `vtysh` commands) or by specifying the configuration in the PTM `topology.dot` file. However, the topology file has some limitations:

- The topology file supports BFD IPv4 and IPv6 *single* hop sessions only; you *cannot* specify IPv4 or IPv6 *multihop* sessions in the topology file.
- The topology file supports BFD sessions for only link-local IPv6 peers; BFD sessions for global IPv6 peers discovered on the link are not created.

Use FRRouting to register multihop peers with PTM and BFD as well as to monitor the connectivity to the remote BGP multihop peer. FRRouting can dynamically register and unregister both IPv4 and IPv6 peers with BFD when the BFD-enabled peer connectivity is established or de-established. Also, you can configure BFD parameters for each BGP or OSPF peer.

> **NOTE**
>
> The BFD parameter configured in the topology file is given higher precedence over the client-configured BFD parameters for a BFD session that has been created by both the topology file and FRRouting.

> **NOTE**
>
> BFD requires an IP address for any interface on which it is configured. The neighbor IP address for a single hop BFD session must be in the ARP table before BFD can start sending control packets.

When you configure BFD, you can set the following parameters for both IPv4 and IPv6 sessions. If you do not set these parameters, the default values are used.

- The required minimum interval between the received BFD control packets. The default value is 300ms.
- The minimum interval for transmitting BFD control packets. The default value is 300ms.
- The detection time multiplier. The default value is 3.

BFD in BGP

When you configure BFD in BGP, neighbors are registered and deregistered with PTM dynamically.

To configure BFD in BGP, run the following commands.

> **NOTE**
>
> You can configure BFD for a peer group or for an individual neighbor.

NCLU Commands vtysh Commands

The following example configures BFD for swp1 and uses the default intervals.

```
cumulus@switch:~$ net add bgp neighbor swp1 bfd
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The following example configures BFD for the peer group `fabric` and sets the interval multiplier to 4, the minimum interval between received BFD control packets to 400, and the minimum interval for sending BFD control packets to 400.

```
cumulus@switch:~$ net add bgp neighbor fabric bfd 4 400 400
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
router bgp 65000
neighbor fabric bfd 4 400 400
...
```

To see neighbor information in BGP, including BFD status, run the NCLU `net show bgp neighbor <interface>` command or the `vtysh show ip bgp neighbor <interface>` command. For example:

```
cumulus@switch:~$ net show bgp neighbor swp1
...
BFD: Type: single hop
  Detect Mul: 3, Min Rx interval: 300, Min Tx interval: 300
  Status: Down, Last update: 0:00:00:08
...
```

BFD in OSPF

When you enable or disable BFD in OSPF, neighbors are registered and de-registered dynamically with PTM. When BFD is enabled on the interface, a neighbor is registered with BFD when two-way adjacency is established and deregistered when adjacency goes down. The BFD configuration is per interface and any IPv4 and IPv6 neighbors discovered on that interface

inherit the configuration.

To configure BFD in OSPF, run the following commands.

NCLU Commands | vtysh Commands

The following example configures BFD in OSPFv3 for interface swp1 and sets interval multiplier to 4, the minimum interval between *received* BFD control packets to 400, and the minimum interval for *sending* BFD control packets to 400.

```
cumulus@switch:~$ net add interface swp1 ospf6 bfd 4 400 400
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU and `vtysh` commands save the configuration in the `/etc/frr/frr.conf` file. For example:

```
...
interface swp1
  ipv6 ospf6 bfd 4 400 400
  ...
```

You can run different commands to show neighbor information in OSPF,

including BFD status.

- To show IPv6 OSPF interface information, run the NCLU `net show ospf6 interface <interface>` command or the `vtysh` `show ip ospf6 interface <interface>` command.
- To show IPv4 OSPF interface information, run the NCLU `net show ospf interface <interface>` command or the `vtysh` `show ip ospf interface <interface>` command.

  The following example shows IPv6 OSPF interface information.

```
cumulus@switch:~$ net show ospf6 interface swp2s0
  swp2s0 is up, type BROADCAST
Interface ID: 4
Internet Address:
  inet : 11.0.0.21/30
  inet6: fe80::4638:39ff:fe00:6c8e/64
Instance ID 0, Interface MTU 1500 (autodetect: 1500)
MTU mismatch detection: enabled
Area ID 0.0.0.0, Cost 10
State PointToPoint, Transmit Delay 1 sec, Priority 1
Timer intervals configured:
  Hello 10, Dead 40, Retransmit 5
DR: 0.0.0.0 BDR: 0.0.0.0
Number of I/F scoped LSAs is 2
```

```
   0 Pending LSAs for LSUpdate in Time 00:00:00 [thread off]
   0 Pending LSAs for LSAck in Time 00:00:00 [thread off]
BFD: Detect Mul: 3, Min Rx interval: 300, Min Tx interval: 300
```

- To show IPv6 OSPF neighbor details, run the NCLU `net show ospf6 neighbor detail` command or the `vtysh` `show ip ospf6 neighbor detail` command.
- To show IPv4 OSPF interface information, run the NCLU `net show ospf neighbor detail` command or the `vtysh` `show ip ospf neighbor detail` command.

  The following example shows IPv6 OSPF neighbor details.

```
cumulus@switch:~$ net show ospf6 neighbor detail
  Neighbor 0.0.0.4%swp2s0
    Area 0.0.0.0 via interface swp2s0 (ifindex 4)
    His IfIndex: 3 Link-local address: fe80::202:ff:fe00:a
    State Full for a duration of 02:32:33
    His choice of DR/BDR 0.0.0.0/0.0.0.0, Priority 1
    DbDesc status: Slave SeqNum: 0x76000000
    Summary-List: 0 LSAs
    Request-List: 0 LSAs
```

```
    Retrans-List: 0 LSAs
    0 Pending LSAs for DbDesc in Time 00:00:00 [thread off]
    0 Pending LSAs for LSReq in Time 00:00:00 [thread off]
    0 Pending LSAs for LSUpdate in Time 00:00:00 [thread off]
    0 Pending LSAs for LSAck in Time 00:00:00 [thread off]
    BFD: Type: single hop
      Detect Mul: 3, Min Rx interval: 300, Min Tx interval:
300
      Status: Up, Last update: 0:00:00:20
```

Scripts

`ptmd` executes scripts at `/etc/ptm.d/bfd-sess-down` when BFD sessions go down and `/etc/ptm.d/bfd-sess-up` when BFD sessions goes up. Modify these default scripts as needed.

Echo Function

Cumulus Linux supports the *echo function* for IPv4 single hops only, and with the asynchronous operating mode only (Cumulus Linux does not support demand mode).

Use the echo function to test the forwarding path on a remote system. To enable the echo function, set `echoSupport` to *1* in the topology file.

After the echo packets are looped by the remote system, the BFD control packets can be sent at a much lower rate. You configure this lower rate by setting the `slowMinTx` parameter in the topology file to a non-zero value in milliseconds.

You can use more aggressive detection times for echo packets because the round-trip time is reduced; echo packets access the forwarding path. You can configure the detection interval by setting the `echoMinRx` parameter in the topology file. The minimum setting is 50 milliseconds. After configured, BFD control packets are sent out at this required minimum echo Rx interval. This indicates to the peer that the local system can loop back the echo packets. Echo packets are transmitted if the peer supports receiving echo packets.

About the Echo Packet

BFD echo packets are encapsulated into UDP packets over destination and source UDP port number 3785. The BFD echo packet format is vendor-specific and has not been defined in the RFC. BFD echo packets that originate from Cumulus Linux are 8 bytes long and have the following format:

0	1	2	3
Version	Length	Reserved	Reserved
My Discriminator			

Where:

- **Version** is the version of the BFD echo packet.
- **Length** is the length of the BFD echo packet.
- **My Discriminator** is a non-zero value that uniquely identifies a BFD session on the transmitting side. When the originating node receives the packet after being looped back by the receiving system, this value uniquely identifies the BFD session.

Transmit and Receive Echo Packets

BFD echo packets are transmitted for a BFD session only when the peer has advertised a non-zero value for the required minimum echo Rx interval (the `echoMinRx` setting) in the BFD control packet when the BFD session starts. The transmit rate of the echo packets is based on the peer advertised echo receive value in the control packet.

BFD echo packets are looped back to the originating node for a BFD session only if locally the `echoMinRx` and `echoSupport` are configured to a non-zero values.

Echo Function Parameters

You configure the echo function by setting the following parameters in the topology file at the global, template and port level:

- **echoSupport** enables and disables echo mode. Set to 1 to enable the echo function. It defaults to 0 (disable).
- **echoMinRx** is the minimum interval between echo packets the local system is capable of receiving. This is advertised in the BFD control packet. When the echo function is enabled, it defaults to 50. If you

disable the echo function, this parameter is automatically set to 0, which indicates the port or the node cannot process or receive echo packets.

- **slowMinTx** is the minimum interval between transmitting BFD control packets when the echo packets are being exchanged.

Troubleshooting

To troubleshoot BFD, run the NCLU `net show bfd sessions` or `net show bfd sessions detail` command.

```
cumulus@switch:~$ net show bfd sessions detail


-------------------------------------------------------------------------------
port  peer                 state  local  type       diag  det
tx_timeout  rx_timeout
                                                          mult
-------------------------------------------------------------------------------
swp1  fe80::202:ff:fe00:1  Up     N/A    singlehop  N/A   3
300         900
swp1  3101:abc:bcad::2     Up     N/A    singlehop  N/A   3
300         900


#continuation of output
-----------------------------------------------------------------
echo        echo        max      rx_ctrl  tx_ctrl  rx_echo
```

```
tx_echo
tx_timeout   rx_timeout   hop_cnt
-------------------------------------------------------------------------
0            0            N/A       187172    185986    0         0
0            0            N/A       501       533       0         0
```

You can also run the Linux `ptmctl -b` command.

Related Information

- RFC 5880 - Bidirectional Forwarding Detection
- RFC 5881 - BFD for IPv4 and IPv6 (Single Hop)
- RFC 5882 - Generic Application of BFD
- RFC 5883 - Bidirectional Forwarding Detection (BFD) for Multihop Paths

Address Resolution Protocol - ARP

Address Resolution Protocol (ARP) is a communication protocol used for discovering the link layer address, such as a MAC address, associated with a given network layer address. ARP is defined by RFC 826. The Cumulus Linux ARP implementation differs from standard Debian Linux ARP behavior in a few ways because Cumulus Linux is an operating system for routers/switches rather than servers.

Standard Debian ARP Behavior and the Tunable ARP Parameters

Debian has these five tunable ARP parameters:

- `arp_accept`
- `arp_announce`
- `arp_filter`
- `arp_ignore`
- `arp_notify`

These parameters are described in the Linux documentation, but snippets for each parameter description are included in the table below and are highlighted in *italics*.

In a standard Debian installation, all of these ARP parameters are set to *0*, leaving the router as wide open and unrestricted as possible. These settings

are based on the assertion made long ago that Linux IP addresses are a property of the device, not a property of an individual interface. Therefore, an ARP request or reply could be sent on one interface containing an address residing on a different interface. While this unrestricted behavior makes sense for a server, it is not the normal behavior of a router. Routers expect the MAC/IP address mappings supplied by ARP to match the physical topology, with the IP addresses matching the interfaces on which they reside. With these tunable ARP parameters, Cumulus Linux is able to specify the behavior to match the expectations of a router.

ARP Tunable Parameter Settings in Cumulus Linux

The ARP tunable parameters are set to the following values by default in Cumulus Linux.

Parameter	Setting	Type	Description
`arp_accept`	0	BOOL	Define behavior for gratuitous ARP frames whose IP is not already present in the ARP table: • 0: Do not create new entries in the ARP

Parameter	Setting	Type	Description
			table. • 1: Create new entries in the ARP table. Cumulus Linux uses the default `arp_accept` behavior of not creating new entries in the ARP table when a gratuitous ARP is seen on an interface or when an ARP reply packet is received. However, an individual interface can have the arp_accept behavior set differently than the remainder of the switch if

Parameter	Setting	Type	Description
			needed. For information on how to apply this port-specific behavior, see below.
`arp_announce`	2	INT	Define different restriction levels for announcing the local source IP address from IP packets in ARP requests sent on interface: • 0: (default) Use any local address, configured on any interface. • 1: Try to avoid local addresses that are not in the

Parameter	Setting	Type	Description
			target's subnet for this interface. This mode is useful when target hosts reachable via this interface require the source IP address in ARP requests to be part of their logical network configured on the receiving interface. When Cumulus Linux generates the request, it checks all subnets

Parameter	Setting	Type	Description
			that include the target IP and preserves the source address if it is from such a subnet. If there is no such subnet. Cumulus Linux selects the source address according to the rules for level 2. • 2: Always use the best local address for this target. In this mode Cumulus Linux ignores the source address in

Parameter	Setting	Type	Description
			the IP packet and tries to select local address preferred for talks with the target host. Such local address is selected by looking for primary IP addresses on all the subnets on the outgoing interface that include the target IP address. If no suitable local address is found, Cumulus Linux selects the first local

Parameter	Setting	Type	Description
			address on the outgoing interface or on all other interfaces, so that a reply for the request is received no matter the source IP address announced. The default Debian behavior with `arp_announce` set to 0 is to send gratuitous ARPs or ARP requests using any local source IP address, not limiting the IP source of the ARP packet to an address residing on

Parameter	Setting	Type	Description
			the interface used to send the packet. This reflects the historically held view in Linux that IP addresses reside inside the device and are not considered a property of a specific interface. Routers expect a different relationship between the IP address and the physical network. Adjoining routers look for MAC/IP addresses to reach a next hop residing on a

Parameter	Setting	Type	Description
			connecting interface for transiting traffic. By setting the `arp_announce` parameter to 2, Cumulus Linux uses the best local address for each ARP request, preferring primary addresses on the interface used to send the ARP. This most closely matches traditional router ARP request behavior.
`arp_filter`	0	BOOL	• 0: (default) The kernel can respond to ARP requests

Parameter	Setting	Type	Description
			with addresses from other interfaces to increase the chance of successful communication. IP addresses are owned by the complete host on Linux, not by particular interfaces. Only for more complex setups like load balancing, does this behavior cause problems. • 1: Allows you to have multiple

Parameter	Setting	Type	Description
			network interfaces on the same subnet and to have the ARPs for each interface answered based on whether or not the kernel routes a packet from the ARPd IP address out of that interface (therefore you must use source based routing for this to work). In other words, it allows control of

Parameter	Setting	Type	Description
			which cards (usually 1) will respond to an ARP request. `arp_filter` for the interface is enabled if at least one of `conf/{all,interface}/arp_fi` is set to TRUE, it is disabled otherwise. Cumulus Linux uses the default Debian Linux `arp_filter` setting of 0. The `arp_filter` is primarily used when multiple interfaces reside in the same subnet

Parameter	Setting	Type	Description
			and is used to allow or disallow which interfaces respond to ARP requests. For OSPF using IP unnumbered interfaces, many interfaces appear to be in the same subnet, and so actually contain the same address. If multiple interfaces are used between a pair of routers, having `arp_filter` set to 1 causes forwarding to fail. The

Parameter	Setting	Type	Description
			`arp_filter` parameter is set to allow a response on any interface in the subnet, where the `arp_ignore` setting (below) to limit cross-interface ARP behavior.
`arp_ignore`	2	INT	Define different modes for sending replies in response to received ARP requests that resolve local target IP addresses: • 0: (default) Reply for any local target IP address, configured on any

Parameter	Setting	Type	Description
			interface. • 1: Reply only if the target IP address is local address configured on the incoming interface. • 2: Reply only if the target IP address is local address configured on the incoming interface and both with the sender's IP address are part from same subnet on this interface. • 3: Do not reply for

Parameter	Setting	Type	Description
			local addresses configured with scope host, only resolutions for global and link addresses are replied. • 4-7: Reserved. • 8: Do not reply for all local addresses. The maximum value from `conf/{all,interface}/arp_ig` is used when the ARP request is received on the {interface}. The default Debian `arp_ignore` parameter allows the device to

Parameter	Setting	Type	Description
			reply to an ARP request for any IP address on any interface. While this matches the expectation that an IP address belongs to the device, not an interface, it can cause some unexpected and undesirable behavior on a router. For example, if the `arp_ignore` parameter is set to 0 and an ARP request is received on one interface for the IP address

Parameter	Setting	Type	Description
			residing on a different interface, the switch responds with an ARP reply even if the interface of the target address is down. This can cause a loss of traffic due to incorrect understanding about the reachability of next hops, and also makes troubleshooting extremely challenging for some failure conditions. In Cumulus Linux, the `arp_ignore` value is set to 2 so that it

Parameter	Setting	Type	Description
			only replies to ARP requests if the target IP address is a local address and both the sender's and target's IP addresses are part of the same subnet on the incoming interface. This should prevent the creation of stale neighbor entries when a peer device sends an ARP request from a source IP address that is not on the connected subnet. Eventually, the switch sends ARP requests to

Parameter	Setting	Type	Description
			the host in an attempt to keep the entry fresh. If the host responds, the switch now has reachable neighbor entries for hosts that are not on the connected subnet.
`arp_notify`	1	BOOL	Define mode for notification of address and device changes. • 0: (default) Do nothing. • 1: Generate gratuitous arp requests when device is brought up or hardware

Parameter	Setting	Type	Description
			address changes. The default Debian arp_notify setting is to remain silent when an interface is brought up or the hardware address is changed. Since Cumulus Linux often acts as a next-hop for many end hosts, it immediately notifies attached devices when an interface comes up or the address changes. This speeds up convergence on the new information

Parameter	Setting	Type	Description
			and provides the most rapid support for changes.

Change Tunable ARP Parameters

You can change the ARP parameter settings in several places, including:

- `/proc/sys/net/ipv4/conf/all/arp*` (all interfaces)
- `/proc/sys/net/ipv4/conf/default/arp*` (default for future interfaces)
- `/proc/sys/net/ipv4/conf/swp*/arp*` (individual interfaces)

The ARP parameter changes in Cumulus Linux use the *default* file locations.

The *all* and *default* locations sound similar, with the exception of which interfaces are impacted, but they operate in significantly different ways. The all location can **potentially** change the value for **all** interfaces running IP, both now and in the future. The reason for this uncertainty is that the *all* value is applied to each parameter using either *MAX* or *OR* logic between the *all* and any *port-specific* settings, as the following table shows:

ARP Parameter	Condition
arp_accept	OR
arp_announce	MAX
arp_filter	OR

ARP Parameter	Condition
arp_ignore	MAX
arp_notify	MAX

For example, if you set the `/proc/sys/net/conf/all/arp_ignore` value to *1* and the `/proc/sys/net/conf/swp1/arp_ignore` value to *0* to try to disable it on a per-port basis, interface swp1 still uses the value of *1*; the port-specific setting does not override the global *all* setting. Instead, the MAX value between the *all* value and port-specific value defines the actual behavior.

The *default* location `/proc/sys/net/ipv4/conf/default/arp*` defines the values for all future IP interfaces. Changing the *default* setting of an ARP parameter does not impact interfaces that already contain an IP address. If changes are being made to a running system that already has IP addresses assigned to it, port-specific settings should be used instead.

NOTE

In Cumulus Linux, the value of the *default* parameter is copied to every port-specific location, excluding those that already have an IP address assigned. Therefore, there is no complicated logic between the *default* setting and the *port-specific* setting like there is when using the *all* location. This makes the application of particular port-specific policies much simpler and more deterministic.

To determine the current ARP parameter settings for each of the locations, run the following commands:

```
cumulus@switch:~$ sudo grep . /proc/sys/net/ipv4/conf/all/arp*
/proc/sys/net/ipv4/conf/all/arp_accept:0
/proc/sys/net/ipv4/conf/all/arp_announce:0
/proc/sys/net/ipv4/conf/all/arp_filter:0
/proc/sys/net/ipv4/conf/all/arp_ignore:0
/proc/sys/net/ipv4/conf/all/arp_notify:0
```

```
cumulus@switch:~$ sudo grep . /proc/sys/net/ipv4/conf/default/
arp*
/proc/sys/net/ipv4/conf/default/arp_accept:0
/proc/sys/net/ipv4/conf/default/arp_announce:2
/proc/sys/net/ipv4/conf/default/arp_filter:0
/proc/sys/net/ipv4/conf/default/arp_ignore:2
/proc/sys/net/ipv4/conf/default/arp_notify:1
```

```
cumulus@switch:~$ sudo grep . /proc/sys/net/ipv4/conf/swp1/arp*
/proc/sys/net/ipv4/conf/swp1/arp_accept:0
/proc/sys/net/ipv4/conf/swp1/arp_announce:2
/proc/sys/net/ipv4/conf/swp1/arp_filter:0
```

```
/proc/sys/net/ipv4/conf/swp1/arp_ignore:2
/proc/sys/net/ipv4/conf/swp1/arp_notify:1
cumulus@switch:~$
```

Cumulus Linux implements this change at boot time using the `arp.conf` file in the following location:

```
cumulus@switch:~$ cat /etc/sysctl.d/arp.conf
net.ipv4.conf.default.arp_announce = 2
net.ipv4.conf.default.arp_notify = 1
net.ipv4.conf.default.arp_ignore=1
cumulus@switch:~$
```

Change Port-specific ARP Parameters

To configure port-specific ARP parameters in a running device, run the following command:

```
cumulus@switch:~$ sudo sh -c "echo 0 > /proc/sys/net/ipv4/conf/
swp1/arp_ignore"
cumulus@switch:~$ sudo grep . /proc/sys/net/ipv4/conf/swp1/arp*
```

```
/proc/sys/net/ipv4/conf/swp1/arp_accept:0
/proc/sys/net/ipv4/conf/swp1/arp_announce:2
/proc/sys/net/ipv4/conf/swp1/arp_filter:0
/proc/sys/net/ipv4/conf/swp1/arp_ignore:0
/proc/sys/net/ipv4/conf/swp1/arp_notify:1
cumulus@switch:~$
```

To make the change persist through reboots, edit the `/etc/sysctl.d/arp.conf` file and add your port-specific ARP setting.

Configure Proxy ARP

When you enable proxy ARP, if the switch receives an ARP request for which it has a route to the destination IP address, the switch sends a proxy ARP reply that contains its own MAC address. The host that sent the ARP request then sends its packets to the switch and the switch forwards the packets to the intended host.

> **NOTE**
>
> Proxy ARP works with IPv4 only; ARP is an IPv4-only protocol.

To enable proxy ARP:

NCLU Commands | Linux Commands

The following example commands enable proxy ARP on swp1.

```
cumulus@switch:~$ net add interface swp1 post-up "echo 1 >
/proc/sys/net/ipv4/conf/swp1/proxy_arp"
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

If you are running two interfaces in the same broadcast domain (typically seen when using VRR, which creates a `-v0` interface in the same broadcast domain), set `/proc/sys/net/ipv4/conf/<INTERFACE>/medium_id` to *2* on both the interface and the -v0 interface so that both interfaces do not respond with proxy ARP replies.

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add interface swp1 post-up "echo 2 >
/proc/sys/net/ipv4/conf/swp1/medium_id"
cumulus@switch:~$ net add interface swp1-v0 post-up "echo 1
> /proc/sys/net/ipv4/conf/swp1-v0/proxy_arp"
cumulus@switch:~$ net add interface swp1-v0 post-up "echo 2
> /proc/sys/net/ipv4/conf/swp1-v0/medium_id"
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

If you are running proxy ARP on a VRR interface, add a post-up line to the VRR interface stanza similar to the following. For example, if vlan100 is the VRR interface for the configuration above:

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add vlan 100 post-up "echo 1 > /proc/
sys/net/ipv4/conf/swp1/proxy_arp"
cumulus@switch:~$ net add vlan 100 post-up "echo 1 > /proc/
sys/net/ipv4/conf/swp1-v0/proxy_arp"
cumulus@switch:~$ net add vlan 100 post-up "echo 2 > /proc/
sys/net/ipv4/conf/swp1/medium_id"
cumulus@switch:~$ net add vlan 100 post-up "echo 2 > /proc/
sys/net/ipv4/conf/swp1-v0/medium_id"
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Duplicate Address Detection (Windows Hosts)

In centralized VXLAN environments, where ARP/ND suppression is enabled and SVIs exist on the leaf switches but are not assigned an address within the subnet, problems with the Duplicate Address Detection process on Microsoft Windows hosts can occur. For example, in a pure layer 2 scenario or with SVIs that have the `ip-forward` option set to off, the IP address is not assigned to the SVI. The `neighmgrd` service selects a source IP address for an ARP probe based on the subnet match on the neighbor IP address. Because the SVI on which this neighbor is learned does not contiain an IP address, the subnet match fails. This results in `neighmgrd` using UNSPEC (0.0.0.0 for IPv4) as the source IP address in the ARP probe.

To work around this issue, run the `neighmgrctl setsrcipv4 <ipaddress>` command to specify a non-0.0.0.0 address for the source; for example:

```
cumulus@switch:~$ neighmgrctl setsrcipv4 10.1.0.2
```

The configuration above takes effect immediately but does not persist if you reboot the switch. To make the changes apply persistently:

1. Create a new file called `/etc/cumulus/neighmgr.conf` and add the `setsrcipv4 <ipaddress>` option; for example:

```
cumulus@switch:~$  sudo nano /etc/cumulus/neighmgr.conf


[main]
setsrcipv4: 10.1.0.2
```

2. Restart the `neighmgrd` service:

```
cumulus@switch:~$ sudo systemctl restart neighmgrd
```

Monitoring and Troubleshooting

This chapter introduces the basics for monitoring and troubleshooting Cumulus Linux.

Serial Console

The serial console is a useful tool for debugging issues, especially when you find yourself rebooting the switch often or if you do not have a reliable network connection.

The default serial console baud rate is 115200, which is the baud rate ONIE uses.

Configure the Serial Console on ARM Switches

On ARM switches, the U-Boot environment variable `baudrate` identifies the baud rate of the serial console. To change the `baudrate` variable, use the `fw_setenv` command:

```
cumulus@switch:~$ sudo fw_setenv baudrate 9600
Updating environment variable: `baudrate'
Proceed with update [N/y]? y
```

You must reboot the switch for the `baudrate` change to take effect.

The valid values for `baudrate` are:

- 300
- 600
- 1200
- 2400
- 4800
- 9600
- 19200
- 38400
- 115200

Configure the Serial Console on x86 Switches

On x86 switches, you configure serial console baud rate by editing `grub`.

WARNING

Incorrect configuration settings in `grub` can cause the switch to be inaccessible via the console. Review `grub` changes carefully before you implement them.

The valid values for the baud rate are:

- 300
- 600
- 1200
- 2400
- 4800
- 9600
- 19200
- 38400
- 115200

To change the serial console baud rate:

1. Edit the `/etc/default/grub` file. The two relevant lines in `/etc/default/grub` are as follows; replace the *115200* value with a valid value specified above in the `--speed` variable in the first line and in the `console` variable in the second line:

```
GRUB_SERIAL_COMMAND="serial --port=0x2f8 --speed=115200 --
word=8 --parity=no --stop=1"
GRUB_CMDLINE_LINUX="console=ttyS1,115200n8
cl_platform=accton_as5712_54x"
```

2. After you save your changes to the grub configuration, type the following at the command prompt:

```
cumulus@switch:~$ update-grub
```

3. If you plan on accessing the switch BIOS over the serial console, you need to update the baud rate in the switch BIOS. For more information, see this knowledge base article.

4. Reboot the switch.

Change the Console Log Level

By default, the console prints all log messages except debug messages. To tune console logging to be less verbose so that certain levels of messages are not printed, run the `dmesg -n <level>` command, where the log levels are:

Level	Description
0	Emergency messages (the system is about to crash or is unstable).
1	Serious conditions; you must take action immediately.
2	Critical conditions (serious hardware or software failures).
3	Error conditions (often used by drivers to indicate difficulties with the hardware).

Level	Description
4	Warning messages (nothing serious but might indicate problems).
5	Message notifications for many conditions, including security events.
6	Informational messages.
7	Debug messages.

Only messages with a value lower than the level specified are printed to the console. For example, if you specify level **3**, only level 2 (critical conditions), level 1 (serious conditions), and level 0 (emergency messages) are printed to the console:

```
cumulus@switch:~$ sudo dmesg -n 3
```

Alternatively, you can run `dmesg --console-level <level>` command, where the log levels are `emerg`, `alert`, `crit`, `err`, `warn`, `notice`, `info`, or `debug`. For example, to print critical conditions, run the following command:

```
cumulus@switch:~$ sudo dmesg --console-level crit
```

The `dmesg` command is applied until the next reboot.

For more details about the `dmesg` command, run `man dmesg`.

Show General System Information

Two commands are helpful for getting general information about the switch and the version of Cumulus Linux you are running. These are helpful with system diagnostics and if you need to submit a support request.

For information about the version of Cumulus Linux running on the switch, run the `net show version`,command which displays the contents of `/etc/lsb-release`:

```
cumulus@switch:~$ net show version
NCLU_VERSION=1.0-cl4u1
DISTRIB_ID="Cumulus Linux"
DISTRIB_RELEASE=4.1.0
DISTRIB_DESCRIPTION="Cumulus Linux 4.1.0"
```

For general information about the switch, run `net show system`, which gathers information about the switch from a number of files in the system:

```
cumulus@switch:~$ net show system
```

```
Hostname......... celRED


Build............ Cumulus Linux 4.1.0
Uptime........... 8 days, 12:24:01.770000


Model............ Cel REDSTONE
CPU.............. x86_64 Intel Atom C2538 2.4 GHz
Memory........... 4GB
Disk............. 14.9GB
ASIC............. Broadcom Trident2 BCM56854
Ports............ 48 x 10G-SFP+ & 6 x 40G-QSFP+
Base MAC Address. a0:00:00:00:00:50
```

Diagnostics Using cl-support

You can use `cl-support` to generate a single export file that contains various details and the configuration from a switch. This is useful for remote debugging and troubleshooting. For more information about `cl-support`, read Understanding the cl-support Output File.

Run `cl-support` before you submit a support request as this file helps in the investigation of issues.

```
cumulus@switch:~$ sudo cl-support -h
Usage: [-h (help)] [-cDjlMsv] [-d m1,m2,...] [-e m1,m2,...]
  [-p prefix] [-r reason] [-S dir] [-T Timeout_seconds] [-t tag]
  -h: Display this help message
  -c: Run only modules matching any core files, if no -e modules
  -D: Display debugging information
  -d: Disable (do not run) modules in this comma separated list
  -e: Enable (only run) modules in this comma separated list; "-
e all" runs
      all modules and sub-modules, including all optional
modules
...
```

Send Log Files to a syslog Server

You can configure the remote syslog server on the switch using the following configuration:

```
cumulus@switch:~$ net add syslog host ipv4 192.168.0.254 port
udp 514
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

This creates a file called `/etc/rsyslog.d/11-remotesyslog.conf` in the `rsyslog` directory. The file has the following content:

```
cumulus@switch:~$ cat /etc/rsyslog.d/11-remotesyslog.conf
# This file was automatically generated by NCLU.
*.*   @192.168.0.254:514   # UDP
```

> **NOTE**
>
> NCLU cannot configure a remote syslog if management VRF is enabled on the switch. Refer to Monitoring and Troubleshooting below.

Log Technical Details

Logging on Cumulus Linux is done with rsyslog. `rsyslog` provides both local logging to the `syslog` file as well as the ability to export logs to an external `syslog` server. High precision timestamps are enabled for all `rsyslog` log files; for example:

```
2015-08-14T18:21:43.337804+00:00 cumulus switchd[3629]:
switchd.c:1409 switchd version 1.0-cl2.5+5
```

There are applications in Cumulus Linux that can write directly to a log file without going through `rsyslog`. These files are typically located in `/var/log/`.

> **NOTE**
>
> All Cumulus Linux rules are stored in separate files in `/etc/rsyslog.d/`, which are called at the end of the `GLOBAL DIRECTIVES` section of `/etc/rsyslog.conf`. As a result, the `RULES` section at the end of `rsyslog.conf` is ignored because the messages have to be processed by the rules in `/etc/rsyslog.d` and then dropped by the last line in `/etc/rsyslog.d/99-syslog.conf`.

Local Logging

Most logs within Cumulus Linux are sent through `rsyslog`, which writes them to files in the `/var log` directory. There are default rules in the `/etc/rsyslog.d/` directory that define where the logs are written:

Rule	Purpose
10-rules.conf	Sets defaults for log messages, include log format and log rate limits.
15-crit.conf	Logs `crit`, `alert` or `emerg` log messages to `/var/log/crit.log` to ensure they are not

Rule	Purpose
	rotated away rapidly.
20-clagd.conf	Logs `clagd` messages to `/var/log/clagd.log` for MLAG.
22-linkstate.conf	Logs link state changes for all physical and logical network links to `/var/log/linkstate`.
25-switchd.conf	Logs `switchd` messages to `/var/log/switchd.log`.
30-ptmd.conf	Logs `ptmd` messages to `/var/log/ptmd.log` for Prescription Topology Manager.
35-rdnbrd.conf	Logs `rdnbrd` messages to `/var/log/rdnbrd.log` for Redistribute Neighbor.
40-netd.conf	Logs `netd` messages to `/var/log/netd.log` for NCLU.
45-frr.conf	Logs routing protocol messages to `/var/log/frr/frr.log`. This includes BGP and OSPF log messages.
99-syslog.conf	All remaining processes that use `rsyslog` are sent to `/var/log/syslog`.

Log files that are rotated are compressed into an archive. Processes that do not use `rsyslog` write to their own log files within the `/var/log` directory.

For more information on specific log files, see Troubleshooting Log Files.

Enable Remote syslog

By default, not all log messages are sent to a remote server. To send other log files (such as `switchd` logs) to a `syslog` server, follow these steps:

1. Create a file in `/etc/rsyslog.d/`. Make sure the filename starts with a number lower than 99 so that it executes before log messages are dropped in, such as `20-clagd.conf` or `25-switchd.conf`. The example file below is called `/etc/rsyslog.d/11-remotesyslog.conf`. Add content similar to the following:

```
## Logging switchd messages to remote syslog server


@192.168.1.2:514
```

   This configuration sends log messages to a remote `syslog` server for the following processes: `clagd`, `switchd`, `ptmd`, `rdnbrd`, `netd` and `syslog`. It follows the same syntax as the `/var/log/syslog` file, where `@` indicates UDP, *192.168.12* is the IP address of the `syslog` server, and *514* is the UDP port.

> **NOTE**

- For TCP-based syslog, use two @@ before the IP address *@@192.168.1.2:514*.

- Running `syslog` over TCP places a burden on the switch to queue packets in the `syslog` buffer. This may cause detrimental effects if the remote `syslog` server becomes unavailable.

- The numbering of the files in `/etc/rsyslog.d/` dictates how the rules are installed into `rsyslog.d`. Lower numbered rules are processed first, and `rsyslog` processing *terminates* with the `stop` keyword. For example, the `rsyslog` configuration for FRR is stored in the `45-frr.conf` file with an explicit `stop` at the bottom of the file. FRR messages are logged to the `/var/log/frr/frr.log` file on the local disk only (these messages are not sent to a remote server using the default configuration). To log FRR messages remotely in addition to writing FRR messages to the local disk, rename the `99-syslog.conf` file to `11-remotesyslog.conf`. FRR messages are first processed by the `11-remotesyslog.conf` rule (transmit to remote server), then continue to be processed by the

`45-frr.conf` file (write to local disk in the `/var/log/frr/frr.log` file).

- Do not use the `imfile` module with any file written by `rsyslogd`.

2. Restart `rsyslog`.

```
cumulus@switch:~$ sudo systemctl restart rsyslog.service
```

Write to syslog with Management VRF Enabled

You can write to syslog with management VRF enabled by applying the following configuration; this configuration is commented out in the `/etc/rsyslog.d/11-remotesyslog.conf` file:

```
cumulus@switch:~$ cat /etc/rsyslog.d/11-remotesyslog.conf
## Copy all messages to the remote syslog server at
192.168.0.254 port 514
action(type="omfwd" Target="192.168.0.254" Device="mgmt"
```

```
Port="514" Protocol="udp")
```

For each syslog server, configure a unique `action` line. For example, to configure two syslog servers at 192.168.0.254 and 10.0.0.1:

```
cumulus@switch:~$ cat /etc/rsyslog.d/11-remotesyslog.conf
## Copy all messages to the remote syslog servers at
192.168.0.254 and 10.0.0.1 port 514
action(type="omfwd" Target="192.168.0.254" Device="mgmt"
Port="514" Protocol="udp")
action(type="omfwd" Target="10.0.0.1" Device="mgmt" Port="514"
Protocol="udp")
```

Rate-limit syslog Messages

If you want to limit the number of `syslog` messages that can be written to the `syslog` file from individual processes, add the following configuration to the `/etc/rsyslog.conf` file. Adjust the interval and burst values to rate-limit messages to the appropriate levels required by your environment. For more information, read the rsyslog documentation.

```
module(load="imuxsock"
       SysSock.RateLimit.Interval="2"
SysSock.RateLimit.Burst="50")
```

The following test script shows an example of rate-limit outputvin Cumulus Linux

▼ Example test script

Harmless syslog Error: Failed to reset devices.list

The following message is logged to `/var/log/syslog` when you run `systemctl daemon-reload` and during system boot:

```
systemd[1]: Failed to reset devices.list on /system.slice:
Invalid argument
```

This message is harmless, and can be ignored. It is logged when `systemd` attempts to change group attributes that are read only. The upstream version of `systemd` has been modified to not log this message by default.

The `systemctl daemon-reload` command is often issued when Debian packages are installed, so the message may be seen multiple times when upgrading packages.

Syslog Troubleshooting Tips

You can use the following commands to troubleshoot `syslog` issues.

Verifying that rsyslog is Running

To verify that the `rsyslog` service is running, use the `sudo systemctl status rsyslog.service` command:

```
cumulus@leaf01:mgmt-vrf:~$ sudo systemctl status rsyslog.service
rsyslog.service - System Logging Service
  Loaded: loaded (/lib/systemd/system/rsyslog.service; enabled)
  Active: active (running) since Sat 2017-12-09 00:48:58 UTC;
7min ago
    Docs: man:rsyslogd(8)
          http://www.rsyslog.com/doc/
Main PID: 11751 (rsyslogd)
  CGroup: /system.slice/rsyslog.service
          └─11751 /usr/sbin/rsyslogd -n

Dec 09 00:48:58 leaf01 systemd[1]: Started System Logging
Service.
```

Verify your rsyslog Configuration

After making manual changes to any files in the `/etc/rsyslog.d` directory,

use the `sudo rsyslogd -N1` command to identify any errors in the configuration files that might prevent the `rsyslog` service from starting.

In the following example, a closing parenthesis is missing in the `11-remotesyslog.conf` file, which is used to configure `syslog` for management VRF:

```
cumulus@leaf01:mgmt-vrf:~$ cat /etc/rsyslog.d/
11-remotesyslog.conf
action(type="omfwd" Target="192.168.0.254" Device="mgmt"
Port="514" Protocol="udp"

cumulus@leaf01:mgmt-vrf:~$ sudo rsyslogd -N1
rsyslogd: version 8.4.2, config validation run (level 1),
master config /etc/rsyslog.conf
syslogd: error during parsing file /etc/rsyslog.d/15-crit.conf,
on or before line 3: invalid character '$' in object definition
- is there an invalid escape sequence somewhere? [try http:
/www.rsyslog.com/e/2207 ]
rsyslogd: error during parsing file /etc/rsyslog.d/
15-crit.conf, on or before line 3: syntax error on token
'crit_log' [try http://www.rsyslog.com/e/2207 ]
```

After correcting the invalid syntax, issuing the `sudo rsyslogd -N1` command produces the following output.

```
cumulus@leaf01:mgmt-vrf:~$ cat /etc/rsyslog.d/
11-remotesyslog.conf
action(type="omfwd" Target="192.168.0.254" Device="mgmt"
Port="514" Protocol="udp")
cumulus@leaf01:mgmt-vrf:~$ sudo rsyslogd -N1
rsyslogd: version 8.4.2, config validation run (level 1),
master config /etc/rsyslog.conf
rsyslogd: End of config validation run. Bye.
```

tcpdump

If a syslog server is not accessible to validate that `syslog` messages are being exported, you can use `tcpdump`.

In the following example, a syslog server has been configured at 192.168.0.254 for UDP syslogs on port 514:

```
cumulus@leaf01:mgmt-vrf:~$ sudo tcpdump -i eth0 host
192.168.0.254 and udp port 514
```

A simple way to generate `syslog` messages is to use `sudo` in another session, such as `sudo date`. Using `sudo` generates an `authpriv` log.

```
cumulus@leaf01:mgmt-vrf:~$ sudo tcpdump -i eth0 host
192.168.0.254 and udp port 514
tcpdump: verbose output suppressed, use -v or -vv for full
protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size
262144 bytes
00:57:15.356836 IP leaf01.lab.local.33875 >
192.168.0.254.syslog: SYSLOG authpriv.notice, length: 105
00:57:15.364346 IP leaf01.lab.local.33875 >
192.168.0.254.syslog: SYSLOG authpriv.info, length: 103
00:57:15.369476 IP leaf01.lab.local.33875 >
192.168.0.254.syslog: SYSLOG authpriv.info, length: 85
```

To see the contents of the `syslog` file, use the `tcpdump -X` option:

```
cumulus@leaf01:mgmt-vrf:~$ sudo tcpdump -i eth0 host
192.168.0.254 and udp port 514 -X -c 3
tcpdump: verbose output suppressed, use -v or -vv for full
protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size
262144 bytes
00:59:15.980048 IP leaf01.lab.local.33875 >
192.168.0.254.syslog: SYSLOG authpriv.notice, length: 105
```

```
0x0000: 4500 0085 33ee 4000 4011 8420 c0a8 000b E...3.@.@.......
0x0010: c0a8 00fe 8453 0202 0071 9d18 3c38 353e .....S...q..<85>
0x0020: 4465 6320 2039 2030 303a 3539 3a31 3520 Dec..9.00:59:15.
0x0030: 6c65 6166 3031 2073 7564 6f3a 2020 6375 leaf01.sudo:..cu
0x0040: 6d75 6c75 7320 3a20 5454 593d 7074 732f mulus.:.TTY=pts/
0x0050: 3120 3b20 5057 443d 2f68 6f6d 652f 6375 1.;.PWD=/home/cu
0x0060: 6d75 6c75 7320 3b20 5553 4552 3d72 6f6f mulus.;.USER=roo
0x0070: 7420 3b20 434f 4d4d 414e 443d 2f62 696e t.;.COMMAND=/bin
0x0080: 2f64 6174 65 /date
```

Monitoring System Hardware

You monitor system hardware using the following commands and utilities:

- `decode-syseeprom`
- `smond`
- `sensors`
- Net-SNMP
- watchdog

Retrieve Hardware Information Using decode-syseeprom

The `decode-syseeprom` command enables you to retrieve information about the switch's EEPROM. If the EEPROM is writable, you can set values on the EEPROM.

The following is an example. The command output is different on different switches:

```
cumulus@switch:~$ decode-syseeprom
TlvInfo Header:
    Id String:    TlvInfo
    Version:      1
    Total Length: 114
TLV Name             Code Len Value
```

```
-------------------- ---- --- -----
Product Name         0x21   4 4804
Part Number          0x22  14 R0596-F0009-00
Device Version       0x26   1 2
Serial Number        0x23  19 D1012023918PE000012
Manufacture Date     0x25  19 10/09/2013 20:39:02
Base MAC Address     0x24   6 00:E0:EC:25:7B:D0
MAC Addresses        0x2A   2 53
Vendor Name          0x2D  17 Penguin Computing
Label Revision       0x27   4 4804
Manufacture Country  0x2C   2 CN
CRC-32               0xFE   4 0x96543BC5
(checksum valid)
```

IMPORTANT

Edgecore AS5712-54X, AS5812-54T, AS5812-54X, AS6712-32X and AS6812-32X switches support a second source power supply. This second source device presents at a different I2C address than the primary. As a result, whenever `decode-syseeprom` attempts to read the EEPROM on the PSUs in these systems, both addresses are checked. When the driver reads the location that is not populated,

a warning message like the following is logged:

```
Oct 18 09:54:41 lfc-1ao15 decode-syseeprom: Unable to
find eeprom at /sys/bus/i2c/devices/11-0050/eeprom for
psu2
```

This is expected behavior on these platforms.

Command Options

Usage: `/usr/cumulus/bin/decode-syseeprom [-a][-r][-s [args]][-t <target>][-e][-m]`

Option	Description
`-h`, `-help`	Displays the help message and exits.
`-a`	Prints the base MAC address for switch interfaces.
`-r`	Prints the number of MACs allocated for switch interfaces.

Option	Description
`-s`	Sets the EEPROM content if the EEPROM is writable. args can be supplied in the command line in a comma separated list of the form `<field>=<value>`. `.` `,` and `=` are illegal characters in field names and values. Fields that are not specified default to their current values. If args are supplied in the command line, they will be written without confirmation. If args is empty, the values will be prompted interactively.
`-j`, `--json`	Displays JSON output.
`-t <target>`	Prints the target EEPROM (board, psu2, psu1) information. **Note**: Some systems that use a BMC to manage sensors (such as the Dell Z9264 and EdgeCore Minipack AS8000) do not provide the PSU EEPROM contents. This is because the BMC connects to the PSUs via I2C and the main CPU of the switch has no direct access.
`--serial`, `-e`	Prints the device serial number.
`-m`	Prints the base MAC address for

Option	Description
	management interfaces.
`--init`	Clears and initializes the board EEPROM cache.

Related Commands

You can also use the `dmidecode` command to retrieve hardware configuration information that is populated in the BIOS.

You can use `apt-get` to install the `lshw` program on the switch, which also retrieves hardware configuration information.

Monitor System Units Using smond

The `smond` daemon monitors system units like power supply and fan, updates their corresponding LEDs, and logs the change in the state. Changes in system unit state are detected via the `cpld` registers. `smond` utilizes these registers to read all sources, which impacts the health of the system unit, determines the unit's health, and updates the system LEDs.

Use `smonctl` to display sensor information for the various system units:

```
cumulus@switch:~$ sudo smonctl
```

```
Board                                                :  OK
Fan                                                  :  OK
PSU1                                                 :  OK
PSU2                                                 :  BAD
Temp1     (Networking ASIC Die Temp Sensor          ):  OK
Temp10    (Right side of the board                  ):  OK
Temp2     (Near the CPU (Right)                     ):  OK
Temp3     (Top right corner                         ):  OK
Temp4     (Right side of Networking ASIC            ):  OK
Temp5     (Middle of the board                      ):  OK
Temp6     (P2020 CPU die sensor                     ):  OK
Temp7     (Left side of the board                   ):  OK
Temp8     (Left side of the board                   ):  OK
Temp9     (Right side of the board                  ):  OK
```

NOTE

When the switch is not powered on, `smonctl` shows the PSU status as *BAD* instead of *POWERED OFF* or *NOT DETECTED*. This is a known limitation.

NOTE

On the Dell S4148 switch, `smonctl` shows PSU1 and PSU2; however in the sensors output, both PSUs are listed as PSU1.

NOTE

Some switch models lack the sensor for reading voltage information, so this data is not output from the `smonctl` command. For example, the Dell S4048 series has this sensor and displays power and voltage information:

```
cumulus@dell-s4048-ON:~$ sudo smonctl -v -s PSU2
PSU2:  OK
power:8.5 W   (voltages = ['11.98', '11.87'] V currents
= ['0.72'] A)
```

The Penguin Arctica 3200c does not have this sensor:

```
cumulus@cel-sea:~/tmp$ sudo smonctl -v -s PSU1
PSU1:  OK
```

The following table shows the `smonctl` command options.

Usage: `smonctl [OPTION]... [CHIP]...`

Option	Description
`-s <sensor>`, `--sensor <sensor>`	Displays data for the specified sensor.
`-v`, `--verbose`	Displays detailed hardware sensors data.

For more information, read `man smond` and `man smonctl`.

You can also run these NCLU commands to show sensor information: `net show system sensors`, `net show system sensors detail`, and `net show system sensors json`.

Monitor Hardware Using sensors

Use the `sensors` command to monitor the health of your switch hardware, such as power, temperature and fan speeds. This command executes `lm-sensors`.

> ⓘ **NOTE**

Even though you can use the `sensors` command to monitor the health of your switch hardware, the `smond` daemon is the recommended method for monitoring hardware health. See Monitor System Units Using smond above.

For example:

```
cumulus@switch:~$ sensors
tmp75-i2c-6-48
Adapter: i2c-1-mux (chan_id 0)
temp1:        +39.0 C  (high = +75.0 C, hyst = +25.0 C)

tmp75-i2c-6-49
Adapter: i2c-1-mux (chan_id 0)
temp1:        +35.5 C  (high = +75.0 C, hyst = +25.0 C)

ltc4215-i2c-7-40
Adapter: i2c-1-mux (chan_id 1)
in1:         +11.87 V
in2:         +11.98 V
power1:       12.98 W
curr1:        +1.09 A
```

```
max6651-i2c-8-48
Adapter: i2c-1-mux (chan_id 2)
fan1:        13320 RPM  (div = 1)
fan2:        13560 RPM
```

NOTE

- Output from the `sensors` command varies depending upon the switch hardware you use, as each platform ships with a different type and number of sensors.
- On a Mellanox switch, if only one PSU is plugged in, the fan is at maximum speed.

The following table shows the `sensors` command options.

Usage: `sensors [OPTION]... [CHIP]...`

Option	Description
`-c`, `--config-file`	Specify a config file; use `-` after `-c` to read the config file from

Option	Description
	`stdin`; by default, `sensors` references the configuration file in `/etc/sensors.d/`.
`-s`, `--set`	Executes set statements in the config file (root only); `sensors -s` is run once at boot time and applies all the settings to the boot drivers.
`-f`, `--fahrenheit`	Show temperatures in degrees Fahrenheit.
`-A`, `--no-adapter`	Do not show the adapter for each chip.
`--bus-list`	Generate bus statements for `sensors.conf`.

If `[CHIP]` is not specified in the command, all chip information is printed.

Example chip names include:

- lm78-i2c-0-2d *-i2c-0-2d
- lm78-i2c-0-* *-i2c-0-*
- lm78-i2c-*-2d *-i2c-*-2d
- lm78-i2c-*-* *-i2c-*-*
- lm78-isa-0290 *-isa-0290
- lm78-isa-* *-isa-*
- lm78-*

Monitor Switch Hardware Using SNMP

The Net-SNMP documentation is discussed here.

Keep the Switch Alive Using the Hardware Watchdog

Cumulus Linux includes a simplified version of the `wd_keepalive(8)` daemon from the standard `watchdog` Debian package. `wd_keepalive` writes to a file called `/dev/watchdog` periodically to keep the switch from resetting, at least once per minute. Each write delays the reboot time by another minute. After one minute of inactivity where `wd_keepalive` doesn't write to `/dev/watchdog`, the switch resets itself.

The watchdog is enabled by default on all supported switches, and starts when you boot the switch, before `switchd` starts.

To disable the watchdog, disable and stop the `wd_keepalive` service:

```
cumulus@switch:~$ sudo systemctl disable wd_keepalive ;
systemctl stop wd_keepalive
```

You can modify the settings for the watchdog — like the timeout setting and scheduler priority — in the configuration file, `/etc/watchdog.conf`. Here is the default configuration file:

```
cumulus@switch:~$ cat /etc/watchdog.conf

watchdog-device         = /dev/watchdog

# Set the hardware watchdog timeout in seconds
watchdog-timeout = 30

# Kick the hardware watchdog every 'interval' seconds
interval = 5

# Log a status message every (interval * logtick) seconds.
Requires
# --verbose option to enable.
logtick = 240

# Run the daemon using default scheduler SCHED_OTHER with
slightly
# elevated process priority.  See man setpriority(2).
realtime = no
priority = -2
```

Related Information

- packages.debian.org/search?keywords=lshw
- lm-sensors.org

- Net-SNMP tutorials

Network Switch Port LED and Status LED Guidelines

Data centers today have a large number of network switches manufactured by different hardware vendors running network operating systems (NOS) from different providers. This chapter provides a set of guidelines for how network port and status LEDs should appear on the front panel of a network switch. This provides a network operator with a standard way to identify the state of a switch and its ports by looking at its front panel, irrespective of the hardware vendor or NOS.

Network Port LEDs

A network port LED indicates the state of the link, such as link UP or Tx/Rx activity. Here are the requirements for these LEDs:

- **Number of LEDs per port** - Ports that cannot be split; for example, 1G ports must have 1 LED per port. Ports that can be split should have 1 LED per split port. So a 40G port that can be split into 4 10G ports has 4 LEDs, one per split port.
- **Location** - A port LED should be placed right above the port. This prevents the LEDs from being hidden by drooping cables. If the port can be split, the LED for each split port should also be placed above the port. The LEDs should be evenly spaced and be inside the edges of the ports to prevent confusion.
- **Port Number Label** - The port number must be printed in white on the

switch front panel directly under the corresponding LED.

- **Colors** - As network port technology improves with smaller ports and higher speeds, having different colors for different types of ports or speeds is confusing. The focus should be on giving a network operator a simple set of indications that provide the operator with basic information about the port. Hence, green and amber colors must be used on the LED to differentiate between good and bad states. These colors are commonly found on network port LEDs and should be easy to implement on future switches.
- **Signaling** - The table below indicates the information that can be conveyed via port LEDs and how it should be done.
  - **Max Speed** indicates the maximum speed at which the port can be run. For a 10G port, if the port speed is 10G, then it is running at its maximum speed. If the 10G port is running at 1G speed then its running at a *lower speed*.
  - **Physical Link Up/Down** displays layer 2 link status.
  - **Beaconing** provides a way for a network operator to identify a particular link. The administrator can *beacon* that port from a remote location so the network operator has visual indication for that port.
  - **Fault** can also be considered a form of beaconing or vice versa. Both try to draw attention of the network operator towards the port, thus they are signaled the same way.
  - **Blinking amber** implies a blink rate of 33ms. *Slow blinking amber* indicates a blink rate of 500ms, with a 50% on/off duty cycle. In other words, a slow blinking amber LED is amber for 500 ms and then off for 500ms.

Activity	Max Speed indication	Lower Speed Indication
Physical Link Down	Off	Off
Physical Link UP	Solid Green	Solid Amber
Link Tx/Rx Activity	Blinking Green	Blinking Amber
Beaconing	Slow Blinking Amber	Slow Blinking Amber
Fault	Slow Blinking Amber	Slow Blinking Amber

Status LEDs

A set of status LEDs are typically located on one side of a network switch. The status LEDs provide a visual indication on what is physically wrong with the network switch. Typical LEDs on the front panel are for PSUs (power supply units), fans and system. Locator LEDs are also found on the front panel of a switch. Each component that has an LED is known as a *unit* below.

- **Number of LEDs per unit** - Each unit should have only 1 LED.
- **Location** - All units should have their LEDs on the right-hand side of the switch after the physical ports.
- **Unit label** - The label should be printed on the front panel directly above the LED.

- **Colors** - The focus should be on giving a network operator a simple set of indications that provide basic information about the unit. The following section has more information about the indications, but colors are standardized on green and amber. These colors are universally found on all status LEDs and should be easy to implement on future switches.
- **Defined LED** - Every network switch must have LEDs for the following:
  - PSU
  - Fans
  - System LED
  - Locator LED
- **PSU LEDs** - Each PSU must have its own LED. PSU faults are difficult to debug. If a network operator knows which PSU is faulty, he or she can quickly check if it is powered up correctly and, if that fault persists, replace the PSU.

Unit Activity	Indication
Installed and power OK	Solid Green
Installed, but no power	Slow Blinking Amber
Installed, powered, but has faults.	Slow Blinking Amber

- **Fan LED** - A network switch may have multiple fan trays (3 - 6). It is difficult to put an LED for each fan tray on the front panel, given the limited real estate. Hence, the recommendation is one LED for all fans.

Unit Activity	Indication
All fans running OK	Solid Green
Fault on any one of the fans.	Slow Blinking Amber

- **System LED** - A network switch must have a system LED that indicates the general state of a switch. This state could be of hardware, software, or both. It is up to the individual switch NOS to decide what this LED indicates. But the LED can have only the following indications:

Unit Activity	Indication
All OK	Solid Green
Not OK	Slow Blinking Amber

- **Locator LED** - The locator LED helps locate a particular switch in a data center full of switches. Thus, it should have a different color and predefined location. It must be located at the top right corner on the front panel of the switch and its color must be blue.

Unit Activity	Indication
Locate enabled	Blinking Blue
Locate disabled	Off

Locate a Switch

Cumulus Linux supports the locator LED functionality for identifying a switch, by blinking a single LED on a specified network port, on the following switches:

- Dell Z9100-ON
- Edgecore AS7712-32X
- Penguin Arctica 3200c
- Quanta QuantaMesh BMS T4048-IX2
- Supermicro SSE-C3632S

To use the locator LED functionality, run:

```
cumulus@switch:~$ ethtool -p --identify PORT_NAME TIME
```

In the example above, `INTERFACE_NAME` should be replaced with the name of the port, and `TIME` should be replaced with the length of time, in seconds, that the port LED should blink.

> **NOTE**
>
> This functionality is only supported on swp* ports, not eth* management interfaces.

Considerations

Dell-N3048EP-ON LED Colors at Low Speeds

Across all 48 ports on a Dell-N3048EP-ON switch, if the link speed of a device is 10Mbps, the link light does not come on and only the activity light is seen. Traffic does work properly at this speed.

> **NOTE**
>
> Cumulus Linux does not support 10M speeds.

If you set the ports to 100M, the link lights for ports 1-46 are orange, while the lights for ports 47 and 48 are green.

When all of the ports are set to 1G, all the link lights are green.

EdgeCore Minipack-AS8000 Port LED Blink State

The port LED blink state that indicates link activity is not implemented; the ports only have ON/OFF states.

Penguin Arctica 3200c Front Panel ALARM LED

On the Penguin Arctica 3200c switch, the front panel ALARM LED is not functional and remains off when you remove or insert a power module. The rear panel ALARM always flashes yellow.

TDR Cable Diagnostics

Cumulus Linux provides the Time Domain Reflectometer (TDR) cable diagnostic tool, which enables you to isolate cable faults on unshielded twisted pair (UTP) cable runs.

> **NOTE**
>
> TDR is supported on the EdgeCore AS4610 and Dell N3248PXE switches. Pluggable modules are not supported.

Run Cable Diagnostics

Cumulus Linux TDR runs, checks, and reports on the status of the cable diagnostic circuitry for specified ports.

> **WARNING**
>
> Running TDR is disruptive to an active link; If the link is up on an enabled port when you start diagnostics on the port, the link is brought down, then brought back up when the diagnostics are complete.

> **NOTE**
>
> To obtain the most accurate results, make sure that auto-negotiation is enabled on both the switch port and the link partner (for fixed copper ports, auto-negotiation is enabled by default in Cumulus Linux and cannot be disabled).

To run cable diagnostics and report results, issue the `cl-tdr <port-list>` command. You must have root permissions to run the command. Because the test is disruptive, a warning message displays and you are prompted to continue.

The following example command runs cable diagnostics on swp39:

```
cumulus@switch:~$ sudo cl-tdr swp39


Time Domain Reflectometer (TDR) diagnostics tests are
disruptive.
When TDR is run on an interface, it will cause the interface to
go down momentarily during the test. The interface will be
restarted
at the conclusion of the test.
```

```
The following interfaces may be affected:
swp39

Are you sure you want to continue? [yes/NO]yes
swp39 current results @ 2019-08-05 09:37:53 EDT
      cable(4 pairs)
      pair A Ok, length 15 meters (+/-10)
      pair B Ok, length 15 meters (+/-10)
      pair C Ok, length 17 meters (+/-10)
      pair D Ok, length 13 meters (+/-10)
```

Command Options

The `cl-tdr` command includes several options, described below:

Option	Description
`-h`	Displays this list of command options.
`-d <delay>`	The delay in seconds between diagnostics on different ports when you run the command on multiple ports. You can specify 0 through 30 seconds. The default is 2 seconds.
`-j`	Displays diagnostic results in

Option	Description
	JSON format.
`-y`	Proceeds automatically without the warning or prompt.

Understanding Diagnostic Results

The TDR tool reports diagnostic test results per pair for each port. For example:

```
swp39 current results @ 2019-08-05 09:37:53 EDT
     cable(4 pairs)
     pair A Ok, length 15 meters (+/-10)
     pair B Ok, length 15 meters (+/-10)
     pair C Ok, length 17 meters (+/-10)
     pair D Ok, length 13 meters (+/-10)
```

Possible cable pair states are as follows:

State	Description
`Ok`	No cable fault is detected.
`Open`	A lack of continuity is detected between the pins at each end of the cable.

State	Description
`Short`	A short-circuit is detected on the cable.
`Open/Short`	Either a lack of continuity between the pins at each end of the cable or a short-circuit is detected on the cable.
`Crosstalk`	A signal transmitted on one pair is interfering with and degrading the transmission on another pair.
`Unknown`	An unknown issue is detected.

Per pair cable faults are detected within plus or minus 5 meters. Good cable accuracy is detected within plus or minus 10 meters.

Cable Diagnostic Logs

Cable diagnostic results are also logged to the `/var/log/switchd.log` file. For example:

```
2019-08-05T10:02:30.691513-04:00 act-4610p-53 switchd[3037]:
hal_bcm_port.c:3495 swp39 Enhanced Cable Diagnostics (TDR)
started
2019-08-05T10:02:31.466523-04:00 act-4610p-53 switchd[3037]:
hal_bcm_port.c:3446 swp39 TDR state=Ok npairs=4 +/- 10
```

```
2019-08-05T10:02:31.468735-04:00 act-4610p-53 switchd[3037]:
hal_bcm_port.c:3449 swp39 TDR pair A state=Ok len=17
2019-08-05T10:02:31.471958-04:00 act-4610p-53 switchd[3037]:
hal_bcm_port.c:3453 swp39 TDR pair B state=Ok len=18
2019-08-05T10:02:31.475047-04:00 act-4610p-53 switchd[3037]:
hal_bcm_port.c:3457 swp39 TDR pair C state=Ok len=18
2019-08-05T10:02:31.477109-04:00 act-4610p-53 switchd[3037]:
hal_bcm_port.c:3461 swp39 TDR pair D state=Ok len=15
```

Example Commands

The following command runs diagnostics on ports swp39, swp40, and swp32 and sets the delay to one second:

```
cumulus@switch:~$  sudo cl-tdr swp39-40,swp32 -d 1
```

The following command example runs diagnostics on swp39 and reports the results in json format:

```
cumulus@switch:~$  sudo cl-tdr swp39 -j
```

The following command runs diagnostics on ports swp39 and swp40

without displaying the warning or prompting to continue:

```
cumulus@switch:~$   sudo cl-tdr swp39-40 -y
```

Understanding the cl-support Output File

The `cl-support` script generates a compressed archive file of useful information for troubleshooting. The system either creates the archive file automatically or you can create the archive file manually.

Automatic cl-support File

The system creates the `cl-support` archive file automatically for the following reasons:

- When there is a core file dump of any application (not specific to Cumulus Linux, but something all Linux distributions support), located in `/var/support/core`.
- After the first failure of one of several monitored services since the switch was rebooted or power cycled.

Manual cl-support File

To create the `cl-support` archive file manually, run the `cl-support` command:

```
cumulus@switch:~$ sudo cl-support
```

If the support team requests that you submit the output from `cl-support` to help with the investigation of issues you might experience with Cumulus Linux and you need to include security-sensitive information, such as the `sudoers` file, use the `-s` option:

```
cumulus@switch:~$ sudo cl-support -s
```

> **NOTE**
>
> On ARM switches, the cl-support FRR module might time out even when FRR is not running. To disable the timeout, run the `cl-support` command with the `-M` option; for example:
>
> ```
> cumulus@switch:~$ sudo cl-support -M
> ```

For information on the directories included in the `cl-support` archive, see:

- Troubleshooting the etc Directory. The `/etc` directory contains the largest number of files; however, log files might be significantly larger in file size.
- Troubleshooting Log Files. This guide highlights the most important log files to inspect. Keep in mind, `cl-support` includes all of the log files.

Troubleshooting Log Files

The only real unique entity for logging on Cumulus Linux compared to any other Linux distribution is `switchd.log`, which logs the HAL (hardware abstraction layer) from hardware like the Broadcom or Mellanox Spectrum ASIC.

Read this guide on NixCraft to understand how `/var/log` works.

Log File Descriptions

Log	Description
`/var/log/alternatives.log`	Information from update-alternatives.
`/var/log/apt`	Information from the `apt` utility. For example, from `apt-get install` and `apt-get remove`.
`/var/log/audit/*`	Information stored by the Linux audit daemon, `auditd`.
`/var/log/autoprovision`	Output generated by running the zero touch provisioning script (ZTP).
`/var/log/boot.log`	Information that is logged when the system boots.
`/var/log/btmp`	Information about failed login attempts. Use the last command

Log	Description
	to view the `btmp` file. For example: `cumulus@switch:~$ last -f /var/log/btmp | more`
`/var/log/clagd.log`	Status of the `clagd` service.
`/var/log/dpkg.log`	Information logged when a package is installed or removed using the `dpkg` command.
`/var/log/frr/*`	FRRouting - Used to troubleshoots routing, such as an MD5 or MTU mismatch with OSPF.
`/var/log/gunicorn`	Error and access events in Gunicorn.
`/var/log/installer/*`	Directory containing files related to the installation of Cumulus Linux.
`/var/log/lastlog`	Formats and prints the contents of the last login log file.
`/var/log/netd.log`	Log file for NCLU.
`/var/log/netd-history.log`	Log file for NCLU configuration commits.

Log	Description
`/var/log/nginx`	Errors and processed requests in NGINX.
`/var/log/ntpstats`	Logs for network configuration protocol.
`/var/log/openvswitch/*`	ovsdb-server logs.
`/var/log/ptmd`	Prescriptive Topology Manager (PTM) errors and information.
`/var/log/switchd.log`	The HAL log for Cumulus Linux. This is specific to Cumulus Linux. Any switchd crashes are logged here.
`/var/log/syslog`	The main system log, which logs everything except auth-related messages. The primary log; grep this file to see what problem occurred.
`/var/log/wtmp`	Login records file.

Troubleshooting the etc Directory

The `cl-support` script replicates the `/etc` directory, however, it deliberately excludes certain files, such as `/etc/nologin`, which prevents unprivileged users from logging into the system.

This is the alphabetical list of the output from running `ls -l` on the `/etc` directory structure created by `cl-support`.

etc Directory Contents

File
acpi
adduser.conf
alternatives
apparmor.d
apt
audisp
audit
bash.bashrc
bash_completion

File
bash_completion.d
bcm.d
bindresvport.blacklist
binfmt.d
ca-certificates
ca-certificates.conf
calendar
console-setup
cron.d
cron.daily
cron.hourly
cron.monthly
crontab
cron.weekly
cruft
cumulus
dbus-1
debconf.conf
debian_version

File
debsums-ignore
default
deluser.conf
dhcp
discover.conf.d
discover-modprobe.conf
dnsmasq.conf
dnsmasq.d
dpkg
e2fsck.conf
emacs
environment
etckeeper
ethertypes
fonts
freeipmi
frr
fstab
gai.conf

File
groff
grub.d
gshadow
gshadow-
gss
gunicorn.conf.py
hostapd
hostapd.conf
host.conf
hostname
hsflowd
hsflowd.conf
hw_init.d
image-release
init
init.d
initramfs-tools
inputrc
insserv

File
insserv.conf
insserv.conf.d
iproute2
issue
issue.net
kernel
ldap
ld.so.cache
ld.so.conf
ld.so.conf.d
libaudit.conf
libnl
linuxptp
lldpd.d
locale.alias
locale.gen
localtime
logcheck
login.defs

File
login.defs.cumulus
login.defs.cumulus-orig
logrotate.conf
logrotate.conf.cumulus
logrotate.conf.cumulus-orig
logrotate.d
lsb-release
lvm
machine-id
magic
magic.mime
mailcap
mailcap.order
manpath.config
mime.types
mke2fs.conf
modprobe.d
modules
modules-load.d

File
motd
motd.distrib
mtab
mysql
nanorc
netd.conf
netq
network
networks
nginx
nsswitch.conf
ntp.conf
openvswitch
opt
os-release
perl
profile
profile.cumulus
profile.cumulus-orig

File
profile.d
protocols
ptm.d
ptp4l.conf
python
python2.7
python3
python3.7
ras
rc0.d
rc1.d
rc2.d
rc3.d
rc4.d
rc5.d
rc6.d
rcS.d
rdnbrd.conf
resolv.conf

File
resolvconf
resolv.conf.bak
restapi.conf
rmt
rpc
rsyslog.conf
rsyslog.conf.cumulus
rsyslog.conf.cumulus-orig
rsyslog.d
runit
screenrc
securetty
security
selinux
sensors3.conf
sensors.d
services
sgml
shells

File
skel
smartd.conf
smartmontools
snmp
ssh
subgid
subgid-
subuid
subuid-
sv
sysctl.conf
sysctl.d
systemd
terminfo
timezone
tmpfiles.d
ucf.conf
udev
ufw

File
update-motd.d
vim
vrf
watchdog.conf
wgetrc
X11
xattr.conf
xdg
xml

Troubleshooting Network Interfaces

The following sections describe various ways you can troubleshoot `ifupdown2` and network interfaces.

Enable Logging for Networking

To obtain verbose logs when you run `systemctl [start|restart] networking.service` as well as when the switch boots, create an overrides file with the `systemctl edit networking.service` command and add the following lines:

```
[Service]
# remove existing ExecStart rule
ExecStart=
# start ifup with verbose option
ExecStart=/sbin/ifup -av
```

> **NOTE**
>
> When you run the `systemctl edit` command, you do *not* need to run `systemctl daemon-reload`.

To disable logging, either:

- Remove the overrides file. Run the `systemctl cat networking` command to show the name of the overrides file.
- Run the `systemctl edit networking.service` command and remove the lines you added.

Exclude Certain Interfaces from Coming Up

To exclude an interface so that it does not come up when you boot the switch or start/stop/reload the networking service:

1. Create a file in the `/etc/systemd/system/networking.service.d` directory (for example, `/etc/systemd/system/networking.service.d/override.conf`).

2. Add the following lines to the file, where `<interface>` is the interface you want to exclude (such as eth0).

```
[Service]
ExecStart=
ExecStart=/sbin/ifup -a -X <interface>
ExecStop=
ExecStop=/sbin/ifdown -a -X <interface>
```

You can exclude any interface specified in the `/etc/network/interfaces` file.

Use ifquery to Validate and Debug Interface Configurations

You use `ifquery` to print parsed `interfaces` file entries.

To use `ifquery` to pretty print `iface` entries from the `interfaces` file, run:

```
cumulus@switch:~$ sudo ifquery bond0
auto bond0
iface bond0
    address 14.0.0.9/30
    address 2001:ded:beef:2::1/64
    bond-slaves swp25 swp26
```

Use `ifquery --check` to check the current running state of an interface within the `interfaces` file. It will return exit code *0* or *1* if the configuration does not match. The line `bond-xmit-hash-policy layer3+7` below fails because it should read `bond-xmit-hash-policy layer3+4`.

```
cumulus@switch:~$ sudo ifquery --check bond0
iface bond0
    bond-xmit-hash-policy layer3+7  [fail]
```

```
    bond-slaves swp25 swp26          [pass]
    address 14.0.0.9/30              [pass]
    address 2001:ded:beef:2::1/64    [pass]
```

> **NOTE**
>
> `ifquery --check` is an experimental feature.

Use `ifquery --running` to print the running state of interfaces in the `interfaces` file format:

```
cumulus@switch:~$ sudo ifquery --running bond0
auto bond0
iface bond0
    bond-slaves swp25 swp26
    address 14.0.0.9/30
    address 2001:ded:beef:2::1/64
```

`ifquery --syntax-help` provides help on all possible attributes supported in the `interfaces` file. For complete syntax on the `interfaces` file, see `man interfaces` and `man ifupdown-addons-interfaces`.

You can use `ifquery --print-savedstate` to check the `ifupdown2` state database. `ifdown` works only on interfaces present in this state database.

```
cumulus@leaf1$ sudo ifquery --print-savedstate eth0
auto eth0
iface eth0 inet dhcp
```

Mako Template Errors

An easy way to debug and get details about template errors is to use the `mako-render` command on your interfaces template file or on `/etc/network/interfaces` itself.

```
cumulus@switch:~$ sudo mako-render /etc/network/interfaces
# This file describes the network interfaces available on your
system
# and how to activate them. For more information, see
interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
```

```
auto eth0
iface eth0 inet dhcp
#auto eth1
#iface eth1 inet dhcp

# Include any platform-specific interface configuration
source /etc/network/interfaces.d/*.if

# ssim2 added
auto swp45
iface swp45

auto swp46
iface swp46

cumulus@switch:~$ sudo mako-render /etc/network/
interfaces.d/<interfaces_stub_file>
```

ifdown Cannot Find an Interface that Exists

If you are trying to bring down an interface that you know exists, use `ifdown` with the `--use-current-config` option to force `ifdown` to check the current `/etc/network/interfaces` file to find the interface. This can solve issues where the `ifup` command issues for that interface are interrupted before it

updates the state database. For example:

```
cumulus@switch:~$ sudo ifdown br0
error: cannot find interfaces: br0 (interface was probably
never up ?)

cumulus@switch:~$ sudo brctl show
bridge name   bridge id      STP enabled interfaces
br0      8000.44383900279f   yes     downlink
                             peerlink

cumulus@switch:~$ sudo ifdown br0 --use-current-config
```

Remove All References to a Child Interface

If you have a configuration with a child interface, whether it is a VLAN, bond, or another physical interface and you remove that interface from a running configuration, you must remove every reference to it in the configuration. Otherwise, the parent interface continues to use the interface.

For example, consider the following configuration:

```
auto lo
```

```
iface lo inet loopback


auto eth0
iface eth0 inet dhcp


auto bond1
iface bond1
    bond-slaves swp2 swp1


auto bond3
iface bond3
    bond-slaves swp8 swp6 swp7


auto br0
iface br0
    bridge-ports swp3 swp5 bond1 swp4 bond3
    bridge-pathcosts  swp3=4 swp5=4 swp4=4
    address 11.0.0.10/24
    address 2001::10/64
```

Notice that bond1 is a member of br0. If bond1 is removed, you must remove the reference to it from the br0 configuration. Otherwise, if you reload the configuration with `ifreload -a`, bond1 is still part of br0.

MTU Set on a Logical Interface Fails with Error: “Numerical result out of range”

This error occurs when the MTU you are trying to set on an interface is higher than the MTU of the lower interface or dependent interface. Linux expects the upper interface to have an MTU less than or equal to the MTU on the lower interface.

In the example below, the swp1.100 VLAN interface is an upper interface to physical interface swp1. If you want to change the MTU to 9000 on the VLAN interface, you must include the new MTU on the lower interface swp1 as well.

```
auto swp1.100
iface swp1.100
    mtu 9000

auto swp1
iface swp1
    mtu 9000
```

iproute2 batch Command Failures

`ifupdown2` batches `iproute2` commands for performance reasons. A batch command contains `ip -force -batch -` in the error message. The

command number that failed is at the end of this line: `Command failed -:1`.

Below is a sample error for the command 1: `link set dev host2 master bridge`. There was an error adding the bond *host2* to the bridge named *bridge* because host2 did not have a valid address.

```
error: failed to execute cmd 'ip -force -batch - [link set dev
host2 master bridge
addr flush dev host2
link set dev host1 master bridge
addr flush dev host1
]'(RTNETLINK answers: Invalid argument
Command failed -:1)
warning: bridge configuration failed (missing ports)
```

“RTNETLINK answers: Invalid argument” Error when Adding a Port to a Bridge

This error can occur when the bridge port does not have a valid hardware address.

This occurs typically when the interface being added to the bridge is an incomplete bond; a bond without slaves is incomplete and does not have a valid hardware address.

MLAG Peerlink Interface Drops Many Packets

Losing a large number of packets across an MLAG peerlink interface might not be a problem. This can occur to prevent looping of BUM (broadcast, unknown unicast and multicast) packets. For more details, and for information on how to detect these drops, read the MLAG chapter.

Monitoring Interfaces and Transceivers Using ethtool

The `ethtool` command enables you to query or control the network driver and hardware settings. It takes the device name (like swp1) as an argument. When the device name is the only argument to `ethtool`, it prints the current settings of the network device. See `man ethtool(8)` for details. Not all options are currently supported on switch port interfaces.

Monitor Interface Status Using ethtool

To check the status of an interface using `ethtool`:

```
cumulus@switch:~$ ethtool swp1
Settings for swp1:
        Supported ports: [ FIBRE ]
        Supported link modes:   1000baseT/Full
                                10000baseT/Full
        Supported pause frame use: No
        Supports auto-negotiation: No
        Advertised link modes:  1000baseT/Full
        Advertised pause frame use: No
        Advertised auto-negotiation: No
        Speed: 10000Mb/s
```

```
        Duplex: Full
        Port: FIBRE
        PHYAD: 0
        Transceiver: external
        Auto-negotiation: off
        Current message level: 0x00000000 (0)

Link detected: yes
```

> **NOTE**
>
> The switch hardware contains the active port settings. The output of `ethtool swpXX` shows the port settings stored in the kernel. The `switchd` process keeps the hardware and kernel in sync for the important port settings (speed, auto-negotiation, and link detected) when they change. However, many of the fields in `ethtool`, like Supported Link Modes and Advertised Link Modes are not updated based on the actual module inserted in the port and therefore are incorrect or misleading.

To query interface statistics:

```
cumulus@switch:~$ sudo ethtool -S swp1
NIC statistics:
        HwIfInOctets: 1435339
        HwIfInUcastPkts: 11795
        HwIfInBcastPkts: 3
        HwIfInMcastPkts: 4578
        HwIfOutOctets: 14866246
        HwIfOutUcastPkts: 11791
        HwIfOutMcastPkts: 136493
        HwIfOutBcastPkts: 0
        HwIfInDiscards: 0
        HwIfInL3Drops: 0
        HwIfInBufferDrops: 0
        HwIfInAclDrops: 28
        HwIfInDot3LengthErrors: 0
        HwIfInErrors: 0
        SoftInErrors: 0
        SoftInDrops: 0
        SoftInFrameErrors: 0
        HwIfOutDiscards: 0
        HwIfOutErrors: 0
        HwIfOutQDrops: 0
        HwIfOutNonQDrops: 0
        SoftOutErrors: 0
```

```
        SoftOutDrops: 0
        SoftOutTxFifoFull: 0
        HwIfOutQLen: 0
```

View and Clear Interface Counters

Interface counters contain information about an interface. You can view this information when you run `cl-netstat`, `ifconfig`, or `cat /proc/net/dev`. You can also use `cl-netstat` to save or clear this information:

```
cumulus@switch:~$ sudo cl-netstat
Kernel Interface table
Iface   MTU Met        RX_OK RX_ERR RX_DRP RX_OVR        TX_OK
TX_ERR TX_DRP TX_OVR    Flg
-------------------------------------------------------------------------------
eth0   1500   0          611      0      0      0
487      0      0      0   BMRU
lo    16436   0            0      0      0      0
0      0      0      0    LRU
swp1   1500   0            0      0      0      0
0      0      0      0    BMU


cumulus@switch:~$ sudo cl-netstat -c
```

```
Cleared counters
```

Option	Description
`-c`	Copies and clears statistics. It does not clear counters in the kernel or hardware. **Note**: The -c argument is applied per user ID by default. You can override it by using the `-t` argument to save statistics to a different directory.
`-d`	Deletes saved statistics, either the uid or the specified tag. **Note**: The `-d` argument is applied per user ID by default. You can override it by using the `-t` argument to save statistics to a different directory.
`-D`	Deletes all saved statistics.
`-l`	Lists saved tags.
`-r`	Displays raw statistics (unmodified output of `cl-netstat`).
`-t <tag name>`	Saves statistics with `<tag`

Option	Description
	`name>`.
`-v`	Prints `cl-netstat` version and exits.

Monitor Switch Port SFP/QSFP Hardware Information Using ethtool

To see hardware capabilities and measurement information on the SFP or QSFP module installed in a particular port, use the `ethtool -m` command. If the SFP/QSFP supports Digital Optical Monitoring (that is, the `Optical diagnostics support` field in the output below is set to *Yes*), the optical power levels and thresholds are also printed below the standard hardware details.

In the sample output below, you can see that this module is a 1000BASE-SX short-range optical module, manufactured by JDSU, part number PLRXPL-VI-S24-22. The second half of the output displays the current readings of the Tx power levels (`Laser output power`) and Rx power (`Receiver signal average optical power`), temperature, voltage and alarm threshold settings.

```
cumulus@switch$ sudo ethtool -m swp3
        Identifier                                : 0x03 (SFP)
```

```
        Extended identifier                      : 0x04 (GBIC/
SFP defined by 2-wire interface ID)
        Connector                                : 0x07 (LC)
        Transceiver codes                        : 0x00 0x00
0x00 0x01 0x20 0x40 0x0c 0x05
        Transceiver type                         : Ethernet:
1000BASE-SX
        Transceiver type                         : FC:
intermediate distance (I)
        Transceiver type                         : FC:
Shortwave laser w/o OFC (SN)
        Transceiver type                         : FC:
Multimode, 62.5um (M6)
        Transceiver type                         : FC:
Multimode, 50um (M5)
        Transceiver type                         : FC: 200
MBytes/sec
        Transceiver type                         : FC: 100
MBytes/sec
        Encoding                                 : 0x01 (8B/
10B)
        BR, Nominal                              : 2100MBd
        Rate identifier                          : 0x00
(unspecified)
```

```
        Length (SMF,km)                         : 0km
        Length (SMF)                            : 0m
        Length (50um)                           : 300m
        Length (62.5um)                         : 150m
        Length (Copper)                         : 0m
        Length (OM3)                            : 0m
        Laser wavelength                        : 850nm
        Vendor name                             : JDSU
        Vendor OUI                              : 00:01:9c
        Vendor PN                               : PLRXPL-VI-
S24-22
        Vendor rev                              : 1
        Optical diagnostics support             : Yes
        Laser bias current                      : 21.348 mA
        Laser output power                      : 0.3186 mW /
-4.97 dBm
        Receiver signal average optical power   : 0.3195 mW /
-4.96 dBm
        Module temperature                      : 41.70
degrees C / 107.05 degrees F
        Module voltage                          : 3.2947 V
        Alarm/warning flags implemented         : Yes
        Laser bias current high alarm           : Off
        Laser bias current low alarm            : Off
```

```
        Laser bias current high warning          : Off
        Laser bias current low warning           : Off
        Laser output power high alarm            : Off
        Laser output power low alarm             : Off
        Laser output power high warning          : Off
        Laser output power low warning           : Off
        Module temperature high alarm            : Off
        Module temperature low alarm             : Off
        Module temperature high warning          : Off
        Module temperature low warning           : Off
        Module voltage high alarm                : Off
        Module voltage low alarm                 : Off
        Module voltage high warning              : Off
        Module voltage low warning               : Off
        Laser rx power high alarm                : Off
        Laser rx power low alarm                 : Off
        Laser rx power high warning              : Off
        Laser rx power low warning               : Off
        Laser bias current high alarm threshold  : 10.000 mA
        Laser bias current low alarm threshold   : 1.000 mA
        Laser bias current high warning threshold : 9.000 mA
         Laser bias current low warning threshold  : 2.000 mA
        Laser output power high alarm threshold  : 0.8000 mW /
-0.97 dBm
```

```
        Laser output power low alarm threshold    : 0.1000 mW /
-10.00 dBm
        Laser output power high warning threshold : 0.6000 mW /
-2.22 dBm
        Laser output power low warning threshold  : 0.2000 mW /
-6.99 dBm
        Module temperature high alarm threshold   : 90.00
degrees C / 194.00 degrees F
        Module temperature low alarm threshold    : -40.00
degrees C / -40.00 degrees F
        Module temperature high warning threshold : 85.00
degrees C / 185.00 degrees F
        Module temperature low warning threshold  : -40.00
degrees C / -40.00 degrees F
        Module voltage high alarm threshold       : 4.0000 V
        Module voltage low alarm threshold        : 0.0000 V
        Module voltage high warning threshold     : 3.6450 V
        Module voltage low warning threshold      : 2.9550 V
        Laser rx power high alarm threshold       : 1.6000 mW /
2.04 dBm
        Laser rx power low alarm threshold        : 0.0100 mW /
-20.00 dBm
        Laser rx power high warning threshold     : 1.0000 mW /
0.00 dBm
```

```
        Laser rx power low warning threshold      : 0.0200 mW /
-16.99 dBm
```

Network Troubleshooting

Cumulus Linux includes a number of command line and analytical tools to help you troubleshoot issues with your network.

Check Reachability Using ping

Use `ping` to check reachability of a host. `ping` also calculates the time it takes for packets to travel the round trip. See `man ping` for details.

To test the connection to an IPv4 host:

```
cumulus@switch:~$ ping 192.0.2.45
PING 192.0.2.45 (192.0.2.45) 56(84) bytes of data.
64 bytes from 192.0.2.45: icmp_req=1 ttl=53 time=40.4 ms
64 bytes from 192.0.2.45: icmp_req=2 ttl=53 time=39.6 ms
...
```

To test the connection to an IPv6 host:

```
cumulus@switch:~$ ping6 -I swp1 2001::db8:ff:fe00:2
PING 2001::db8:ff:fe00:2(2001::db8:ff:fe00:2) from
2001::db8:ff:fe00:1 swp1: 56 data bytes
```

```
64 bytes from 2001::db8:ff:fe00:2: icmp_seq=1 ttl=64 time=1.43
ms
64 bytes from 2001::db8:ff:fe00:2: icmp_seq=2 ttl=64 time=0.927
ms
```

When troubleshooting intermittent connectivity issues, it is helpful to send continuous pings to a host.

Print Route Trace Using traceroute

`traceroute` tracks the route that packets take from an IP network on their way to a given host. See `man traceroute` for details.

To track the route to an IPv4 host:

```
cumulus@switch:~$ traceroute www.google.com
traceroute to www.google.com (74.125.239.49), 30 hops max, 60
byte packets
1  cumulusnetworks.com (192.168.1.1)  0.614 ms  0.863 ms  0.932
ms
...
5  core2-1-1-0.pao.net.google.com (198.32.176.31)  22.347 ms
22.584 ms  24.328 ms
```

```
6  216.239.49.250 (216.239.49.250)  24.371 ms  25.757 ms
25.987 ms
7  72.14.232.35 (72.14.232.35)  27.505 ms  22.925 ms  22.323 ms
8  nuq04s19-in-f17.1e100.net (74.125.239.49)  23.544 ms  21.851
ms  22.604 ms
```

Run Commands in a Non-default VRF

You can use `ip vrf exec` to run commands in a non-default VRF context. This is particularly useful for network utilities like `ping`, `traceroute`, and `nslookup`.

The full syntax is `ip vrf exec <vrf-name> <command> <arguments>`. For example:

```
cumulus@switch:~$ sudo ip vrf exec Tenant1 nslookup google.com
- 8.8.8.8
```

By default, `ping`/`ping6` and `traceroute`/`traceroute6` all use the default VRF. This is done using a mechanism that checks the VRF context of the current shell - which can be seen when you run `ip vrf id` - at the time one of these commands is run. If the shell's VRF context is *mgmt*, then these commands are run in the default VRF context.

`ping` and `traceroute` have additional arguments that you can use to specify an egress interface and/or a source address. In the default VRF, the source interface flag (`ping -I` or `traceroute -i`) specifies the egress interface for the `ping`/`traceroute` operation. However, you can use the source interface flag instead to specify a non-default VRF to use for the command. Doing so causes the routing lookup for the destination address to occur in that VRF.

With `ping -I`, you can specify the source interface or the source IP address, but you cannot use the flag more than once. Thus, you can choose either an egress interface/VRF or a source IP address. For `traceroute`, you can use `traceroute -s` to specify the source IP address.

You gain some additional flexibility if you run `ip vrf exec` in combination with `ping`/`ping6` or `traceroute`/`traceroute6`, as the VRF context is specified outside of the `ping` and `traceroute` commands. This allows for the most granular control of `ping` and `traceroute`, as you can specify both the VRF and the source interface flag.

For `ping`, use the following syntax:

```
ip vrf exec <vrf-name> [ping|ping6] -I [<egress_interface> |
<source_ip>] <destination_ip>
```

For example:

```
cumulus@switch:~$ sudo ip vrf exec Tenant1 ping -I swp1 8.8.8.8
cumulus@switch:~$ sudo ip vrf exec Tenant1 ping -I 192.0.1.1
8.8.8.8
cumulus@switch:~$ sudo ip vrf exec Tenant1 ping6 -I swp1
2001:4860:4860::8888
cumulus@switch:~$ sudo ip vrf exec Tenant1 ping6 -I 2001:db8::1
2001:4860:4860::8888
```

For `traceroute`, use the following syntax:

```
ip vrf exec <vrf-name> [traceroute|traceroute6] -i
<egress_interface> -s <source_ip> <destination_ip>
```

For example:

```
cumulus@switch:~$ sudo ip vrf exec Tenant1 traceroute -i swp1
-s 192.0.1.1 8.8.8.8
cumulus@switch:~$ sudo ip vrf exec Tenant1 traceroute6 -i swp1
-s 2001:db8::1 2001:4860:4860::8888
```

Because the VRF context for `ping` and `traceroute` commands is

automatically shifted to the default VRF context, you must use the source interface flag to specify the management VRF. Typically, this is not an issue since there is only a single interface in the management VRF - eth0 - and in most situations only a single IPv4 address or IPv6 global unicast address is assigned to it. But it is worth mentioning since, as stated earlier, you cannot specify both a source interface and a source IP address with `ping -I`.

Manipulate the System ARP Cache

`arp` manipulates or displays the kernel's IPv4 network neighbor cache. See `man arp` for details.

To display the ARP cache:

```
cumulus@switch:~$ arp -a
? (11.0.2.2) at 00:02:00:00:00:10 [ether] on swp3
? (11.0.3.2) at 00:02:00:00:00:01 [ether] on swp4
? (11.0.0.2) at 44:38:39:00:01:c1 [ether] on swp1
```

To delete an ARP cache entry:

```
cumulus@switch:~$ arp -d 11.0.2.2
cumulus@switch:~$ arp -a
? (11.0.2.2) at <incomplete> on swp3
```

```
? (11.0.3.2) at 00:02:00:00:00:01 [ether] on swp4
? (11.0.0.2) at 44:38:39:00:01:c1 [ether] on swp1
```

To add a static ARP cache entry:

```
cumulus@switch:~$ arp -s 11.0.2.2 00:02:00:00:00:10
cumulus@switch:~$ arp -a
? (11.0.2.2) at 00:02:00:00:00:10 [ether] PERM on swp3
? (11.0.3.2) at 00:02:00:00:00:01 [ether] on swp4
? (11.0.0.2) at 44:38:39:00:01:c1 [ether] on swp1
```

If you need to flush or remove an ARP entry for a specific interface, you can disable dynamic ARP learning:

```
cumulus@switch:~$ ip link set arp off dev INTERFACE
```

Generate Traffic Using mz

`mz` (or `mausezahn`) is a fast traffic generator. It can generate a large variety of packet types at high speed. See `man mz` for details.

For example, to send two sets of packets to TCP port 23 and 24, with

source IP address 11.0.0.1 and destination IP address 11.0.0.2:

```
cumulus@switch:~$ sudo mz swp1 -A 11.0.0.1 -B 11.0.0.2 -c 2 -v
-t tcp "dp=23-24"

Mausezahn 0.40 - (C) 2007-2010 by Herbert Haas -
https://packages.debian.org/unstable/mz
Use at your own risk and responsibility!
-- Verbose mode --

This system supports a high resolution clock.
  The clock resolution is 4000250 nanoseconds.
Mausezahn will send 4 frames...
  IP:  ver=4, len=40, tos=0, id=0, frag=0, ttl=255, proto=6,
sum=0, SA=11.0.0.1, DA=11.0.0.2,
       payload=[see next layer]
  TCP: sp=0, dp=23, S=42, A=42, flags=0, win=10000, len=20,
sum=0,
       payload=

  IP:  ver=4, len=40, tos=0, id=0, frag=0, ttl=255, proto=6,
sum=0, SA=11.0.0.1, DA=11.0.0.2,
       payload=[see next layer]
  TCP: sp=0, dp=24, S=42, A=42, flags=0, win=10000, len=20,
sum=0,
```

```
        payload=

  IP:  ver=4, len=40, tos=0, id=0, frag=0, ttl=255, proto=6,
sum=0, SA=11.0.0.1, DA=11.0.0.2,
        payload=[see next layer]
  TCP: sp=0, dp=23, S=42, A=42, flags=0, win=10000, len=20,
sum=0,
        payload=

  IP:  ver=4, len=40, tos=0, id=0, frag=0, ttl=255, proto=6,
sum=0, SA=11.0.0.1, DA=11.0.0.2,
        payload=[see next layer]
  TCP: sp=0, dp=24, S=42, A=42, flags=0, win=10000, len=20,
sum=0,
        payload=
```

Create Counter ACL Rules

In Linux, all ACL rules are always counted. To create an ACL rule for counting purposes only, set the rule action to ACCEPT. See the Netfilter chapter for details on how to use `cl-acltool` to set up iptables-/ip6tables-/ebtables-based ACLs.

> **NOTE**
>
> Always place your rules files under `/etc/cumulus/acl/policy.d/`.

To count all packets going to a Web server:

```
cumulus@switch:~$ cat sample_count.rules

  [iptables]
  -A FORWARD -p tcp --dport 80 -j ACCEPT

  cumulus@switch:~$ sudo cl-acltool -i -p sample_count.rules
  Using user provided rule file sample_count.rules
  Reading rule file sample_count.rules ...
  Processing rules in file sample_count.rules ...
  Installing acl policy... done.

  cumulus@switch:~$ sudo iptables -L -v
  Chain INPUT (policy ACCEPT 16 packets, 2224 bytes)
  pkts bytes target     prot opt in     out     source
destination

  Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
```

```
  pkts bytes target     prot opt in     out     source
destination
    2   156 ACCEPT     tcp  --  any    any     anywhere
anywhere           tcp dpt:http

  Chain OUTPUT (policy ACCEPT 44 packets, 8624 bytes)
  pkts bytes target     prot opt in     out     source
destination
```

WARNING

The `-p` option clears out all other rules. The `-i` option reinstalls all the rules.

Configure SPAN and ERSPAN

SPAN (Switched Port Analyzer) provides for the mirroring of all packets coming in from or going out of an interface (the *SPAN source*), and being copied and transmitted out of a local port or CPU (the *SPAN destination*) for monitoring. The SPAN destination port is also referred to as a mirror-to-port (MTP). The original packet is still switched, while a mirrored copy of the packet is sent out of the MTP.

ERSPAN (Encapsulated Remote SPAN) enables the mirrored packets to be sent to a monitoring node located anywhere across the routed network. The switch finds the outgoing port of the mirrored packets by doing a lookup of the destination IP address in its routing table. The original L2 packet is encapsulated with GRE for IP delivery. The encapsulated packets have the following format:

```
 ------------------------------------------------------------
| MAC_HEADER | IP_HEADER | GRE_HEADER | L2_Mirrored_Packet |
 ------------------------------------------------------------
```

NOTE

- Mirrored traffic is not guaranteed. If the MTP is congested, mirrored packets might be discarded.
- A SPAN and ERSPAN destination interface that is oversubscribed might result in data plane buffer depletion and buffer drops. Exercise caution when enabling SPAN and ERSPAN when the aggregate speeds of all source ports exceeds the destination port. Selective SPAN is recommended when possible to limit traffic in this scenario.

SPAN and ERSPAN are configured via `cl-acltool`, the same utility for

security ACL configuration. The match criteria for SPAN and ERSPAN is usually an interface; for more granular match terms, use selective spanning. The SPAN source interface can be a port, a subinterface, or a bond interface. Ingress traffic on interfaces can be matched, and on switches with Spectrum ASICs, egress traffic can be matched. See the list of limitations below.

Cumulus Linux supports a maximum of two SPAN destinations. Multiple rules (SPAN sources) can point to the same SPAN destination, although a given SPAN source cannot specify two SPAN destinations. The SPAN destination (MTP) interface can be a physical port, subinterface, bond interface or CPU. The SPAN and ERSPAN action is independent of security ACL actions. If packets match both a security ACL rule and a SPAN rule, both actions are carried out.

> **NOTE**
>
> Always place your rules files under `/etc/cumulus/acl/policy.d/`.

Limitations for SPAN and ERSPAN

- For Broadcom switches, Cumulus Linux supports a maximum of two SPAN destinations.
- Because SPAN and ERSPAN is done in hardware, eth0 is not supported as a destination.
- For Mellanox Spectrum switches, Cumulus Linux supports only a single SPAN destination in atomic mode or three SPAN destinations in non-

atomic mode.

- Multiple rules (SPAN sources) can point to the same SPAN destination, but a given SPAN source *cannot* specify two SPAN destinations.
- To configure SPAN or ERSPAN on a Tomahawk or Trident3 switch, you must enable non-atomic update mode.
- Mellanox Spectrum switches reject SPAN ACL rules for an output interface that is a subinterface.
- Mirrored traffic is not guaranteed. If the MTP is congested, mirrored packets might be discarded.
- Cut-through mode is not supported for ERSPAN in Cumulus Linux on switches using Broadcom Tomahawk, Trident II+ and Trident II ASICs.
- On Broadcom switches, SPAN does not capture egress traffic.
- Cumulus Linux does not support IPv6 ERSPAN destinations.
- ERSPAN does not cause the kernel to send ARP requests to resolve the next hop for the ERSPAN destination. If an ARP entry for the destination/next hop does not already exist in the kernel, you need to manually resolve this before mirrored traffic is sent (using ping or arping).
- Mirroring to the same interface that is being monitored will cause a recursive flood of traffic and may impact traffic on other interfaces.

Configure SPAN for Switch Ports

This section describes how to set up, install, verify and uninstall SPAN rules. In the examples that follow, you span (mirror) switch port swp4 input traffic and swp4 output traffic to destination switch port swp19.

First, create a rules file in `/etc/cumulus/acl/policy.d/`:

```
cumulus@switch:~$ sudo bash -c 'cat <<EOF > /etc/cumulus/acl/
policy.d/span.rules
[iptables]
-A FORWARD --in-interface swp4 -j SPAN --dport swp19
-A FORWARD --out-interface swp4 -j SPAN --dport swp19
EOF'
```

> **NOTE**
>
> Using `cl-acltool` with the `--out-interface` rule applies to transit traffic only; it does not apply to traffic sourced from the switch.

Next, verify all the rules that are currently installed:

```
cumulus@switch:~$ sudo iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out
source               destination
     0     0 DROP       all  --  swp+   any     240.0.0.0/
5          anywhere
     0     0 DROP       all  --  swp+   any     loopback/
```

```
8           anywhere
     0     0 DROP       all  --  swp+   any     base-
address.mcast.net/8  anywhere
     0     0 DROP       all  --  swp+   any
255.255.255.255      anywhere
     0     0 SETCLASS   ospf --  swp+   any
anywhere             anywhere            SETCLASS  class:7
     0     0 POLICE     ospf --  any    any
anywhere             anywhere            POLICE  mode:pkt
rate:2000 burst:2000
     0     0 SETCLASS   tcp  --  swp+   any
anywhere             anywhere            tcp dpt:bgp SETCLASS
class:7
     0     0 POLICE     tcp  --  any    any
anywhere             anywhere            tcp dpt:bgp POLICE
mode:pkt rate:2000 burst:2000
     0     0 SETCLASS   tcp  --  swp+   any
anywhere             anywhere            tcp spt:bgp SETCLASS
class:7
     0     0 POLICE     tcp  --  any    any
anywhere             anywhere            tcp spt:bgp POLICE
mode:pkt rate:2000 burst:2000
     0     0 SETCLASS   tcp  --  swp+   any
anywhere             anywhere            tcp dpt:5342
```

```
SETCLASS  class:7
     0     0 POLICE     tcp  --  any    any
anywhere             anywhere             tcp dpt:5342 POLICE
mode:pkt rate:2000 burst:2000
     0     0 SETCLASS   tcp  --  swp+   any
anywhere             anywhere             tcp spt:5342
SETCLASS  class:7
     0     0 POLICE     tcp  --  any    any
anywhere             anywhere             tcp spt:5342 POLICE
mode:pkt rate:2000 burst:2000
     0     0 SETCLASS   icmp --  swp+   any
anywhere             anywhere             SETCLASS  class:2
     0     0 POLICE     icmp --  any    any
anywhere             anywhere             POLICE  mode:pkt
rate:100 burst:40
    15  5205 SETCLASS   udp  --  swp+   any
anywhere             anywhere             udp
dpts:bootps:bootpc SETCLASS  class:2
    11  3865 POLICE     udp  --  any    any
anywhere             anywhere             udp dpt:bootps
POLICE  mode:pkt rate:100 burst:100
     0     0 POLICE     udp  --  any    any
anywhere             anywhere             udp dpt:bootpc
POLICE  mode:pkt rate:100 burst:100
```

```
     0     0 SETCLASS   tcp  --  swp+   any
anywhere             anywhere             tcp
dpts:bootps:bootpc SETCLASS  class:2
     0     0 POLICE     tcp  --  any    any
anywhere             anywhere             tcp dpt:bootps
POLICE  mode:pkt rate:100 burst:100
     0     0 POLICE     tcp  --  any    any
anywhere             anywhere             tcp dpt:bootpc
POLICE  mode:pkt rate:100 burst:100
    17  1088 SETCLASS   igmp --  swp+   any
anywhere             anywhere             SETCLASS class:6
    17  1156 POLICE     igmp --  any    any
anywhere             anywhere             POLICE  mode:pkt
rate:300 burst:100
    394 41060 POLICE    all  --  swp+   any
anywhere             anywhere             ADDRTYPE match dst-
type LOCAL POLICE  mode:pkt rate:1000 burst:1000 class:0
     0     0 POLICE     all  --  swp+   any
anywhere             anywhere             ADDRTYPE match dst-
type IPROUTER POLICE  mode:pkt rate:400 burst:100 class:0
    988  279K SETCLASS  all  --  swp+   any
anywhere            anywhere             SETCLASS  class:0

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
```

```
  pkts bytes target     prot opt in     out
source               destination
     0     0 DROP       all  --  swp+   any     240.0.0.0/
5          anywhere
     0     0 DROP       all  --  swp+   any     loopback/
8           anywhere
     0     0 DROP       all  --  swp+   any     base-
address.mcast.net/8  anywhere
     0     0 DROP       all  --  swp+   any
255.255.255.255      anywhere
 26864 4672K SPAN       all  --  swp4   any
anywhere             anywhere             dport:swp19  <----
input packets on swp4

40722   47M  SPAN       all  --  any    swp4
anywhere             anywhere             dport:swp19  <----
output packets on swp4

Chain OUTPUT (policy ACCEPT 67398 packets, 5757K bytes)
  pkts bytes target     prot opt in     out
source               destination
```

Install the rules:

```
cumulus@switch:~$ sudo cl-acltool -i
[sudo] password for cumulus:
Reading rule file /etc/cumulus/acl/policy.d/
00control_plane.rules ...
Processing rules in file /etc/cumulus/acl/policy.d/
00control_plane.rules ...
Reading rule file /etc/cumulus/acl/policy.d/
99control_plane_catch_all.rules ...
Processing rules in file /etc/cumulus/acl/policy.d/
99control_plane_catch_all.rules ...
Reading rule file /etc/cumulus/acl/policy.d/span.rules ...
Processing rules in file /etc/cumulus/acl/policy.d/span.rules
...
Installing acl policy
done.
```

WARNING

Running the following command is incorrect and will remove **all** existing control-plane rules or other installed rules and only install the rules defined in `span.rules`:

```
cumulus@switch:~$ sudo cl-acltool -i  -P /etc/cumulus/
acl/policy.d/span.rules
```

Verify that the SPAN rules are installed:

```
cumulus@switch:~$ sudo cl-acltool -L all | grep SPAN
38025 7034K SPAN       all  --  swp4   any
anywhere             anywhere             dport:swp19
50832   55M SPAN       all  --  any    swp4
anywhere             anywhere             dport:swp19
```

SPAN Sessions that Reference an Outgoing Interface

SPAN sessions that reference an outgoing interface create the mirrored packets based on the ingress interface before the routing/switching decision. For example, the following rule captures traffic that is ultimately destined to leave swp2 but mirrors the packets when they arrive on swp3. The rule transmits packets that reference the original VLAN tag and source/destination MAC address at the time the packet is originally

received on swp3.

```
-A FORWARD --out-interface swp2 -j SPAN --dport swp1
```

Configure SPAN for Bonds

This section describes how to configure SPAN for all packets going out of `bond0` locally to `bond1`.

First, create a rules file in `/etc/cumulus/acl/policy.d/`:

```
cumulus@switch:~$ sudo bash -c 'cat <<EOF > /etc/cumulus/acl/
policy.d/span_bond.rules
[iptables]
-A FORWARD --out-interface bond0 -j SPAN --dport bond1
EOF'
```

> **NOTE**
>
> Using `cl-acltool` with the `--out-interface` rule applies to transit traffic only; it does not apply to traffic sourced from the switch.

Install the rules:

```
cumulus@switch:~$ sudo cl-acltool -i
[sudo] password for cumulus:
Reading rule file /etc/cumulus/acl/policy.d/
00control_plane.rules ...
Processing rules in file /etc/cumulus/acl/policy.d/
00control_plane.rules ...
Reading rule file /etc/cumulus/acl/policy.d/
99control_plane_catch_all.rules ...
Processing rules in file /etc/cumulus/acl/policy.d/
99control_plane_catch_all.rules ...
Reading rule file /etc/cumulus/acl/policy.d/span_bond.rules ...
Processing rules in file /etc/cumulus/acl/policy.d/
span_bond.rules ...
Installing acl policy
done.
```

Verify that the SPAN rules are installed:

```
cumulus@switch:~$ sudo iptables -L -v | grep SPAN
    19  1938 SPAN       all  --  any    bond0
anywhere             anywhere             dport:bond1
```

Use the CPU port as the SPAN Destination

You can set the CPU port as a SPAN destination interface to mirror data plane traffic to the CPU. The SPAN traffic is sent to a separate network interface mirror where you can analyze it with `tcpdump`. This is a useful feature if you do not have any free external ports on the switch for monitoring purposes. SPAN traffic does not appear on switch ports.

Cumulus Linux controls how much traffic reaches the CPU so that mirrored traffic does not overwhelm the CPU.

> **NOTE**
>
> - CPU port as a SPAN destination interface is supported on Mellanox switches only.
> - Egress Mirroring for control plane generated traffic to the CPU port is not supported.

To use the CPU port as the SPAN destination, create a file in the `/etc/cumulus/acl/policy.d/` directory and add the rules. The following example rule matches on swp1 ingress traffic that has the source IP Address 10.10.1.1. When a match occurs, the traffic is mirrored to the CPU:

```
[iptables]
     -A FORWARD -i swp1 -s 10.10.1.1 -j SPAN --dport cpu
```

This example rule matches on swp1 egress traffic that has the source IP Address 10.10.1.1. When a match occurs, the traffic is is mirrored to the CPU:

```
[iptables]
     -A FORWARD -o swp1 -s 10.10.1.1 -j SPAN --dport cpu
```

Install the rules:

```
cumulus@switch:~$ sudo cl-acltool -i
```

You can use `tcpcdump` to monitor traffic mirrored to the CPU on the switch. You can also use filters for `tcpdump`. To use `tcpcdump` to monitor traffic mirrored to the CPU, run the following command:

```
cumulus@switch:~$ sudo tcpdump -i mirror
```

Configure ERSPAN

This section describes how to configure ERSPAN for all packets coming in from `swp1` to 12.0.0.2.

> **NOTE**
>
> Cut-through mode is **not** supported for ERSPAN in Cumulus Linux on switches using Broadcom Tomahawk, Trident II+, and Trident II ASICs.
>
> Cut-through mode **is** supported for ERSPAN in Cumulus Linux on switches using Mellanox Spectrum ASICs.

1. First, create a rules file in `/etc/cumulus/acl/policy.d/`:

```
cumulus@switch:~$ sudo bash -c 'cat <<EOF > /etc/cumulus/acl/
policy.d/erspan.rules
[iptables]
-A FORWARD --in-interface swp1 -j ERSPAN --src-ip 12.0.0.1 --
dst-ip 12.0.0.2  --ttl 64
EOF'
```

2. Install the rules:

```
cumulus@switch:~$ sudo cl-acltool -i
Reading rule file /etc/cumulus/acl/policy.d/
00control_plane.rules ...
Processing rules in file /etc/cumulus/acl/policy.d/
00control_plane.rules ...
Reading rule file /etc/cumulus/acl/policy.d/
99control_plane_catch_all.rules ...
Processing rules in file /etc/cumulus/acl/policy.d/
99control_plane_catch_all.rules ...
Reading rule file /etc/cumulus/acl/policy.d/erspan.rules ...
Processing rules in file /etc/cumulus/acl/policy.d/
erspan.rules ...
Installing acl policy
done.
```

3. Verify that the ERSPAN rules are installed:

```
cumulus@switch:~$ sudo iptables -L -v | grep SPAN
69  6804 ERSPAN     all  --  swp1   any
anywhere             anywhere             ERSPAN src-
ip:12.0.0.1 dst-ip:12.0.0.2
```

The `src-ip` option can be any IP address, whether it exists in the routing

table or not. The `dst-ip` option must be an IP address reachable via the routing table. The destination IP address must be reachable from a front-panel port, and not the management port. Use `ping` or `ip route get <ip>` to verify that the destination IP address is reachable. Setting the `--ttl` option is recommended.

> **TIP**
>
> If a SPAN destination IP address is not available, or if the interface type or types prevent using a laptop as a SPAN destination, read this knowledge base article for a workaround.

ERSPAN and Wireshark

- When using Wireshark to review the ERSPAN output, Wireshark may report the message “Unknown version, please report or test to use fake ERSPAN preference”, and the trace is unreadable. To resolve this, go into the General preferences for Wireshark, then go to **Protocols** > **ERSPAN** and check the **Force to decode fake ERSPAN frame** option.
- To set up a capture filter on the destination switch that filters for a specific IP protocol, use `ip.proto == 47` to filter for GRE-encapsulated (IP protocol 47) traffic.

Selective Spanning

SPAN and ERSPAN traffic rules can be configured to limit the traffic that is spanned, to reduce the volume of copied data.

> **NOTE**
>
> Cumulus Linux supports selective spanning for `iptables` only. `ip6tables` and `ebtables` are not supported.

The following matching fields are supported:

- IPv4 SIP/DIP
- IP protocol
- L4 (TCP/UDP) src/dst port
- TCP flags
- An ingress port/wildcard (swp+) can be specified in addition

> **NOTE**
>
> With ERSPAN, a maximum of two `--src-ip --dst-ip` pairs are supported. Exceeding this limit produces an error when you install the rules with `cl-acltool`.

SPAN Examples

To mirror forwarded packets from all ports matching SIP 20.0.1.0 and DIP 20.0.1.2 to port swp1s1:

```
-A FORWARD --in-interface swp+ -s 20.0.0.2 -d 20.0.1.2 -j SPAN
--dport swp1s2
```

To mirror icmp packets from all ports to swp1s2:

```
-A FORWARD --in-interface swp+ -s 20.0.0.2 -p icmp -j SPAN --
dport swp1s2
```

To mirror forwarded UDP packets received from port swp1s0, towards DIP 20.0.1.2 and destination port 53:

```
-A FORWARD --in-interface swp1s0 -d 20.0.1.2 -p udp --dport 53
-j SPAN --dport swp1s2
```

To mirror all forwarded TCP packets with only SYN set:

```
-A FORWARD --in-interface swp+ -p tcp --tcp-flags ALL SYN -j
SPAN --dport swp1s2
```

To mirror all forwarded TCP packets with only FIN set:

```
-A FORWARD --in-interface swp+ -p tcp --tcp-flags ALL FIN -j
SPAN --dport swp1s2
```

ERSPAN Examples

To mirror forwarded packets from all ports matching SIP 20.0.1.0 and DIP 20.0.1.2:

```
-A FORWARD --in-interface swp+ -s 20.0.0.2 -d 20.0.1.2 -j
ERSPAN --src-ip 90.0.0.1 --dst-ip 20.0.2.2
```

To mirror ICMP packets from all ports:

```
-A FORWARD --in-interface swp+ -s 20.0.0.2 -p icmp -j ERSPAN --
src-ip 90.0.0.1 --dst-ip 20.0.2.2
```

To mirror forwarded UDP packets received from port swp1s0, towards DIP 20.0.1.2 and destination port 53:

```
-A FORWARD --in-interface swp1s0 -d 20.0.1.2 -p udp --dport 53
-j ERSPAN --src-ip 90.0.0.1 --dst-ip 20.0.2.2
```

To mirror all forwarded TCP packets with only SYN set:

```
-A FORWARD --in-interface swp+ -p tcp --tcp-flags ALL SYN -j
ERSPAN --src-ip 90.0.0.1 --dst-ip 20.0.2.2
```

To mirror all forwarded TCP packets with only FIN set:

```
-A FORWARD --in-interface swp+ -p tcp --tcp-flags ALL FIN -j
ERSPAN --src-ip 90.0.0.1 --dst-ip 20.0.2.2
```

Remove SPAN Rules

To remove your SPAN rules, run:

```
#Remove rules file:
```

```
cumulus@switch:~$ sudo rm  /etc/cumulus/acl/policy.d/span.rules
#Reload the default rules
cumulus@switch:~$ sudo cl-acltool -i
cumulus@switch:~$
```

To verify that the SPAN rules were removed:

```
cumulus@switch:~$ sudo cl-acltool -L all | grep SPAN
cumulus@switch:~$
```

Monitor Control Plane Traffic with tcpdump

You can use `tcpdump` to monitor control plane traffic - traffic sent to and coming from the switch CPUs. `tcpdump` does **not** monitor data plane traffic; use `cl-acltool` instead (see above).

For more information on `tcpdump`, read the documentation and the man page.

The following example incorporates a few `tcpdump` options:

- `-i bond0`, which captures packets from bond0 to the CPU and from the CPU to bond0
- `host 169.254.0.2`, which filters for this IP address

- `-c 10`, which captures 10 packets then stops

```
cumulus@switch:~$ sudo tcpdump -i bond0 host 169.254.0.2 -c 10
tcpdump: WARNING: bond0: no IPv4 address assigned
tcpdump: verbose output suppressed, use -v or -vv for full
protocol decode
listening on bond0, link-type EN10MB (Ethernet), capture size
65535 bytes
16:24:42.532473 IP 169.254.0.2 > 169.254.0.1: ICMP echo
request, id 27785, seq 6, length 64
16:24:42.532534 IP 169.254.0.1 > 169.254.0.2: ICMP echo reply,
id 27785, seq 6, length 64
16:24:42.804155 IP 169.254.0.2.40210 > 169.254.0.1.5342: Flags
[.], seq 266275591:266277039, ack 3813627681, win 58, options
[nop,nop,TS val 590400681 ecr 530346691], length 1448
16:24:42.804228 IP 169.254.0.1.5342 > 169.254.0.2.40210: Flags
[.], ack 1448, win 166, options [nop,nop,TS val 530348721 ecr
590400681], length 0
16:24:42.804267 IP 169.254.0.2.40210 > 169.254.0.1.5342: Flags
[P.], seq 1448:1836, ack 1, win 58, options [nop,nop,TS val
590400681 ecr 530346691], length 388
16:24:42.804293 IP 169.254.0.1.5342 > 169.254.0.2.40210: Flags
[.], ack 1836, win 165, options [nop,nop,TS val 530348721 ecr
590400681], length 0
```

```
16:24:43.532389 IP 169.254.0.2 > 169.254.0.1: ICMP echo
request, id 27785, seq 7, length 64
16:24:43.532447 IP 169.254.0.1 > 169.254.0.2: ICMP echo reply,
id 27785, seq 7, length 64
16:24:43.838652 IP 169.254.0.1.59951 > 169.254.0.2.5342: Flags
[.], seq 2555144343:2555145791, ack 2067274882, win 58, options
[nop,nop,TS val 530349755 ecr 590399688], length 1448
16:24:43.838692 IP 169.254.0.1.59951 > 169.254.0.2.5342: Flags
[P.], seq 1448:1838, ack 1, win 58, options [nop,nop,TS val
530349755 ecr 590399688], length 390
10 packets captured
12 packets received by filter
0 packets dropped by kernel
```

Related Information

- Wikipedia page on ping
- Wikipedia page on traceroute
- `tcpdump` website

Using NCLU to Troubleshoot Your Network Configuration

The Network Command Line Utility (NCLU) can quickly return a lot of information about your network configuration.

net show Commands

Running `net show` and pressing TAB displays all available command line arguments usable by `net`. The output looks like this:

```
cumulus@switch:~$ net show <TAB>
bfd             :  Bidirectional forwarding detection
bgp             :  Border Gateway Protocol
bridge          :  a layer2 bridge
clag            :  Multi-Chassis Link Aggregation
commit          :  apply the commit buffer to the system
configuration   :  settings, configuration state, etc
counters        :  net show counters
debugs          :  Debugs
dot1x           :  Configure, Enable, Delete or Show IEEE 802.1X
EAPOL
evpn            :  Ethernet VPN
hostname        :  local hostname
```

```
igmp            :  Internet Group Management Protocol
interface       :  An interface, such as swp1, swp2, etc.
ip              :  Internet Protocol version 4/6
ipv6            :  Internet Protocol version 6
lldp            :  Link Layer Discovery Protocol
mpls            :  Multiprotocol Label Switching
mroute          :  Static unicast routes in MRIB for multicast
RPF lookup
msdp            :  Multicast Source Discovery Protocol
ospf            :  Open Shortest Path First (OSPFv2)
ospf6           :  Open Shortest Path First (OSPFv3)
package         :  A Cumulus Linux package name
pbr             :  Policy Based Routing
pim             :  Protocol Independent Multicast
ptp             :  Precision Time Protocol
rollback        :  revert to a previous configuration state
route           :  Static routes
route-map       :  Route-map
snmp-server     :  Configure the SNMP server
system          :  System information
time            :  Time
version         :  Version number
vrf             :  Virtual Routing and Forwarding
vrrp            :  Virtual Router Redundancy Protocol
```

Show Interfaces

To show all available interfaces that are physically UP, run `net show interface`:

```
cumulus@switch:~$ net show interface

    Name    Speed    MTU    Mode           Summary
--  ------  -------  -----  -------------
---------------------------------------
UP  lo      N/A      65536  Loopback       IP: 10.0.0.11/32,
127.0.0.1/8, ::1/128
UP  eth0    1G       1500   Mgmt           IP: 192.168.0.11/
24(DHCP)
UP  swp1    1G       1500   Access/L2      Untagged: br0
UP  swp2    1G       1500   NotConfigured
UP  swp51   1G       1500   NotConfigured
UP  swp52   1G       1500   NotConfigured
UP  blue    N/A      65536  NotConfigured
UP  br0     N/A      1500   Bridge/L3      IP: 172.16.1.1/24
                                           Untagged Members:
swp1
                                           802.1q Tag: Untagged
                                           STP:
```

```
RootSwitch(32768)
UP  red     N/A      65536  NotConfigured
```

To show every interface regardless of state, run `net show interface all`:

```
cumulus@leaf01:~$ net show interface all
State  Name     Spd  MTU    Mode
LLDP                    Summary
-----  -------  ---  -----  -------------
----------------------  -------------------------
UP     lo       N/A  65536
Loopback                                IP: 127.0.0.1/8


lo                                                          IP:
10.0.0.11/32


lo                                                          IP:
::1/128
UP     eth0     1G   1500   Mgmt           oob-mgmt-switch
(swp6)  IP: 192.168.0.11/24(DHCP)
UP     swp1     1G   1500   Access/L2      server01
(eth1)         Master: br0(UP)
ADMDN  swp2     N/A  1500   NotConfigured
```

```
ADMDN  swp45    N/A  1500   NotConfigured
ADMDN  swp46    N/A  1500   NotConfigured
ADMDN  swp47    N/A  1500   NotConfigured
ADMDN  swp48    N/A  1500   NotConfigured
ADMDN  swp49    N/A  1500   NotConfigured
ADMDN  swp50    N/A  1500   NotConfigured
UP     swp51    1G   1500   Default        spine01 (swp1)
UP     swp52    1G   1500   Default        spine02 (swp1)
UP     br0      N/A  1500   Bridge/
L3                              IP: 172.16.1.1/24
ADMDN  vagrant  N/A  1500   NotConfigured
```

To get information about the switch itself, run `net show system`:

```
cumulus@switch:~$ net show system
Hostname......... celRED

Build............ Cumulus Linux 4.1.0
Uptime........... 8 days, 12:24:01.770000

Model............ Cel REDSTONE
CPU.............. x86_64 Intel Atom C2538 2.4 GHz
Memory........... 4GB
```

```
Disk.............. 14.9GB
ASIC.............. Broadcom Trident2 BCM56854
Ports............. 48 x 10G-SFP+ & 6 x 40G-QSFP+
Base MAC Address. a0:00:00:00:00:50
Serial Number.... A1010B2A011212AB000001
```

network-docopt Package

NCLU uses the `python network-docopt` package. This is inspired by `docopt` and enables you to specify partial commands without tab completion or running the complete option. For example, `net show int` runs the `net show interface` command and `net show sys` runs the `net show system` command.

Mellanox What Just Happened (WJH)

Cumulus Linux supports the *What Just Happened* (WJH) feature for Mellanox switches to stream detailed and contextual telemetry for off-box analysis with tools, such as Cumulus NetQ. This advanced streaming telemetry technology provides real time visibility into problems in the network, such as hardware packet drops due to buffer congestion, incorrect routing, ACL or layer 1 problems.

When WJH capabilities are combined with the analytics engine of Cumulus NetQ, you have the ability to hone in on any loss, anywhere in the fabric, from a single management console. You can view any current or historic drops and specific drop reasons, and also identify any flow or endpoints and pin-point exactly where communication is failing in the network.

WJH is enabled by default on a Mellanox switch; no configuration is required in Cumulus Linux.

Monitoring System Statistics and Network Traffic with sFlow

sFlow is a monitoring protocol that samples network packets, application operations, and system counters. sFlow collects both interface counters and sampled 5-tuple packet information, so that you can monitor your network traffic as well as your switch state and performance metrics. An outside server, known as an *sFlow collector*, is required to collect and analyze this data.

`hsflowd` is the daemon that samples and sends sFlow data to configured collectors. By default, `hsflowd` is disabled and does *not* start automatically when the switch boots up.

NOTE

- sFlow is not supported on Broadcom switches with the Hurricane2 ASIC.
- If you intend to run this service within a VRF, including the management VRF, follow these steps for configuring the service.

Configure sFlow

To configure `hsflowd` to send to the designated collectors, either:

- Use DNS service discovery (DNS-SD)
- Manually configure the `/etc/hsflowd.conf` file

Configure sFlow with DNS-SD

You can configure your DNS zone to advertise the collectors and polling information to all interested clients.

Add the following content to the zone file on your DNS server:

```
_sflow._udp SRV 0 0 6343 collector1
_sflow._udp SRV 0 0 6344 collector2
_sflow._udp TXT (
"txtvers=1"
"sampling.100M=100"
"sampling.1G=1000"
"sampling.10G=10000"
"sampling.40G=40000"
"sampling.100G=100000"
"polling=20"
)
```

The above snippet instructs `hsflowd` to send sFlow data to collector1 on port 6343 and to collector2 on port 6344. `hsflowd` will poll counters every 20 seconds and sample 1 out of every 2048 packets.

NOTE

The maximum samples per second delivered from the hardware is limited to 16K. You can configure the number of samples per second in the `/etc/cumulus/datapath/traffic.conf` file, as shown below:

```
# Set sflow/sample ingress cpu packet rate and burst in
packets/sec
# Values: {0..16384}
#sflow.rate = 16384
#sflow.burst = 16384
```

Start the sFlow daemon:

```
cumulus@switch:~$ sudo systemctl start hsflowd.service
```

No additional configuration is required in the `/etc/hsflowd.conf` file.

Manually Configure /etc/hsflowd.conf

You can set up the collectors and variables on each switch.

Edit the `/etc/hsflowd.conf` file to set up your collectors and sampling rates in `/etc/hsflowd.conf`. For example:

```
sflow {
# ====== Sampling/Polling/Collectors ======
  # EITHER: automatic (DNS SRV+TXT from _sflow._udp):
  #   DNS-SD { }
  # OR: manual:
  #   Counter Polling:
        polling = 20
  #   default sampling N:
  #     sampling = 400
  #   sampling N on interfaces with ifSpeed:
        sampling.100M = 100
        sampling.1G = 1000
        sampling.10G = 10000
        sampling.40G = 40000
  #   sampling N for apache, nginx:
  #     sampling.http = 50
  #     sampling N for application (requires json):
  #     sampling.app.myapp = 100
```

```
  #   collectors:
  collector { ip=192.0.2.100 udpport=6343 }
  collector { ip=192.0.2.200 udpport=6344 }
}
```

This configuration polls the counters every 20 seconds, samples 1 of every 40000 packets for 40G interfaces, and sends this information to a collector at 192.0.2.100 on port 6343 and to another collector at 192.0.2.200 on port 6344.

> **NOTE**
>
> Some collectors require each source to transmit on a different port, others listen on only one port. Refer to the documentation for your collector for more information.

To configure the IP address for the sFlow agent, configure one of the following the `/etc/hsflowd.conf` file (following the recommendations in the sFlow documentation):

- The agent CIDR. For example, `agent.cidr = 10.0.0.0/8`. The IP address should fall within this range.
- The agent interface. For example, if the agent is using eth0, select the IP address associated with this interface.

You can check to see which agent IP was selected using:

```
cumulus@switch:~$ grep agentIP /etc/hsflowd.auto
```

Configure sFlow Visualization Tools

For information on configuring various sFlow visualization tools, read this knowledge base article.

Considerations

The EdgeCore AS4610 switch occasionally sends malformed packets and does not send any flow samples; it sends only counters. This is a known limitation on this Helix4 platform.

Related Information

- sFlow Collectors
- sFlow Wikipedia page

Simple Network Management Protocol - SNMP

SNMP is an IETF standards-based network management architecture and protocol. Cumulus Linux uses the open source Net-SNMP agent `snmpd` version 5.8.1.pre1, which provides support for most of the common industry-wide MIBs, including interface counters and TCP/UDP IP stack data. The version in Cumulus Linux adds custom MIBs and pass-through and pass-persist scripts.

SNMP Components

The main components of SNMP in Cumulus Linux are:

- SNMP network management system (NMS)
- SNMP agents
- The MIBs (management information bases)

SNMP Network Management System

An SNMP network management system (NMS) is a system configured to poll SNMP agents (such as Cumulus Linux switches or routers) that can send query requests to SNMP agents with the correct credentials. The managers poll the agents and the agents respond with the data. There are a variety of command line tools for polling, including `snmpget`, `snmpgetnext`, `snmpwalk`, `snmpbulkget`, and `snmpbulkwalk`. SNMP agents can also send unsolicited traps and inform messages to the NMS based on predefined

criteria, like link changes.

SNMP Agent

The SNMP agent (the `snmpd` daemon) running on a Cumulus Linux switch gathers information about the local system and stores the data in a *management information base*, or MIB. Parts of the MIB tree are available and provided to incoming requests originating from an NMS host that has authenticated with the correct credentials. You can configure the Cumulus Linux switch with usernames and credentials to provide authenticated and encrypted responses to NMS requests. The `snmpd` agent can also proxy requests and act as a *master agent* to sub-agents running on other daemons, like for FRR or LLDP.

Management Information Base (MIB)

The MIB is a database for the `snmpd` daemon that runs on the agent. MIBs adhere to IETF standards but are flexible enough to allow vendor-specific additions. Cumulus Linux includes a number of custom enterprise MIB tables, which are defined in a set text files on the switch; the files are located in `/usr/share/snmp/mibs/` and their names all start with *Cumulus*. They include:

- Cumulus-Counters-MIB.txt
- Cumulus-POE-MIB.txt
- Cumulus-Resource-Query-MIB.txt
- Cumulus-Snmp-MIB.txt

The MIB is structured as a top-down hierarchical tree. Each branch that

forks off is labeled with both an identifying number (starting with 1) and an identifying string that is unique for that level of the hierarchy. The strings and numbers can be used interchangeably. The parent IDs (numbers or strings) are strung together, starting with the most general to form an address for the MIB object. Each junction in the hierarchy is represented by a dot in this notation so that the address ends up being a series of ID strings or numbers separated by dots. This entire address is known as an object identifier (OID).

You can use various online and command line tools to translate between numbers and strings and to also provide definitions for the various MIB objects. For example, you can view the *sysLocation* object (which is defined in `SNMPv2-MIB.txt`) in the system table as either a series of numbers *1.3.6.1.2.1.1.6* or as the string *iso.org.dod.internet.mgmt.mib-2.system.sysLocation*. You view the definition with the `snmptranslate` command, which is part of the `snmp` Debian package in Cumulus Linux.

```
cumulus@switch:~$ snmptranslate -Td -On SNMPv2-MIB::sysLocation
.1.3.6.1.2.1.1.6
sysLocation OBJECT-TYPE
  -- FROM       SNMPv2-MIB
  -- TEXTUAL CONVENTION DisplayString
  SYNTAX        OCTET STRING (0..255)
  DISPLAY-HINT  "255a"
```

```
  MAX-ACCESS    read-write
  STATUS        current
  DESCRIPTION   "The physical location of this node (e.g.,
'telephone
            closet, 3rd floor').  If the location is unknown,
the
            value is the zero-length string."
::= { iso(1) org(3) dod(6) internet(1) mgmt(2) mib-2(1)
system(1) 6 }
```

In the last line above, the section *1.3.6.1* or *iso.org.dod.internet* is the OID that defines internet resources. The *2* or *mgmt* that follows is for a management subcategory. The *1* or *mib-2* under that defines the MIB-2 specification. And finally, the *1* or *system* is the parent for a number of child objects *sysDescr*, *sysObjectID*, *sysUpTime*, *sysContact*, *sysName*, *sysLocation*, *sysServices*, and so on, as seen in the tree output from the second `snmptranslate` command below, where *sysLocation* is defined as *6*.

```
cumulus@leaf01:mgmt:~$ snmptranslate -Tp -IR system
+--system(1)
   |
   +-- -R-- String    sysDescr(1)
   |        Textual Convention: DisplayString
```

```
|        Size: 0..255
+-- -R-- ObjID     sysObjectID(2)
+-- -R-- TimeTicks sysUpTime(3)
|  |
|  +--sysUpTimeInstance(0)
|
+-- -RW- String    sysContact(4)
|        Textual Convention: DisplayString
|        Size: 0..255
+-- -RW- String    sysName(5)
|        Textual Convention: DisplayString
|        Size: 0..255
+-- -RW- String    sysLocation(6)
|        Textual Convention: DisplayString
|        Size: 0..255
+-- -R-- INTEGER   sysServices(7)
|        Range: 0..127
+-- -R-- TimeTicks sysORLastChange(8)
|        Textual Convention: TimeStamp
|
+--sysORTable(9)
   |
   +--sysOREntry(1)
      |  Index: sysORIndex
```

```
        |
        +-- ---- INTEGER   sysORIndex(1)
        |        Range: 1..2147483647
        +-- -R-- ObjID     sysORID(2)
        +-- -R-- String    sysORDescr(3)
        |        Textual Convention: DisplayString
        |        Size: 0..255
        +-- -R-- TimeTicks sysORUpTime(4)
                 Textual Convention: TimeStamp
```

Configure SNMP

The most basic SNMP configuration requires you to specify:

- One or more IP addresses on which the SNMP agent listens.
- Either a username (for SNMPv3) or a read-only community string (a password, for SNMPv1 or SNMPv2c).

By default, the SNMP configuration has a listening address of localhost (127.0.0.1), which allows the agent (the `snmpd` daemon) to respond to SNMP requests originating on the switch itself. This is a secure method that allows checking the SNMP configuration without exposing the switch to outside attacks. In order for an external SNMP NMS to poll a Cumulus Linux switch, you must configure the `snmpd` daemon running on the switch to listen to one or more IP addresses on interfaces that have a link state UP.

The SNMPv3 username is the recommended option instead of the read-only community name, as it is more secure; it does not expose the user credentials and can also encrypt packet contents. However, a read-only community password is required for SNMPv1 or SNMPv2c environments so that the `snmpd` daemon can respond to requests. The read-only community string allows polling of the various MIB objects on the device itself.

Start the SNMP Daemon

Before you can use SNMP, you need to enable and start the `snmpd` service.

> **NOTE**
>
> If you intend to run this service within a VRF, including the management VRF, follow these steps for configuring the service.

To start the SNMP daemon:

1. Start the `snmpd` daemon:

```
cumulus@switch:~$ sudo systemctl start snmpd.service
```

2. Enable the `snmpd` daemon to start automatically after reboot:

```
cumulus@switch:~$ sudo systemctl enable snmpd.service
```

3. To enable `snmpd` to restart automatically after failure, create a file called `/etc/systemd/system/snmpd.service.d/restart.conf` and add the following lines:

```
[Service]
```

```
Restart=always
RestartSec=60
```

4. Run the `sudo systemctl daemon-reload` command.

After the service starts, you can use SNMP to manage various components on the switch.

Configure SNMP

Cumulus Networks recommends that you use NCLU to configure `snmpd` even though NCLU does not provide functionality to configure every `snmpd` feature. You are not restricted to using NCLU for configuration and can edit the `/etc/snmp/snmpd.conf` file and control `snmpd` with `systemctl` commands.

IMPORTANT

If you need to manually edit the SNMP configuration — for example, if the necessary option has not been implemented in NCLU — you need to edit the configuration directly in the `/etc/snmp/snmpd.conf` file.

Use caution when editing this file. Be aware that `snmpd` caches SNMPv3 usernames and passwords in the `/var/lib/snmp/snmpd.conf` file. Make sure you stop `snmpd` and remove the old entries when making changes. Otherwise, Cumulus Linux uses the old usernames and passwords in the `/var/lib/snmp/snmpd.conf` file instead of the ones in the `/etc/snmp/snmpd.conf` file.

The next time you use NCLU to update your SNMP configuration, if NCLU is unable to correctly parse the syntax, some of the options might be overwritten.

Make sure you do not delete the `snmpd.conf` file; this can cause issues with the package manager the next time you update Cumulus Linux.

The `snmpd` daemon uses the `/etc/snmp/snmpd.conf` configuration file for most of its configuration. The syntax of the most important keywords are defined in the following table.

Configure the Listening IP Addresses

For security reasons, the listening address is set to the localhost by default

so that the SNMP agent only responds to requests originating on the switch itself. You can also configure listening only on the IPv6 localhost address. When using IPv6 addresses or localhost, you can use a `readonly-community-v6` for SNMPv1 and SNMPv2c requests. For SNMPv3 requests, you can use the `username` command to restrict access. See Configure the SNMPv3 Username below.

The IP address must exist on an interface that has link UP on the switch where `snmpd` is being used. By default, this is set to `udp:127.0.0.1:161`, so `snmpd` only responds to requests (such as `snmpwalk`, `snmpget`, `snmpgetnext`) originating from the switch. A wildcard setting of *udp:161,udp6:161* forces `snmpd` to listen on all IPv4 and IPv6 interfaces for incoming SNMP requests.

You can configure multiple IP addresses and bind to a particular IP address within a particular VRF table.

NCLU Commands | Linux Commands

To configure the `snmpd` daemon to listen on the localhost IPv4 and IPv6 interfaces, run:

```
cumulus@switch:~$ net add snmp-server listening-address
localhost
cumulus@switch:~$ net add snmp-server listening-address
localhost-v6
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

TIP

If you configure the listening address on the loopback interface, since it is not a change from the default, a message appears in the console stating that the configuration has not changed.

```
cumulus@switch:~$ net add snmp-server listening-
address localhost
Cannot add 127.0.0.1. It is already a listener-
address
The configuration has not changed.
```


To configure the `snmpd` daemon to listen on all interfaces for either IPv4

SNMP and VRFs

Cumulus Linux provides a listening address for VRFs along with trap and inform support. You can configure `snmpd` to listen to a specific IPv4 or IPv6 address on an interface within a particular VRF. With VRFs, identical IP addresses can exist in different VRF tables. This command restricts listening to a particular IP address within a particular VRF. If the VRF name is not given, the default VRF is used.

NCLU Commands | Linux Commands

The following command configures `snmpd` to listen to IP address 10.10.10.10 on eth0, the management interface in the management VRF:

```
cumulus@switch:~$ net add snmp-server listening-address
10.10.10.10 vrf mgmt
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

By default, `snmpd` does not cross VRF table boundaries. To listen on IP addresses in different VRF tables, use multiple listening-address commands each with a VRF name, as shown below.

```
cumulus@switch:~$ net add snmp-server listening-address
10.10.10.10 vrf rocket
cumulus@switch:~$ net add snmp-server listening-address
10.10.10.20 vrf turtle
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Configure the SNMPv3 Username

As mentioned above, Cumulus Networks recommends you use an SNMPv3 username and password instead of the read-only community string as the more secure way to use SNMP, since SNMPv3 does not expose the password in the `GetRequest` and `GetResponse` packets and can also encrypt packet contents. You can configure multiple usernames for different user roles with different levels of access to various MIBs.

SNMPv3 usernames are added to the `/etc/snmp/snmpd.conf` file, along with plaintext authentication and encryption pass phrases.

> **NOTE**
>
> The default `snmpd.conf` file contains a default user, *_snmptrapusernameX*. This username cannot be used for authentication, but is required for SNMP traps.

You have three choices for authenticating the user:

- No authentication password (if you specify `auth-none`)
- MD5 password
- SHA password

NCLU Commands | Linux Commands

For no authentication, run:

```
cumulus@switch:~$ net add snmp-server username
testusernoauth auth-none
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

For MD5 authentication, run:

```
cumulus@switch:~$ net add snmp-server username testuserauth
auth-md5 myauthmd5password
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

For SHA authentication, run:

```
cumulus@switch:~$ net add snmp-server username limiteduser1
auth-sha SHApassword1
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

If you specify MD5 or SHA authentication, you can also specify an AES or DES encryption password to encrypt the contents of the request and response packets.

Configure an SNMP View Definition

To restrict MIB tree exposure, you can define a view for an SNMPv3 username or community password, and a host from a restricted subnet. In doing so, any SNMP request with that username and password must have a source IP address within the configured subnet.

You can define a specific view multiple times and fine tune to provide or restrict access using the `included` or `excluded` command to specify branches of certain MIB trees.

By default, the `snmpd.conf` file contains numerous views within the *systemonly* view.

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add snmp-server viewname cumulusOnly
included .1.3.6.1.4.1.40310
cumulus@switch:~$ net add snmp-server viewname
cumulusCounters included .1.3.6.1.4.1.40310.2
cumulus@switch:~$ net add snmp-server readonly-community
simplepassword access any view cumulusOnly
cumulus@switch:~$ net add snmp-server username
testusernoauth auth-none view cumulusOnly
cumulus@switch:~$ net add snmp-server username limiteduser1
auth-md5 md5password1 encrypt-aes myaessecret view
cumulusCounters
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Configure the Community String

The `snmpd` authentication for SNMPv1 and SNMPv2c is disabled by default in Cumulus Linux. You enable it by providing a password (called a community string) for SNMPv1 or SNMPv2c environments so that the `snmpd` daemon can respond to requests. By default, this provides access to the full OID tree for such requests, regardless of from where they were sent. No default password is set, so `snmpd` does not respond to any requests that arrive unless you set the read-only community password.

For SNMPv1 and SNMPv2c you can specify a read-only community string. For SNMPv3, you can specify a read-only or a read-write community string (provided you are not using the preferred username method described above), but you must configure the read-write community string directly in the `snmpd.conf` file; you cannot use NCLU to configure it. If you configure a read-write community string, then edit the SNMP configuration later with NCLU, the read-write community configuration is preserved.

You can specify a source IP address token to restrict access to only that host or network given.

You can also specify a view to restrict the subset of the OID tree.

NCLU Commands | Linux Commands

The following example configuration:

- Sets the read only community string to *simplepassword* for SNMP requests
- Restricts requests to only those sourced from hosts in the 192.168.200.10/24 subnet
- Restricts viewing to the *mysystem* view defined with the `viewname` command

```
cumulus@switch:~$ net add snmp-server viewname mysystem
included 1.3.6.1.2.1.1
cumulus@switch:~$ net add snmp-server readonly-community
simplepassword access 192.168.200.10/24 view mysystem
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

This example creates a read-only community password *showitall* that allows access to the entire OID tree for requests originating from any source IP address.

```
cumulus@switch:~$ net add snmp-server readonly-community
showitall access any
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Configure System Settings

You can configure system settings for the SNMPv2 MIB. The example commands here set:

- The system physical location for the node in the SNMPv2-MIB system table (the `syslocation`).
- The username and email address of the contact person for this managed node (the `syscontact`).
- An administratively-assigned name for the managed node (the `sysname`).

NCLU Commands | Linux Commands

For example, to set the system physical location for the node in the SNMPv2-MIB system table, run:

```
cumulus@switch:~$ net add snmp-server system-location My
private bunker
cumulus@switch:~$ net commit
```

To set the username and email address of the contact person for this managed node, run:

```
cumulus@switch:~$ net add snmp-server system-contact user X
at myemail@example.com
cumulus@switch:~$ net commit
```

To set an administratively-assigned name for the managed node, run the following command. Typically, this is the fully-qualified domain name of the node.

```
cumulus@switch:~$ net add snmp-server system-name
CumulusBox number 1,543,567
cumulus@switch:~$ net commit
```

These commands append the following content to the `/etc/snmp/snmpd.conf` file:

Enable SNMP Support for FRRouting

SNMP supports routing MIBs in FRRouting. To enable SNMP support for FRRouting, you need to configure AgentX (ASX) access in FRR.

The default `/etc/snmp/snmpd.conf` configuration already enables AgentX and sets the correct permissions.

Enabling FRR includes support for BGP. However, if you plan on using the BGP4 MIB, be sure to provide access to the MIB tree 1.3.6.1.2.1.15.

> **NOTE**
>
> At this time, SNMP does not support monitoring BGP unnumbered neighbors.

> **TIP**
>
> If you plan on using the OSPFv2 MIB, provide access to 1.3.6.1.2.1.14 and to 1.3.6.1.2.1.191 for the OSPv3 MIB.

To enable SNMP support for FRR:

1. Configure AgentX access in FRR:

```
cumulus@switch:~$ net add routing agentx
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

2. Update the SNMP configuration to enable FRR to respond to SNMP requests. Open the `/etc/snmp/snmpd.conf` file in a text editor and verify that the following configuration exists:

```
agentxsocket /var/agentx/master
agentxperms 777 777 snmp snmp
master agentx
```

> **NOTE**
>
> Make sure that the `/var/agentx` directory is world-readable and world-searchable (octal mode 755).
>
> ```
> cumulus@switch:~$ ls -la /var/
> ...
> ```

```
drwxr-xr-x  2 root root  4096 Nov 11 12:06 agentx
...
```

3. Optionally, you might need to expose various MIBs:

   - For the BGP4 MIB, allow access to `1.3.6.1.2.1.15`
   - For the OSPF MIB, allow access to `1.3.6.1.2.1.14`
   - For the OSPFV3 MIB, allow access to `1.3.6.1.2.1.191`

To verify the configuration, run `snmpwalk`. For example, if you have a running OSPF configuration with routes, you can check this OSPF-MIB first from the switch itself with:

```
cumulus@switch:~$ sudo snmpwalk -v2c -cpublic localhost
1.3.6.1.2.1.14
```

Enable the .1.3.6.1.2.1 Range

Some MIBs, including storage information, are not included by default in `snmpd.conf` in Cumulus Linux. This results in some default views on common network tools (like `librenms`) to return less than optimal data. You can

include more MIBs by enabling the complete .1.3.6.1.2.1 range. This simplifies the configuration file, removing the concern that any required MIBs might be missed by the monitoring system. Various MIBs included were added to the default SNMPv3 configuration and include the following:

- ENTITY-MIB
- ENTITY-SENSOR MIB
- Parts of the BRIDGE-MIB and Q-BRIDGE-MIBs

WARNING

This configuration grants access to a large number of MIBs, including all SNMPv2-MIB, which might reveal more data than expected. In addition to being a security vulnerability, it might consume more CPU resources.

To enable the .1.3.6.1.2.1 range, make sure the view commands include the required MIB objects.

Restore the Default SNMP Configuration

The following command removes all custom entries in the `/etc/snmp/snmpd.conf` file and replaces them with defaults, including for all SNMPv3 usernames and readonly-communities. A `listening-address` for the

localhost is configured in its place.

```
cumulus@switch:~$ net del snmp-server all
cumulus@switch:~$ net commit
```

Set up the Custom Cumulus Networks MIBs on the NMS

No changes are required in the `/etc/snmp/snmpd.conf` file on the switch to support the custom Cumulus Networks MIBs. The following lines are already included by default and provide support for both the Cumulus Counters and the Cumulus Resource Query MIBs.

```
cumulus@switch:~$ cat /etc/snmp/snmpd.conf
...
sysObjectID 1.3.6.1.4.1.40310
pass_persist .1.3.6.1.4.1.40310.1 /usr/share/snmp/resq_pp.py
pass_persist .1.3.6.1.4.1.40310.2 /usr/share/snmp/
cl_drop_cntrs_pp.py
...
```

However, you need to copy several files to the NMS server for the custom Cumulus MIB to be recognized on the NMS server.

- `/usr/share/snmp/mibs/Cumulus-Snmp-MIB.txt`
- `/usr/share/snmp/mibs/Cumulus-Counters-MIB.txt`
- `/usr/share/snmp/mibs/Cumulus-Resource-Query-MIB.txt`

Pass Persist Scripts

The pass persist scripts in Cumulus Linux use the pass_persist extension to Net-SNMP. The scripts are stored in `/usr/share/snmp` and include:

- `bgp4_pp.py`
- `bridge_pp.py`
- `cl_drop_cntrs_pp.py`
- `cl_poe_pp.py`
- `entity_pp.py`
- `entity_sensor_pp.py`
- `ieee8023_lag_pp.py`
- `resq_pp.py`
- `snmpifAlias_pp.py`
- `sysDescr_pass.py`

All the scripts are enabled by default in Cumulus Linux, except for:

- `bgp4_pp.py`, which is handled by FRRouting.
- `cl_poe_pp.py`, which is disabled by default as only certain platforms that Cumulus Linux supports are capable of doing Power over Ethernet.

Example Configuration

The following example configuration:

- Enables an SNMP agent to listen on all IPv4 addresses with a community string password.
- Sets the trap destination host IP address.
- Creates four types of SNMP traps.

You can find a working example configuration on the NVIDIA Networking GitLab project, which you can try for free with Cumulus AIR.

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add snmp-server listening-address all
cumulus@switch:~$ net add snmp-server readonly-community
tempPassword access any
cumulus@switch:~$ net add snmp-server trap-destination
1.1.1.1 community-password mypass version 2c
cumulus@switch:~$ net add snmp-server trap-link-up check-
frequency 15
cumulus@switch:~$ net add snmp-server trap-link-down check-
frequency 10
cumulus@switch:~$ net add snmp-server trap-cpu-load-average
one-minute 7.45 five-minute 5.14
cumulus@switch:~$ net add snmp-server trap-snmp-auth-
failures
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following `/etc/snmp/snmpd.conf` file:

```
cumulus@switch:~$ sudo nano /etc/snmp/snmpd.conf
agentaddress udp:161
agentxperms 777 777 snmp snmp
agentxsocket /var/agentx/master
+authtrapenable 1
createuser _snmptrapusernameX
iquerysecname _snmptrapusernameX
load 7.45 5.14 0
master agentx
monitor -r 60 -o laNames -o laErrMessage "laTable"
```

Configure SNMP Traps

SNMP *traps* are alert notification messages sent from SNMP agents to the SNMP manager. These messages are generated whenever any failure or fault occurs in a monitored device or service. An SNMPv3 inform is an acknowledged SNMPv3 trap.

You configure the following for SNMPv3 trap and inform messages:

- The trap destination IP address; the VRF name is optional.
- The authentication type and password. The encryption type and password are optional.
- The engine ID/username pair for the Cumulus Linux switch sending the traps. The *inform* keyword specifies an inform message where the SNMP agent waits for an acknowledgement. You can find this at the end of the `/var/lib/snmp/snmpd.conf` file labeled *oldEngineID*. Configure this same engine ID/username (with authentication and encryption passwords) for the trap daemon receiving the trap to validate the received trap.

Generate Event Notification Traps

The Net-SNMP agent provides a method to generate SNMP trap events using the Distributed Management (DisMan) Event MIB for various system events, including:

- Link up/down.
- Exceeding the temperature sensor threshold, CPU load, or memory threshold.

- Other SNMP MIBs.

To enable specific types of traps, create the following configurations in `/etc/snmp/snmpd.conf`.

Define Access Credentials

Although the traps are sent to an SNMPv2c receiver, the SNMPv3 username is still required to authorize the DisMan service. Starting with Net-SNMP 5.3, `snmptrapd` no longer accepts all traps by default. `snmptrapd` must be configured with authorized SNMPv1/v2c community strings and/or SNMPv3 users. Non-authorized traps/informs are dropped.

Follow the steps in Configure SNMP to define the username. You can refer to the snmptrapd.conf(5) manual page for more information.

> **NOTE**
>
> You may need to install the `snmptrapd` Debian package before you can configure the username.
>
> ```
> cumulus@switch:~$ sudo apt-get install snmptrapd
> ```

Define Trap Receivers

The following configuration defines the trap receiver IP address where

SNMPv1 and SNMPv2c traps are sent. For SNMP versions 1 and 2c, you must set at least one SNMP trap destination IP address; multiple destinations can exist. Removing all settings disables SNMP traps. The default version is 2c, unless otherwise configured. You must include a VRF name with the IP address to force traps to be sent in a non-default VRF table.

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add snmp-server trap-destination
localhost vrf rocket community-password
mymanagementvrfpassword version 1
cumulus@switch:~$ net add snmp-server trap-destination
localhost-v6 community-password mynotsosecretpassword
version 2c
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following configuration in the `/etc/snmp/snmpd.conf` file:

```
cumulus@switch:~$ cat /etc/snmp/snmpd.conf
...
trap2sink [::1] mynotsosecretpassword
trapsink 127.0.0.1@rocket mymanagementvrfpassword
...
```

SNMPv3 Trap and Inform Messages

The SNMP trap receiving daemon must have usernames, authentication passwords, and encryption passwords created with its own EngineID. You

must configure this trap server EngineID in the switch `snmpd` daemon sending the trap and inform messages. You specify the level of authentication and encryption for SNMPv3 trap and inform messages with `-l` (`NoauthNoPriv, authNoPriv,` or `authPriv`).

NCLU Commands | Linux Commands

For inform messages, the engine ID/username creates the username on the receiving trap daemon server. The trap receiver sends the response for the trap message using its own engine ID/username. In practice, the trap daemon generates the usernames with its own engine ID and after these are created, the SNMP server (or agent) needs to use these engine ID/usernames when configuring the inform messages so that they are correctly authenticated and the correct response is sent to the `snmpd` agent that sent it.

```
cumulus@switch:~$ net add snmp-server trap-destination
localhost username myv3user auth-md5 md5password1 encrypt-
aes myaessecret engine-id
0x80001f888070939b14a514da5a00000000 inform
cumulus@switch:~$ net add snmp-server trap-destination
localhost vrf mgmt username mymgmtvrfusername auth-md5
md5password2 encrypt-aes myaessecret2 engine-id
0x80001f888070939b14a514da5a00000000 inform
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Source Traps from a Different Source IP Address

When client SNMP programs (such as `snmpget`, `snmpwalk`, or `snmptrap`) are

run from the command line, or when `snmpd` is configured to send a trap (based on `snmpd.conf`), you can configure a `clientaddr` in `snmpd.conf` that allows the SNMP client programs or `snmpd` (for traps) to source requests from a different source IP address.

For more information about `clientaddr`, read the `snmpd.conf` man page.

> **NOTE**
>
> `snmptrap`, `snmpget`, `snmpwalk` and `snmpd` itself must be able to bind to this address.

> **NOTE**
>
> There is no NCLU command for this configuration.

Edit the `/etc/snmp/snmpd.conf` file and add the `clientaddr` option. In the following example, spine01 is used as the client (IP address 192.168.200.21).

```
cumulus@switch:~$ sudo nano /etc/snmp/snmpd.conf
...
trapsess -Ci --clientaddr=192.168.200.21 -v 2c
...
```

Restart the `snmpd` service to apply the changes.

```
cumulus@switch:~$ sudo systemctl restart snmpd.service
```

Monitor Fans, Power Supplies, Temperature and Transformers

An SNMP agent (`snmpd`) waits for incoming SNMP requests and responds to them. If no requests are received, an agent does not initiate any actions. However, various commands can configure `snmpd` to send traps based on preconfigured settings (`load`, `file`, `proc`, `disk`, or `swap` commands), or customized `monitor` directives.

See the `snmpd.conf` man page for details on the `monitor` directive.

You can configure `snmpd` to monitor the operational status of either the Entity MIB or Entity-Sensor MIB by adding the `monitor` directive to the `snmpd.conf` file. Once you know the OID, you can determine the operational status — which can be a value of *ok(1)*, *unavailable(2)* or *nonoperational(3)* — by adding a configuration like the following example to `/etc/snmp/snmpd.conf` and adjusting the values:

- Using the `entPhySensorOperStatus` integer:

```
cumulus@switch:~$ sudo nano /etc/snmp/snmpd.conf
```

```
...
# without installing extra MIBS we can check the check Fan1
status
# if the Fan1 index is 100011001, monitor this specific OID (-
I) every 10 seconds (-r), and defines additional information to
be included in the trap (-o).
monitor -I -r 10  -o 1.3.6.1.2.1.47.1.1.1.1.7.100011001 "Fan1
Not OK"  1.3.6.1.2.1.99.1.1.1.5.100011001 > 1
# Any Entity Status non OK (greater than 1)
monitor  -r 10  -o 1.3.6.1.2.1.47.1.1.1.1.7  "Sensor Status
Failure"  1.3.6.1.2.1.99.1.1.1.5 > 1
```

- Using the OID name. You can use the OID name if the `snmp-mibs-downloader` package is installed (see below).

```
cumulus@switch:~$ sudo nano /etc/snmp/snmpd.conf
...
# for a specific fan called Fan1 with an index 100011001
monitor -I -r 10  -o entPhysicalName.100011001 "Fan1 Not OK"
entPhySensorOperStatus.100011001 > 1
# for any Entity Status not OK ( greater than 1)
monitor  -r 10  -o entPhysicalName  "Sensor Status Failure"
```

```
entPhySensorOperStatus > 1
```

> **NOTE**
>
> The `entPhySensorOperStatus` integer can be found by walking the `entPhysicalName` table.

To get all sensor information, run `snmpwalk` on the `entPhysicalName` table. For example:

```
cumulus@leaf01:~$ snmpwalk -v 2c -cpublic localhost
.1.3.6.1.2.1.47.1.1.1.1.7
iso.3.6.1.2.1.47.1.1.1.1.7.100000001 = STRING: "PSU1Temp1"
iso.3.6.1.2.1.47.1.1.1.1.7.100000002 = STRING: "PSU2Temp1"
iso.3.6.1.2.1.47.1.1.1.1.7.100000003 = STRING: "Temp1"
iso.3.6.1.2.1.47.1.1.1.1.7.100000004 = STRING: "Temp2"
iso.3.6.1.2.1.47.1.1.1.1.7.100000005 = STRING: "Temp3"
iso.3.6.1.2.1.47.1.1.1.1.7.100000006 = STRING: "Temp4"
iso.3.6.1.2.1.47.1.1.1.1.7.100000007 = STRING: "Temp5"
iso.3.6.1.2.1.47.1.1.1.1.7.100011001 = STRING: "Fan1"
iso.3.6.1.2.1.47.1.1.1.1.7.100011002 = STRING: "Fan2"
```

```
iso.3.6.1.2.1.47.1.1.1.1.7.100011003 = STRING: "Fan3"
iso.3.6.1.2.1.47.1.1.1.1.7.100011004 = STRING: "Fan4"
iso.3.6.1.2.1.47.1.1.1.1.7.100011005 = STRING: "Fan5"
iso.3.6.1.2.1.47.1.1.1.1.7.100011006 = STRING: "Fan6"
iso.3.6.1.2.1.47.1.1.1.1.7.100011007 = STRING: "PSU1Fan1"
iso.3.6.1.2.1.47.1.1.1.1.7.100011008 = STRING: "PSU2Fan1"
iso.3.6.1.2.1.47.1.1.1.1.7.110000001 = STRING: "PSU1"
iso.3.6.1.2.1.47.1.1.1.1.7.110000002 = STRING: "PSU2"
```

Restart the `snmpd` service to apply the changes.

```
cumulus@switch:~$ sudo systemctl restart snmpd.service
```

NOTE

There is no NCLU command for monitoring hardware.

NOTE

In earlier versions of Cumulus Linux, you could use the LM-

SENSORS MIB to monitor temperature, but that MIB has been deprecated.

Configure Link Up/Down Notifications

The `linkUpDownNotifications` directive is used to configure link up/down notifications when the operational status of the link changes.

NOTE

The default frequency for checking link up/down is 60 seconds. You can change the default frequency using the directly instead of the `linkUpDownNotifications` directive. See `man snmpd.conf` for details.

NCLU Commands | Linux Commands

To enable notifications for interface link-up events to be sent to SNMP trap destinations every 15 seconds, run:

```
cumulus@switch:~$ net add snmp-server trap-link-up check-
frequency 15
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Similarly, to enable notifications for interface link-down events to be sent to SNMP trap destinations every 10 seconds, run:

```
cumulus@switch:~$ net add snmp-server trap-link-down check-
frequency 10
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Configure Free Memory Notifications

You can monitor free memory using the following directives. The example below generates a trap when free memory drops below 1,000,000KB. The free memory trap also includes the amount of total real memory:

```
cumulus@switch:~$ sudo nano /etc/snmp/snmpd.conf
...
monitor MemFreeTotal -o memTotalReal memTotalFree <  1000000
...
```

Restart the `snmpd` service to apply the changes.

```
cumulus@switch:~$ sudo systemctl restart snmpd.service
```

> **NOTE**
>
> There is no NCLU command for monitoring free memory.

Configure Processor Load Notifications

To enable a trap when the CPU load average exceeds a configured threshold, run the following commands. You can only use integers or floating point numbers.

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add snmp-server trap-cpu-load-average
one-minute 4.34 five-minute 2.32 fifteen-minute 6.5
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Configure Disk Utilization Notifications

To monitor disk utilization for all disks, use the `includeAllDisks` directive together with the `monitor` directive. The example code below generates a trap when a disk is 99% full:

```
cumulus@switch:~$ sudo nano /etc/snmp/snmpd.conf
...
includeAllDisks 1%
monitor -r 60 -o dskPath -o DiskErrMsg "dskTable" diskErrorFlag
!=0
...
```

Restart the `snmpd` service to apply the changes.

```
cumulus@switch:~$ sudo systemctl restart snmpd.service
```

> **NOTE**
>
> There is no NCLU command for monitoring disk utilization.

Configure Authentication Notifications

To enable SNMP trap notifications to be sent for every SNMP authentication failure, run the following commands.

NCLU Commands | Linux Commands

```
cumulus@switch:~$ net add snmp-server trap-snmp-auth-
failures
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

Monitor UCD-SNMP-MIB Tables

To configure the Event MIB tables to monitor the various UCD-SNMP-MIB tables for problems (as indicated by the appropriate xxErrFlag column objects) and send a trap, add `defaultMonitors yes` to the `snmpd.conf` file

and provide a configuration. You must first download the `snmp-mibs-downloader` Debian package and comment out the `mibs` line from the `/etc/snmp/snmpd.conf` file (see below). Then add a configuration like the following example:

```
cumulus@switch:~$ sudo nano /etc/snmp/snmpd.conf
...
defaultMonitors yes


monitor   -o prNames -o prErrMessage "process table"
prErrorFlag != 0
monitor   -o memErrorName -o memSwapErrorMsg "memory"
memSwapError != 0
monitor   -o extNames -o extOutput "extTable" extResult !=
0<br>monitor   -o dskPath -o dskErrorMsg "dskTable"
dskErrorFlag != 0
monitor   -o laNames -o laErrMessage  "laTable" laErrorFlag !=
0<br>monitor   -o fileName -o fileErrorMsg  "fileTable"
fileErrorFlag != 0
...
```

Restart the `snmpd` service to apply the changes.

```
cumulus@switch:~$ sudo systemctl restart snmpd.service
```

Enable MIB to OID Translation

MIB names can be used instead of OIDs, which greatly improves the readability of the `snmpd.conf` file. You enable this by installing the `snmp-mibs-downloader`, which downloads SNMP MIBs to the switch prior to enabling traps.

1. Open `/etc/apt/sources.list` in a text editor.
2. Add the `non-free` repository, then save the file:

```
cumulus@switch:~$ sudo deb http://ftp.us.debian.org/debian/
buster main non-free
```

3. Update the switch:

```
cumulus@switch:~$ sudo -E apt-get update
```

4. Install the `snmp-mibs-downloader`:

```
cumulus@switch:~$ sudo -E apt-get install snmp-mibs-downloader
```

5. Open the `/etc/snmp/snmp.conf` file to verify that the `mibs :` line is commented out:

```
#
# As the snmp packages come without MIB files due to license
reasons, loading
# of MIBs is disabled by default. If you added the MIBs you
can reenable
# loading them by commenting out the following line.
#mibs :
```

6. Open the `/etc/default/snmpd` file to verify that the `export MIBS=` line is commented out:

```
# This file controls the activity of snmpd and snmptrapd


# Don't load any MIBs by default.
# You might comment this lines once you have the MIBs
Downloaded.
```

```
#export MIBS=
```

7. After you confirm the configuration, remove or comment out the `non-free` repository in `/etc/apt/sources.list`.

```
#deb http://ftp.us.debian.org/debian/ buster main non-free
```

Configure Incoming SNMP Traps

The Net-SNMP trap daemon configured in `/etc/snmp/snmpd.conf` *receives* SNMP traps. You configure how *incoming* traps are processed in the `/etc/snmp/snmptrapd.conf` file. Starting with Net-SNMP release 5.3, you must specify who is authorized to send traps and informs to the notification receiver (and what types of processing these are allowed to trigger). You can specify three processing types:

- *log* logs the details of the notification in a specified file to standard output (or stderr), or through syslog (or similar).
- *execute* passes the details of the trap to a specified handler program, including embedded Perl.
- *net* forwards the trap to another notification receiver.

Typically, you configure all three — *log,execute,net* — to cover any style of processing for a particular category of notification. But you can limit certain

notification sources to selected processing only.

`authCommunity TYPES COMMUNITY [SOURCE [OID | -v VIEW ]]` authorizes traps and SNMPv2c INFORM requests with the specified community to trigger the types of processing listed. By default, this allows any notification using this community to be processed. You can use the SOURCE field to specify that the configuration only applies to notifications received from particular sources. For more information about specific configuration options within the file, look at the snmptrapd.conf(5) man page with the `man 5 snmptrapd.conf` command.

NOTE

You may need to install the `snmptrapd` Debian package before you can configure incoming traps.

```
cumulus@switch:~$ sudo apt-get install snmptrapd
```

Supported MIBs

Below are the MIBs supported by Cumulus Linux, as well as suggested uses for them. The overall Cumulus Linux MIB is defined in the `/usr/share/snmp/mibs/Cumulus-Snmp-MIB.txt` file.

MIB Name	Suggested Uses
BGP4-MIB OSPFv2-MIB OSPFv3-MIB RIPv2-MIB	You can enable FRRouting SNMP support to provide support for OSPF-MIB (RFC-1850), OSPFV3-MIB (RFC-5643), and BGP4-MIB (RFC-1657). See the FRRouting section above.
CUMULUS-COUNTERS-MIB	Discard counters: Cumulus Linux also includes its own counters MIB, defined in `/usr/share/snmp/mibs/Cumulus-Counters-MIB.txt`. It has the OID `.1.3.6.1.4.1.40310.2`.
CUMULUS-POE-MIB	The custom Power over Ethernet PoE MIB defined in the `/usr/share/snmp/mibs/Cumulus-POE-MIB.txt` file. For devices that provide PoE, this provides users with the system wide power information in `poeSystemValues` as well as per interface `PoeObjectsEntry` values for the `poeObjectsTable`. Most of this information comes

MIB Name	Suggested Uses
	from the `poectl` command. To enable this MIB, uncomment the following line in `/etc/snmp/snmpd.conf`: `#pass_persist .1.3.6.1.4.1.40310.3 /usr/share/snmp/cl_poe_pp.py`
CUMULUS-RESOURCE-QUERY-MIB	Cumulus Linux includes its own resource utilization MIB, which is similar to using `cl-resource-query`. This MIB monitors layer 3 entries by host, route, nexthops, ECMP groups, and layer 2 MAC/BDPU entries. The MIB is defined in `/usr/share/snmp/mibs/Cumulus-Resource-Query-MIB.txt` and has the OID `.1.3.6.1.4.1.40310.1`.
CUMULUS-SNMP-MIB	SNMP counters. For information on exposing CPU and memory information with SNMP, see this knowledge base article.
DISMAN-EVENT-MIB	Trap monitoring.
ENTITY-MIB	From RFC 4133, the temperature

MIB Name	Suggested Uses
	sensors, fan sensors, power sensors, and ports are covered. **Note:** The ENTITY-MIB does not show the chassis information in Cumulus Linux.
ENTITY-SENSOR-MIB	Physical sensor information (temperature, fan, and power supply) from RFC 3433.
HOST-RESOURCES-MIB	Users, storage, interfaces, process info, run parameters.
BRIDGE-MIB Q-BRIDGE-MIB	The `dot1dBasePortEntry` and `dot1dBasePortIfIndex` tables in the BRIDGE-MIB and `dot1qBase`, `dot1qFdbEntry`, `dot1qTpFdbEntry`, `dot1qTpFdbStatus`, and `dot1qVlanStaticName` tables in the Q-BRIDGE-MIB tables. You must uncomment the `bridge_pp.py pass_persist` script in `/etc/snmp/snmpd.conf`.
IEEE8023-LAG-MIB	Implementation of the IEEE 8023-LAG-MIB includes the `dot3adAggTable` and `dot3adAggPortListTable` tables. To enable this, edit `/etc/snmp/snmpd.conf` and

MIB Name	Suggested Uses

	uncomment or add the following lines:
	
	`view systemonly included .1.2.840.10006.300.43`
	`pass_persist .1.2.840.10006.300.43 /usr/share/snmp/ieee8023_lag_pp.py`

IF-MIB	Interface description, type, MTU, speed, MAC, admin, operation status, counters. **Note**: The IF-MIB cache is disabled by default. The non-caching code path in the IF-MIB treats 64-bit counters like 32-bit counters (a 64-bit counter rolls over after the value increments to a value that extends beyond 32 bits). To enable the counter to reflect traffic statistics using 64-bit counters, remove the `-y` option from the `SNMPDOPTS` line in the `/etc/default/snmpd` file. The example below first shows the original line, commented out, then the modified line without

MIB Name	Suggested Uses

	the `-y` option:
	
	`cumulus@switch:~$ cat /etc/default/snmpd`
	`# SNMPDOPTS='-y -LS 0-4 d -Lf /dev/null -u snmp -g snmp -I -smux -p /run/snmpd.pid'`
	`SNMPDOPTS='-LS 0-4 d -Lf /dev/null -u snmp -g snmp -I -smux -p /run/snmpd.pid`

IP-FORWARD-MIB	IP routing table.
IP-MIB (includes ICMP)	IPv4, IPv4 addresses counters, netmasks.
IPv6-MIB	IPv6 counters.
LLDP-MIB	Layer 2 neighbor information from `lldpd` (you need to enable the SNMP subagent in LLDP). You need to start `lldpd` with the `-x` option to enable connectivity to `snmpd`(AgentX).
LM-SENSORS MIB	Fan speed, temperature sensor values, voltages. This is deprecated since the ENTITY-SENSOR MIB has been added.

MIB Name	Suggested Uses
NET-SNMP-AGENT-MIB	Agent timers, user, group config.
NET-SNMP-VACM-MIB	Agent timers, user, group config.
NOTIFICATION-LOG-MIB	Local logging.
SNMP-FRAMEWORK-MIB	Users, access.
SNMP-MPD-MIB	Users, access.
SNMP-TARGET-MIB	SNMP-TARGET-MIB.
SNMP-USER-BASED-SM-MIBS	Users, access.
SNMP-VIEW-BASED-ACM-MIB	Users, access.
TCP-MIB	TCP-related information.
UCD-SNMP-MIB	System memory, load, CPU, disk IO.
UDP-MIB	UDP-related information.

List All Installed MIBs

Due to licensing restrictions, not all supported MIBs are installed in Cumulus Linux. The MIBs that are not installed require the “non-free” archive to be added to `/etc/apt/sources.list`. To see which MIBs are installed on your switch, run `ls /usr/share/snmp/mibs/`.

To install more MIBs, you need to install `snmp-mibs-downloader` and then either remove or comment out the “non-free” repository in `/etc/apt/`

`sources.list`. This is described here.

▼ Installed MIBs

Troubleshoot SNMP

Use the following commands to troubleshoot potential SNMP issues.

Troubleshoot with NCLU

To check the status of `snmpd` using NCLU, run the `net show snmp-server status` command. If there are issues, you might see errors like the following:

```
cumulus@switch:~$ net show snmp-server status
Simple Network Management Protocol (SNMP) Daemon.
---------------------------------
---------------------------------------------------------------------------------
Current Status                    failed (failed)
Reload Status                     enabled
Listening IP Addresses            localhost 9.9.9.9
Main snmpd PID                    0
Version 1 and 2c Community String  Configured
Version 3 Usernames               Not Configured
Last Logs (with Errors)           -- Logs begin at Thu
2017-08-03 16:23:05 UTC, end at Fri 2017-08-04 18:17:24 UTC. --
                                  Aug 04 18:17:19 cel-redxp-01
snmpd[8389]: Error opening specified endpoint "9.9.9.9"
                                  Aug 04 18:17:19 cel-redxp-01
snmpd[8389]: Server Exiting with code 1
```

```
---------------------------------
-------------------------------------------------------------------------------
```

You can review the SNMP server configuration when you run:

```
cumulus@switch:~$ net show configuration snmp-server
snmp-server
  listening-address 127.0.0.1
  readonly-community public access default
  readonly-community allpass access any
  readonly-community temp2 access 1.1.1.1
  readonly-community temp2 access 2.2.2.2
  trap-destination 1.1.1.1 community-password public version 2c
  trap-link-up check-frequency 10
  trap-snmp-auth-failures
```

You can see which NCLU commands were used to configure SNMP. Look for `snmp-server` in the output when you run:

```
cumulus@switch:~$ net show configuration commands
...
```

```
net add snmp-server listening-address all
net add snmp-server readonly-community allpass access any
net add snmp-server readonly-community temp2 access 1.1.1.1
net add snmp-server readonly-community temp2 access 2.2.2.2
net add snmp-server trap-destination 1.1.1.1 community-password
public version 2c
net add snmp-server trap-link-up check-frequency 10
net add snmp-server trap-snmp-auth-failures
...
```

Troubleshoot with SNMP Commands

The `snmp` Debian package contains `snmpget`, `snmpwalk` and other programs that are useful for checking daemon functionality from the switch itself or from another workstation.

From a client, you access the MIB with the correct credentials.

```
cumulus@switch:~$ snmpwalk -v 3 -u userMD5withDES -l authPriv
-a MD5 -x DES -A md5authpass -X desprivpass localhost
1.3.6.1.2.1.1.1
cumulus@switch:~$ snmpwalk -v 3 -u userSHAwithAES -l authPriv
-a SHA -x AES -A shaauthpass -X aesprivpass localhost
```

```
1.3.6.1.2.1.1.1
```

This command gets the first MIB object in the system table; in this case, the SNMPv2 system name specified above:

```
cumulus@switch:~$ snmpgetnext -v 2c -c mynotsosecretpassword
localhost SNMPv2-MIB::sysName
SNMPv2-MIB::sysName.0 = STRING: my little router
```

The following commands check the access for each user from the localhost.

To check user1, which has no authentication or encryption (*NoauthNoPriv*):

```
cumulus@switch:~$ snmpget -v 3 -u user1 -l NoauthNoPriv
localhost 1.3.6.1.2.1.1.1.0
cumulus@switch:~$ snmpwalk -v 3 -u user1 -l NoauthNoPriv
localhost 1.3.6.1.2.1.1
```

To check user2, which has authentication but no encryption (*authNoPriv*):

```
cumulus@switch:~$ snmpget -v 3 -u user2 -l authNoPriv -a MD5 -A
user2password localhost 1.3.6.1.2.1.1.1.0
cumulus@switch:~$ snmpget -v 3 -u user2 -l authNoPriv -a MD5 -A
user2password localhost 1.3.6.1.2.1.2.1.0
cumulus@switch:~$ snmpwalk -v 3 -u user2 -l authNoPriv -a MD5
-A user2password localhost 1.3.6.1.2.1
```

To check user3, which has both authentication and encryption (*authPriv*):

```
cumulus@switch:~$ snmpget -v 3 -u user3 -l authPriv -a MD5 -A
user3password -x DES -X user3encryption localhost
.1.3.6.1.2.1.1.1.0
cumulus@switch:~$ snmpwalk -v 3 -u user3 -l authPriv -a MD5 -A
user3password -x DES -X user3encryption localhost .1.3.6.1.2.1
cumulus@switch:~$ snmpwalk -v 3 -u user666 -l authPriv -a SHA
-x AES -A user666password -X user666encryption localhost
1.3.6.1.2.1.1
cumulus@switch:~$ snmpwalk -v 3 -u user999 -l authPriv -a MD5
-x DES -A user999password -X user999encryption localhost
1.3.6.1.2.1.1
```

NOTE

As mentioned in Configure SNMP, SNMP is VRF-aware. To run commands like `snmpget` or `snmpwalk` in a VRF, preface the command with `sudo ip vrf exec <VRF>`, like this:

```
cumulus@switch:~$ sudo ip vrf exec default snmpgetnext
-v 2c -c mynotsosecretpassword localhost
SNMPv2-MIB::sysName
SNMPv2-MIB::sysName.0 = STRING: my little router
```

Using Nutanix Prism as a Monitoring Tool

Nutanix Prism is a graphical user interface (GUI) for managing infrastructure and virtual environments. You need to take special steps within Cumulus Linux before you can configure Prism.

Configure Cumulus Linux

1. SSH to the Cumulus Linux switch that needs to be configured, replacing `[switch]` below as appropriate:

```
cumulus@switch:~$ ssh cumulus@[switch]
```

2. Open the `/etc/snmp/snmpd.conf` file in an editor.

3. Uncomment the following 3 lines in the `/etc/snmp/snmpd.conf` file, then save the file:

   - bridge_pp.py

```
pass_persist .1.3.6.1.2.1.17 /usr/share/snmp/bridge_pp.py
```

- Community

```
 rocommunity public  default    -V systemonly
```

- Line directly below the Q-BRIDGE-MIB (.1.3.6.1.2.1.17)

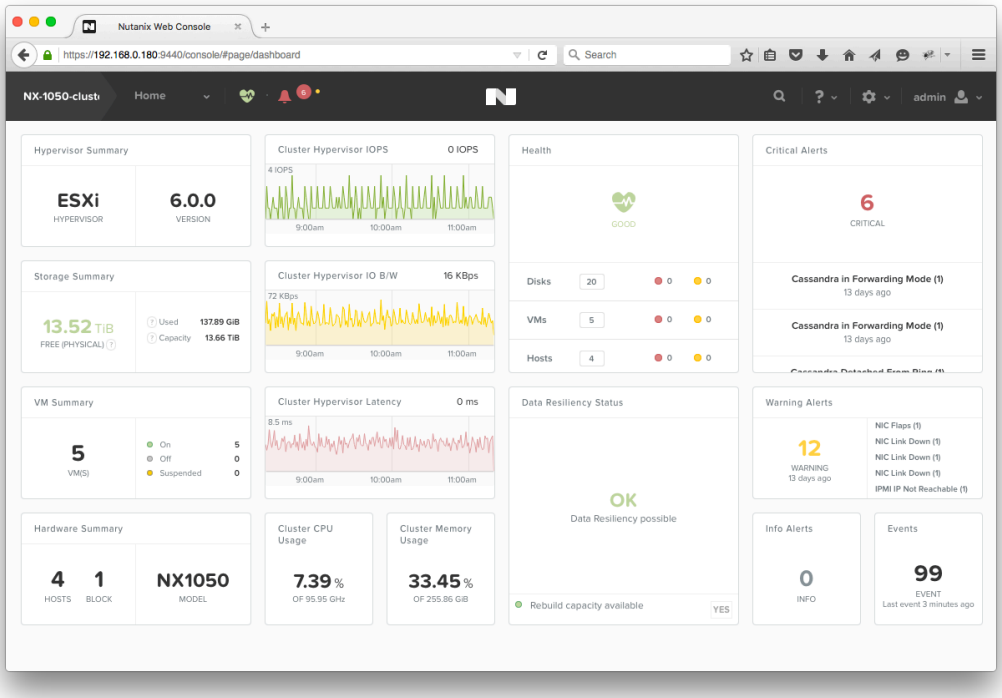
```
\# BRIDGE-MIB and Q-BRIDGE-MIB tables
view   systemonly  included   .1.3.6.1.2.1.17
```

4. Restart `snmpd`:

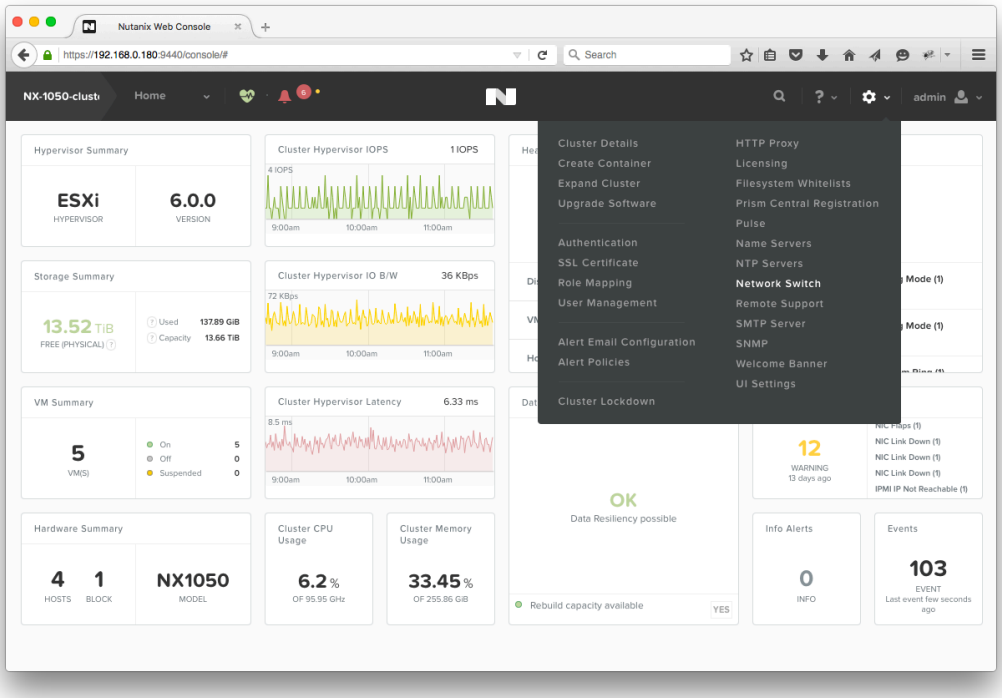
```
cumulus@switch:~$ sudo systemctl restart snmpd.service
Restarting network management services: snmpd.
```

Configure Nutanix

1. Log into the Nutanix Prism. Nutanix defaults to the Home menu, referred to as the Dashboard:

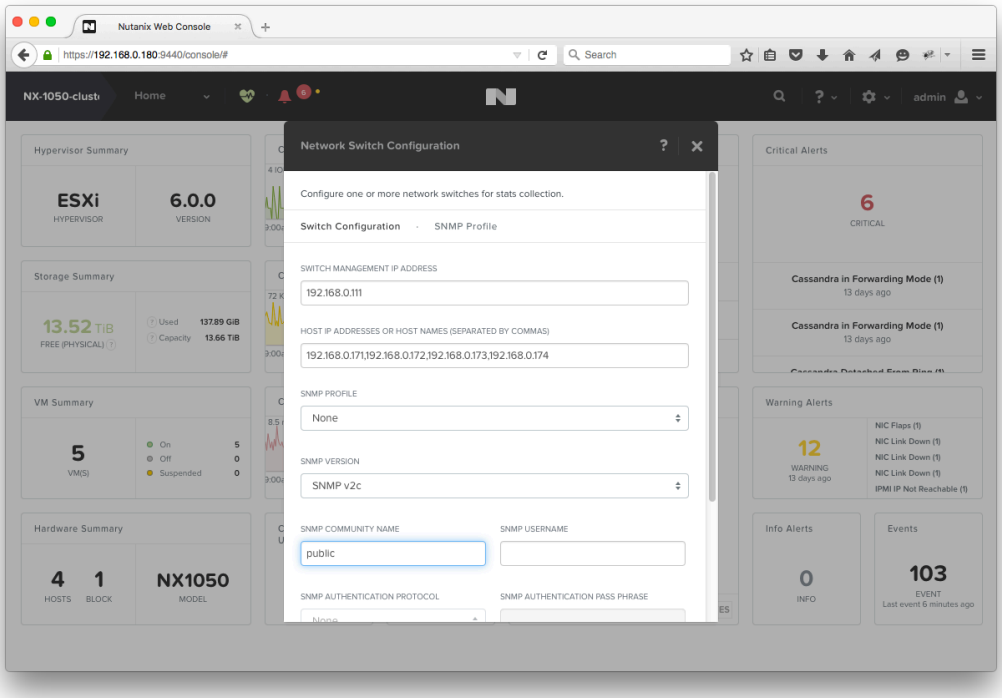


2. Click on the gear icon in the top right corner of the dashboard, then select NetworkSwitch:



3. Click the **+Add Switch Configuration** button in the **Network Switch Configuration** pop up window.

4. Fill out the **Network Switch Configuration** for the Top of Rack (ToR) switch configured for snmpd in the previous section:



Configuration Parameter	Description	Value Used in Example
Switch Management IP Address	This can be any IP address on the box. In the screenshot above, the eth0 management IP is used.	192.168.0.111
Host IP Addresses or Host Names	IP addresses of Nutanix hosts connected to that particular ToR switch.	192.168.0.171,192.168.0.172,192.168.0.173

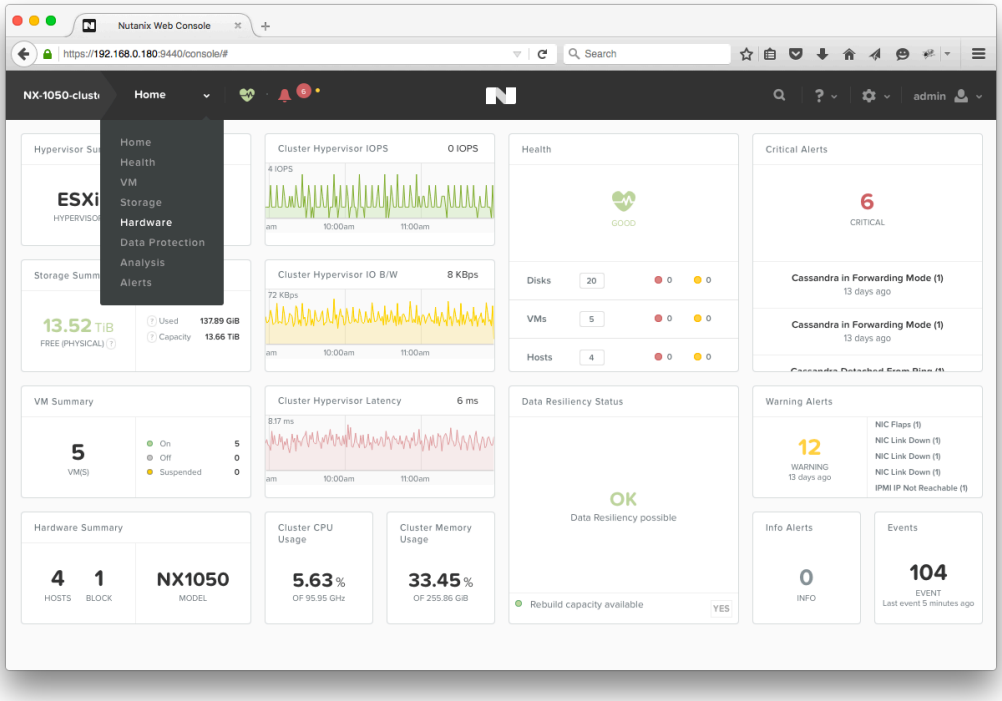
Configuration Parameter	Description	Value Used in Example
SNMP Profile	Saved profiles, for easy configuration when hooking up to multiple switches.	None
SNMP Version	SNMP v2c or SNMP v3. Cumulus Linux has only been tested with SNMP v2c for Nutanix integration.	SNMP v2c
SNMP Community Name	SNMP v2c uses communities to share MIBs. The default community for snmpd is 'public'.	public

> **NOTE**
>
> The rest of the values were not touched for this demonstration.
>
> They are usually used with SNMP v3.

5. Save the configuration. The switch will now be present in the **Network Switch Configuration** menu now.

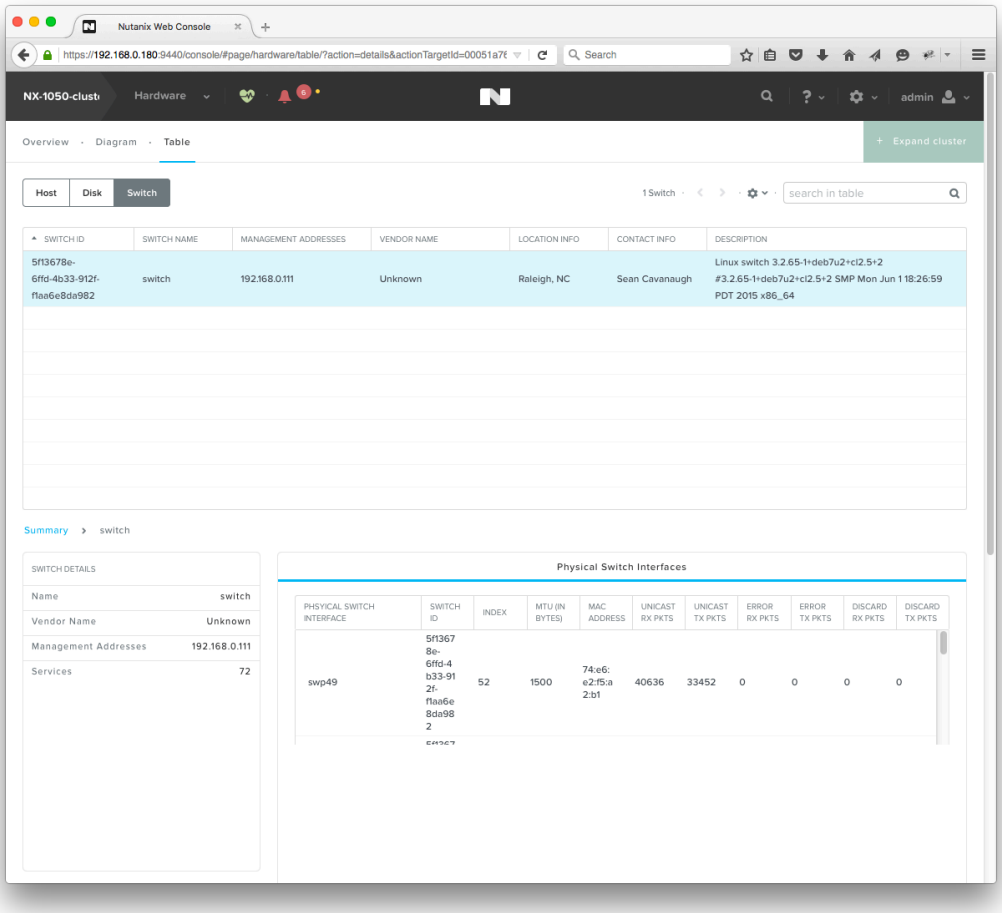
6. Close the pop up window to return to the dashboard.

7. Open the **Hardware** option from the **Home** dropdown menu:



8. Click the **Table** button.

9. Click the **Switch** button. Configured switches are shown in the table, as indicated in the screenshot below, and can be selected in order to view interface statistics:



> **NOTE**
>
> The switch has been added correctly when interfaces hooked up to the Nutanix hosts are visible.

Switch Information Displayed on Nutanix Prism

- Physical interface (for example, swp1, swp2). This only displays switch port interfaces connected to Nutanix hosts by default.
- Switch ID: Unique identifier that Nutanix keeps track of each port ID (see below).
- Index: Interface index, in the above demonstration swp49 maps to Index 52 because there is a loopback and two ethernet interface before the swp starts.
- MTU of interface.
- MAC address of interface.
- Unicast RX packets (received).
- Unicast TX packets (transmitted).
- Error RX packets (received).
- Error TX packets (transmitted).
- Discard RX packets (received).
- Discard TX packets (transmitted).

The Nutanix appliance will use Switch IDs that can also be viewed on the Prism CLI (by SSHing to the box). To view information from the Nutanix CLI, login using the default username *nutanix*, and the password *nutanix/4u*.

```
nutanix@NTNX-14SM15270093-D-CVM:192.168.0.184:~$ ncli network
list-switch
    Switch ID                 :
```

```
00051a76-f711-89b6-0000-000000003bac::5f13678e-6ffd-4b33-912f-
f1aa6e8da982
    Name                        : switch
    Switch Management Address : 192.168.0.111
    Description                 : Linux switch
3.2.65-1+deb7u2+cl2.5+2 #3.2.65-1+deb7u2+cl2.5+2 SMP Mon Jun 1
18:26:59 PDT 2015 x86_64
    Object ID                   : enterprises.40310
    Contact Information         : Admin <admin@company.com>
    Location Information        : Raleigh, NC
    Services                    : 72
    Switch Vendor Name          : Unknown
    Port Ids                    :
00051a76-f711-89b6-0000-000000003bac::5f13678e-6ffd-4b33-912f-
f1aa6e8da982:52,
00051a76-f711-89b6-0000-000000003bac::5f13678e-6ffd-4b33-912f-
f1aa6e8da982:53,
00051a76-f711-89b6-0000-000000003bac::5f13678e-6ffd-4b33-912f-
f1aa6e8da982:54,
00051a76-f711-89b6-0000-000000003bac::5f13678e-6ffd-4b33-912f-
f1aa6e8da982:55
```

Enable LLDP/CDP on VMware ESXi (Hypervisor on Nutanix)

1. Follow the directions on one of the following websites to enable CDP:

   - VMware knowledge base article: Configuring the Cisco Discovery Protocol (CDP) with ESX/ESXi (1003885)
   - Wahl Network: Utilizing CDP and LLDP with vSphere Networking

   For example, switch CDP on:

```
root@NX-1050-A:~] esxcli network vswitch standard set -c both
-v vSwitch0
```

   Then confirm it is running:

```
root@NX-1050-A:~] esxcli network vswitch standard list -v
vSwitch0
vSwitch0
    Name: vSwitch0
    Class: etherswitch
    Num Ports: 4082
    Used Ports: 12
    Configured Ports: 128
```

```
    MTU: 1500
    CDP Status: both
    Beacon Enabled: false
    Beacon Interval: 1
    Beacon Threshold: 3
    Beacon Required By:
    Uplinks: vmnic3, vmnic2, vmnic1, vmnic0
    Portgroups: VM Network, Management Network
```

both means CDP is now running and the `lldp` dameon on Cumulus Linux is capable of *seeing* CDP devices.

2. After the next CDP interval, the Cumulus Linux switch picks up the interface via the `lldp` daemon:

```
cumulus@switch:~$ lldpctl show neighbor swp49
-------------------------------------------------------------------------------
LLDP neighbors:
-------------------------------------------------------------------------------
Interface:    swp49, via: CDPv2, RID: 6, Time: 0 day, 00:34:58
Chassis:
    ChassisID:    local NX-1050-A
    SysName:      NX-1050-A
```

```
    SysDescr:     Releasebuild-2494585 running on VMware ESX
    MgmtIP:       0.0.0.0
    Capability:   Bridge, on
Port:
    PortID:       ifname vmnic2
    PortDescr:    vmnic2
-------------------------------------------------------------------------------
```

3. Use `net show` to look at `lldp` information:

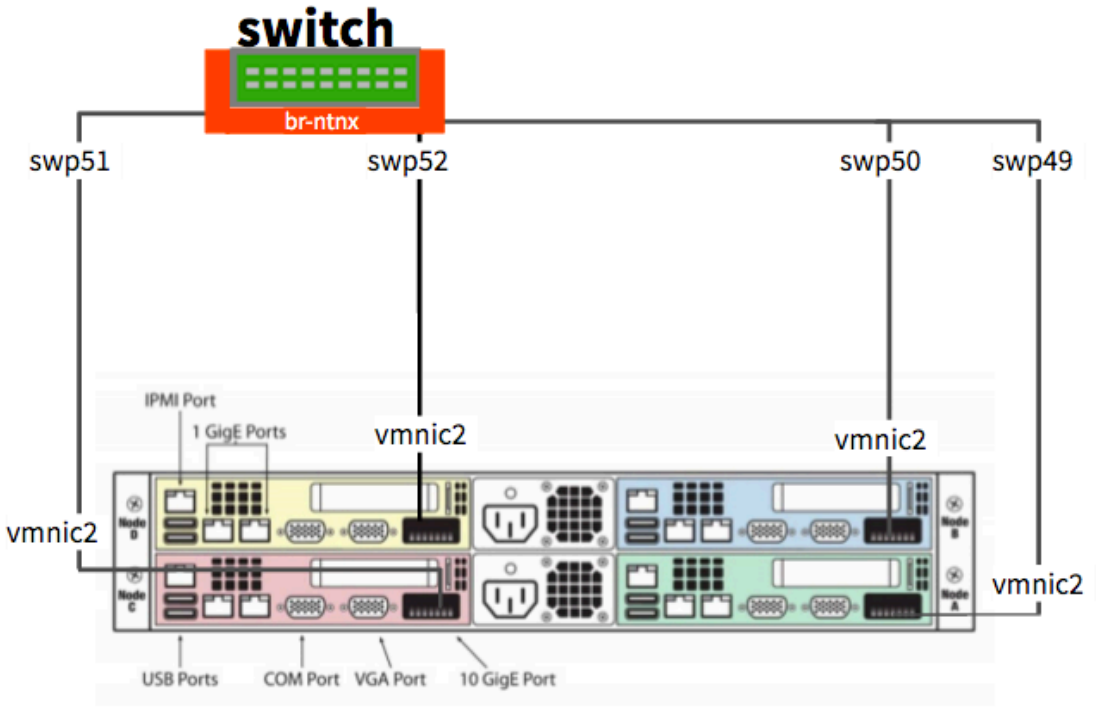
```
cumulus@switch:~$ net show lldp


Local Port    Speed    Mode                 Remote
Port        Remote  Host     Summary
------------  -------  -------------  ----
-----------------  ---------------  -------------------------
eth0          1G       Mgmt           ====
swp6               oob-mgmt-switch  IP: 192.168.0.11/24(DHCP)
swp1          1G       Access/L2      ====
44:38:39:00:00:03  server01         Untagged: br0
swp51         1G       NotConfigured  ====
swp1               spine01
swp52         1G       NotConfigured  ====
```

```
swp1                spine02
```

Nutanix Acropolis is an alternate hypervisor that Nutanix supports. **Acropolis Hypervisor** uses the `yum` packaging system and is capable of installing normal Linux `lldp` daemons to operating just like Cumulus Linux. LLDP should be enabled for each interface on the host. Refer to this article from Mellanox, https://portal.nutanix.com/page/documents/kbs/details/?targetId=kA032000000TVfiCAG, for setup instructions.

Troubleshooting

To help visualize the following diagram is provided:



Nutanix Node	Physical Port	Cumulus Linux Port
Node A (Green)	vmnic2	swp49
Node B (Blue)	vmnic2	swp50
Node C (Red)	vmnic2	swp51
Node D (Yellow)	vmnic2	swp52

Troubleshoot Connections without LLDP or CDP

1. Find the MAC address information in the Prism GUI, located in: **Hardware > Table > Host > Host NICs**

2. Select a MAC address to troubleshoot (for example, 0c:c4:7a:09:a2:43 represents vmnic0 which is tied to NX-1050-A).

3. List out all the MAC addresses associated with the bridge:

```
cumulus@switch:~$ brctl showmacs br-ntnx
port name mac addr        vlan    is local?   ageing timer
swp9      00:02:00:00:00:06   0   no        66.94
swp52     00:0c:29:3e:32:12   0   no         2.73
swp49     00:0c:29:5a:f4:7f   0   no         2.73
swp51     00:0c:29:6f:e1:e4   0   no         2.73
swp49     00:0c:29:74:0c:ee   0   no         2.73
swp50     00:0c:29:a9:36:91   0   no         2.73
swp9      08:9e:01:f8:8f:0c   0   no        13.56
swp9      08:9e:01:f8:8f:35   0   no         2.73
swp4      0c:c4:7a:09:9e:d4   0   no        24.05
swp1      0c:c4:7a:09:9f:8e   0   no        13.56
swp3      0c:c4:7a:09:9f:93   0   no        13.56
swp2      0c:c4:7a:09:9f:95   0   no        24.05
swp52     0c:c4:7a:09:a0:c1   0   no         2.73
swp51     0c:c4:7a:09:a2:35   0   no         2.73
swp49     0c:c4:7a:09:a2:43   0   no         2.73
swp9      44:38:39:00:82:04   0   no         2.73
swp9      74:e6:e2:f5:a2:80   0   no         2.73
swp1      74:e6:e2:f5:a2:81   0   yes        0.00
swp2      74:e6:e2:f5:a2:82   0   yes        0.00
swp3      74:e6:e2:f5:a2:83   0   yes        0.00
```

```
swp4      74:e6:e2:f5:a2:84   0   yes        0.00
swp5      74:e6:e2:f5:a2:85   0   yes        0.00
swp6      74:e6:e2:f5:a2:86   0   yes        0.00
swp7      74:e6:e2:f5:a2:87   0   yes        0.00
swp8      74:e6:e2:f5:a2:88   0   yes        0.00
swp9      74:e6:e2:f5:a2:89   0   yes        0.00
swp10     74:e6:e2:f5:a2:8a   0   yes        0.00
swp49     74:e6:e2:f5:a2:b1   0   yes        0.00
swp50     74:e6:e2:f5:a2:b2   0   yes        0.00
swp51     74:e6:e2:f5:a2:b3   0   yes        0.00
swp52     74:e6:e2:f5:a2:b4   0   yes        0.00
swp9      8e:0f:73:1b:f8:24   0   no         2.73
swp9      c8:1f:66:ba:60:cf   0   no        66.94
```

Alternatively, you can use `grep`:

```
cumulus@switch:~$ brctl showmacs br-ntnx | grep
0c:c4:7a:09:a2:43
swp49     0c:c4:7a:09:a2:43   0   no         4.58
```

vmnic1 is now hooked up to swp49. This matches what is seen in `lldp`:

```
cumulus@switch:~$ lldpctl show neighbor swp49
-------------------------------------------------------------------------------
LLDP neighbors:
-------------------------------------------------------------------------------
Interface:    swp49, via: CDPv2, RID: 6, Time: 0 day, 01:11:12
    Chassis:
    ChassisID:      local NX-1050-A
    SysName:        NX-1050-A
    SysDescr:       Releasebuild-2494585 running on VMware ESX
    MgmtIP:         0.0.0.0
    Capability:     Bridge, on
    Port:
    PortID:         ifname vmnic2
    PortDescr:      vmnic2
-------------------------------------------------------------------------------
```

Single User Mode - Password Recovery

Use single user mode to assist in troubleshooting system boot issues or for password recovery.

To enter single user mode:

1. Boot the switch, then as soon as you see the GRUB menu, use the arrow keys to select **Advanced options for Cumulus Linux GNU/Linux**.

⚠ **IMPORTANT**

Before the GRUB menu appears, the switch goes through the boot cycle. Do **not** interrupt this autoboot process when you see the following lines; wait until you see the GRUB menu.

```
...
CLOCKS:ARM Core=1000Hz, AXI=500Hz, APB=125Hz,
Peripheral=500Hz
USB0:  Bringing USB2 host out of reset...
Net:   eth-0
SF:    MX25L6405D with page size 4 KiB, total 8 MiB
```

```
Hit any key to stop autoboot:  2
```

```
             GNU GRUB  version 2.02+dfsg1-20

+----------------------------------------------------------------------------
+
|*Cumulus Linux GNU/
Linux                                                        |
| Advanced options for Cumulus Linux GNU/
Linux                                  |
|
ONIE
|
|
|
+----------------------------------------------------------------------------
+
```

2. Select **Cumulus Linux GNU/Linux, with Linux 4.19.0-cl-1-amd64**

(recovery mode).

```
             GNU GRUB  version 2.02+dfsg1-20

+----------------------------------------------------------------------------
+
| Cumulus Linux GNU/Linux, with Linux
4.19.0-cl-1-amd64                          |
|*Cumulus Linux GNU/Linux, with Linux 4.19.0-cl-1-amd64
(recovery mode)        |
|
|
+----------------------------------------------------------------------------
+
```

3. Press **ctrl-d** to reboot.

4. After the system reboots, set a new **root** password. The root user provides complete control over the switch.

```
root@switch:~# passwd
Enter new UNIX password:
Retype new UNIX password:
```

```
passwd: password updated successfully
```

TIP

You can take this opportunity to reset the password for the *cumulus* account.

```
root@switch:~# passwd cumulus
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

5. Sync the `/etc` directory, then reboot the system:

```
root@switch:~# sync
root@switch:~# reboot -f
Restarting the system.
```

Resource Diagnostics Using cl-resource-query

You can use the `cl-resource-query` command or the NCLU `net show system asic` command to retrieve information about host entries, MAC entries, layer 2 and layer 3 routes, and ECMP routes that are in use. Because Cumulus Linux synchronizes routes between the kernel and the switching silicon, if the required resource pools in hardware fill up, new kernel routes can cause existing routes to move from being fully allocated to being partially allocated. To avoid this, monitor the routes in the hardware to keep them below the ASIC limits. For example, on a Broadcom Tomahawk switch, the limits are as follows:

```
routes: 8192 <<<< if all routes are IPv6, or 65536 if all
routes are IPv4
route mask limit 64
host_routes: 73728
ecmp_nhs: 16327
ecmp_nhs_per_route: 52
```

This translates to about 314 routes with ECMP nexthops, if every route has the maximum ECMP nexthops.

To monitor the routes in Cumulus Linux hardware, use the `cl-resource-query` command. The results ary between switches running on different

chipsets.

The example below shows `cl-resource-query` results for a Broadcom Tomahawk switch:

```
cumulus@switch:~$ sudo cl-resource-query
IPv4/IPv6 host entries:                 0,   0% of maximum
value  40960
IPv4 neighbors:                         0
IPv6 neighbors:                         0
IPv4 route entries:                     4,   0% of maximum
value  65536
IPv6 route entries:                     8,   0% of maximum
value   8192
IPv4 Routes:                            4
IPv6 Routes:                            8
Total Routes:                          12,   0% of maximum
value  65536
ECMP nexthops:                          0,   0% of maximum
value  16327
MAC entries:                            1,   0% of maximum
value  40960
Total Mcast Routes:                     0,   0% of maximum
value  20480
Ingress ACL entries:                  195,  12% of maximum
```

```
value   1536
Ingress ACL counters:                195,  12% of maximum
value   1536
Ingress ACL meters:                   21,   1% of maximum
value   2048
Ingress ACL slices:                    6, 100% of maximum
value      6
Egress ACL entries:                   58,  11% of maximum
value    512
Egress ACL counters:                  58,   5% of maximum
value   1024
Egress ACL meters:                    29,   5% of maximum
value    512
Egress ACL slices:                     2, 100% of maximum
value      2
Ingress ACL ipv4_mac filter table:    36,  14% of maximum
value    256 (allocated: 256)
Ingress ACL ipv6 filter table:        29,  11% of maximum
value    256 (allocated: 256)
Ingress ACL mirror table:              0,   0% of maximum
value      0 (allocated: 0)
Ingress ACL 8021x filter table:        0,   0% of maximum
value      0 (allocated: 0)
Ingress PBR ipv4_mac filter table:     0,   0% of maximum
```

```
value      0 (allocated: 0)
Ingress PBR ipv6 filter table:          0,   0% of maximum
value      0 (allocated: 0)
Ingress ACL ipv4_mac mangle table:      0,   0% of maximum
value      0 (allocated: 0)
Ingress ACL ipv6 mangle table:          0,   0% of maximum
value      0 (allocated: 0)
Egress ACL ipv4_mac filter table:      29,  11% of maximum
value    256 (allocated: 256)
Egress ACL ipv6 filter table:           0,   0% of maximum
value      0 (allocated: 0)
ACL L4 port range checkers:             2,   6% of maximum
value     32
```

The example below shows `cl-resource-query` results for a Broadcom Trident II switch:

```
cumulus@switch:~$ sudo cl-resource-query
IPv4/IPv6 host entries:                 0,   0% of maximum
value  16384
IPv4 neighbors:                         0
IPv6 neighbors:                         0
IPv4 route entries:                     0,   0% of maximum
```

```
value 131072
IPv6 route entries:                      1,   0% of maximum
value  20480
IPv4 Routes:                             0
IPv6 Routes:                             1
Total Routes:                            1,   0% of maximum
value 131072
ECMP nexthops:                           0,   0% of maximum
value  16346
MAC entries:                             0,   0% of maximum
value  32768
Total Mcast Routes:                      0,   0% of maximum
value   8192
Ingress ACL entries:                   130,   6% of maximum
value   2048
Ingress ACL counters:                   86,   4% of maximum
value   2048
Ingress ACL meters:                     21,   0% of maximum
value   4096
Ingress ACL slices:                      4,  66% of maximum
value      6
Egress ACL entries:                     58,  11% of maximum
value    512
Egress ACL counters:                    58,   5% of maximum
```

```
value   1024
Egress ACL meters:                    29,   5% of maximum
value    512
Egress ACL slices:                     2, 100% of maximum
value      2
Ingress ACL ipv4_mac filter table:    36,   7% of maximum
value    512 (allocated: 256)
Ingress ACL ipv6 filter table:        29,   3% of maximum
value    768 (allocated: 512)
Ingress ACL mirror table:              0,   0% of maximum
value      0 (allocated: 0)
Ingress ACL 8021x filter table:        0,   0% of maximum
value      0 (allocated: 0)
Ingress PBR ipv4_mac filter table:     0,   0% of maximum
value      0 (allocated: 0)
Ingress PBR ipv6 filter table:         0,   0% of maximum
value      0 (allocated: 0)
Ingress ACL ipv4_mac mangle table:     0,   0% of maximum
value      0 (allocated: 0)
Ingress ACL ipv6 mangle table:         0,   0% of maximum
value      0 (allocated: 0)
Egress ACL ipv4_mac filter table:     29,  11% of maximum
value    256 (allocated: 256)
Egress ACL ipv6 filter table:          0,   0% of maximum
```

```
value      0 (allocated: 0)
ACL L4 port range checkers:              2,   8% of maximum
value     24
```

> **NOTE**
>
> On a switch with a Spectrum ASIC, the `cl-resource-query` command shows the number of TCAM entries used by the different types of ACL resources.

The example below shows `cl-resource-query` results for a NVIDIA Mellanox Spectrum switch:

```
cumulus@switch:~$ sudo cl-resource-query
IPv4 host entries:                       0,   0% of maximum
value  32768
IPv6 host entries:                       0,   0% of maximum
value  16384
IPv4 neighbors:                          0
IPv6 neighbors:                          0
IPv4 route entries:                      0,   0% of maximum
```

```
value  65536
IPv6 route entries:                     7,   0% of maximum
value  28672
IPv4 Routes:                            0
IPv6 Routes:                            7
Total Routes:                           7,   0% of maximum
value  94208
ECMP nexthops:                          0,   0% of maximum
value   4101
MAC entries:                            0,   0% of maximum
value  40960
Total Mcast Routes:                     0,   0% of maximum
value    400
Ingress ACL entries:                    0,   0% of maximum
value      0
Ingress ACL counters:                   0,   0% of maximum
value      0
Ingress ACL meters:                     0,   0% of maximum
value      0
Ingress ACL slices:                     0,   0% of maximum
value      0
Egress ACL entries:                     0,   0% of maximum
value      0
Egress ACL counters:                    0,   0% of maximum
```

```
value      0
Egress ACL meters:                      0,   0% of maximum
value      0
Egress ACL slices:                      0,   0% of maximum
value      0
Ingress ACL ipv4_mac filter table:      0,   0% of maximum
value      0 (allocated: 0)
Ingress ACL ipv6 filter table:          0,   0% of maximum
value      0 (allocated: 0)
Ingress ACL mirror table:               0,   0% of maximum
value      0 (allocated: 0)
Ingress ACL 8021x filter table:         0,   0% of maximum
value      0 (allocated: 0)
Ingress PBR ipv4_mac filter table:      0,   0% of maximum
value      0 (allocated: 0)
Ingress PBR ipv6 filter table:          0,   0% of maximum
value      0 (allocated: 0)
Ingress ACL ipv4_mac mangle table:      0,   0% of maximum
value      0 (allocated: 0)
Ingress ACL ipv6 mangle table:          0,   0% of maximum
value      0 (allocated: 0)
Egress ACL ipv4_mac filter table:       0,   0% of maximum
value      0 (allocated: 0)
Egress ACL ipv6 filter table:           0,   0% of maximum
```

```
value      0 (allocated: 0)
ACL L4 port range checkers:              0,   0% of maximum
value      0
ACL Regions:                             4,   1% of maximum
value    400
ACL 18B Rules Key:                       2,   0% of maximum
value   2256
ACL 32B Rules Key:                       0,   0% of maximum
value   1024
ACL 54B Rules Key:                       2,   0% of maximum
value   1024
```

NOTE

Ingress ACL and Egress ACL entries show the counts in single wide (*not* double-wide). For information about ACL entries, see Estimate the Number of ACL Rules.

Monitoring Virtual Device Counters

Cumulus Linux gathers statistics for VXLANs and VLANs using virtual device counters. These counters are supported on Tomahawk, Trident II+ and Trident II-based platforms only; see the Cumulus Linux HCL for a list of supported platforms.

You can retrieve the data from these counters using tools like `ip -s link show`, `ifconfig`, `/proc/net/dev` or `netstat -i`.

Sample VXLAN Statistics

VXLAN statistics are available as follows:

- Aggregate statistics are available per VNI; this includes access and network statistics.
- Network statistics are available for each VNI and displayed against the VXLAN device. This is independent of the VTEP used, so this is a summary of the VNI statistics across all tunnels.
- Access statistics are available per VLAN subinterface.

To show interface information about the VXLAN bridge:

```
cumulus@switch:~$ brctl show br-vxln16757104
```

```
bridge name        bridge id            STP enabled
interfaces
-vxln16757104      8000.443839006988    no             swp2s0.6
                                                       swp2s1.6
                                                       swp2s2.6
                                                       swp2s3.6


vxln16757104
```

To show VNI statistics, run:

```
cumulus@switch:~$ ip -s link show br-vxln16757104
62: br-vxln16757104: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500
qdisc noqueue state UP mode DEFAULT
    link/ether 44:38:39:00:69:88 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    10848      158      0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    27816      541      0       0       0       0
```

To show access statistics, run:

```
cumulus@switch:~$ ip -s link show swp2s0.6
63: swp2s0.6@swp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500
qdisc noqueue master br-vxln16757104 state UP mode DEFAULT
    link/ether 44:38:39:00:69:88 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    2680       39       0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    7558       140      0       0       0       0
```

To show network statistics, run:

```
cumulus@switch:~$ ip -s link show vxln16757104
61: vxln16757104: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500
qdisc noqueue master br-vxln16757104 state UNKNOWN mode DEFAULT
    link/ether e2:37:47:db:f1:94 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    0          0        0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    0          0        0       9       0       0
```

Sample VLAN Statistics

For VLANs Using the VLAN-aware Bridge Mode Driver

For a bridge using the VLAN-aware bridge mode driver, the bridge is a just a container and each VLAN (VID/PVID) in the bridge is an independent layer 2 broadcast domain. As there is no `netdev` available to display these VLAN statistics, the `switchd` nodes are used instead:

```
cumulus@switch:~$ ifquery bridge
auto bridge
iface bridge inet static
  bridge-vlan-aware yes
  bridge-ports swp2s0 swp2s1
  bridge-stp on
  bridge-vids 2000-2002 4094

cumulus@switch:~$ ls /cumulus/switchd/run/stats/vlan/
2  2000  2001  2002  all

cumulus@switch:~$ cat /cumulus/switchd/run/stats/vlan/2000/
aggregate
Vlan id                         : 2000
L3 Routed In Octets             : -
L3 Routed In Packets            : -
```

```
L3 Routed Out Octets            : -
L3 Routed Out Packets           : -
Total In Octets                 : 375
Total In Packets                : 3
Total Out Octets                : 387
Total Out Packets               : 3
```

For VLANs Using the Traditional Bridge Mode Driver

For a bridge using the traditional bridge mode driver, each bridge is a single L2 broadcast domain and is associated with an internal VLAN. This internal VLAN's counters are displayed as bridge netdev stats.

```
cumulus@switch:~$ brctl show br0
bridge name   bridge id            STP enabled   interfaces
br0           8000.443839006989    yes           bond0.100
                                                 swp2s2.100
cumulus@switch:~$ ip -s link show br0
42: br0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
noqueue state UP mode DEFAULT
    link/ether 44:38:39:00:69:89 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    23201498   227514   0       0       0       0
```

```
    TX: bytes  packets  errors  dropped carrier collsns
    18198262   178443   0       0       0       0
```

Configure the Counters in switchd

These counters are enabled by default. To configure them, use `cl-cfg` and configure them as you would any other `switchd` parameter. The `switchd` parameters are:

- `stats.vlan.aggregate`, which controls the statistics available for each VLAN. Its value defaults to *BRIEF*.
- `stats.vxlan.aggregate`, which controls the statistics available for each VNI (access and network). Its value defaults to *DETAIL*.
- `stats.vxlan.member`, which controls the statistics available for each local/access port in a VXLAN bridge. Its value defaults to *BRIEF*.

The values for each parameter can be one of the following:

- NONE: This disables the counter.
- BRIEF: This provides tx/rx packet/byte counters for the associated parameter.
- DETAIL: This provides additional feature-specific counters. In the case of `stats.vxlan.aggregate`, DETAIL provides access vs. network statistics. For the other types, DETAIL has the same effect as BRIEF.

> **NOTE**
>
> If you change one of these settings on the fly, the new configuration applies only to those VNIs or VLANs set up after the configuration changed; previously allocated counters remain as is.

Configure the Poll Interval

The virtual device counters are polled periodically. This can be CPU intensive, so the interval is configurable in `switchd`, with a default of 2 seconds.

```
# Virtual devices hw-stat poll interval (in seconds)
#stats.vdev_hw_poll_interval = 2
```

Configure Internal VLAN Statistics

For debugging purposes, you can access packet statistics associated with internal VLAN IDs. These statistics are hidden by default, but you can configure them in `switchd`:

```
#stats.vlan.show_internal_vlans = FALSE
```

Clear Statistics

Because `ethtool` is not supported for virtual devices, you *cannot* clear the statistics cache maintained by the kernel. You can clear the hardware statistics via `switchd`:

```
cumulus@switch:~$ sudo echo 1 > /cumulus/switchd/clear/stats/
vlan
cumulus@switch:~$ sudo echo 1 > /cumulus/switchd/clear/stats/
vxlan
```

Considerations

- Currently the CPU port is internally added as a member of all VLANs.Therefore, packets sent to the CPU are counted against the corresponding VLAN's tx packets/bytes. There is no workaround.
- When checking the virtual counters for the bridge, the TX count is the number of packets destined to the CPU before any hardware policers take effect. For example, if 500 broadcast packets are sent into the bridge, the CPU is also sent 500 packets. These 500 packets are policed by the default ACLs in Cumulus Linux, so the CPU might receive fewer than the 500 packets if the incoming packet rate is too high. The TX counter for the bridge should be equal to 500*(number of ports in the bridge - incoming port + CPU port) or just 500 * number of ports in the bridge.

- You cannot use `ethtool -S` for virtual devices. This is because the counters available via `netdev` are sufficient to display the VLAN/VXLAN counters currently supported in the hardware (only rx/tx packets/bytes are supported currently).

ASIC Monitoring

Cumulus Linux provides an ASIC monitoring tool that collects and distributes data about the state of the ASIC. The monitoring tool polls for data at specific intervals and takes certain actions so that you can quickly identify and respond to problems, such as:

- Microbursts that result in longer packet latency
- Packet buffer congestion that might lead to packet drops
- Network problems with a particular switch, port, or traffic class

You can collect the following type of statistics with the ASIC monitoring tool:

- A fine-grained history of queue lengths using histograms maintained by the ASIC
- Packet counts per port, priority and size
- Dropped packet, pause frame, and ECN-marked packet counts
- Buffer congestion occupancy per port, priority and buffer pool, and at input and output ports

> **NOTE**
>
> ASIC monitoring is currently supported on switches with Spectrum ASICs only.

Collecting Queue Lengths in Histograms

The Mellanox Spectrum ASIC provides a mechanism to measure and report egress queue lengths in histograms (a graphical representation of data, which is divided into intervals or bins). You can configure the ASIC to measure up to 64 egress queues. Each queue is reported through a histogram with 10 bins, where each bin represents a range of queue lengths.

You configure the histogram with a minimum size boundary (Min) and a histogram size. You then derive the maximum size boundary (Max) by adding the minimum size boundary and the histogram size.

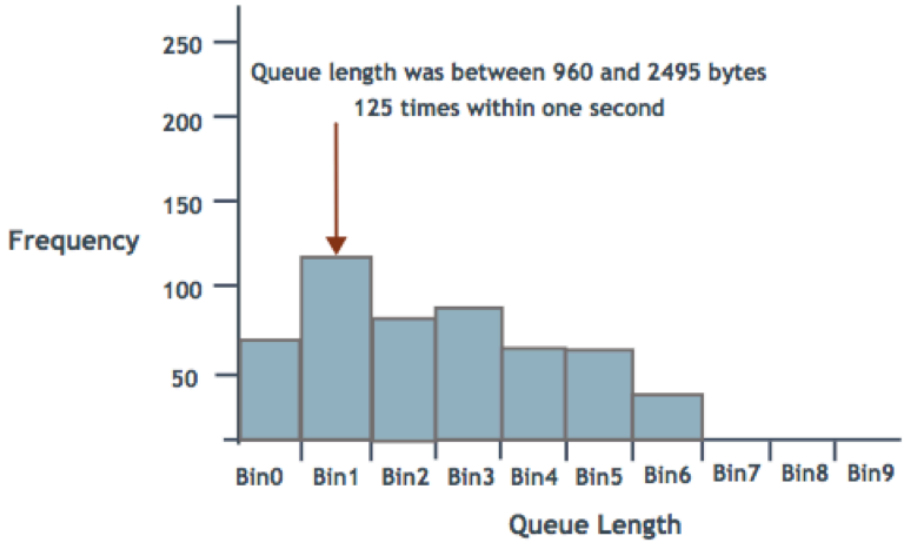
The 10 bins are numbered 0 through 9. Bin 0 represents queue lengths up to the Min specified, including queue length 0. Bin 9 represents queue lengths of Max and above. Bins 1 through 8 represent equal-sized ranges between the Min and Max, which is determined by dividing the histogram size by 8.

For example, consider the following histogram queue length ranges, in bytes:

- Min = 960
- Histogram size = 12288
- Max = 13248
- Range size = 1536
- Bin 0: 0:959
- Bin 1: 960:2495

- Bin 2: 2496:4031
- Bin 3: 4032:5567
- Bin 4: 5568:7103
- Bin 5: 7104:8639
- Bin 6: 8640:10175
- Bin 7: 10176:11711
- Bin 8: 11712:13247
- Bin 9: 13248:*

The following illustration demonstrates a histogram showing how many times the queue length for a port was in the ranges specified by each bin. The example shows that the queue length was between 960 and 2495 bytes 125 times within one second.



Bin0 = 0:959
Bin1 = 960:2495
Bin2 = 2496:4031
Bin3 = 4032:5567
Bin4 = 5568:7103
Bin5 = 7104:8639
Bin6 = 8640:10175
Bin7 = 10176:11711
Bin8 = 11712:13247
Bin9 = 13248:*

Configure ASIC Monitoring

The ASIC monitoring tool is managed by the `asic-monitor` service, (which is managed by `systemd`). The `asic-monitor` service reads the `/etc/cumulus/datapath/monitor.conf` configuration file to determine what statistics to collect and when to trigger. The service always starts; however, if the configuration file is empty, the service exits.

The `monitor.conf` configuration file provides information about the type of data to collect, the switch ports to monitor, how and when to start reading the ASIC (such as when a specific queue length or number of packets dropped is reached), and what actions to take (create a snapshot file, send a message to the `/var/log/syslog` file, or collect more data).

To configure ASIC monitoring, edit the `/etc/cumulus/datapath/monitor.conf` file and restart the `asic-monitor` service. The `asic-monitor` service reads the new configuration file and then runs until it is stopped.

The following procedure describes how to monitor queue lengths using a histogram. The settings are configured to collect data every second and write the results to a snapshot file. When the size of the queue reaches 500 bytes, the system sends a message to the `/var/log/syslog` file.

To monitor queue lengths using a histogram:

1. Open the `/etc/cumulus/datapath/monitor.conf`file in a text editor.

```
cumulus@switch:~$ sudo nano /etc/cumulus/datapath/monitor.conf
```

2. At the end of the file, add the following line to specify the name of the histogram monitor (port group). The example uses `histogram_pg`; however, you can use any name you choose. You must use the same name with all histogram settings.

```
monitor.port_group_list = [histogram_pg]
```

3. Add the following line to specify the ports you want to monitor. The following example sets swp1 through swp50.

```
monitor.histogram_pg.port_set = swp1-swp50
```

4. Add the following line to set the data type to `histogram`. This is the data type for histogram monitoring.

```
monitor.histogram_pg.stat_type = histogram
```

5. Add the following line to set the trigger type to `timer`. Currently, the only

trigger type available is timer.

```
monitor.histogram_pg.trigger_type = timer
```

6. Add the following line to set the frequency at which data collection starts. In the following example, the frequency is set to one second.

```
monitor.histogram_pg.timer = 1s
```

7. Add the following line to set the actions you want to take when data is collected. In the following example, the system writes the results of data collection to a snapshot file and sends a message to the `/var/log/syslog` file.

```
monitor.histogram_pg.action_list = [snapshot,log]
```

8. Add the following line to specify a name and location for the snapshot file. In the following example, the system writes the snapshot to a file called `histogram_stats` in the `/var/lib/cumulus` directory and adds a suffix to the file name with the snapshot file count (see the following step).

```
monitor.histogram_pg.snapshot.file = /var/lib/cumulus/
histogram_stats
```

9. Add the following line to set the number of snapshots that are taken before the system starts overwriting the earliest snapshot files. In the following example, because the snapshot file count is set to 64, the first snapshot file is named `histogram_stats_0` and the 64th snapshot is named `histogram_stats_63`. When the 65th snapshot is taken, the original snapshot file (`histogram_stats_0`) is overwritten and the sequence continues until `histogram_stats_63` is written. Then, the sequence restarts.

```
monitor.histogram_pg.snapshot.file_count = 64
```

10. Add the following line to include a threshold, which determines how to collect data. Setting a threshold is optional. In the following example, when the size of the queue reaches 500 bytes, the system sends a message to the `/var/log/syslog` file.

```
monitor.histogram_pg.log.queue_bytes = 500
```

11. Add the following lines to set the size, minimum boundary, and sampling time of the histogram. Adding the histogram size and the minimum boundary size together produces the maximum boundary size. These settings are used to represent the range of queue lengths per bin.

```
monitor.histogram_pg.histogram.minimum_bytes_boundary = 960
monitor.histogram_pg.histogram.histogram_size_bytes   = 12288
monitor.histogram_pg.histogram.sample_time_ns         = 1024
```

12. Save the file, then restart the `asic-monitor` service with the following command:

```
cumulus@switch:~$ systemctl restart asic-monitor.service
```

> **NOTE**
>
> Restarting the `asic-monitor` service does not disrupt traffic or require you to restart `switchd`. The service is enabled by default when you boot the switch and restarts when you restart `switchd`.

> **NOTE**
>
> Overhead is involved in collecting the data, which uses both the CPU and SDK process and can affect execution of `switchd`. Snapshots and logs can occupy a lot of disk space if you do not limit their number.

To collect other data, such as all packets per port, buffer congestion, or packet drops due to error, follow the procedure above but change the port group list setting to include the port group name you want to use. For example, to monitor packet drops due to buffer congestion:

```
monitor.port_group_list = [buffers_pg]
monitor.buffers_pg.port_set  = swp1-swp50
monitor.buffers_pg.stat_type = buffer
...
```

Certain settings in the procedure above (such as the histogram size, boundary size, and sampling time) only apply to the histogram monitor. All ASIC monitor settings are described in ASIC Monitoring.

ASIC Monitoring Settings

The following table provides descriptions of the ASIC monitor settings.

Setting	Description
`port_group_list`	Specifies the names of the monitors (port groups) you want to use to collect data, such as `discards_pg`, `histogram_pg`, `all_packet_pg`, `buffers_pg`. You can provide any name you want for the port group; the names above are just examples. You must use the same name for all the settings of a particular port group. Example: `monitor.port_group_list = [histogram_pg,discards_pg,buffers_pg, all_packets_pg]` **Note**: You must specify at least one port group. If the port group list is empty, systemd shuts down the asic-monitor service.
`<port_group_name>.port_set`	Specifies the range of ports monitored. You can specify GLOBs and comma-separated lists; for example, `swp1 swp4,swp8,swp10-swp50`. Example:

Setting	Description
	`monitor.histogram_pg.port_set = swp1-swp50`

`<port_group_name>.stat_type`	Specifies the type of data that the port group collects.
	
	For histograms, specify histogram. For example:
	
	`monitor.histogram_pg.stat_type = histogram`
	
	For packet drops due to errors, specify packet. For example:
	
	`monitor.discards_pg.stat_type = packet`
	
	For packet occupancy statistics, specify buffer. For example:

Setting	Description

	`monitor.buffers_pg.stat_type = buffer`
	
	For all packets per port, specify `packet_all`.
	Example:
	
	`monitor.all_packet_pg.stat_type = packet_all`

`<port_group_name>.cos_list`	For histogram monitoring, each CoS (Class of Service) value in the list has its own histogram on each port. The global limit on the number of histograms is an average of one histogram per port. Example: `monitor.histogram_pg.cos_list = [0]`

Setting	Description
`<port_group_name>.trigger_type`	Specifies the type of trigger that initiates data collection. Currently, the only option is timer. At least one port group must have a timer configured, otherwise no data is ever collected. Example: `monitor.histogram_pg.trigger_type = timer`
`<port_group_name>.timer`	Specifies the frequency at which data is collected; for example, a setting of 1s indicates that data is collected once per second. You can set the timer to the following: 1 to 60 seconds: 1s, 2s, and so on up to 60s 1 to 60 minutes: 1m, 2m, and so on up to 60m 1 to 24 hours: 1h, 2h, and so on up to 24h 1 to 7 days: 1d, 2d and so on up to 7d Example:

Setting	Description
	`monitor.histogram_pg.timer = 4s`

`<port_group_name>.action_list`	Specifies one or more actions that occur when data is collected:
	`snapshot` writes a snapshot of the data collection results to a file. If you specify this action, you must also specify a snapshot file (described below). You can also specify a threshold that initiates the snapshot action, but this is not required.
	
	Example:
	
	`monitor.histogram_pg.action_list = [snapshot]`
	`monitor.histogram_pg.snapshot.file = /var/lib/cumulus/histogram_stats`
	
	`collect` gathers additional data. If you specify this action, you must also specify the port

Setting	Description

	groups for the additional data you want to collect.
	
	Example:
	
	`monitor.histogram_pg.action_list = [collect monitor.histogram_pg.collect.port_group_l = [buffers_pg,all_packet_pg]`
	
	`log` sends a message to the `/var/log/syslog` file. If you specify this action, you must also specify a threshold that initiates the log action.
	Example:
	
	`monitor.histogram_pg.action_list = [log] monitor.histogram_pg.log.queue_bytes = 500`
	
	You can use all three of these actions in one monitoring step. For example

Setting	Description
	`monitor.histogram_pg.action_list = [snapshot,collect,log]` **Note**: If an action appears in the action list but does not have the required settings (such as a threshold for the log action), the ASIC monitor stops and reports an error.
`<port_group_name>.snapshot.file`	Specifies the name for the snapshot file. All snapshots use this name, with a sequential number appended to it. See the `snapshot.file_count` setting. Example: `monitor.histogram_pg.snapshot.file = /var/lib/cumulus/histogram_stats`
`<port_group_name>.snapshot.file_count`	Specifies the number of snapshots that can be created before the first snapshot file is overwritten. In the following

Setting	Description
	example, because the snapshot file count is set to 64, the first snapshot file is named histogram_stats_0 and the 64th snapshot is named histogram_stats_63. When the 65th snapshot is taken, the original snapshot file (histogram_stats_0) is overwritten and the sequence restarts. Example: `monitor.histogram_pg.snapshot.file_count = 64` **Note**: While more snapshots provide you with more data, they can occupy a lot of disk space on the switch.
`<port_group_name>.<action>.queue_bytes`	*For histogram monitoring*. Specifies a threshold for the histogram monitor. This is the length of the queue in bytes that initiates a specified action (snapshot, log, collect). Examples:

Setting	Description

	`monitor.histogram_pg.snapshot.queue_bytes = 500`
	`monitor.histogram_pg.log.queue_bytes = 500`
	`monitor.histogram_pg.collect.queue_bytes = 500`

`<port_group_name>.<action>.packet_error_drops`	*For monitoring packet drops due to error.*
	
	Specifies a threshold for the packet drops due to error monitor. This is the number of packet drops due to error that initiates a specified action (snapshot, log, collect).
	
	Examples:
	
	`monitor.discards_pg.snapshot.packet_error = 500`
	`monitor.discards_pg.log.packet_error_drop = 500`
	`monitor.discards_pg.collect.packet_error_ = 500`

Setting	Description

`<port_group_name>.<action>.packet_congestion_packets`	*For monitoring packet drops due to buffer congestion.*
	
	Specifies a threshold for the packet drops due to buffer congestion monitor. This is the number of packet drops due to buffer congestion that initiates a specified action (log or collect).
	
	Examples:
	
	`monitor.buffers_pg.log.packet_congestion_ = 500`
	`monitor.buffers_pg.snapshot.packet_conges = 500`
	`monitor.buffers_pg.collect.packet_congest = 500`

`<port_group_name>.histogram.minimum_histogram_boundary`	*For histogram monitoring.* The minimum boundary size for the histogram in bytes. On a Spectrum switch, this number must be a multiple of 96. Adding this number to the size of the histogram produces the maximum boundary size. These values are used to represent the range of queue lengths per bin.

Setting	Description

	Example:
	
	`monitor.histogram_pg.histogram.minimum_by`
	`= 960`

`<port_group_name>.histogram.histogram_bytes`	*For histogram monitoring.*
	
	The size of the histogram in bytes. Adding this number and the minimum_bytes_boundary value together produces the maximum boundary size. These values are used to represent the range of queue lengths per bin.
	
	Example:
	
	`monitor.histogram_pg.histogram.histogram_`
	`= 12288`

`<port_group_name>.histogram.sample_histogram`	*For histogram monitoring.* The sampling time of the histogram in nanoseconds. Example:

Setting	Description
	`monitor.histogram_pg.histogram.sample_tim = 1024`

Example Configurations

Several configuration examples are provided below.

Queue Length Histograms

In the following example:

- Queue length histograms are collected every second for swp1 through swp50.
- The results are written to the `/var/lib/cumulus/histogram_stats` snapshot file.
- The size of the histogram is set to 12288 bytes, the minimum boundary to 960 bytes, and the sampling time to 1024 nanoseconds.
- A threshold is set so that when the size of the queue reaches 500 bytes, the system sends a message to the `/var/log/syslog` file.

```
monitor.port_group_list                                =
```

```
[histogram_pg]
monitor.histogram_pg.port_set                          =
swp1-swp50
monitor.histogram_pg.stat_type                         =
histogram
monitor.histogram_pg.cos_list                          = [0]
monitor.histogram_pg.trigger_type                      = timer
monitor.histogram_pg.timer                             = 1s
monitor.histogram_pg.action_list                       =
[snapshot,log]
monitor.histogram_pg.snapshot.file                     = /var/
lib/cumulus/histogram_stats
monitor.histogram_pg.snapshot.file_count               = 64
monitor.histogram_pg.log.queue_bytes                   = 500
monitor.histogram_pg.histogram.minimum_bytes_boundary = 960
monitor.histogram_pg.histogram.histogram_size_bytes    = 12288
monitor.histogram_pg.histogram.sample_time_ns          = 1024
```

Packet Drops Due to Errors

In the following example:

- Packet drops on swp1 through swp50 are collected every two seconds.
- If the number of packet drops is greater than 100, the results are written to the `/var/lib/cumulus/discard_stats snapshot` file and the system

sends a message to the `/var/log/syslog` file.

```
monitor.port_group_list                              =
[discards_pg]
monitor.discards_pg.port_set                         = swp1-swp50
monitor.discards_pg.stat_type                        = packet
monitor.discards_pg.action_list                      =
[snapshot,log]
monitor.discards_pg.trigger_type                     = timer
monitor.discards_pg.timer                            = 2s
monitor.discards_pg.log.packet_error_drops           = 100
monitor.discards_pg.snapshot.packet_error_drops      = 100
monitor.discards_pg.snapshot.file                    = /var/lib/
cumulus/discard_stats
monitor.discards_pg.snapshot.file_count              = 16
```

Queue Length (Histogram) with Collect Actions

A collect action triggers the collection of additional information. You can daisy chain multiple monitors (port groups) into a single collect action.

In the following example:

- Queue length histograms are collected for swp1 through swp50 every second.
- The results are written to the `/var/lib/cumulus/histogram_stats`

snapshot file.

- When the queue length reaches 500 bytes, the system sends a message to the `/var/log/syslog` file and collects additional data; buffer occupancy and all packets per port.
- Buffer occupancy data is written to the `/var/lib/cumulus/buffer_stats` snapshot file and all packets per port data is written to the `/var/lib/cumulus/all_packet_stats` snapshot file.
- In addition, packet drops on swp1 through swp50 are collected every two seconds. If the number of packet drops is greater than 100, the results are written to the `/var/lib/cumulus/discard_stats` snapshot file and a message is sent to the `/var/log/syslog` file.

```
monitor.port_group_list                                =
[histogram_pg,discards_pg]

monitor.histogram_pg.port_set                          =
swp1-swp50
monitor.histogram_pg.stat_type                         = buffer
monitor.histogram_pg.cos_list                          = [0]
monitor.histogram_pg.trigger_type                      = timer
monitor.histogram_pg.timer                             = 1s
monitor.histogram_pg.action_list                       =
[snapshot,collect,log]
monitor.histogram_pg.snapshot.file                     = /var/
```

```
lib/cumulus/histogram_stats
monitor.histogram_pg.snapshot.file_count              = 64
monitor.histogram_pg.histogram.minimum_bytes_boundary = 960
monitor.histogram_pg.histogram.histogram_size_bytes   = 12288
monitor.histogram_pg.histogram.sample_time_ns         = 1024
monitor.histogram_pg.log.queue_bytes                  = 500
monitor.histogram_pg.collect.queue_bytes              = 500
monitor.histogram_pg.collect.port_group_list          =
[buffers_pg,all_packet_pg]

monitor.buffers_pg.port_set                           =
swp1-swp50
monitor.buffers_pg.stat_type                          = buffer
monitor.buffers_pg.action_list                        =
[snapshot]
monitor.buffers_pg.snapshot.file                      = /var/
lib/cumulus/buffer_stats
monitor.buffers_pg.snapshot.file_count                = 8

monitor.all_packet_pg.port_set                        =
swp1-swp50
monitor.all_packet_pg.stat_type                       =
packet_all
monitor.all_packet_pg.action_list                     =
```

```
[snapshot]
monitor.all_packet_pg.snapshot.file                    = /var/
lib/cumulus/all_packet_stats
monitor.all_packet_pg.snapshot.file_count              = 8

monitor.discards_pg.port_set                           =
swp1-swp50
monitor.discards_pg.stat_type                          = packet
monitor.discards_pg.action_list                        =
[snapshot,log]
monitor.discards_pg.trigger_type                       = timer
monitor.discards_pg.timer                              = 2s
monitor.discards_pg.log.packet_error_drops             = 100
monitor.discards_pg.snapshot.packet_error_drops        = 100
monitor.discards_pg.snapshot.file                      = /var/
lib/cumulus/discard_stats
monitor.discards_pg.snapshot.file_count                = 16
```

NOTE

Certain actions require additional settings. For example, if the `snapshot` action is specified, a snapshot file is also required. If the `log` action is specified, a log threshold is also required. See

action_list for additional settings required for each *action*.

Example Snapshot File

A snapshot action writes a snapshot of the current state of the ASIC to a file. Because parsing the file and finding the information can be tedious, you can use a third-party analysis tool to analyze the data in the file. The following example shows a snapshot of queue lengths.

```
{"timestamp_info": {"start_datetime": "2017-03-16
21:36:40.775026", "end_datetime": "2017-03-16
21:36:40.775848"}, "buffer_info": null, "packet_info": null,
"histogram_info": {"swp2": {"0": 55531}, "swp32": {"0": 48668},
"swp1": {"0": 64578}}}
```

Example Log Message

A log action writes out the ASIC state to the `/var/log/syslog` file. In the following example, when the size of the queue reaches 500 bytes, the system sends this message to the `/var/log/syslog` file:

```
2018-02-26T20:14:41.560840+00:00 cumulus asic-monitor-module
INFO:  2018-02-26 20:14:41.559967: Egress queue(s) greater than
500 bytes in monitor port group histogram_pg.
```

Monitoring Best Practices

The following monitoring processes are considered best practices for reviewing and troubleshooting potential issues with Cumulus Linux environments. In addition, several of the more common issues have been listed, with potential solutions included.

This document describes:

- Metrics that you can poll from Cumulus Linux and use in trend analysis
- Critical log messages that you can monitor for triggered alerts

Trend Analysis Using Metrics

A metric is a quantifiable measure that is used to track and assess the status of a specific infrastructure component. It is a check collectedover time. Examples of metrics include bytes on an interface, CPU utilization, and total number of routes.

Metrics are more valuable when used for trend analysis.

Generate Alerts with Triggered Logging

Triggered issues are normally sent to `syslog`, but can go to another log file depending on the feature. In Cumulus Linux, `rsyslog` handles all logging, including local and remote logging. Logs are the best method to use for generating alerts when the system transitions from a stable steady state.

Sending logs to a centralized collector, then creating alerts based on critical logs is an optimal solution for alerting.

Log Formatting

Most log files in Cumulus Linux use a standard presentation format. For example, consider this `syslog` entry:

```
2017-03-08T06:26:43.569681+00:00 leaf01 sysmonitor: Critically
high CPU use: 99%
```

- *2017-03-08T06:26:43.569681+00:00* is the timestamp.
- *leaf01* is the hostname.
- *sysmonitor* is the process that is the source of the message.
- *Critically high CPU use: 99%* is the message.

For brevity and legibility, the timestamp and hostname have been omitted from the examples in this chapter.

Hardware

The `smond` process provides monitoring functionality for various switch hardware elements. Minimum or maximum values are output depending on the flags applied to the basic command. The hardware elements and applicable commands and flags are listed in the table below.

Hardware Element	Monitoring Commands	Interval Poll
Temperature	`cumulus@switch:~$ smonctl -j` `cumulus@switch:~$ smonctl -j -s TEMP[X]`	10 seconds
Fan	`cumulus@switch:~$ smonctl -j` `cumulus@switch:~$ smonctl -j -s FAN[X]`	10 seconds
PSU	`cumulus@switch:~$ smonctl -j` `cumulus@switch:~$ smonctl -j -s PSU[X]`	10 seconds

Hardware Element	Monitoring Commands	Interval Poll
PSU Fan	`cumulus@switch:~$ smonctl -j` `cumulus@switch:~$ smonctl -j -s PSU[X]Fan[X]`	10 seconds
PSU Temperature	`cumulus@switch:~$ smonctl -j` `cumulus@switch:~$ smonctl -j -s PSU[X]Temp[X]`	10 seconds
Voltage	`cumulus@switch:~$ smonctl -j` `cumulus@switch:~$ smonctl -j -s Volt[X]`	10 seconds

Hardware Element	Monitoring Commands	Interval Poll

Front Panel LED	`cumulus@switch:~$ ledmgrd -d`
	`cumulus@switch:~$ ledmgrd -j`
	
	You can also run `net show system leds`, which is the NCLU command equivalent of `ledmgrd -d`.
		5 seconds

NOTE

Not all switch models include a sensor for monitoring power consumption and voltage. See this note for details.

Hardware Logs	Log Location	Log Entries

High temperature	`/var/log/syslog`	`/usr/sbin/smond : : Temp1(Board Sensor near CPU): state changed from UNKNOWN to OK`
		`/usr/sbin/smond : : Temp2(Board Sensor Near Virtual Switch): state changed from UNKNOWN to OK`
		`/usr/sbin/smond : : Temp3(Board Sensor at Front Left Corner): state changed from`

Hardware Logs	Log Location	Log Entries
		`UNKNOWN to OK /usr/sbin/ smond : : Temp4(Board Sensor at Front Right Corner): state changed from UNKNOWN to OK /usr/sbin/ smond : : Temp5(Board Sensor near Fan): state changed from UNKNOWN to OK`
Fan speed issues	`/var/log/`	`/usr/sbin/`

Hardware Logs	Log Location	Log Entries
	`syslog`	`smond : : Fan1(Fan Tray 1, Fan 1): state changed from UNKNOWN to OK /usr/sbin/ smond : : Fan2(Fan Tray 1, Fan 2): state changed from UNKNOWN to OK /usr/sbin/ smond : : Fan3(Fan Tray 2, Fan 1): state changed from UNKNOWN to OK /usr/sbin/ smond : : Fan4(Fan`

Hardware Logs	Log Location	Log Entries
		Tray 2, Fan 2): state changed from UNKNOWN to OK /usr/sbin/ smond : : Fan5(Fan Tray 3, Fan 1): state changed from UNKNOWN to OK /usr/sbin/ smond : : Fan6(Fan Tray 3, Fan 2): state changed from UNKNOWN to OK

Hardware Logs	Log Location	Log Entries
PSU failure	`/var/log/syslog`	`/usr/sbin/smond : : PSU1Fan1(PSU1 Fan): state changed from UNKNOWN to OK` `/usr/sbin/smond : : PSU2Fan1(PSU2 Fan): state changed from UNKNOWN to BAD`

System Data

Cumulus Linux includes a number of ways to monitor various aspects of system data. In addition, alerts are issued in high risk situations.

CPU Idle Time

When a CPU reports five high CPU alerts within a span of five minutes, an alert is logged.

WARNING

Short bursts of high CPU can occur during `switchd` churn or routing protocol startup. Do not set alerts for these short bursts.

System Element	Monitoring Commands	Interval Poll
CPU utilization	`cumulus@switch:~$ cat /proc/stat` `cumulus@switch:~$ top -b -n 1`	30 seconds

CPU Logs	Log Location	Log Entries
High CPU	`/var/log/syslog`	`sysmonitor: Critically high CPU`

CPU Logs	Log Location	Log Entries
		use: 99% systemd[1]: Starting Monitor system resources (cpu, memory, disk)… systemd[1]: Started Monitor system resources (cpu, memory, disk). sysmonitor: High CPU use: 89% systemd[1]: Starting Monitor system resources (cpu, memory, disk)… systemd[1]:

CPU Logs	Log Location	Log Entries
		`Started Monitor system resources (cpu, memory, disk). sysmonitor: CPU use no longer high: 77%`

Cumulus Linux 3.0 and later monitors CPU, memory, and disk space via `sysmonitor`. The configurations for the thresholds are stored in `/etc/cumulus/sysmonitor.conf`. More information is available with `man sysmonitor`.

CPU measure	Thresholds
Use	Alert: 90% Crit: 95%
Process Load	Alarm: 95% Crit: 125%

Disk Usage

When monitoring disk utilization, you can exclude `tmpfs` from monitoring.

System Element	Monitoring Commands	Interval Poll
Disk utilization	`cumulus@switch:~$ /bin/df -x tmpfs`	300 seconds

Process Restart

In Cumulus Linux, `systemd` is responsible for monitoring and restarting processes.

Process Element	Monitoring Commands
View processes monitored by systemd	`cumulus@switch:~$ systemctl status`

Layer 1 Protocols and Interfaces

Link and port state interface transitions are logged to `/var/log/syslog` and `/var/log/switchd.log`.

Interface Element	Monitoring Commands
Link state	`cumulus@switch:~$ cat /sys/class/net/[iface]/operstate` `cumulus@switch:~$ net show interface all json`
Link speed	`cumulus@switch:~$ cat /sys/class/net/[iface]/speed` `cumulus@switch:~$ net show interface all json`
Port state	`cumulus@switch:~$ ip link show` `cumulus@switch:~$ net show interface all json`

Interface Element	Monitoring Commands
Bond state	`cumulus@switch:~$ cat /proc/net/bonding/[bond]` `cumulus@switch:~$ net show interface all json`

Interface counters are obtained from either querying the hardware or the Linux kernel. The two outputs should align, but the Linux kernel aggregates the output from the hardware.

Interface Counter Element	Monitoring Commands	Interval Poll

Interface counters	`cumulus@switch:~$ cat /sys/class/net/[iface]/statistics/[stat_name]`
	`cumulus@switch:~$ net show counters json`
	`cumulus@switch:~$ cl-netstat`
		10 seconds

Interface Counter Element	Monitoring Commands	Interval Poll
	-j cumulus@switch:~$ ethtool -S [iface]	

Layer 1 Logs	L og Location	Log Entries
Link failure/Link flap	/var/log/ switchd.log	switchd[5692]: nic.c:213 nic_set_carrier: swp17: setting kernel carrier: down switchd[5692]: netlink.c:291 libnl: swp1, family 0, ifi 20, oper down switchd[5692]:

Layer 1 Logs	L og Location	Log Entries
		`nic.c:213 nic_set_carrier: swp1: setting kernel carrier: up switchd[5692]: netlink.c:291 libnl: swp17, family 0, ifi 20, oper up`
Unidirectional link	`/var/log/ switchd.log /var/log/ ptm.log`	`ptmd[7146]: ptm_bfd.c:2471 Created new session 0x1 with peer 10.255.255.11 port swp1 ptmd[7146]: ptm_bfd.c:2471 Created new`

Layer 1 Logs	L og Location	Log Entries
		`session 0x2 with peer fe80::4638:39ff:fe00:5b port swp1 ptmd[7146]: ptm_bfd.c:2471 Session 0x1 down to peer 10.255.255.11, Reason 8 ptmd[7146]: ptm_bfd.c:2471 Detect timeout on session 0x1 with peer 10.255.255.11, in state 1`
Bond Negotiation Working	`/var/log/ syslog`	`kernel: [85412.763193] bonding: bond0 is`

Layer 1 Logs	L og Location	Log Entries
		being created… kernel: [85412.770014] bond0: Enslaving swp2 as a backup interface with an up link kernel: [85412.775216] bond0: Enslaving swp1 as a backup interface with an up link kernel: [85412.797393] IPv6: ADDRCONF(NETDEV_UP): bond0: link is not ready kernel: [85412.799425] IPv6:

Layer 1 Logs	L og Location	Log Entries
		`ADDRCONF(NETDEV_CHANGE): bond0: link becomes ready`
Bond Negotiation Failing	`/var/log/syslog`	`kernel: [85412.763193] bonding: bond0 is being created… kernel: [85412.770014] bond0: Enslaving swp2 as a backup interface with an up link kernel: [85412.775216] bond0: Enslaving`

Layer 1 Logs	L og Location	Log Entries
		`swp1 as a backup interface with an up link kernel: [85412.797393] IPv6: ADDRCONF(NETDEV_UP): bond0: link is not ready`
MLAG peerlink negotiation Working	`/var/log/ syslog`	`lldpd[998]: error while receiving frame on swp50: Network is down lldpd[998]: error while receiving frame on swp49:`

Layer 1 Logs	L og Location	Log Entries
		Network is down kernel: [76174.262893] peerlink: Setting ad_actor_system to 44:38:39:00:00:11 kernel: [76174.264205] 8021q: adding VLAN 0 to HW filter on device peerlink mstpd: one_clag_cmd: setting (1) peer link: peerlink mstpd: one_clag_cmd: setting (1) clag state: up mstpd: one_clag_cmd:

Layer 1 Logs	L og Location	Log Entries
		`setting system-mac 44:38:39:ff:40:94` `mstpd: one_clag_cmd: setting clag-role secondary`

	`/var/log/clagd.log`	`clagd[14003]: Cleanup is executing.`
		`clagd[14003]: Cannot open file “/tmp/pre-clagd.q7XiO`
		`clagd[14003]: Cleanup is finished`
		`clagd[14003]: Beginning execution of clagd`

Layer 1 Logs	L og Location	Log Entries
		version 1 clagd[14003]: Invoked with: /usr/sbin/clagd –daemon clagd[14003]: Role is now secondary clagd[14003]: HealthCheck: role via backup is second clagd[14003]: HealthCheck: backup active clagd[14003]: Initial config loaded clagd[14003]: The peer switch is active. clagd[14003]: Initial data sync

Layer 1 Logs	L og Location	Log Entries
		`from peer done. clagd[14003]: Initial handshake done. clagd[14003]: Initial data sync to peer done.`
MLAG peerlink negotiation Failing	`/var/log/ syslog`	`lldpd[998]: error while receiving frame on swp50: Network is down lldpd[998]: error while receiving frame on swp49:`

Layer 1 Logs	L og Location	Log Entries
		Network is down kernel: [76174.262893] peerlink: Setting ad_actor_system to 44:38:39:00:00:11 kernel: [76174.264205] 8021q: adding VLAN 0 to HW filter on device peerlink mstpd: one_clag_cmd: setting (1) peer link: peerlink mstpd: one_clag_cmd: setting (1) clag state: down mstpd: one_clag_cmd:

Layer 1 Logs	L og Location	Log Entries
		`setting system-mac 44:38:39:ff:40:94 mstpd: one_clag_cmd: setting clag-role secondary`
	`/var/log/ clagd.log`	`clagd[26916]: Cleanup is executing. clagd[26916]: Cannot open file “/tmp/ pre-clagd.6M527vvGX0/ brbatch” for reading: No such file or directory clagd[26916]: Cleanup is`

Layer 1 Logs	L og Location	Log Entries
		finished clagd[26916]: Beginning execution of clagd version 1.3.0 clagd[26916]: Invoked with: /usr/sbin/clagd -daemon 169.254.1.2 peerlink.4094 44:38:39:FF:01:01 -priority 1000 -backupIp 10.0.0.2 clagd[26916]: Role is now secondary clagd[26916]: Initial config loaded

Layer 1 Logs	L og Location	Log Entries
MLAG port negotiation Working	`/var/log/syslog`	`kernel: [77419.112195] bonding: server01 is being created… lldpd[998]: error while receiving frame on swp1: Network is down kernel: [77419.122707] 8021q: adding VLAN 0 to HW filter on device swp1 kernel: [77419.126408] server01: Enslaving swp1 as a backup interface with a down link`

Layer 1 Logs	L og Location	Log Entries
		kernel: [77419.177175] server01: Setting ad_actor_system to 44:38:39:ff:40:94 kernel: [77419.190874] server01: Warning: No 802.3ad response from the link partner for any adapters in the bond kernel: [77419.191448] IPv6: ADDRCONF(NETDEV_UP): server01: link is not ready kernel: [77419.191452] 8021q:

Layer 1 Logs	L og Location	Log Entries
		adding VLAN 0 to HW filter on device server01 kernel: [77419.192060] server01: link status definitely up for interface swp1, 1000 Mbps full duplex kernel: [77419.192065] server01: now running without any active interface! kernel: [77421.491811] IPv6: ADDRCONF(NETDEV_CHANGE): server01: link becomes

Layer 1 Logs	L og Location	Log Entries
		`ready mstpd: one_clag_cmd: setting (1) mac 44:38:39:00:00:17 <server01, None>`
	`/var/log/ clagd.log`	`clagd[14003]: server01 is now dual connected.`
MLAG port negotiation Failing	`/var/log/ syslog`	`kernel: [79290.290999] bonding: server01 is being created…`

Layer 1 Logs	L og Location	Log Entries
		kernel: [79290.299645] 8021q: adding VLAN 0 to HW filter on device swp1 kernel: [79290.301790] server01: Enslaving swp1 as a backup interface with a down link kernel: [79290.358294] server01: Setting ad_actor_system to 44:38:39:ff:40:94 kernel: [79290.373590] server01: Warning: No 802.3ad response

Layer 1 Logs	L og Location	Log Entries
		from the link partner for any adapters in the bond kernel: [79290.374024] IPv6: ADDRCONF(NETDEV_UP): server01: link is not ready kernel: [79290.374028] 8021q: adding VLAN 0 to HW filter on device server01 kernel: [79290.375033] server01: link status definitely up for interface swp1, 1000

Layer 1 Logs	L og Location	Log Entries
		`Mbps full duplex kernel: [79290.375037] server01: now running without any active interface!`
	`/var/log/ clagd.log`	`clagd[14291]: Conflict (server01): matching clag-id (1) not configured on peer… clagd[14291]: Conflict cleared (server01): matching clag-id (1)`

Layer 1 Logs	L og Location	Log Entries
		`detected on peer`
MLAG port negotiation Flapping	`/var/log/syslog`	`mstpd: one_clag_cmd: setting (0) mac 00:00:00:00:00:00 <server01, None>` `mstpd: one_clag_cmd: setting (1) mac 44:38:39:00:00:03 <server01, None>`
	`/var/log/clagd.log`	`clagd[14291]: server01 is`

Layer 1 Logs	L og Location	Log Entries
		`no longer dual connected clagd[14291]: server01 is now dual connected.`

Prescriptive Topology Manager (PTM) uses LLDP information to compare against a `topology.dot` file that describes the network. It has built in alerting capabilities, so it is preferable to use PTM on box rather than polling LLDP information regularly. The PTM code is available in the Cumulus Networks GitHub repository. Additional PTM, BFD, and associated logs are documented in the code.

> **NOTE**
>
> Tracking peering information through PTM is highly recommended. For more information, refer to the Prescriptive Topology Manager documentation.

Neighbor Element	Monitoring Commands	Interval Poll
LLDP Neighbor	`cumulus@switch:~$ lldpctl -f json`	300 seconds
Prescriptive Topology Manager	`cumulus@switch:~$ ptmctl -j [-d]`	Triggered

Layer 2 Protocols

Spanning tree is a protocol that prevents loops in a layer 2 infrastructure. In a stable state, the spanning tree protocol should stably converge. Monitoring the Topology Change Notifications (TCN) in STP helps identify when new BPDUs are received.

Interface Counter Element	Monitoring Commands	Interval Poll
STP TCN Transitions	`cumulus@switch:~$ mstpctl showbridge json` `cumulus@switch:~$ mstpctl showport json`	60 seconds

MLAG peer state	`cumulus@switch:~$ clagctl status`
	`cumulus@switch:~$ clagd -j`
	`cumulus@switch:~$ cat /var/log/clagd.log`
		60 seconds

Interface Counter Element	Monitoring Commands	Interval Poll
MLAG peer MACs	`cumulus@switch:~$ clagctl dumppeermacs` `cumulus@switch:~$ clagctl dumpourmacs`	300 seconds

Layer 2 Logs	Log Location	Log Entries

Spanning Tree Working	`/var/log/syslog`	`kernel: [1653877.190724] device swp1 entered promiscuous mode`
		`kernel: [1653877.190796] device swp2 entered promiscuous mode`
		`mstpd: create_br:`

Layer 2 Logs	Log Location	Log Entries
		Add bridge bridge mstpd: clag_set_sys_mac_br: set bridge mac 00:00:00:00:00:00 mstpd: create_if: Add iface swp1 as port#2 to bridge bridge mstpd: set_if_up: Port swp1 : up mstpd: create_if: Add iface swp2 as port#1 to bridge bridge mstpd: set_if_up: Port swp2 : up

Layer 2 Logs	Log Location	Log Entries
		mstpd: set_br_up: Set bridge bridge up mstpd: MSTP_OUT_set_state: bridge:swp1:0 entering blocking state(Disabled) mstpd: MSTP_OUT_set_state: bridge:swp2:0 entering blocking state(Disabled) mstpd: MSTP_OUT_flush_all_fids: bridge:swp1:0 Flushing forwarding database mstpd: MSTP_OUT_flush_all_fids: bridge:swp2:0 Flushing forwarding database mstpd:

Layer 2 Logs	Log Location	Log Entries
		MSTP_OUT_set_state: bridge:swp1:0 entering learning state(Designated) mstpd: MSTP_OUT_set_state: bridge:swp2:0 entering learning state(Designated) sudo: pam_unix(sudo:session): session closed for user root mstpd: MSTP_OUT_set_state: bridge:swp1:0 entering forwarding state(Designated) mstpd: MSTP_OUT_set_state: bridge:swp2:0 entering forwarding state(Designated) mstpd:

Layer 2 Logs	Log Location	Log Entries
		`MSTP_OUT_flush_all_fids: bridge:swp2:0 Flushing forwarding database mstpd: MSTP_OUT_flush_all_fids: bridge:swp1:0 Flushing forwarding database`
Spanning Tree Blocking	`/var/log/ syslog`	`mstpd: MSTP_OUT_set_state: bridge:swp2:0 entering blocking state(Designated) mstpd: MSTP_OUT_set_state: bridge:swp2:0 entering learning state(Designated)`

Layer 2 Logs	Log Location	Log Entries
		mstpd: MSTP_OUT_set_state: bridge:swp2:0 entering forwarding state(Designated) mstpd: MSTP_OUT_flush_all_fids: bridge:swp2:0 Flushing forwarding database mstpd: MSTP_OUT_flush_all_fids: bridge:swp2:0 Flushing forwarding database mstpd: MSTP_OUT_set_state: bridge:swp2:0 entering blocking state(Alternate) mstpd: MSTP_OUT_flush_all_fids: bridge:swp2:0 Flushing forwarding

Layer 2 Logs	Log Location	Log Entries
		`database`

Layer 3 Protocols

When FRRouting boots up for the first time, there is a different log file for each daemon that is activated. If the log file is ever edited (for example, through `vtysh` or `frr.conf`), the integrated configuration sends all logs to the same file.

To send FRRouting logs to syslog, apply the configuration `log syslog` in `vtysh`.

BGP

When monitoring BGP, check if BGP peers are operational. There is not much value in alerting on the current operational state of the peer; monitoring the transition is more valuable, which you can do by monitoring `syslog`.

Monitoring the routing table provides trending on the size of the infrastructure. This is especially useful when integrated with host-based solutions (such as Routing on the Host) when the routes track with the number of applications available.

BGP Element	Monitoring Commands	Interval Poll
BGP peer failure	`cumulus@switch:~$ sudo vtysh -c "show ip bgp summary json"` `cumulus@switch:~$ net show bgp summary json`	60 seconds
BGP route table	`cumulus@switch:~$ sudo vtysh -c "show ip bgp json"` `cumulus@switch:~$ net show route bgp json`	600 seconds

BGP Logs	Log Location	Log Entries
BGP peer down	`/var/log/syslog` `/var/log/frr/*.log`	`bgpd[3000]: %NOTIFICATION: sent to neighbor swp1 4/0 (Hold Timer Expired) 0 bytes` `bgpd[3000]: %ADJCHANGE: neighbor swp1 Down BGP Notification send`

OSPF

When monitoring OSPF, check if OSPF peers are operational. There is not much value in alerting on the current operational state of the peer; monitoring the transition is more valuable, which you can do by monitoring `syslog`.

Monitoring the routing table provides trending on the size of the infrastructure. This is especially useful when integrated with host-based solutions (such as Routing on the Host) when the routes track with the

number of applications available.

OSPF Element	Monitoring Commands	Interval Poll
OSPF protocol peer failure	`cumulus@switch:~$ sudo vtysh -c "show ip ospf neighbor all json"` `cumulus@switch:~$ cl-ospf summary show json`	60 seconds
OSPF link state database	`cumulus@switch:~$ sudo vtysh - c "show ip ospf database"`	600 seconds

Route and Host Entries

Route Element	Monitoring Commands	Interval Poll
Host Entries	`cumulus@switch:~$ cl-resource-query` `cumulus@switch:~$ cl-resource-query -k`	600 seconds
Route Entries	`cumulus@switch:~$ cl-resource-query` `cumulus@switch:~$ cl-resource-query -k`	600 seconds

NOTE

You can also run the `net show system asic` command, which is the NCLU command equivalent of `cl-resource-query`.

Routing Logs

Layer 3 Logs	Log Location	Log Entries
Routing protocol process crash	`/var/log/syslog`	`frrouting[1824]: Starting FRRouting daemons (prio:10):. zebra. bgpd. bgpd[1847]: BGPd 1.0.0+cl3u7 starting: vty@2605, bgp@:179 zebra[1840]: client 12 says hello and bids fair to announce`

Layer 3 Logs	Log Location	Log Entries
		only bgp routes watchfrr[1853]: watchfrr 1.0.0+cl3u7 watching [zebra bgpd], mode [phased zebra restart] watchfrr[1853]: bgpd state -> up : connect succeeded watchfrr[1853]: bgpd state -> down : read returned EOF cumulus-core: Running cl-support for core files bgpd.3030.1470341944.core.core_ core_check.sh[4992]: Please send

Layer 3 Logs	Log Location	Log Entries
		/var/ support/ cl_support__spine01_20160804_2 to Cumulus support watchfrr[1853]: Forked background command [pid 6665]: /usr/sbin/ service frr restart bgpd watchfrr[1853]: watchfrr 0.99.24+cl3u2 watching [zebra bgpd ospfd], mode [phased zebra restart] watchfrr[1853]: zebra state -> up : connect succeeded watchfrr[1853]:

Layer 3 Logs	Log Location	Log Entries
		`bgpd state -> up : connect succeeded watchfrr[1853]: watchfrr: Notifying Systemd we are up and running`

Logging

The table below describes the various log files.

Logging Element	Monitoring Commands	Log Location
syslog	Catch all log file. Identifies memory leaks and CPU spikes.	`/var/log/syslog`

Logging Element	Monitoring Commands	Log Location
switchd functionality	Hardware Abstraction Layer (HAL).	`/var/log/switchd.log`
Routing daemons	FRRouting zebra daemon details.	`/var/log/daemon.log`
Routing protocol	The log file is configurable in FRRouting. When FRRouting first boots, it uses the non-integrated configuration so each routing protocol has its own log file. After booting up, FRRouting switches over to using the integrated configuration, so that all logs go to a single place.	`/var/log/frr/zebra.log` `/var/log/frr/{protocol}.log` `/var/log/frr/frr.log`

Logging Element	Monitoring Commands	Log Location
	To edit the location of the log files, use the log file command. By default, FRRouting logs are not sent to syslog. Use the log syslog command to send logs through rsyslog and into /var/log/syslog. **Note**: To write syslog debug messages to the log file, you must run the log syslog debug command to configure FRR with syslog severity 7 (debug); otherwise, when you issue a debug command such as, debug bgp neighbor-events, no output is sent to /var/log/frr/frr.log. However, when you manually define a log target with the log file /var/log/frr/	

Logging Element	Monitoring Commands	Log Location
	debug.log command, FRR automatically defaults to severity 7 (debug) logging and the output is logged to /var/log/frr/frr.log.	

Protocols and Services

Run the following command to confirm that the NTP process is working correctly and that the switch clock is in sync with NTP:

```
cumulus@switch:~$ /usr/bin/ntpq -p
```

Device Management

Device Access Logs

Access Logs	Log Location	Log Entries
User Authentication and Remote Login	`/var/log/syslog`	`sshd[31830]: Accepted publickey for cumulus from 192.168.0.254 port 45582 ssh2: RSA 38:e6:3b:cc:04:ac:41:5e:c9:e3:` `sshd[31830]: pam_unix(sshd:session): session opened for user cumulus by (uid=0)`

Device Super User Command Logs

Super User Command Logs	Log Location	Log Entries

Executing commands using sudo	`/var/log/syslog`	`sudo: cumulus: TTY=unknown ; PWD=/home/cumulus ; USER=root ; COMMAND=/tmp/script_9938.sh -v`
		`sudo: pam_unix(sudo:session): session opened for user root by (uid=0)`
		`sudo: pam_unix(sudo:session): session closed for user root`

switchd Log Message Reference

The following table lists the log messages generated by `switchd`, organized by severity, then message text. These messages appear in `/var/log/switchd.log`.

Severity	Message Text	Explanation	Recommended Action
CRITICAL	_port_group_config_create: hal_list_get failed on [str]	hal_list_get failed.	File a ticket with Cumulus Support.
CRITICAL	_range_limits_get: start linux interface name buffer is NULL	Invalid parameter.	File a ticket with Cumulus Support.
CRITICAL	_range_limits_get: end linux interface name buffer is NULL	Invalid parameter.	File a ticket with Cumulus Support.
CRITICAL	_range_limits_get: [str]-[str] not recognized	Invalid port set configuration.	Check QoS configuration file.
CRITICAL	_range_limits_get: port range [str] not	Invalid port set configuration.	Check QoS configuration file.

Severity	Message Text	Explanation	Recommended Action
	recognized		
CRITICAL	_port_group_ports_set: hal_list_get failed on [str]	Port set list create failed.	Check QoS configuration file.
CRITICAL	_port_group_name_list_get: hal_list_get failed on [str]	List create failed.	Check QoS configuration file.
CRITICAL	_port_group_range_validate _get_range_limits failed on [str]	Invalid port set configuration.	Check QoS configuration file.
CRITICAL	_priority_group_config_get: hal_list_get failed on [str]	Configuration list create failed.	File a ticket with Cumulus Support.
CRITICAL	hal_list_get: list string [str] contains more elements than the maximum allowed ([int])	List capacity exceeded.	File a ticket with Cumulus Support.
CRITICAL	hal_sh_datapath_file_read: could not load config file [str]	Could not load the back end QoS configuration file.	Check backend QoS configuration file.
CRITICAL	Unable to	Memory	File a ticket

Severity	Message Text	Explanation	Recommended Action
	reallocate [int] bytes of memory	allocation failed.	with Cumulus Support.
CRITICAL	No backends found.	No back ends found.	File a ticket with Cumulus Support.
CRITICAL	License: email is longer than [int] characters	Email length exceeds maximum.	Modify email address.
CRITICAL	License: license data is longer than [int]	License data exceeds maximum.	Check license.
CRITICAL	License: Invalid format	Invalid license format.	Check license.
CRITICAL	No license file.	No license file found.	Check license file.
CRITICAL	The Cumulus Linux license appears to be invalid. This WILL NOT affect your system operations at the moment. Future	Invalid license.	Check license.

Severity	Message Text	Explanation	Recommended Action
	versions will enforce fully valid licenses on the system. Please contact licensing@cumulusnetworks.com at your convenience so we can validate and assist you with this licensing issue.		
CRITICAL	No license file.	No license file found.	Check license file.
CRITICAL	Incomplete license.	Incomplete license.	Check license.
CRITICAL	License is expired!	License is expired.	Renew license.
CRITICAL	unable to get tap_name for port [uint]	Port config failed: no port name.	File a ticket with Cumulus Support.
CRITICAL	Voluntary restart by timestamp	Voluntary switchd restart.	None.

Severity	Message Text	Explanation	Recommended Action
	check requested		
CRITICAL	Couldn't write ready file [str]	Could not mark switchd startup complete.	File a ticket with Cumulus Support.
CRITICAL	Could not open [str] to record error type	Could not report restart reason.	File a ticket with Cumulus Support.
CRITICAL	Error setting signal handlers.	Signal handler initialization failed.	File a ticket with Cumulus Support.
CRITICAL	No license to run switchd!	No switchd license is installed.	Install switchd license.
CRITICAL	daemon call failed with rv [int]	switchd could not be daemonized.	File a ticket with Cumulus Support.
CRITICAL	Couldn't write pid file [str]	Could not write out the process ID.	File a ticket with Cumulus Support.
CRITICAL	Switchd fs init failed.	Failed to initialize the switchd file system.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
CRITICAL	Switchd config failed.	Could not load the switchd configuration file.	Check switchd configuration file.
CRITICAL	Netlink init failed.	Netlink initialization failed.	File a ticket with Cumulus Support.
CRITICAL	HAL init failed.	HAL initialization failed.	File a ticket with Cumulus Support.
CRITICAL	NIC init failed.	NC initialization failed.	File a ticket with Cumulus Support.
CRITICAL	Port init failed.	Port initialization failed.	File a ticket with Cumulus Support.
CRITICAL	Bridges init failed.	Bridges initialization failed.	File a ticket with Cumulus Support.
CRITICAL	Bonds init failed.	Bonds initialization failed.	File a ticket with Cumulus Support.
CRITICAL	Logical networks init failed.	Logical networks initialization failed.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
CRITICAL	Interface list init failed.	Interface list initialization failed.	File a ticket with Cumulus Support.
CRITICAL	Switchd fs mount failed.	Could not mount switchd file system.	File a ticket with Cumulus Support.
CRITICAL	Failed to add route [str]	Failure in VRF route leak feature. This message notifies that a route entry could not be properly added to one of the software tables.	File a ticket with Cumulus Support.
CRITICAL	MAC address [str] couldn't be added to or retrieved from hash	Relates to merging MAC tables. The message notifies that an entry expected in a MAC address software table is not found therein.	This should never be seen. File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
CRITICAL	Failed to add route [str]	Add MPLS transit LSP to a software table failed.	File a ticket with Cumulus Support.
CRITICAL	[str]: hal port list malloc failed	Memory exhausted.	File a ticket with Cumulus Support.
CRITICAL	Failed to add [str] to [str]	“Failed to add to ”. Issue happens when addition of the port to a software table fails.	File a ticket with Cumulus Support.
CRITICAL	Failed to add hal mroute for [str]	Failed to add multicast route to a software table.	File a ticket with Cumulus Support.
CRITICAL	Failed to add [str] to [str]	“Failed to add to ”. Issue happens when addition of the port to a software table fails.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
CRITICAL	Failed to add grp [str] to mroute	A multicast route for a group could not be added to a software hash table.	File a ticket with Cumulus Support.
CRITICAL	Maximum number of bonds exceeded, max is [int]	Maximum number of bonds exceeded.	Reduce the number of bonds on the switch.
CRITICAL	Maximum number of slaves per bond exceeded, max is [int]	Maximum number of bond members per bond exceeded.	Reduce the number of bond members configured for a bond.
CRITICAL	rtnl slave state get failed for bond:[int] port: [int]	Could not get the bond member state from Netlink.	File a ticket with Cumulus Support.
CRITICAL	Failed to add route [str]	Failed to add route to a software table.	File a ticket with Cumulus Support.
CRITICAL	Failed to add [str] to	Failed to add a port to the	File a ticket with Cumulus

Severity	Message Text	Explanation	Recommended Action
	bridge [int]	bridge.	Support.
CRITICAL	Failed to add [str] to grp [str]	Failed to add a port to the MDB group.	File a ticket with Cumulus Support.
CRITICAL	Failed to add [str] to bridge [int]	Failed to add an MDB group to the bridge.	File a ticket with Cumulus Support.
CRITICAL	Failed to add bridge [int] to mdb	Failed to add a given bridge to MDB.	File a ticket with Cumulus Support.
CRITICAL	Failed to add port [str] in grp [str], bridge [int]	Failed to add a port to a group for the specific bridge.	File a ticket with Cumulus Support.
CRITICAL	Failed to add [str] to bridge [int]	Failed to add a port to the bridge.	File a ticket with Cumulus Support.
CRITICAL	Failed to add port [str] in grp [str]	Failed to add a port to the MDB group.	File a ticket with Cumulus Support.
CRITICAL	Failed to add [str] to bridge [int]	Failed to add a port to the bridge.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
CRITICAL	Failed to add bridge [str] to mdb	Failed to add a given bridge to MDB.	File a ticket with Cumulus Support.
CRITICAL	Failed to add [str] to bridge [int]	Failed to add a port to the bridge.	File a ticket with Cumulus Support.
CRITICAL	arptables: Memory allocation for rules failed,malloc: [str]	ACL out of memory resource.	File a ticket with Cumulus Support.
CRITICAL	Failed to create kernel bridge	L2: Bridge hash table add failed.	File a ticket with Cumulus Support.
CRITICAL	Open of /dev/net/tun failed: [str]	Failed to create a net device.	File a ticket with Cumulus Support.
CRITICAL	TUNSETIFF failed: [str]	Failed to create a net device.	File a ticket with Cumulus Support.
CRITICAL	SIOCGIFHWADDR failed: [str]	Failed to create a net device.	File a ticket with Cumulus Support.
CRITICAL	SIOCSIFHWADDR failed: [str]	Failed to create a net	File a ticket with Cumulus

Severity	Message Text	Explanation	Recommended Action
		device.	Support.
CRITICAL	TUNSETPERSIST failed: [str]	Failed to create a net device.	File a ticket with Cumulus Support.
CRITICAL	TUNSETOFFLOAD failed: [str]	Failed to create a net device.	File a ticket with Cumulus Support.
CRITICAL	Couldn't create tuntap ioctl socket.	Failed to create a net device.	File a ticket with Cumulus Support.
CRITICAL	Couldn't get netdev flags.	Failed to create a net device.	File a ticket with Cumulus Support.
CRITICAL	Couldn't Set netdev flags.	Failed to create a net device.	File a ticket with Cumulus Support.
CRITICAL	[str]: rtnl_link_alloc failed for family [int]	Failed to create filters.	File a ticket with Cumulus Support.
CRITICAL	[str]: rtnl_neigh_alloc failed for family [int]	Failed to create filters.	File a ticket with Cumulus Support.
CRITICAL	Failed to blacklist	Failed to block	Check block interfaces.

Severity	Message Text	Explanation	Recommended Action
	interface [int]	interfaces.	
CRITICAL	F ailed to blacklist interface [int]	Failed to block interfaces.	Check block interfaces.
CRITICAL	Failed to add [str], ifindex [int] to sw_intfs	Failed to create interfaces.	Recreate the interface.
CRITICAL	Failed to delete ifindex [int] from sw_intfs	Failed to create interfaces.	Recreate the interface.
CRITICAL	[str]: could not load interface config	Syncing database failed between kernel and switchd.	File a ticket with Cumulus Support.
CRITICAL	bogus filesystem path: [str]	Failed to add a file in SFS (simple file system).	File a ticket with Cumulus Support.
CRITICAL	Need file spec	Failed to add a file in SFS (simple file system).	File a ticket with Cumulus Support.
CRITICAL	can’t replace	Failed to add	File a ticket

Severity	Message Text	Explanation	Recommended Action
	existing directory with file: [str]	a file in SFS (simple file system).	with Cumulus Support.
CRITICAL	filesystem already initialized	Failed to initialize in SFS (simple file system).	File a ticket with Cumulus Support.
CRITICAL	filesystem hash table alloc failed	Failed to allocate hash table in SFS init.	File a ticket with Cumulus Support.
CRITICAL	filesystem mount failed	Failed to mount SFS in swtichd init.	File a ticket with Cumulus Support.
CRITICAL	filesystem new failed	Failed to mount SFS in swtichd init.	File a ticket with Cumulus Support.
CRITICAL	bogus filesystem path: [str]	Failed to delete filesystem in SFS.	File a ticket with Cumulus Support.
CRITICAL	pthread_create failed: [str]	Failed to create a thread in NIC init.	File a ticket with Cumulus Support.
CRITICAL	pthread_detach failed: [str]	Failed to detach a	File a ticket with Cumulus

Severity	Message Text	Explanation	Recommended Action
		thread in NIC init.	Support.
CRITICAL	TX Ring allocation failed: [str]	Failed to alloc packt buffer in NIC init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't increase netlink rbuf size: [str]	Failed to init buffer size in Netlink socket.	File a ticket with Cumulus Support.
CRITICAL	Couldn't increase netlink wbuf size: [str]	Failed to init buffer size in Netlink socket.	File a ticket with Cumulus Support.
CRITICAL	Couldn't allocate netlink socket.	Failed to create a Netlink socket.	File a ticket with Cumulus Support.
CRITICAL	Couldn't connect netlink socket: [str]	Failed to create a Netlink socket.	File a ticket with Cumulus Support.
CRITICAL	nl_resync_route failed for cache [int]: [str]	Failied to create a resync router function in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't set	Failied to	File a ticket

Severity	Message Text	Explanation	Recommended Action
	bufsize for manager netlink socket.	reinitialize buffer size in Netlink socket.	with Cumulus Support.
CRITICAL	invalid cache mngrinfo.	Failied to create a resync in Netlink idle callback.	File a ticket with Cumulus Support.
CRITICAL	[str]: failed to close socket: [str]	Failed to configure FD in Netlink socket.	File a ticket with Cumulus Support.
CRITICAL	[str]: nl_cache_mngr_data_ready failed: [str]	Failed to configure FD in Netlink socket.	File a ticket with Cumulus Support.
CRITICAL	Couldn't allocate netlink socket.	Failed to create a socket in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't allocate netlink socket.	Failed to create a socket in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't allocate manager	Failed to create a socket in	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
	netlink socket.	Netlink init.	
CRITICAL	Couldn't create cache manager: [str]	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't set bufsize for manager netlink socket.	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't add link cache: [str]	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't add link cache: [str]	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't add route cache: [str]	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't add mdb cache: [str]	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
CRITICAL	Couldn't alloc neigh cache: [str]	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't add mroute cache: [str]	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't alloc tcqdisc cache: [str]	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't add tcqdisc cache: [str]	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't alloc tcclass cache: [str]	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't add tcclass cache: [str]	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't alloc tccls cache: [str]	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
CRITICAL	Couldn't add tccls cache: [str]	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't alloc tcact cache: [str]	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't add tcact cache: [str]	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't alloc rule cache: [str]	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't add rule cache: [str]	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't add neigh cache: [str]	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Couldn't alloc netconf cache: [str]	Failed to allocate a cache in Netlink init.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
CRITICAL	Couldn't initialize genl/ port interface	Failed to initialize port interface in Netlink init.	File a ticket with Cumulus Support.
CRITICAL	Failed to create kernel bridge	Failed to allocate a hash entry in bridge sync.	File a ticket with Cumulus Support.
CRITICAL	Port msg [str] failure: err [int]	Failed to configure PORT FEC parameter.	File a ticket with Cumulus Support.
CRITICAL	Port msg [str] reply failure: err [int]	Failed to configure PORT FEC parameter.	File a ticket with Cumulus Support.
CRITICAL	Failed run recvmsg_default on port socket, err [int], [str]	Failed to initialize PORT receiving messages.	File a ticket with Cumulus Support.
CRITICAL	vlan stats send failure: err [int]	Failed to configure status in VLAN.	File a ticket with Cumulus Support.
CRITICAL	mroute hitbits send failure: err [int]	Failed to configure hit bit status in	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		MCAST router.	
CRITICAL	Port send stats failure: err [int]	Failed to configure status in PORT.	File a ticket with Cumulus Support.
CRITICAL	Port send settings failure: err [int]	Failed to configure settings in PORT.	File a ticket with Cumulus Support.
CRITICAL	Port send carrier failure: err [int]	Failed to configure carrier in PORT.	File a ticket with Cumulus Support.
CRITICAL	ifindex [int] already registered for port ops	Failed to register interface options in PORT.	File a ticket with Cumulus Support.
CRITICAL	ifindex [int] not registered for port ops	Failed to unregister interface options in PORT.	File a ticket with Cumulus Support.
CRITICAL	Failed to allocate port hash table	Failed to initialize PORT.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
CRITICAL	Failed to allocate port socket	Failed to initialize PORT.	File a ticket with Cumulus Support.
CRITICAL	Failed to genl connect to port socket	Failed to initialize PORT.	File a ticket with Cumulus Support.
CRITICAL	Failed to allocate port sync socket	Failed to initialize PORT.	File a ticket with Cumulus Support.
CRITICAL	Failed to genl connect to port socket	Failed to initialize PORT.	File a ticket with Cumulus Support.
CRITICAL	Failed to set genl port sync socket to non-blocking	Failed to initialize PORT.	File a ticket with Cumulus Support.
CRITICAL	Failed to resolve port ops, err [int]	Failed to initialize PORT.	File a ticket with Cumulus Support.
CRITICAL	Failed to resolve port multicast group	Failed to initialize PORT.	File a ticket with Cumulus Support.
CRITICAL	Failed to register port ops, err [int]	Failed to initialize PORT.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
CRITICAL	Failed to add port group membership, err [int]	Failed to initialize PORT.	File a ticket with Cumulus Support.
CRITICAL	Failed to modify port socket notify cb, err [int]	Failed to initialize PORT.	File a ticket with Cumulus Support.
CRITICAL	[str]:[int]: [str][str]Assertion [str] failed	Failed to find the function name in switchd.	File a ticket with Cumulus Support.
ERROR	priority group [int] headroom count [int] exceeds the maximum value [int]	P ort headroom buffers exceed ASIC limit.	File a ticket with Cumulus Support.
ERROR	shared buffer type [int] not recognized	Invalid buffer type.	File a ticket with Cumulus Support.
ERROR	sx_api_cos_prio_to_tc_set failed: [str]	IEEE priority map configuration write failed.	File a ticket with Cumulus Support.
ERROR	_hal_mlx_packet_2_switch: priority field	Packet priority field	Check QoS configuration

Severity	Message Text	Explanation	Recommended Action
	[int] not supported	not supported for source.	file.
ERROR	priority field [int] not supported	P acket priority field not supported for remark.	Check QoS configuration file.
ERROR	cos list length [int] is longer than maximum value [int]	ECN/RED configuration: list is too long.	Check QoS configuration file.
ERROR	hash params get failed: [str]	ASIC ECMP hash seed configuration read failed.	File a ticket with Cumulus Support.
ERROR	hash params set failed: [str]	ASIC ECMP hash seed configuration write failed.	File a ticket with Cumulus Support.
ERROR	hal_sh_datapath_pfc_set: PFC configuration not supported on the CPU port	CPU port does not support priority flow control.	File a ticket with Cumulus Support.
ERROR	hal_sh_datapath_init:	Back end	Check for

Severity	Message Text	Explanation	Recommended Action
	datapath init failed: rv [int]: [str]	QoS initialization failed.	detailed log messages.
ERROR	_priority_field_list_get: Packet priority field [str] not supported	Invalid packet priority field.	Check QoS configuration file.
ERROR	_sfs_init: could not load traffic config file [str]	Traffic configuration file missing or is unreadable.	Check QoS configuration file.
ERROR	_sfs_port_init: could not load traffic config file [str]	Traffic configuration file missing or is unreadable.	Check QoS configuration file.
ERROR	_add_port_group: port group [str] exceeds max port group count [int]	Too many port groups configured.	Check QoS configuration file.
ERROR	_add_port_group: memory allocation failed for port	Memory allocation failed.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
	group [str]		
ERROR	_switch_priority_config: [str]	ASIC scheduler configuration failed.	File a ticket with Cumulus Support.
ERROR	_priority_map_config: map function, hal port [int]: [str]	ASIC priority map configuration failed.	File a ticket with Cumulus Support.
ERROR	_priority_map_config: enable function: [str]	ASIC priority map enable configuration failed.	File a ticket with Cumulus Support.
ERROR	_port_group_range_validate: invalid port list: range length is 0, id list is 0	Invalid port set configuration.	Check QoS configuration file.
ERROR	_port_group_range_validate: failed: port list not created from range [str] to [str]	Invalid port set configuration.	Check QoS configuration file.
ERROR	hal_datapath_init: priority field initialization expects 3	Invalid inputs.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
	three priority fields, got [int]		
ERROR	hal_datapath_init: priority map direction initialization expects two priority map directions, got [int]	Invalid inputs.	File a ticket with Cumulus Support.
ERROR	hal_datapath_init: DOS config failed: [str]	ASIC DoS config failed.	File a ticket with Cumulus Support.
ERROR	_cutthrough_config: Cutthrough config failed on HAL port [int]: [str]	ASIC cut-through configuration failed.	File a ticket with Cumulus Support.
ERROR	_source_priority_map_init: packet priority map size [int] is larger than array length [int]	Configured priority map list is too large.	Check QoS configuration file.
ERROR	_source_priority_map_populate: packet	Configured priority map	Check QoS configuration

Severity	Message Text	Explanation	Recommended Action
	priority map entry index [int] is larger than array length [int]	list is too large.	file.
ERROR	_remark_priority_map_ipv4: packet priority map entry index [int] is larger than array length [int]	Configured priority map list is too large.	Check QoS configuration file.
ERROR	_remark_priority_map_ipv6: packet priority map entry index [int] is larger than array length [int]	Configured priority map list is too large.	Check QoS configuration file.
ERROR	_priority_map_get: field index [int] is out of bounds: [int] available field entries	Invalid packet priority field.	Check QoS configuration file.
ERROR	_priority_map_get: cos ID [int] is out of bounds: [int]	Invalid priority value.	Check QoS configuration file.

Severity	Message Text	Explanation	Recommended Action
	cos ID values		
ERROR	_priority_map_get: old syntax flag [int] should be less than [int]	Invalid configuration syntax.	File a ticket with Cumulus Support.
ERROR	hal_list_get: strdup returned NULL	Memory allocation failed.	File a ticket with Cumulus Support.
ERROR	hal_port_pause_set: RX pause not allowed on port [int]	Invalid operation for the current port configuration.	Check QoS configuration file.
ERROR	vlan_range_confi: incorrect format, revert to default	Invalid configuration.	Correct configured VLAN range.
ERROR	vlan_range_confi: incorrect format, revert to default	Invalid configuration.	Correct configured VLAN range.
ERROR	vlan_range_confi: incorrect range, revert to default	Invalid configuration.	Correct configured VLAN range.

Severity	Message Text	Explanation	Recommended Action
ERROR	vlan_range_confi: minimum range is [int], revert to default	Invalid configuration.	Correct configured VLAN range.
ERROR	backend_enum_info_key unsupported type [uint]	Invalid backend type.	File a ticket with Cumulus Support.
ERROR	hal_init [str] enum_fn [str]: dlerror [str]	No back end enum function found.	File a ticket with Cumulus Support.
ERROR	hal_init failed to open [str]: [str]	Could not open the back end DLL.	File a ticket with Cumulus Support.
ERROR	hal_init unsupported type [uint]	Invalid back end type.	File a ticket with Cumulus Support.
ERROR	hal_init: backend function at offset [int] non populated: [address]	Back end function pointer is not set.	File a ticket with Cumulus Support.
ERROR	Unable to setup	Signal handler	File a ticket with Cumulus

Severity	Message Text	Explanation	Recommended Action
	handling of SIGHUP for log rotation: [str]	initiailzation failed.	Support.
ERROR	Couldn't delete pid file [str], [str]	Could not delete process ID file.	File a ticket with Cumulus Support.
ERROR	Failed to update VRF [str] to table id [uint]		
ERROR	Failed to add VRF [str] with table id [uint]		
ERROR	Failed to get VRF table id for index [int]	VRF table ID not found for the master link.	File a ticket with Cumulus Support.
ERROR	[int] hosts were ignored due to capacity.	Kernel neighbors did not fit in the hardware table.	Modify neighbor configuration.
ERROR	[int] routes were ignored due to total	Kernel routes did not fit in the hardware	Modify route configuration.

Severity	Message Text	Explanation	Recommended Action
	capacity.	table.	
ERROR	[int] [str] routes were ignored due to capacity.	Kernel routes did not fit in the hardware table.	Modify route configuration.
ERROR	Ignoring VRF [str]; table id 0 is reserved for default VRF	Invalid table ID for the VRF.	Modify VRF configuration.
ERROR	Failed to update VRF [str] to table id [uint]	Failed to update the VRF table ID.	File a ticket with Cumulus Support.
ERROR	Failed to add VRF [str] with table id [uint]	Failed to add the VRF.	File a ticket with Cumulus Support.
ERROR	Failed to set type to ‘vrf’ in link filter	Failed to update the Netlink cache link type.	File a ticket with Cumulus Support.
ERROR	Found [int] VRF entries after sync. cleaning up.	VRF entries remaining after sync operation.	File a ticket with Cumulus Support.
ERROR	Ignoring	Failed to	File a ticket

Severity	Message Text	Explanation	Recommended Action
	attempts to delete default route for table 0	remove the default route.	with Cumulus Support.
ERROR	Master interface not found in nl cache for index [int]	Link object not found in Netlink cache.	File a ticket with Cumulus Support.
ERROR	Maximum number of VRFs already exist. Can not add VRF for table [uint]	More VRFs have been configured than the maximum number supported by the platform.	Reduce the number of configured VRFs.
ERROR	Failed to delete REPL route from Hash Table	Failure in VRF route leak feature. This message notifies that an entry could not be properly deleted from one of the software tables.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	Failed to add route [str]	Failure in VRF route leak feature. This message notifies that a route entry could not be properly added to one of the software tables.	File a ticket with Cumulus Support.
ERROR	Route [str] in HW, but not in HAL cache. Adding.	Happens when HAL resync is triggered. Route discovered in hardware but it is not in the software database.	No action is required as software auto-corrects.
ERROR	HW route [str] doesn't match HAL route [str]. Updating.	Happens when HAL resync is triggered. Route discovered in hardware but it is not matching the	No action is required as software auto-corrects.

Severity	Message Text	Explanation	Recommended Action
		software database.	
ERROR	Route [str] in HAL cache, but not in HW. Deleting.	Happens when HAL resync is triggered. Route discovered in software but it is not seen in the hardware.	No action is required as software auto-corrects.
ERROR	No parent interface for [str]	The parent interface of an interface could not be derived. This is a software issue.	File a ticket with Cumulus Support.
ERROR	[str]: sfs backing pointer is NULL	FUSE file pointer is NULL.	File a ticket with Cumulus Support.
ERROR	port sample rate string not found	Port sampling rate is not set.	Correct the configuration.
ERROR	[str]: hal port pointer is	Port pointer is incorrect	File a ticket with Cumulus

Severity	Message Text	Explanation	Recommended Action
	NULL for port sample string [str]	when port sampling is set.	Support.
ERROR	port sample rate string [str] not recognized	Port sampling rate is not configured properly.	Correct the configuration.
ERROR	port sample interface [str] in port sample string '[str]' not recognized	Port sampling port-name is not configured properly.	Correct the configuration.
ERROR	port sample rate value not recognized in string [str]	Port sampling value is not configured properly.	Correct the configuration.
ERROR	port sample rate value [str] does not contain a valid integer	Port sampling value is not configured properly using integer values.	Correct the configuration.
ERROR	[int] mroutes were ignored due to total capacity.	Maximum number of multicast routes exceeded.	Reduce the number of multicast routres in the switch/fabric.

Severity	Message Text	Explanation	Recommended Action
ERROR	local_ip format in fuse node is incorrect [str]	Local_IP value formatting in FUSE file is incorrect.	Correct the configuration.
ERROR	local_ip fuse node read failed	Local_IP value formatting in FUSE file is invalid.	Correct the configuration.
ERROR	vxlan encap dscp action invalid	VXLAN encap DSCP action in FUSE file is invalid.	Correct the configuration.
ERROR	vxlan encap dscp action [[str]] invalid	VXLAN encap DSCP action formatting in FUSE file is invalid.	Correct the configuration.
ERROR	vxlan decap dscp action invalid	VXLAN decap DSCP action in FUSE file is invalid.	Correct the configuration.
ERROR	vxlan decap dscp action invalid	VXLAN decap DSCP action formatting in FUSE file is invalid.	Correct the configuration.

Severity	Message Text	Explanation	Recommended Action
ERROR	sx_api_lag_hash_flow_based_hash_set failed: [str]	Set hash flow parameters failed in the SDK.	
ERROR	bond IDs exhausted	Maximum number of bonds exhausted.	Reduce the numbert of configured bonds.
ERROR	bond_id [uint] swid [uint] lag create failed: [str]	Setting a LAG port group failed in the SDK.	
ERROR	bond_id [uint] lag_id 0x%x port state set failed: [str]	Port state could not be set to ADMIN_UP in the SDK.	
ERROR	bond_id [uint] lag_id 0x%x ingr_filter set failed: [str]	Ingress filter set failed for specified bond_id and lag_id in the SDK.	
ERROR	bond_id [uint] old lag_id 0x%x not cleaned	Could not find the mapping of the bond_id	

Severity	Message Text	Explanation	Recommended Action
	up	with the old lag_id, hence cleanup was not successful.	
ERROR	lag_id 0x%x swid [uint] failed: [str]	Removal of the LAG port group for the specified lag_id failed.	
ERROR	cannot find bond slave port [uint]	During addition of a bond slave port to a bond, the slave port entry in the software tables could not be found.	
ERROR	ifp not found for bond_id [uint]	Software entry for a bond_id could not be found in the database.	
ERROR	[str] member [str] add failed: [str]	Adding a member to a LAG group	

Severity	Message Text	Explanation	Recommended Action
		failed in the SDK.	
ERROR	[str] member [str] delete failed: [str]	Deleting a member in a LAG group failed in the SDK.	
ERROR	invalid port_storm_ctrl_type [uint]	Invalid Storm-Control-Type (invalid value) detected in the software. Internal error.	
ERROR	unexpected duplicate bond if_key [str]	Existing duplicate bond key found in the software table.	
ERROR	lag_id 0x%x with no corresponding bond id	bond_id could not be located for the given lag_id.	
ERROR	[str] collector set failed for [str]: [str]	Adding the specified port to the collector set	

Severity	Message Text	Explanation	Recommended Action
		for the bond failed.	
ERROR	[str] distributor set failed for [str]: [str]	Adding the specified port to the distributor set for the bond failed.	
ERROR	cannot find base bond slave [str] for bond_id [int]	During update of a bond, the slave port entry in the software tables could not be found.	
ERROR	unexpected duplicate bond interface [str]	Existing duplicate bond interface found in the software table.	
ERROR	unexpected duplicate lag_id 0x%x	Existing duplicate lag_id found in the software table.	

Severity	Message Text	Explanation	Recommended Action
ERROR	info not found for bond_id [uint]	An expected entry could not be found for a given bond_id in a software table.	
ERROR	[str] unexpected duplicate member [str]	Duplicate entry for the specified interface found for a bond in a software table.	
ERROR	info not found for bond_id [uint]	An expected entry could not be found for a given bond_id in a software table.	
ERROR	initialization failed: [str]	SDK API call for tunnel initialization failed.	File a ticket with Cumulus Support.
ERROR	logical network type [uint] key	Logical network of given type	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
	[uint] not found	and key was not found in the software tables.	
ERROR	logical network type [uint] key [uint] not found	Logical network of given type and key was not found in the software tables.	File a ticket with Cumulus Support.
ERROR	failed to create keys tunnel type [uint] and key [uint]	Failed to create logical network keys of given tunnel type and interface key in the software tables.	File a ticket with Cumulus Support.
ERROR	failed to update the decap key for gre entry tunnel_id:	Failed to update the GRE decap keys for the given tunnel.	File a ticket with Cumulus Support.
ERROR	failed to find the decap key for tunnel_id : (0x%x)	Failed to find the GRE decap keys for the given	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		tunnel.	
ERROR	failed to update a decap entry tunnel_id : (0x%x)	Failed to update the GRE decap keys for the given tunnel.	File a ticket with Cumulus Support.
ERROR	unexpected duplicate entry in gre_tunnel_key_ht	GRE tunnel key table has an existing duplicate entry.	File a ticket with Cumulus Support.
ERROR	unexpected duplicate entry in gre_tunnel_id_ht	GRE tunnel ID table has an existing duplicate entry.	File a ticket with Cumulus Support.
ERROR	unexpected duplicate entry in gre_olay_ulay_ht	GRE tunnel overlay/ underlay table has an existing duplicate entry.	File a ticket with Cumulus Support.
ERROR	failed to create the hw decap key for gre entry tunnel_id:	Failed to create the GRE decap keys for the given tunnel.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	failed to create a decap entry tunnel_id : (0x%x)	Failed to create a decap entry in the software table for the given tunnel_id.	File a ticket with Cumulus Support.
ERROR	unexpected duplicate entry in gre_decap_ht tunnel_id	GRE decap table has an existing duplicate entry for the given tunnel.	File a ticket with Cumulus Support.
ERROR	failed to create	Failed to set up a tunnel between the given local and remote IP addresses.	File a ticket with Cumulus Support.
ERROR	unable to find overlay overlay info from overlay ifindex:	While removing a GRE tunnel, the overlay info could not be retrieved from the software tables for the given ifindex.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	failed to create key for type [uint]	Failed to create the GRE keys for the given tunnel type.	File a ticket with Cumulus Support.
ERROR	failed to find gre entry	Failed to find a logical network entry for a tunnel in the software tables during tunnel removal.	File a ticket with Cumulus Support.
ERROR	failed to find gre entry	Failed to find a GRE decap entry for a tunnel during tunnel removal.	File a ticket with Cumulus Support.
ERROR	Failed to open SX-API, error: [str]	SDK API open failed.	File a ticket with Cumulus Support.
ERROR	Failed to set SDK VERBOSITY level, error: [str]	Failed to set SDK VERBOSITY level.	File a ticket with Cumulus Support.
ERROR	num_devices	SDK found	File a ticket

Severity	Message Text	Explanation	Recommended Action
	[uint] is invalid	zero (asic) devices.	with Cumulus Support.
ERROR	Failed to initialize SDK ([str])	Failed to initialize SDK.	File a ticket with Cumulus Support.
ERROR	*** failed to configure the requested setup ***	The board configuration could not be enforced.	File a ticket with Cumulus Support.
ERROR	more devices encountered than configured [uint]	More devices encountered than configured.	File a ticket with Cumulus Support.
ERROR	duplicate device ID [uint]	Found an existing duplicate device when adding a new device.	File a ticket with Cumulus Support.
ERROR	invalid unit [uint] num_devices [uint]	Internal error: given unit number is out of range.	File a ticket with Cumulus Support.
ERROR	invalid port [uint] num_ports [uint]	Internal error: given port number is out of range.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	ERROR: Fail to extract data from XML file	The topology map could not be extracted from the XML file.	File a ticket with Cumulus Support.
ERROR	Device ID already added	A device is already added to the topology database.	File a ticket with Cumulus Support.
ERROR	Device ID [uint] NOT found in the XML file	A device could not be found in the databse.	File a ticket with Cumulus Support.
ERROR	Failed to add device [uint] to the SDK (sx_api_topo_device_set)	Failed to add a device to the topology database.	File a ticket with Cumulus Support.
ERROR	Failed to allocate memory for tree tree_info array,	Memory exhausted.	Restart switchd, if possible. File a ticket with Cumulus Support.
ERROR	ERROR: Fail to add topo tree	Top tree could not be added.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	Failed to set topo device ready for device [uint]: [str]	Could not set DEVICE_READY in the SDK.	File a ticket with Cumulus Support.
ERROR	Unable to load file [str] (file not exists or corrupted)	The specified XML configuration file could not be loaded as it either does not exist or is possibly corrupted.	File a ticket with Cumulus Support.
ERROR	Unable to parse file (error #[int]: [str])	The specified XML configuration file could not be parsed.	File a ticket with Cumulus Support.
ERROR	Expat error #[int] (line [int] , column [int]): [str]	A parsing error encountered when parsing the configuration file.	File a ticket with Cumulus Support.
ERROR	Error parsing number of devices	Error parsing the number of child	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		devices in the XML configuration file.	
ERROR	Error parsing device mac address	Error parsing the device MAC address in the XML configuration file.	File a ticket with Cumulus Support.
ERROR	Error parsing device MAC address	Error parsing the device MAC address in the XML configuration file.	File a ticket with Cumulus Support.
ERROR	Error mac cannot be 00:00:00:00:00:00	Found a device MAC address of all zeores in the XML configuration file.	File a ticket with Cumulus Support.
ERROR	Failed running [str]	Specified system command failed.	File a ticket with Cumulus Support.
ERROR	Error parsing	Error parsing	File a ticket

Severity	Message Text	Explanation	Recommended Action
	dev_number	the device number in the XML configuration file.	with Cumulus Support.
ERROR	Error parsing device mac address	Error parsing the device MAC address in the XML configuration file.	File a ticket with Cumulus Support.
ERROR	Error parsing device MAC address	Error parsing the device MAC address in the XML configuration file.	File a ticket with Cumulus Support.
ERROR	Error parsing number of physical ports value	Error parsing the number of PHY ports in the XML configuration file.	File a ticket with Cumulus Support.
ERROR	Invalid number of physical ports [uint]	Found zero or more than the allowed maximum port numbers in the XML	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		configuration file.	
ERROR	Error parsing ports list section	Error parsing the port list section in the XML configuration file.	File a ticket with Cumulus Support.
ERROR	Error parsing local port number	Error parsing the local port number in the XML configuration file.	File a ticket with Cumulus Support.
ERROR	Error parsing mapping mode	Error parsing the mapping mode in the XML configuration file.	File a ticket with Cumulus Support.
ERROR	Error parsing label port	Error parsing the label port in the XML configuration file.	File a ticket with Cumulus Support.
ERROR	Error parsing width value	Error parsing the port width in the	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		XML configuration file.	
ERROR	Error parsing RX lanes value , local port: [[int]]	Error parsing the RX lanes value in the XML configuration file.	File a ticket with Cumulus Support.
ERROR	Error parsing lanes value , local port: [[int]]	Error parsing the lanes value in the XML configuration file.	File a ticket with Cumulus Support.
ERROR	Error parsing lanes value, local port: [[int]]	Error parsing the lanes value in the XML configuration file.	File a ticket with Cumulus Support.
ERROR	Error parsing lane to module value, local port: [[int]]	Error parsing the lane to module mapping value in the XML configuration file.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	Error parsing lane to module value , local port: [[int]]	Error parsing the lane to module mapping value in the XML configuration file.	File a ticket with Cumulus Support.
ERROR	Error parsing port mode value	Error parsing the in the port mode value XML configuration file.	File a ticket with Cumulus Support.
ERROR	Error parsing port speed value	Error parsing the in the port speed value XML configuration file.	File a ticket with Cumulus Support.
ERROR	Error parsing swid value	Error parsing the swid value in the XML configuration file.	File a ticket with Cumulus Support.
ERROR	Error parsing autoneg value	Error parsing the autoneg value in the	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		XML configuration file.	
ERROR	Conflict detected in lid 0x%0x port [uint] dev_id [uint]	lid already detected in a software table.	File a ticket with Cumulus Support.
ERROR	port_device_set has failed for device [uint] ([str])	Could not set a port device to the SDK.	File a ticket with Cumulus Support.
ERROR	topo_xml_device_add failed for device [uint] ([str])	A device could not be added to a topo_xml_device database.	File a ticket with Cumulus Support.
ERROR	mode set dev [int] port 0x%x failed: [str]	Port mode could not be set to STACKING mode in the SDK.	File a ticket with Cumulus Support.
ERROR	binding dev [int] port 0x%x to swid [int] failed: [str]	Binding the device and port to the swid failed in the SDK.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	port_init dev [int] port 0x%x failed: [str]	Specified device/port could not be initialized in the SDK.	File a ticket with Cumulus Support.
ERROR	swid set failed: [str]	swid set failed in the SDK.	File a ticket with Cumulus Support.
ERROR	Failed to set port 0x%x mapping: [str]	Port mapping failed for the specified port in the SDK.	File a ticket with Cumulus Support.
ERROR	Failed to set port 0x%x mapping: [str]	Port mapping failed for the specified port in the SDK.	File a ticket with Cumulus Support.
ERROR	invalid bridge_vlan [uint] for bridge_id [int]	The specified bridge_vlan is not valid for the given bridge_id.	File a ticket with Cumulus Support
ERROR	vfid not set for vlan [uint]	VFID is not set for the specified VLAN.	File a ticket with Cumulus Support.
ERROR	new group check failed for vlan [uint]	The specified MAC address and VLAN	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
	mac [str]: [str]	already exist in a group.	
ERROR	old port list get failed for vlan [uint] mac [str]: [str]	Could not find the existing port list for the given MAC address and VLAN.	File a ticket with Cumulus Support.
ERROR	port delete failed for vlan [uint] mac [str]: [str]	Could not delete the existing port list for the given MAC address and VLAN.	File a ticket with Cumulus Support.
ERROR	create failed for vlan [uint] mac [str]: [str]	Could not create a new multicast group for the given MAC address and VLAN.	File a ticket with Cumulus Support.
ERROR	port add failed for vlan [uint] mac [str]: [str]	Could not add a port to the port list for the given MAC address and VLAN.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	invalid bridge_vlan [uint] for bridge_id [int]	The specified bridge_vlan is not valid for the given bridge_id.	File a ticket with Cumulus Support.
ERROR	vfid not set for vlan [uint]	The specified MAC address and VLAN already exist in a group.	File a ticket with Cumulus Support.
ERROR	group delete failed for vlan [uint] mac [str]: [str]	Could not delete a multicast group for the given MAC address and VLAN.	File a ticket with Cumulus Support.
ERROR	invalid bridge_vlan [uint] for bridge_id [int]	The specified bridge_vlan is not valid for the given bridge_id.	File a ticket with Cumulus Support.
ERROR	vfid not set for vlan [uint]	VFID is not set for the specified VLAN.	File a ticket with Cumulus Support.
ERROR	flood_mode_get failed for swid	Flood mode value could	File a ticket with Cumulus

Severity	Message Text	Explanation	Recommended Action
	[int] vfid [int] [str]	not be retrieved for the given swid/port/ vfid.	Support.
ERROR	unreg_mc_flood_ports fail for swid [int], vfid [int], [str]	Unregistered multicast flood ports setting for given swid/ port/vfid failed in the SDK.	File a ticket with Cumulus Support.
ERROR	mroute ports [int] exceeds [int]	A multicast route has a larger number of ports than the RIFs on the switch.	File a ticket with Cumulus Support.
ERROR	no container id retrieved for [str]	Egress container could not be retrieved.	File a ticket with Cumulus Support.
ERROR	route cmd [str] failed: [str]	Setting multicast route in the SDK failed.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	router table_id [uint] vrid [int] set failed: [str]	Setting a VRID for a router failed.	File a ticket with Cumulus Support.
ERROR	unexpected duplicate key list size [uint]	Unexpected duplicate entry found in next-hop list.	File a ticket with Cumulus Support.
ERROR	unexpected duplicate key type [str] min_mtu [uint] fid [uint]	Unexpected duplicate entry in container anchor.	File a ticket with Cumulus Support.
ERROR	unexpected duplicate nh_list num_elems [uint]	Unexpcted duplicate next-hop list in container.	File a ticket with Cumulus Support.
ERROR	failed for nh_list num_elems [uint]: [str]	Could not create a new container for the next-hop list in the SDK.	File a ticket with Cumulus Support.
ERROR	failed for type [str]	Could not free a	File a ticket with Cumulus

Severity	Message Text	Explanation	Recommended Action
	container_id [uint] num_elems [uint]: [str]	container for the next-hop list in the SDK.	Support.
ERROR	unsupported chip type [uint]	Chip type in table is not supported.	File a ticket with Cumulus Support.
ERROR	invalid parse depth	VXLAN parsing depth setting is invalid.	File a ticket with Cumulus Support.
ERROR	unexpected duplicate ln_type [uint] ln_key [uint]	Unexpected duplicate entry in logical VPN key table.	File a ticket with Cumulus Support.
ERROR	unexpected duplicate ID 0x%x	Unexpected duplicate entry in logical VPN ID table.	File a ticket with Cumulus Support.
ERROR	unsupported ln_type [int] or ln_key [int]	Unsupported logical network type encountered.	File a ticket with Cumulus Support.
ERROR	unexpected duplicate ln_type [uint]	Unexpected duplicate entry in	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
	ln_key [uint]	logical network table.	
ERROR	lid 0x%x remote_ip [str] not found	Local ID to remote_ip mapping retrieval failed.	File a ticket with Cumulus Support.
ERROR	unexpected duplicate key [uint]	Unexpected duplicate entry in the VPN map table.	File a ticket with Cumulus Support.
ERROR	invalid VPN or vlan [uint]	Invalid VPN/ VLAN value provided for creating a VPM map entry.	File a ticket with Cumulus Support.
ERROR	unexpected duplicate entry	Unexpected duplicate entry in VPN decap table.	File a ticket with Cumulus Support.
ERROR	invalid VPN	Invalid VPN value provided for creating a VPN decap	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		entry.	
ERROR	unexpected duplicate key [str] type [uint]	Unexpected duplicate entry in VPN next-hop table.	File a ticket with Cumulus Support.
ERROR	tunnel get failed: [str]	Tunnel attribute retrieval from the SDK failed.	File a ticket with Cumulus Support.
ERROR	tunnel update failed: [str]	Tunnel attribute update in the SDK failed.	File a ticket with Cumulus Support.
ERROR	creation failed: [str]	VXLAN tunnel creation failed.	File a ticket with Cumulus Support.
ERROR	tunnel_id 0x%x hash set failed: [str]	VXLAN UDP header src port control setting in SDK for enabling better entropy failed.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	unsupported type [uint]	Unsupported tunnel type encountered.	File a ticket with Cumulus Support.
ERROR	unexpected duplicate key for tunnel_id 0x%x	Unexpected duplicate entry in logical VPN tunnel key table.	File a ticket with Cumulus Support.
ERROR	unexpected duplicate ID for tunnel_id 0x%x	Unexpected duplicate entry in logical VPN ID table.	File a ticket with Cumulus Support.
ERROR	unexpected duplicate ln_type [uint] ln_key [uint]	Unexpected duplicate entry in logical network table.	File a ticket with Cumulus Support.
ERROR	update failed: [str]	VXLAN tunnel update failed.	File a ticket with Cumulus Support.
ERROR	unsupported type [uint]	Unsupported tunnel type encountered.	File a ticket with Cumulus Support.
ERROR	tunnnel_id 0x%x failed:	VXLAN tunnel	File a ticket with Cumulus

Severity	Message Text	Explanation	Recommended Action
	[str]	destroy failed.	Support.
ERROR	tunnel_id 0x%x ttl [uint] failed: [str]	VXLAN tunnel TTL set failed.	File a ticket with Cumulus Support.
ERROR	failed: [str]	VXLAN tunnel map set failed.	File a ticket with Cumulus Support.
ERROR	vfid not available for vlan [uint]	VFID is not set for the specified VLAN.	File a ticket with Cumulus Support.
ERROR	failed: [str]	VXLAN tunnel map set failed.	File a ticket with Cumulus Support.
ERROR	failed: [str]	Tunnel decap entry could not be set.	File a ticket with Cumulus Support.
ERROR	failed: [str]	Tunnel decap entry could not be set.	File a ticket with Cumulus Support.
ERROR	[str] group_id [uint] vid [uint] failed: [str]	FDB flood set for a group failed.	File a ticket with Cumulus Support.
ERROR	group_id	FDB flood set	File a ticket

Severity	Message Text	Explanation	Recommended Action
	[uint] vid [uint] failed: [str]	for a group failed.	with Cumulus Support.
ERROR	creation failed: [str]	VPN device addition to the SDK failed.	File a ticket with Cumulus Support.
ERROR	unexpected duplicate entry	VPN port table has an existing duplicate entry.	File a ticket with Cumulus Support.
ERROR	vpn_port 0x%x failed: [str]	VPN device deletion from the SDK failed.	File a ticket with Cumulus Support.
ERROR	tunnel encap dscp action PRESERVE is invalid	PRESERVE is not a valid action.	Check configuration.
ERROR	tunnel_id 0x%x cos set failed: [str]	COS set failed for a tunnel.	File a ticket with Cumulus Support.
ERROR	tunnel decap dscp action SET is invalid	SET is not a valid DSCP action.	Check configuration.
ERROR	tunnel_id	COS set failed	File a ticket

Severity	Message Text	Explanation	Recommended Action
	0x%x cos set failed: [str]	for a tunnel.	with Cumulus Support.
ERROR	tunnel id not found: ln_type [uint] ln_key [uint]	Specified tunnel ID was not found in the VPN table.	File a ticket with Cumulus Support.
ERROR	Error opening socket for grat arp [str], ifi [int]: [str]	Gratuitous ARP socket could not be opened.	File a ticket with Cumulus Support.
ERROR	Error sending grat arp [str], ifi [int]: [str]	Gratuitous ARP socket could not be sent.	File a ticket with Cumulus Support.
ERROR	Failed to update table id for port [int], interface [str]/[int]	table_id could not be updated for a port.	File a ticket with Cumulus Support.
ERROR	interface not found at index [int]	Interface information for a given ifindex could not be found.	File a ticket with Cumulus Support.
ERROR	Failed to open IPv4 socket [str]	IPv4 socket open failed.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	Failed to set SO_RCVBUF [str]	Socket option SO_RCVBUF set failed.	File a ticket with Cumulus Support.
ERROR	No source MAC address ifindex [int], [str]	No source MAC address found for given ifiindex.	File a ticket with Cumulus Support.
ERROR	Failed to open IPv6 socket [str]	IPv6 socket open failed.	File a ticket with Cumulus Support.
ERROR	Failed to set SO_RCVBUF [str]	Socket option SO_RCVBUF set failed.	File a ticket with Cumulus Support.
ERROR	No source MAC address ifindex [int], [str]	No source MAC address found for given ifiindex.	File a ticket with Cumulus Support.
ERROR	Adding ctx to hash table failed	Adding ctx to hash table failed.	File a ticket with Cumulus Support.
ERROR	Getting pktinj failed	Getting pktinj failed.	File a ticket with Cumulus Support.
ERROR	ERSPAN target is supported with the	ACL unsupported ERSPAN target.	See ACL user documentation.

Severity	Message Text	Explanation	Recommended Action
	following field(s): –src-ip –dst-ip .		
ERROR	Inverse flags not supported for UDP	ACL inverse match not supported.	See ACL user documentation.
ERROR	Specified Match [str] not supported	ACL unsupported match.	See ACL user documentation.
ERROR	Specified Target [str] not supported	ACL unsupported target.	See ACL user documentation.
ERROR	Inverse flags not supported for ARP	ACL inverse match not supported.	See ACL user documentation.
ERROR	Inverse flags not supported for IP	ACL inverse match not supported.	See ACL user documentation.
ERROR	Inverse flags not supported for IPv6	ACL inverse match not supported.	See ACL user documentation.

Severity	Message Text	Explanation	Recommended Action
ERROR	Traffic class [hex] not supported for IP	ACL unsupported match.	See ACL user documentation.
ERROR	Range for ICMP Types i.e [hex]-[hex] not supported	ACL unsupported match.	See ACL user documentation.
ERROR	Range for ICMP codes i.e [hex]-[hex] not supported	ACL unsupported match.	See ACL user documentation.
ERROR	Invert flags not supported for mark_m	ACL unsupported match.	See ACL user documentation.
ERROR	OR bitmask [hex] not supported for mark_m match	ACL unsupported match.	See ACL user documentation.
ERROR	Inverse flags not supported for VLAN	ACL inverse match not supported.	See ACL user documentation.
ERROR	Inverse flags	ACL inverse	See ACL user

Severity	Message Text	Explanation	Recommended Action
	not supported for ICMP	match not supported.	documentation.
ERROR	vlanid match not supported for VLAN	ACL vlanid usupported match.	See ACL user documentation.
ERROR	vlan prio match not supported for VLAN	ACL vlan prio unsupported match.	See ACL user documentation.
ERROR	IP, IPv6, ARP options are not supported for LOG	ACL unsupported match for LOG action.	See ACL user documentation.
ERROR	Interface [str] not supported for SPAN	ACL unsupported target.	See ACL user documentation.
ERROR	SPAN target is supported with the following	ACL unsupported SPAN target.	See ACL user documentation.
ERROR	ERSPAN target is supported with the	ACL unsupported match.	See ACL user documentation.

Severity	Message Text	Explanation	Recommended Action
	following		
ERROR	Inverse flags not supported for ICMPv6	ACL inverse match not supported.	See ACL user documentation.
ERROR	target bitmask [hex] not supported for mark	ACL unsupported verdicts continue/ return/etc.	See ACL user documentation.
ERROR	Specified Match [str] not supported	ACL unsupported match.	See ACL user documentation.
ERROR	Specified watcher [str] not supported	ACL unsupported watcher action.	See ACL user documentation.
ERROR	Specified target [str] not supported	ACL unsupported target action.	See ACL user documentation.
ERROR	Inverse flags not supported for Multiport	ACL inverse match not supported.	See ACL user documentation.
ERROR	Inverse flags	ACL inverse	See ACL user

Severity	Message Text	Explanation	Recommended Action
	not supported for TOS	match not supported.	documentation.
ERROR	Inverse flags not supported for DSCP	ACL inverse match not supported.	See ACL user documentation.
ERROR	Inverse flags or Limit Interface or Source not supported for addrtype	ACL inverse match not supported.	See ACL user documentation.
ERROR	Dest addrtypes other than IPROUTER/ LOCAL not supported	ACL unsupported match for addrtype.	See ACL user documentation.
ERROR	IP6 Fragment IDs not supported	ACL unsupported match.	See ACL user documentation.
ERROR	IP6 Length/ Reserved or Last or More fragment Fields not supported	ACL unsupported match.	See ACL user documentation.

Severity	Message Text	Explanation	Recommended Action
ERROR	Greater/ Lesser/Not equal to mode not supported for TTL field	ACL unsupported match.	See ACL user documentation.
ERROR	Greater/ Lesser/Not equal mode not supported for HL field	ACL unsupported match.	See ACL user documentation.
ERROR	Inverse flags not supported for match mark	ACL inverse match not supported.	See ACL user documentation.
ERROR	TCP SEQ, TCP Options or IP options, UID are not supported for LOG	ACL unsupported match.	See ACL user documentation.
ERROR	TCP SEQ, TCP Options or IP options, UID are not supported for LOG	ACL unsupported match.	See ACL user documentation.

Severity	Message Text	Explanation	Recommended Action
ERROR	Inverse flags or options not supported for TCP	ACL inverse match not supported.	See ACL user documentation.
ERROR	Interface [str] not supported for SPAN	ACL unsupported target interface in SPAN rule.	See ACL user documentation.
ERROR	SPAN target is supported with the following " field(s): –dport swp.”	ACL unsupported SPAN target.	See ACL user documentation.
ERROR	Interface [str] not supported for SPAN	ACL unsupported target interface.	See ACL user documentation.
ERROR	SPAN target is supported with the following field(s): –dport swp.	ACL unsupported SPAN target.	See ACL user documentation.
ERROR	resource region [uint]	ACL: Mellanox API for region	File a ticket with Cumulus

Severity	Message Text	Explanation	Recommended Action
	destroy failed: [str]	destroy failed.	Support.
ERROR	resource region [uint] size [uint] create failed: [str]	ACL: Mellanox API for TCAM region create failed.	Check if there are too many rules/tables.
ERROR	unexpected duplicate key [str]	ACL: Internal interface cache has duplicate.	File a ticket with Cumulus Support.
ERROR	unexpected duplicate user: [str] key_idx [val] offset [val]	ACL: Duplicate entry in interface rule hash.	File a ticket with Cumulus Support.
ERROR	expected trap_id [uint](actual [uint]) type [uint] (actual [val])	ACL: INPUT chain trap counter get ID invalid.	File a ticket with Cumulus Support.
ERROR	table [str] chain [str] region [str] [str] size [uint] creation	ACL: Mellanox API for TCAM region create failed.	Check if there are too many rules/tables.

Severity	Message Text	Explanation	Recommended Action
	failed: [str]		
ERROR	table [str] chain [str] region [str] [str] size [uint] acl_id creation failed: [str]	ACL: Mellanox API for ACL ID create failed.	Check if there are too many rules/tables.
ERROR	table [str] chain [str] region [str] [str] rules del @offset [val] num_rules [val] failed: [str]	ACL: Mellanox API for setting region to ACL ID failed.	File a ticket with Cumulus Support.
ERROR	table [str] chain [str] region [str] [str] size [uint] acl_id destroy failed: [str]	ACL: Mellanox API for unsetting region from ACL ID failed.	File a ticket with Cumulus Support.
ERROR	table [str] chain [str] region [str] [str] size [uint] destroy failed: [str]	ACL: Mellanox API for region destroy failed.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	table [str] chain [str] region [str] [str] rules add @offset [val] num_rules [val] failed: [str]	ACL: Mellanox API for setting rule in a region failed.	File a ticket with Cumulus Support.
ERROR	table [str] chain [str] failed to allocate Mark value [val]	ACL: Mark value alloc failed.	File a ticket with Cumulus Support.
ERROR	table [str] chain [str] Mark values must be less than [val] when partial mask is in use	ACL: Masked mark use limitation.	Change mask match rules.
ERROR	PBR: Unsupported route type ecmp	PBR: Invalid ECMP route.	File a ticket with Cumulus Support.
ERROR	PBR: Unsupported route type singe hop	PBR: Invalid non-ECMP route.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	table [str] chain [str] region [str] action [str] is not supported	ACL: Unsupported action.	Remove rule with the action.
ERROR	PBR: couldn't set default forward action for rule	PBR: Default action accept couldn't be set for PBR rules.	File a ticket with Cumulus Support.
ERROR	[str] offset [uint] or key_idx [uint] exceeds rule list size [uint] or descriptor size [val]	ACL: Too may rules in a region.	File a ticket with Cumulus Support.
ERROR	[str] offset [uint] key_idx [uint] invalid key_id [uint]	ACL: Invalid src/dst port/ intf in rule.	Remove the rule.
ERROR	acl group [str] creation failed: [str]	ACL: Mellanox API for ACL group create failed.	Check for too many ACL tables
ERROR	table [str] chain [str]	ACL: Mellanox API to create	File a ticket with Cumulus

Severity	Message Text	Explanation	Recommended Action
	key handle create failed: [str]	a key list handle failed.	Support.
ERROR	table [str] chain [str] key attr query failed: [str]	ACL: Mellanox API to get key attributes failed.	File a ticket with Cumulus Support.
ERROR	table [str] chain [str] key handle delete failed: [str]	ACL: Mellanox API for key handle delete failed.	File a ticket with Cumulus Support.
ERROR	table [str] chain [str] region [str] [str] size [uint] offset too large	ACL: Rule offset in region larger than region size.	File a ticket with Cumulus Support.
ERROR	region set failed: [str]	ACL: Mellanox API for TCAM region create failed.	Check if there are too many rules/tables.
ERROR	acl set failed: [str]	ACL: Mellanox API for ACL ID create failed.	Check if there are too many rules/tables.
ERROR	rules set failed: [str]	ACL: Mellanox API for rule	Check if there are too many

Severity	Message Text	Explanation	Recommended Action
		set failed.	rules.
ERROR	[str] malloc failed	ACL: Memory resource allocation error.	File a ticket with Cumulus Support.
ERROR	table [str] chain [str] too many keys in rule	ACL: Too many keys used in a rule.	Delete the rule.
ERROR	table [str] chain [str] too many actions in rule”	ACL: Too many actions in rule.	Modify the rule.
ERROR	table [str] chain [str] rule can match on a single output interface only	ACL: Rule match limitation.	Delete the rule.
ERROR	table [str] chain [str] number of input interfaces ([uint]) cannot be less than	ACL: Rule replication limitation.	Remove the rule.

Severity	Message Text	Explanation	Recommended Action
	number of output interfaces ([val])		
ERROR	table [str] chain [str] key classification missing for [val] input bridge interface(s)	ACL: Ingress/ egress intf missing.	Check ACL rules.
ERROR	table [str] chain [str] key classification missing for [val] input interface(s)	ACL: Ingress/ egress port/ bond missing.	Check ACL rules.
ERROR	analyzer set failed: [str]	ACL: Mellanox API for setting ERSPAN analyzer port failed.	File a ticket with Cumulus Support.
ERROR	session [uint] [str] failed: [str]	ACL: Mellanox API for en/dis of SPAN session failed.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	session [uint] edit failed: [str]	ACL: Mellanox API for ERSPAN session modify failed.	File a ticket with Cumulus Support.
ERROR	session [uint] [str] lid 0x%x add failed: [str]	ACL: Mellanox API to set SPAN mirror source failed.	File a ticket with Cumulus Support.
ERROR	session [uint] [str] lid 0x%x delete failed: [str]	ACL: Mellanox API to unset SPAN mirror sources failed.	File a ticket with Cumulus Support.
ERROR	span init failed: [str]	ACL: Mellanox API for SPAN engine initialization failed.	File a ticket with Cumulus Support.
ERROR	Unexpected duplicate session key	ACL: Duplicate SPAN session key.	File a ticket with Cumulus Support.
ERROR	session create failed: [str]	ACL: Mellanox API for session create failed.	File a ticket with Cumulus Support.
ERROR	out of SPAN/	ACL: Too	Check and

Severity	Message Text	Explanation	Recommended Action
	ERSPAN sessions	many SPAN/ ERSPAN session targets.	reduce SPAN sesions.
ERROR	session [uint] analyzer delete failed: [str]	ACL: Mellanox API for session delete failed.	File a ticket with Cumulus Support.
ERROR	session [uint] destroy failed: [str]	ACL: Mellanox API for for session destroy failed.	File a ticket with Cumulus Support.
ERROR	Non flow-based SPAN does not support router interface	ACL: Non-flow based cannot SPAN from routed interface.	Remove these SPAN rules.
ERROR	Non flow-based SPAN does not support sub-interface	ACL: Non-flow based cannot SPAN from routed sub-interface.	Remove these SPAN rules.
ERROR	unsupported session_type [uint]	ACL: Mirror session has to be SPAN or ERSPAN.	Remove these rules.
ERROR	router	ACL: SPAN	Remove these

Severity	Message Text	Explanation	Recommended Action
	interface ([str]) is not supported as SPAN dport	destination cannot be router interface.	rules.
ERROR	unsupported chip type [uint]	ACL: Wrong plaform/chip type.	File a ticket with Cumulus Support.
ERROR	policer creation failed: [str]	ACL: Mellanox internal API to allocate a policer resource failed.	Check number of ACL rules using policers.
ERROR	unexpected duplicate ID [val]	ACL: Duplicate policer ID allocated.	File a ticket with Cumulus Support.
ERROR	policer [val] delete failed [str]	ACL: Mellnox internal API to delete policer resource failed.	File a ticket with Cumulus Support.
ERROR	tricolor conversion failed pir [uint] cir [uint] cbs	ACL: Converting tricolor policer rates to kbps failed.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
	[val] ebs [val]		
ERROR	conversion failed mode [uint] rate [uint] burst [uint]	ACL: Converting color bind policer rates to kpbs failed.	File a ticket with Cumulus Support.
ERROR	counter creation failed: [str]	ACL: Mellanox internal API for allocating a counter failed.	Check number of ACL rules.
ERROR	counter [uint] delete failed: [str]	ACL: Mellanox internal API for counter delete failed.	File a ticket with Cumulus Support.
ERROR	counter [uint] failed: [str]	ACL: Mellanox internal API for counter read failed.	File a ticket with Cumulus Support.
ERROR	policer [val] counter failed: [str]	ACL: Mellanox internal API for for policer counter read failed.	File a ticket with Cumulus Support.
ERROR	invalid interface [str]	ACL: Invalid interface specified in ACL rule.	Check ACL rule set.

Severity	Message Text	Explanation	Recommended Action
ERROR	bond id [uint] not fully established	ACL: Bond interface not fully up.	Check bond interface configuration on local/ remote ends.
ERROR	unsupported interface type: [str]	ACL: An ACL rule has an unsupported type of interface specified in match.	Check ACL rule set for interfaces specified.
ERROR	mixing PBS ports in different swids [uint] and [uint] is not allowed	ACL: SPAN using policy-based switching doesn't support target ports in different units.	Change SPAN rule configuration.
ERROR	unexpected duplicate PBS key with [uint] port(s)	ACL: Duplicate PBS key used for SPAN.	File a ticket with Cumulus Support.
ERROR	create failed for PBS record with [uint] port(s):	ACL: Mellanox internal API for PBS set failed.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
	[str]		
ERROR	pbs_id [uint] delete failed: [str]	ACL: Mellanox internal API for PBS ID delete failed.	File a ticket with Cumulus Support.
ERROR	[str] failed for pbs_id [uint]: [str]	ACL: Mellanox internal API for PBS ID delete failed.	File a ticket with Cumulus Support.
ERROR	group [str] set failed: [str]	ACL: Mellanox internal API that binds an ACL group to a hardware resource like port/VLAN failed.	File a ticket with Cumulus Support.
ERROR	user tokens exhausted	ACL: Mellanox internal API for user token alloc failed.	Check number of ACL rules.
ERROR	hardware platform does not support user tokens	ACL: Mellanox plaform doesn't support user tokens.	File a ticket with Cumulus Support.
ERROR	unexpected duplicate	ACL: User token	File a ticket with Cumulus

Severity	Message Text	Explanation	Recommended Action
	mark key [uint]	duplicate allocated.	Support.
ERROR	bind [str] set failed on port 0x%x: [str]	ACL: Mellanox internal API that binds an ACL group to a port failed.	File a ticket with Cumulus Support.
ERROR	bind [str] unset failed on port 0x%x: [str]	ACL: Mellanox internal API that unbinds an ACL group from a port failed.	File a ticket with Cumulus Support.
ERROR	bind [str] cmd [uint] failed on bond 0x%x: [str]	ACL: Mellanox internal API that un/binds an ACL group with a bond interface failed.	File a ticket with Cumulus Support.
ERROR	bind [str] set failed on port 0x%x: [str]	ACL: Mellanox internal API that binds an ACL group to a bond member failed.	File a ticket with Cumulus Support.
ERROR	bind [str]	ACL: Mellanox	File a ticket

Severity	Message Text	Explanation	Recommended Action
	cmd [uint] failed on RIF 0x[uint]: [str]	internal API that binds an ACL group to an L3 ingress interface failed.	with Cumulus Support.
ERROR	group set failed: [str]	ACL: Mellanox internal API that sets the group of an ACL failed.	File a ticket with Cumulus Support.
ERROR	range creation failed: [str]	ACL: Mellanox internal API to allocate an L4 port range resource failed.	Check number of ranges being used in ACLs.
ERROR	range delete failed: [str]	ACL: Mellanox internal API to free an L4 port range resource failed.	File a ticket with Cumulus Support.
ERROR	table [str] chain [str] region [str] [str] size [uint] creation	ACL: Mellanox API for TCAM region create failed.	Check if there are too many rules/tables.

Severity	Message Text	Explanation	Recommended Action
	failed: [str]		
ERROR	table [str] chain [str] region [str] [str] size [uint] acl_id creation failed: [str]	ACL: Mellanox API for ACL ID create failed.	Check if there are too many rules/tables.
ERROR	table [str] chain [str] region [str] [str] rules del @offset [val] num_rules [val] failed: [str]	ACL: Mellanox API for setting region to ACL ID failed.	File a ticket with Cumulus Support.
ERROR	table [str] [str] chain [str] region [str] [str] rules del @offset [val] num_rules [val] failed: [str]	ACL: Mellanox API for setting region to ACL ID failed.	File a ticket with Cumulus Support.
ERROR	table [str] [str] chain [str] region [str] [str] size	ACL: Mellanox API for unsetting region from	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
	[uint] acl_id destroy failed: [str]	ACL ID failed.	
ERROR	table [str] [str] chain [str] region [str] [str] size [uint] destroy failed: [str]	ACL: Mellanox API for region destroy failed.	File a ticket with Cumulus Support.
ERROR	acl group [str] creation failed: [str]	ACL: Mellanox API for ACL group create failed.	Check for too many ACL tables.
ERROR	table [str] [str] chain [str] region [str] [str] size [uint] offset too large	ACL: Rule offset in region larger than region size.	File a ticket with Cumulus Support.
ERROR	region set failed: [str]	ACL: Mellanox API for TCAM region create failed.	Check if there are too many rules/tables.
ERROR	acl set failed: [str]	ACL: Mellanox API for ACL ID create failed.	Check if there are too many rules/tables.
ERROR	rules set	ACL: Mellanox	Check if there

Severity	Message Text	Explanation	Recommended Action
	failed: [str]	API for rule set failed.	are too many rules.
ERROR	iACL action cannot be satisfied with eACL key	ACL: Invalid dependency between ingress and egress ACLs.	Check ACL rules.
ERROR	eACL action cannot be satisfied with iACL key	ACL: Invalid dependency between ingress and egress ACLs.	Check ACL rules.
ERROR	ACL can match one single output interface only	ACL: Don't support multiple out interface match.	Check ACL rules.
ERROR	expected trap_id [uint] (actual [uint]) type [uint] (actual [val])	ACL: INPUT chain trap counter get ID invalid.	File a ticket with Cumulus Support.
ERROR	[str] [str] API [str]: dlerror [str]	nftables unsupported.	Ignore this error message.
ERROR	Memory allocation failed	ACL out of memory resource.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	Rule with LOG must be followed by same rule with DROP	An ACL rule with a LOG action must be followed by same rule with a DROP action.	See ACL user documentaion for more info.
ERROR	Rule with LOG must be followed by same rule with DROP	An ACL rule with a LOG action must be followed by same rule with a DROP action.	See ACL user documentaion for more info.
ERROR	Rule with log watcher must be have DROP action	An ACL rule with a watcher action must be followed by same rule with a DROP action.	See ACL user documentaion for more info.
ERROR	Rule with LOG must be followed by same rule with DROP	An ACL rule with a LOG action must be followed by same rule with a DROP action.	See ACL user documentaion for more info.

Severity	Message Text	Explanation	Recommended Action
ERROR	[str] not found in hal_bonds	ACL: Bond interface in rule not present.	Check bond configuration.
ERROR	Inverse flags for SRC/DST IP, IN/OUT interface, TOS, Protocol not supported	ACL inverse match flags are not supported.	See ACL user documentaion.
ERROR	Target Verdict :[str] not supported	ACL rule target verdict queue/stop/ return/etc not supported.	See ACL user documentation for supported targets.
ERROR	Fall through target not supported	ACL fall through action not supported.	See ACL documentation for supported actions.
ERROR	Jump, target:[val] not supported	ACL jump action not supported.	See ACL documentation for supported actions.
ERROR	Module, target:[str] not	Specified ACL action not supported.	See ACL documentation for supported

Severity	Message Text	Explanation	Recommended Action
	supported		actions.
ERROR	iptables: Invalid argument	iptables rules likely empty.	Check iptables rules list.
ERROR	iptables:Could not open raw IPv4,socket.	Internal socket error.	File a ticket with Cumulus Support.
ERROR	iptables:Error retrieving getsockopt SO_GET_INFO:	Internal socket error.	File a ticket with Cumulus Support.
ERROR	iptables: Memory allocation for counters failed for size	ACL out of memory resource.	File a ticket with Cumulus Support.
ERROR	iptables: Error retrieving getsockopt SO_GET_ENTRIES:	Internal socket error.	File a ticket with Cumulus Support.
ERROR	iptables: Memory allocation for rules failed for size	ACL out of memory resource.	File a ticket with Cumulus Support.
ERROR	Inverse flags for SRC/DST	ACL inverse match flags	See ACL user documentaion.

Severity	Message Text	Explanation	Recommended Action
	IP, IN/OUT interface, TOS, Protocol not supported	are not supported.	
ERROR	Target Verdict :[str] not supported	ACL rule target verdict queue/stop/ return/etc not supported.	See ACL user documentation for supported targets.
ERROR	Fall through target not supported	ACL fall through action not supported.	See ACL documentation for supported actions.
ERROR	Jump, target:[val] not supported	ACL jump action not supported.	See ACL documentation for supported actions.
ERROR	Module, target:[str] not supported	Specified ACL action not supported.	See ACL documentation for supported actions.
ERROR	iptables: Invalid argument	iptables rules likely empty.	Check iptables rules list.
ERROR	ip6tables:Could not open raw	Internal socket error.	File a ticket with Cumulus

Severity	Message Text	Explanation	Recommended Action
	IPv6,socket.		Support.
ERROR	ip6tables:Error retrieving getsockopt SO_GET_INFO:	Internal socket error.	File a ticket with Cumulus Support.
ERROR	ip6tables: Memory allocation for rules failed for size	ACL out of memory resource.	File a ticket with Cumulus Support.
ERROR	ip6tables: Error retrieving getsockopt SO_GET_ENTRIES:	Internal socket error.	File a ticket with Cumulus Support.
ERROR	iptables: Memory allocation for rules failed for size	ACL out of memory resource.	File a ticket with Cumulus Support.
ERROR	arptables:Could not open raw IPv6 socket.	Internal socket error.	File a ticket with Cumulus Support.
ERROR	arptables:Error retrieving getsockopt SO_GET_INFO: [str]	Internal socket error.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	arptables: Error retrieving getsockopt SO_GET_ENTRIES: [str]	Internal socket error.	File a ticket with Cumulus Support.
ERROR	Inverse flags for SRC/DST MAC, IN/ OUT/logical interface, Protocol not supported	ACL unsupported match.	See ACL documentation for supported matches.
ERROR	logical interface in:[str] out:[str] not supported	ACL unsupported match.	See ACL documentation for supported matches.
ERROR	Protocol field: LENGTH not supported	ACL unsupported LENGTH field match for Ethernet packets.	See ACL documentation for supported matches.
ERROR	Policy not supported	ACL unsupported ebtables action continue/	See ACL documentation for supported actions.

Severity	Message Text	Explanation	Recommended Action
		return/etc.	
ERROR	Target verdict: [str] not supported	ACL unsupported ebtables verdict.	See ACL documentation for supported actions.
ERROR	Fall through or Jump target not supported	ACL fall through/jump action not supported.	See ACL documentation for supported actions.
ERROR	Target verdict: [str] not supported	ACL target verdict queue/stop/ return/etc not supported.	See ACL user documentation for supported targets.
ERROR	ebtables: Invalid argument	iptables rules likely empty or read from kernel failed.	Check ebtables rules list or file a ticket with Cumulus Support.
ERROR	ebtables:Could not open raw IPv4,socket.	Internal socket error.	File a ticket with Cumulus Support.
ERROR	ebtables:Error retrieving getsockopt SO_GET_INFO:	Internal socket error.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	ebtables: Memory allocation for rules failed for size	ACL out of memory resource.	File a ticket with Cumulus Support.
ERROR	ebtables: Error retrieving getsockopt SO_GET_ENTRIES:	Internal socket error.	File a ticket with Cumulus Support.
ERROR	sfs_add: [str] failed	ACL: FUSE file system node creation failed.	File a ticket with Cumulus Support.
ERROR	MAX retries reached, stats sync acl failed - %d	ACL: Kernel updation of stats failed after retries.	File a ticket with Cumulus Support.
ERROR	sync_acl hardware installation failed	Hardware offload of ACL rule set failed, typically due to TCAM resource exhaustion and/or unsupported rules.	Check the ACL rule set/ number of rules/user documentation.

Severity	Message Text	Explanation	Recommended Action
ERROR	ACL: Restore of current table failed: sync_acl hardware installation failed	Hardware restore of previously installed ACL rule set failed.	File a ticket with Cumulus Support.
ERROR	kernel tunnel not found for if_key [str]	L2: Failed to lookup a tunnel interface.	Check configuration.
ERROR	[str] duplicate member [str] for bridge [int]	L2: Duplicate bridge in hash table.	File a ticket with Cumulus Support.
ERROR	tc: u32 ip: unknown match: handle: [hex] index:[int] off:[int] offmask:[hex] val:[hex] mask: [hex]	TC rule harware offload unsupported.	Remove TC rules.
ERROR	tc: u32 ip: match parse failed: handle: [hex],	TC rule harware offload unsupported.	Remove TC rule.

Severity	Message Text	Explanation	Recommended Action
	index:[int], rv: [int]		
ERROR	TC: [str]	TC rule harware offload unsupported.	Remove TC rule.
ERROR	TC: sync_clss hardware installation failed	TC rule harware offload unsupported.	Remove TC rule.
ERROR	IPRULE: sync to h/w failed in non atomic mode. IPRULE rules deleted. Please retry	Failed to install ACL rules.	Remove ACL rules and reinstall ACL rules.
ERROR	IPRULE: event handler failed	Failed to install ACL rules.	Remove ACL rules and reinstall ACL rules.
ERROR	TC: sync to h/w failed in non atomic mode. TC rules deleted. Please retry	Failed to install ACL rules.	Remove ACL rules and reinstall ACL rules.
ERROR	TC: event	Syncing	File a ticket

Severity	Message Text	Explanation	Recommended Action
	handler failed	database failed between kernel and switchd.	with Cumulus Support.
ERROR	sigaction failed for signal [int], [str]	Failed to initialize signal handler.	File a ticket with Cumulus Support.
ERROR	Ignoring VRF [str]; table id 0 is reserved for default VRF	Failed to add a new entry in the ARP table.	
ERROR	Unable to setup handling of SIGHUP for log rotation: [str]	Failed to create a polling thread in NIC init.	File a ticket with Cumulus Support.
ERROR	read error on fd errno [int]	Failed to attach a port in NIC.	File a ticket with Cumulus Support.
ERROR	[str] duplicate member [str] for bridge [int]	Failed to allocate a member in the kernel bridge.	Please check there are no duplicate bridge members.

Severity	Message Text	Explanation	Recommended Action
ERROR	[str]: no context	Failed to append the CSV command in Prescriptive Topology Manager.	File a ticket with Cumulus Support.
ERROR	[str]: Could not append key	Failed to append CSV command in Prescriptive Topology Manager	File a ticket with Cumulus Support.
ERROR	[str]: Could not append val	Failed to append CSV command in Prescriptive Topology Manager	File a ticket with Cumulus Support.
ERROR	[str]: Could not allocate csv	Failed to initialize the CSV command in Prescriptive Topology Manager.	File a ticket with Cumulus Support.
ERROR	[str]: Could not allocate record	Failed to initialize the CSV	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		command in Prescriptive Topology Manager.	
ERROR	[str]: Could not allocate context	Failed to initialize the CSV command in Prescriptive Topology Manager.	File a ticket with Cumulus Support.
ERROR	[str]: no context	Failed to complete the CSV command in Prescriptive Topology Manager.	File a ticket with Cumulus Support.
ERROR	[str]: cannot serialize	Failed to complete the CSV command in Prescriptive Topology Manager.	File a ticket with Cumulus Support.
ERROR	fatal recv error([str]), closing connection,	Failed to complete the CSV command in	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
	rc [int]	Prescriptive Topology Manager.	
ERROR	Cannot allocate csv for msg	Failed to read the CSV command in Prescriptive Topology Manager.	File a ticket with Cumulus Support.
ERROR	[str]: Could not allocate context	Failed to decode the CSV command in Prescriptive Topology Manager.	File a ticket with Cumulus Support.
ERROR	STP mode_set failed for port [int]: [str]	Failed to set spanning tree mode in port [uint], error msg [str]. Forwarding behavior would be impacted by this failure.	File a ticket and contact Cumulus Support.
ERROR	failed to set port [int] vlan_ingress_filter	Failed to set VLAN ingress filter for port	File a ticket and contact Cumulus

Severity	Message Text	Explanation	Recommended Action
	enable	[uint], error msg [str].	Support.
ERROR	failed to set FDB polling interval swid [uint]: [str]	Failed to set FDB polling interval for Mellanox SDK switchd id [int], error msg [str]. Failure to do this impacts MAC address learning behavior.	File a ticket and contact Cumulus Support.
ERROR	failed to set FDB notify_params swid [uint]: [str]	Failed to set FDB MAC address learning notification in Mellanox SDK for switch id [uint], error msg [str]. This error impacts the capability of the switch to learn MAC address.	File a ticket and contact Cumulus Support.
ERROR	failed to	Failed to	File a ticket

Severity	Message Text	Explanation	Recommended Action
	create trap group [uint] trap id [uint] swid [uint] group_attr.prio : [int] error: [str]	create the TRAP groups in the Mellanox SDK. Traps groups are used for policing trap IDs, which are used to punt control packets to OS stack. This failure impacts packet forwarding.	and contact Cumulus Support.
ERROR	failed to open host ifc group [uint] trap id [uint] swid [uint] error [str]	Failed to retrieve the file descriptor of the current open channel to the Mellanox SDK, for ifc group [uint] trap ID [uit] swid [uint], error msg [str]. The error is not recoverable.	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	failed to obtain group [uint] FD for polling	Failed to retrieve the FD for a trap group [id]. The error is not recoverable.	File a ticket and contact Cumulus Support.
ERROR	failed to define trap [uint] group [uint] swid [uint] error: [str]	Failed to set trap ID [uint], trap group [uint], switch ID [uint], for user defined trap, error msg [str] .	File a ticket and contact Cumulus Support.
ERROR	failed to set trap [uint] group [uint] swid [uint] error: [str]	Failed to set trap ID [uint], trap group [uint], switch ID [uint], for user defined trap, error msg [str].	File a ticket and contact Cumulus Support.
ERROR	failed to register trap [uint] swid [uint] error: [str]	Failed to register trap ID [uint] in switch ID [uint] in Mellanox SDK, error	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		msg [str].	
ERROR	trap_id [uint] was not installed	Trap ID [uint] was not installed in the Mellanox SDK. This would impact packet forwarding from the switch ASIC to the control plane.	File a ticket and contact Cumulus Support.
ERROR	trap_id [uint] was not installed	Trap ID [uint] was not installed in the Mellanox SDK. This would impact packet forwarding from the switch ASIC to the control plane.	File a ticket and contact Cumulus Support.
ERROR	dflt_trap_parsing_dflt get failed: [str]	Failed to retrieve the Mellanox Spectrum chip parsing	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		depth from Mellanox SDK, error msg [str]. Possibly the parsing depth has not been set correctly. This would impact hardware packet forwarding.	
ERROR	new_depth [uint] failed: [str]	Failed to set the packet parsing depth [uint] in Mellanox SDK, error msg [str]. This failure impacts hadrware packet forwarding.	File a ticket and contact Cumulus Support.
ERROR	failed to set trap [uint] group [uint] swid [uint] action [uint] error: [str]	Failed to set trap ID [uint], trap group [uint], switch ID [uint], trap action. Failure	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		would lead to the respective control packet not reaching the CPU.	
ERROR	[str] failed to convert trap policer attributes	Failed to get the policer unit for policer group name [str]. Policer unit can be metered with unit of packets for bytes.	File a ticket and contact Cumulus Support.
ERROR	[str] failed to create policer: [str]	Failed to create policer for policer group [str], error msg [str]. Failure to set policer would impact packet forwarding from hadrware data path to	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		CPU.	
ERROR	[str] sw_rate_limiter set failed: [str]	Failed to set the software rate limiter for policy group [str] in Mellanox SDK, error msg [str]. This failure could impact rate limiting for packets forwarded to CPU.	File a ticket and contact Cumulus Support.
ERROR	group [str] failed to edit policer: [str]	Failed to modify policer for policer group [str], error msg [str]. Failure to set policer would impact packet forwarding from hadrware data path to CPU.	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	unknown trap group [uint]	A trap group ID [uint] unknown to the Mellanox SDK is being used to configure the Mellanox SDK policer. This is an internal configuration error.	File a ticket and contact Cumulus Support.
ERROR	group [str] failed to bind policer %” PRIu64 “: [str]	Policer group [str] with policer ID [uint64] failed to bind in the Mellanox SDK, error msg [str]. This error would impact policing of packets being forwarded from hardware to CPU.	File a ticket and contact Cumulus Support.
ERROR	group [str] failed to	Policer group [str] with	File a ticket and contact

Severity	Message Text	Explanation	Recommended Action
	unbind policer %” PRIu64 “: [str]	policer ID [uint64] failed to unbind in the Mellanox SDK, error msg [str]. This error would impact policing of packets being forwarded from hardware to CPU.	Cumulus Support.
ERROR	unsupported type [uint]	Failed to create a trap counter type as the trap counter type [uint] does not match one of the well-defined ones in the Mellanox SDK. This is an internal configuration error.	File a ticket and contact Cumulus Support.
ERROR	unsupported	Failed to	File a ticket

Severity	Message Text	Explanation	Recommended Action
	type [uint]	create a trap counter type as the trap counter type [uint] does not match one of the well-defined ones in Mellanox SDK. This is an internal configuration error.	and contact Cumulus Support.
ERROR	type [uint] failed: [str]	Failed to retrieve the host IFC counter for the counter type [int] from the Mellanox SDK, error msg [str].	File a ticket and contact Cumulus Support.
ERROR	unknown meter_type [uint]	Incorrect policer unit [uint] used to find out the policer group meter unit. Policer unit	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		can be metered with unit of packets for bytes. This is an internal error.	
ERROR	unrecognized lid [hex]	Failed to retrieve the interface key from logical port id [uint].	File a ticket and contact Cumulus Support.
ERROR	[str] unexpected duplicate key [uint]	Failed to add an interface [str] vport with internal VLAN ID [uint] in external VLAN vport hash table because of duplicate entry [uint].	File a ticket and contact Cumulus Support.
ERROR	[str] int_vid [uint] ext_vid [uint]: [str]	Failed to create a virtual port from logical port, interface [str], internal	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		vlan id [uint] and external vlan id[uint], error msg [str]	
ERROR	Unexpected duplicate vport_lid [hex] for [str]	Failed to add vport logical interface id [uint], interface [str], in vlan vport hash table because of duplicate entry [uint]	File a ticket and contact Cumulus Support.
ERROR	delete failed for [str] int_vid [uint] ext_vid [uint]: [str]	Failed to delete a virtual port from logical port, interface [str], internal vlan id [uint] and external vlan id [uint], error msg [str]	File a ticket and contact Cumulus Support.
ERROR	[str] vrid not found for table [uint]	virtual router id not found in virtual id table [id] in	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		software	
ERROR	delete failed for int_vid [uint] ext_vid [uint] vport_lid [hex] : [str]	Failed to delete a virtual port from logical port, internal vlan id [uint], virtual port logical if [uint] and external vlan id [uint], error msg [str]	File a ticket and contact Cumulus Support.
ERROR	[str] vrid not found for table [uint]	virtual router id not found in virtual id table [id] in software	File a ticket and contact Cumulus Support.
ERROR	port [int] ext_vlan [int] already exists	port [uint] and external vlan [uint] already exists in the e2i table	File a ticket and contact Cumulus Support.
ERROR	[str] int_vlan [int] already assigned to [str]	interface [str] with internal vlan [uint] is already assigned to	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		interface [str] in the e2i table. This is an internal configration error	
ERROR	failed to get base bond for [str]	Failed to get the bond interface for interface [str]	File a ticket and contact Cumulus Support.
ERROR	failed to add to interface ht s[str]	Failed to add interface [str] to the ifp hash table because an entry already exists	File a ticket and contact Cumulus Support.
ERROR	[str] old_int_vlan [int] inconsistent	interface [str] with vlan [uint] is inconsistent in the e2i table	File a ticket and contact Cumulus Support.
ERROR	[str] new_int_vlan [int] already assigned to [str]	interface [str] with internal vlan [uint] is already assigned to interface [str]	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		in the e2i table. This is an internal configration error	
ERROR	UC flood block [uint] failed for [str] vlan [uint]: [str]	unicast flood block [uint] failed for interface [str] for vlan id [uint], error msg [str]	File a ticket and contact Cumulus Support.
ERROR	learn mode [uint] failed for [str] vlan [uint]: [str]	learn mode [uint] failed for interface [str] and internal vlan [uint], error msg [str]	File a ticket and contact Cumulus Support.
ERROR	error processing bridge vlan information	error processing bridge vlan information	File a ticket and contact Cumulus Support.
ERROR	bond_mbrs_vlan_port_set failed for bond: [int]	failed to set vlan for bond members for bond id [uint]	File a ticket and contact Cumulus Support.
ERROR	unsupported	unsupported	File a ticket

Severity	Message Text	Explanation	Recommended Action
	interface type: [uint]	interface type [uint]	and contact Cumulus Support.
ERROR	cannot find STG for bridge_vlan [uint] vid [uint]	cannot find the spanning tree group for bridge vlan [uint] and vlan id [uint]	File a ticket and contact Cumulus Support.
ERROR	flood_mode_set failed for swid [int] vid [int]	Flood mode could not be set for unregistered multicast in swid [uint] vlan id [uint]	File a ticket and contact Cumulus Support.
ERROR	vlans set failed for [str]: [str]	setting of vlan failed for interface [str], error msg [str]	File a ticket and contact Cumulus Support.
ERROR	qinq mode set failed for [str]: [str]	failed to set qinq mode for interface [str], error msg [str]	File a ticket and contact Cumulus Support.
ERROR	qinq mode set failed for	failed to set qinq mode	File a ticket and contact

Severity	Message Text	Explanation	Recommended Action
	[str]: [str]	for bond for interface [str], error msg [str]	Cumulus Support.
ERROR	unsupported interface type: [uint]	unsupported interface type [uint]	File a ticket and contact Cumulus Support.
ERROR	bond id [uint] not fully created	bond id [uint] creation is not complete	File a ticket and contact Cumulus Support.
ERROR	cannot find bridge vlan for bridge: [int]	unable to find bridge vlan for the bridge id [uint]	File a ticket and contact Cumulus Support.
ERROR	cannot find bond vlan for bond	cannot find bond vlan for the bond	File a ticket and contact Cumulus Support.
ERROR	cannot allocate vlan for bond interface	Failed to allocate vlan for bond interface	File a ticket and contact Cumulus Support.
ERROR	cannot allocate vlan for sub-interface	Failed to allocate vlan for sub interface	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	gre tunnel decap entry creation failed : [str]	Failed to create decapsulation entry in Mellanox SDK. Decapsulation of GRE packet woould not be operational, error msg [str]	File a ticket and contact Cumulus Support.
ERROR	gre tunnel decap destroy failed : [str]	Failed to delete the GRE decapsulation entry in Mellanox SDK, error msg [str]	File a ticket and contact Cumulus Support.
ERROR	gre tunnel curr decap entry delete failed : [str]	Failed to delete the GRE decapsulation entry in Mellanox SDK, error msg [str]	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	gre tunnel new decap entry update failed : [str]	Failed to create decapsulation entry in Mellanox SDK. Decapsulation of GRE packet woould not be operational, error msg [str]	File a ticket and contact Cumulus Support.
ERROR	failed to make logical gre key	Failed to form the logical GRE key from the interface information provided	File a ticket and contact Cumulus Support.
ERROR	failed to make gre decap key	Failed to form the logical decap GRE key from the information provided	File a ticket and contact Cumulus Support.
ERROR	failed to make overlay key from underlay key	Failed to create overlay gre key from underlay	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		information	
ERROR	unable to find gre entry for tunnel id ([hex]	Failed to find gre entry from tunnel id in the gre tunnel key hash table, using tunnel id [uint]	File a ticket and contact Cumulus Support.
ERROR	duplicate entry in overlay ht : ifindex ([int]	Unable to add a duplicate gre entry with ifindex [uint] in the gre overlay hash table. A duplicate config is being attempted	File a ticket and contact Cumulus Support.
ERROR	failed to create overlay rif : ifindex : [int] tunnel type [uint] key [uint]	Unable to create an overlay router interface with ifindex [uint] , tunnel type [uint] and tunel key [uint]	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	gre tunnel creation failed: [str] :	Failed to create tunnel id for GRE in Mellanox SDK, error msg [str]	File a ticket and contact Cumulus Support.
ERROR	invalid argument	GRE update is being called with an invalid GRE information, no operation would be performed	File a ticket and contact Cumulus Support.
ERROR	gre tunnel ([hex]) update failed: [str] :	Failed to update tunnel id [hex] for GRE in Mellanox SDK, error msg [str]	File a ticket and contact Cumulus Support.
ERROR	gre tunnel destroy failed: [str]	Failed to delete tunnel id for GRE in Mellanox SDK, error msg [str]	File a ticket and contact Cumulus Support.
ERROR	loopback rif for ifindex	Failed to add the loopback	File a ticket and contact

Severity	Message Text	Explanation	Recommended Action
	([int]) : [str]	router interface, interface ifindex [uint] in Mellanox SDK, error [str]	Cumulus Support.
ERROR	ifindex ([int]) overlay rif ([int]) : [str]	Failed to delete the loopback router interface, interface ifindex [uint], overlay router interface [uint] in Mellanox SDK, error [str]	File a ticket and contact Cumulus Support.
ERROR	cannot allocate bridge vlan for bridge id [int]	Failed to allocate bridge vlan for bridge id [uint]	Check The Cumulus Linux Configuration guide
ERROR	flood_mode_set failed for swid [int] vid [int]	Flood mode could not be set for unregistered multicast in	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		swid [uint] vlan id [uint]	
ERROR	cannot allocate ln_vlan [uint] for bridge_id [int]	Failed to allocate vlan [uint] for bridge id [uint]	Check The Cumulus Linux Configuration guide
ERROR	flood_mode_set failed for swid [int] vid [int]	Flood mode could not be set for unregistered multicast in swid [uint] vlan id [uint]	File a ticket and contact Cumulus Support.
ERROR	vlan [uint] not yet allocated	vlan [uint] does not exists for the bridge and is not allocated	Check The Cumulus Linux Configuration guide
ERROR	[str] bridge_id [uint] vlan [uint] port [hex] failed: [str]	failed to add a unicast mac address [str] on bridge id [uint] vlan [uint] port [uint]. This could be an internal error or could be	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		because of configuration error	
ERROR	[str] bridge_id [uint] vlan [uint] port [hex] failed: [str]	failed to delete a unicast mac address [str] on bridge id [uint] vlan [uint] port [uint]. This could be an internal error or could be because of configuration error	File a ticket and contact Cumulus Support.
ERROR	num_macs [uint] num_failed_macs [uint] delete failed: [str]	failed to delete a number [uint] of unicast mac address, error msg [str]. This could be an internal error or could be because of configuration error	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	num_macs [uint] learn set failed: [str]	failed to add [uint] unicast mac addressess, error msg [str]. This could be because of resource exhaustion	File a ticket and contact Cumulus Support.
ERROR	num_macs [uint] delete failed: [str]	failed to delete a number [uint] of unicast mac address, error message [str]. This could be an internal error or could be because of configuration error	File a ticket and contact Cumulus Support.
ERROR	age_time set failed [str] on swid [uint]	Failed to set fdb ageing time. This would cause the mac addresses in FDB not to	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		age mac address	
ERROR	cannot find vlan for brmac [str] vfid [uint]	vlan [uint] does not exists for the bridge and so could not find the vlan for bridge mac address	Check The Cumulus Linux Configuration guide
ERROR	vfid not set for vlan [uint]	failed to return a translated vlan id for vlan [uint]	File a ticket and contact Cumulus Support.
ERROR	num_macs [uint] delete failed: [str]	failed to delete a number [uint] of unicast mac address, error msg [str]. This could be an internal error or could be because of configuration error	File a ticket and contact Cumulus Support.
ERROR	bridge_vlan	bridge vlan id	Check The

Severity	Message Text	Explanation	Recommended Action
	[uint] expected swid [uint] but found [uint]	[uint] expected switchd id [uint] for the vlan is [uint] switch id	Cumulus Linux Configuration guide
ERROR	num_macs [uint] delete failed: [str]	failed to delete a number [uint] of unicast mac address, error msg [str]. This could be an internal error or could be because of configuration error	File a ticket and contact Cumulus Support.
ERROR	bridge_vlan [uint] expected swid [uint] but found [uint]	bridge vlan id [uint] expected switchd id [uint] for the vlan is [uint] switch id	Check The Cumulus Linux Configuration guide
ERROR	get failed: [str]	Failed to get fdb unicast mac address from Mellanox	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		SDK, error msg [str]	
ERROR	num_macs [uint] delete failed: [str]	failed to delete a unicast mac address [str] on bridge id [uint] vlan [uint] port [uint]. This could be an internal error or could be because of configuration error	File a ticket and contact Cumulus Support.
ERROR	internal vlans exhausted	total number of internal vlan has exhausted. No morevlans could be addded	Check The Cumulus Linux Configuration guide
ERROR	identity map failed for vlan [uint]: [str]	Failed to map the forwarding id to the vlan id [uint] in Mellanox SDK, error	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		msg [str]	
ERROR	learn mode_failed for vlan [uint]: [str]	failed to set the learning mode for vlan id [uint], error message [str]	File a ticket and contact Cumulus Support.
ERROR	failed to get members for vlan [uint]: [str]	Failed to get member port for vlan [uint], error message [uint]	File a ticket and contact Cumulus Support.
ERROR	vlan [uint] is not an L3 vlan	vlan [uint] entry is not representing a l3 interface	Check The Cumulus Linux Configuration guide
ERROR	unsupported interface type: [uint]	unsupported interface type [uint]	File a ticket and contact Cumulus Support.
ERROR	[hex] int_vlan [uint] failed: [str]	failed to set the internal vlan [uint] for the logical port id [uint], error msg [str]	Check The Cumulus Linux Configuration guide

Severity	Message Text	Explanation	Recommended Action
ERROR	[hex] pvid [uint] failed: [str]	failed to set the logical interface [uint] to van id [uint], error msg [str]	Check The Cumulus Linux Configuration guide
ERROR	[hex] int_vlan [uint] failed: [str]	failed to unset the internal vlan [uint] for the logical port id [uint], error msg [str]	Check The Cumulus Linux Configuration guide
ERROR	[hex] revert pvid: [str]	failed to delete the logical interface [uint] to vlan id [uint], error msg [str]	Check The Cumulus Linux Configuration guide
ERROR	unsupported interface type: [uint]	unsupported interface type [uint]	File a ticket and contact Cumulus Support.
ERROR	unsupported interface type: [uint]	unsupported interface type [uint]	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	failed for lid [hex] int_vlan [uint] STG [uint]: [str]	Failed to set the spanning tree group for logical interface [uint], internal vlan [uint] spanning tree group [uint], error msg [%s]	File a ticket and contact Cumulus Support.
ERROR	unsupported if_type [uint]	unsupported interface type [uint]	File a ticket and contact Cumulus Support.
ERROR	port [str] not established	port interface [str] has not been established yet	File a ticket and contact Cumulus Support.
ERROR	failed for [str] lid [hex]: [str]	Failed to set the port, logical id [uint], interface [str], to accept the frame type, error msg [str]	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	list allocation failed	failed to allocate memory for the ports	File a ticket and contact Cumulus Support.
ERROR	STGs exhausted	Total number of spanning tree group has exhausetd. please consult configuration manual	Check The Cumulus Linux Configuration guide
ERROR	MSTP instance set failed for STG [int]: [str]	Failed to set the MSTP instance for the spanning tree group [uint] in Mellanox SDK, error msg [str]	Check The Cumulus Linux Configuration guide
ERROR	failed to delete STG [uint]: [str]	Failed to delete the MSTP instance for the spanning tree group [uint] in Mellanox	Check The Cumulus Linux Configuration guide

Severity	Message Text	Explanation	Recommended Action
		SDK, error msg [str]	
ERROR	failed to add vlan [int] to STG [int]: [str]	Failed to add vlan [uint] to spanning tree group [uint] in Mellanox SDK, error msg [str]	File a ticket and contact Cumulus Support.
ERROR	failed to remove vlan [int] from STG [int]: [str]	Failed to delete vlan [uint] to spanning tree group [uint] in Mellanox SDK, error msg [str]	File a ticket and contact Cumulus Support.
ERROR	vlan [uint] not yet created	vlan [uint] does not exist and is not allocated	Check The Cumulus Linux Configuration guide
ERROR	STG [int] not yet created	spanning tree group id [uint] is not created	File a ticket and contact Cumulus Support.
ERROR	Duplicate vfid [uint]	Failed to add virtual	File a ticket and contact

Severity	Message Text	Explanation	Recommended Action
		forwarding id [uint] in hash table because of a duplicate entry	Cumulus Support.
ERROR	fdb_uc_mac_addr_get failed: [str]	Failed to get fdb unicast mac address from Mellanox SDK, error msg [str]	File a ticket and contact Cumulus Support.
ERROR	failed to allocate mac_list	Failed to allocate mac address list	Check The Cumulus Linux Configuration guide
ERROR	init set failed: [str]	Initilizition of router module in Mellanox SDK failed.	File a ticket and contact Cumulus Support.
ERROR	hash params set failed: [str]	Initilizition of router ecmp hash module in Mellanox SDK failed.	File a ticket and contact Cumulus Support.
ERROR	router #[uint] set failed:	Initilizition of virtual router	Check The Cumulus

Severity	Message Text	Explanation	Recommended Action
	[str]	id [uint] failed, because of error [str] in Mellanox SDK.	Linux Configuration guide
ERROR	[str] failed cmd [str] vlan [uint] mac [str] fwd_state [str]: [str]	Routed interface description [str] failed to add/update/ delete [str] with vlan [uint] mac [str] forwarding state [str]	Check The Cumulus Linux Configuration guide
ERROR	[str] cmd [str] failed for vlan [uint] mac [str]: [str]	Routed interface description [str] failed to add/update/ delete [str] with vlan [uint] mac [str] error [str]	Check The Cumulus Linux Configuration guide
ERROR	failed for intf [uint]: [str]	Failed to get retrieve router	File a ticket and contact Cumulus

Severity	Message Text	Explanation	Recommended Action
		interface id [uint], errorstr [str]	Support.
ERROR	failed for vlan [uint] intf [uint]: [str]	Failed to delete interface for vlan [uint] interface id [uint] error [str]	File a ticket and contact Cumulus Support.
ERROR	neigh delete all failed for intf [uint]: [str]	Deletion of all the neighbor entries for the interface id [uint] error [str]	File a ticket and contact Cumulus Support.
ERROR	interface state set failed for l3_intf_id [uint]: [str]	Failed to set state for l3 interface id [uint] error [str]	File a ticket and contact Cumulus Support.
ERROR	invalid router mac [str]	check for a unicast mac address [str] failed	Check The Cumulus Linux Configuration guide
ERROR	failed for l3_intf_id	Failed to add mac address	File a ticket and contact

Severity	Message Text	Explanation	Recommended Action
	[int] mac [str] vlan [uint]: [str]	[str] for interface id [uint] vlan [uint] error [str]	Cumulus Support.
ERROR	failed for l3_intf_id [int] mac [str] vlan [uint]: [str]	Failed to delete mac address [str] for interface id [uint] vlan [uint] error [str]	File a ticket and contact Cumulus Support.
ERROR	Invalid table id: must be between [int] and [int]	An Invalid vrf id is being tried to program, vrf id should be in range of vrf id [uint] - [uint]	File a ticket and contact Cumulus Support.
ERROR	calloc failed to allocate a new ECMP entry	Failed to allocate a new entry for ECMP	File a ticket and contact Cumulus Support.
ERROR	unexpected duplicate SDK ecmp id [uint] (HAL ECMP id [int]	Failed to add an ECMP entry id [uint] in hash table because of	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		duplicate ECMP ID [uint]	
ERROR	unable to reconstruct original route [str]	Failed to construct a route [str] from a hardware entry to a hardware abstraction layer entry entry, for the purpose of updating the existing entry	File a ticket and contact Cumulus Support.
ERROR	unable to obtain HW info for original route [str]	Failed to retrieve a route [str] from the hardware, for the purpose of updating the existing entry	File a ticket and contact Cumulus Support.
ERROR	activity get failed: [str]	Failed to check the neighbor activity information,	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		error [str]	
ERROR	route [str] not found in hal_routes	Failed to find the route [str] in software	File a ticket and contact Cumulus Support.
ERROR	unexpected duplicate l3_intf_id [uint]	Found duplicate l3 interface id [uint] in software	File a ticket and contact Cumulus Support.
ERROR	unexpected duplicate RIF param [uint]	Found duplicate l3 interface id [uint] parameters in software	File a ticket and contact Cumulus Support.
ERROR	unexpected duplicate l3_intf_id [uint] vlan [uint]	Found duplicate l3 interface id [uint] having vlan [uint] in software	File a ticket and contact Cumulus Support.
ERROR	failed for vlan [uint] l3_intf_id [uint]: [str]	Failed to delete the router interface, vlan [uint] interface if	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		[uint] in Mellanox SDK, error [str]	
ERROR	unexpected duplicate entry	Failed to add the router interface because of duplicate entry	File a ticket and contact Cumulus Support.
ERROR	vlan not found for l3_intf_id [uint]	Failed retrieve l3 interface [uint] param	File a ticket and contact Cumulus Support.
ERROR	[str] vlan [uint] does not exist for bridge_id [int]	vlan [uint] does not exists for the bridge id [uint]	Check The Cumulus Linux Configuration guide
ERROR	[str] no bridge exists for bridge_id [int]	bridge id [uint] does not exists in the software database	Check The Cumulus Linux Configuration guide
ERROR	[str] cannot allocate vlan [uint] for bridge_id	failed to allocate bridge id [uint] for vlan	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
	[int]	id [uint] in mellaox SDK.	
ERROR	interface [str] not an svi	Interface [str] type is expected to be SVI, but it is not SVI	File a ticket and contact Cumulus Support.
ERROR	[str] vrid not found for table [uint]	virtual router id not found in virtual id table [id] in software	File a ticket and contact Cumulus Support.
ERROR	invalid interface: [str]	An interface check has found that the .	File a ticket and contact Cumulus Support.
ERROR	[str] vrid not found for table [uint]	virtual router id not found in virtual id table [id] in software	File a ticket and contact Cumulus Support.
ERROR	neighbor set failed for netdev_rif [uint]: [str]	Setting up of a neighbor failed on the netdev router interface [uint]	File a ticket and contact Cumulus Support.
ERROR	neighbor	Deletion of	File a ticket

Severity	Message Text	Explanation	Recommended Action
	delete failed: [str]	the neighbor entry failed, error [str]	and contact Cumulus Support.
ERROR	unexpected duplicate:	Addition of the interface to the software table failed because of an already existing interface	File a ticket and contact Cumulus Support.
ERROR	Failed to get vrid for table [uint]	virtual router id not found in virtual id table [id] in software	File a ticket and contact Cumulus Support.
ERROR	Failed to get vrid for table [uint]	virtual router id not found in virtual id table [id] in software	File a ticket and contact Cumulus Support.
ERROR	route_op [uint] neighbor route must have a valid netdev:	route operation to add/del [uint] failed as the route does not have a valid next hop	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		net device	
ERROR	cannot find vlan_if for next hop [str]	vlan interface [str] could not be found for the next hop, as the next hop programming is being done on a vlan interface	File a ticket and contact Cumulus Support.
ERROR	cannot find parent bond info	Failed to retrieve the master bond interface descriptor for port id [uint] and vlan id [uint]	File a ticket and contact Cumulus Support.
ERROR	no RIF found for [str]	Router interface [str] not found for neighbor	File a ticket and contact Cumulus Support.
ERROR	unexpected rif type [str]	A router interface type [str] found and it is an invalid type of router	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		interface	
ERROR	neigh_get vrid [uint] failed: [str]	Failed to retrieve the neighbor from the virtual router id [uint]	File a ticket and contact Cumulus Support.
ERROR	failed to allocate neigh_entry_list	Failed to allocate memory for a list of neighbors	File a ticket and contact Cumulus Support.
ERROR	route cmd [str] failed for vrid [int]: [str]	Failed operation [str] on a unicast route on vrid [uint]	File a ticket and contact Cumulus Support.
ERROR	for [str]	Failed to set a unicast route [str] in Mellanox SDK.	File a ticket and contact Cumulus Support.
ERROR	route delete failed for [str]: [str]	Failed to delete route in route [str] in the Mellanox SDK, error	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		msg [str]	
ERROR	unexpected duplicate:	Found duplicate route in software	File a ticket and contact Cumulus Support.
ERROR	unknown route type [uint]	Route inspected is an unknown route type [uint], unsupported by Mellanox SDK.	File a ticket and contact Cumulus Support.
ERROR	ECMP SDK id [uint] not found	ECMP id [uint] for the route not found in the Mellanox SDK.	File a ticket and contact Cumulus Support.
ERROR	Failed to get vrid for table [uint]	virtual router id not found in virtual id table [id] in software.	File a ticket and contact Cumulus Support.
ERROR	too many next hops [int] for hal_route	number of next hop [uint] exceeded the	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
	[str]	max limit for the route [str].	
ERROR	Cannot allocate an ECMP key	Failed to allocate an ECMP key to program a new route.	File a ticket and contact Cumulus Support.
ERROR	Failed to get vrid for table [uint]	virtual router id not found in virtual id table [id] in software.	File a ticket and contact Cumulus Support.
ERROR	unknown route type [uint]	Route inspected is an unknown route type [uint], unsupported by Mellanox SDK.	File a ticket and contact Cumulus Support.
ERROR	route get failed for prefix_type [uint]: [str]	Failed to retrieve a unicast route from Mellanox SDK, error reason [str].	File a ticket and contact Cumulus Support.
ERROR	Failed to get	Failed to	File a ticket

Severity	Message Text	Explanation	Recommended Action
	table_id for vrid [uint]	retrieve vritual router table id for virtual router id [uint].	and contact Cumulus Support.
ERROR	sx_api_router_uc_route_get failed: [str]	Failed to retrieve a unicast route from Mellanox SDK, error reason [str].	File a ticket and contact Cumulus Support.
ERROR	failed to allocate uc_route_entry_list	Failed to allocate a list for unicast route entries.	File a ticket and contact Cumulus Support.
ERROR	ECMP hal_route_to_hw_ecmp_key failed	Failed to retrieve ecmp key for a route from Mellanox SDK.	File a ticket and contact Cumulus Support.
ERROR	unable to find gre entry for tunnel	Failed to find a GRE tunnel entry.	File a ticket and contact Cumulus Support.
ERROR	unable to make logical gre key from	Failed to create a logical GRE	File a ticket and contact Cumulus

Severity	Message Text	Explanation	Recommended Action
	if_key	key from the interface key.	Support.
ERROR	ECMP route contains one or more unresolvable nexthops	ECMP route has one or more unresolved next hop. So programming for the route is not performed.	File a ticket and contact Cumulus Support.
ERROR	ecmp: can't hold the ECMP entry	ECMP container NULL or invalid, so route entry programming would not be performed.	File a ticket and contact Cumulus Support.
ERROR	onlink host route key setup failed	Failed to create onlink host route key.	File a ticket and contact Cumulus Support.
ERROR	onlink host route creation failed	Failed to create onlink host route.	File a ticket and contact Cumulus Support.
ERROR	sx_api_router_ecmp_ip_del_set	Failed to	File a ticket

Severity	Message Text	Explanation	Recommended Action
	failed on parent SDK ECMP ID [int]: [str]	clone ecmp route to the parent ecmp	and contact Cumulus Support.
ERROR	ecmp: empty ECMP container add failed: [str]	Failed to create a new and empty ecmp container with attriuts in Mellanox SDK, status msg [str].	File a ticket and contact Cumulus Support.
ERROR	ecmp: ECMP [str] failed: [str] num_next_hops is [int].	Failed to (create/ update) operation [str], an ecmp container, status msg [str], with number of nexthops [uint].	File a ticket and contact Cumulus Support.
ERROR	unexpected duplicate ECMP key	Failed to add an ECMP key in ecmp key hash table, as a duplicate entry already	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		exists.	
ERROR	unexpected duplicate ECMP SDK id [uint]	Failed to add an ECMP id in ecmp id hash table, as a duplicate entry already exists.	File a ticket and contact Cumulus Support.
ERROR	ecmp_id [uint] delete failed: [str]	Failed to delete a ecmp entry, ecmp id [uint], status msg [str].	File a ticket and contact Cumulus Support.
ERROR	onlink host route key setup failed.	Failed to create onlink host route key	File a ticket and contact Cumulus Support.
ERROR	ecmp pbr refcount: can’t hold the ECMP sdkid: [int] entry.	ecmp entry not found in the software while programming entries for PBR.	File a ticket and contact Cumulus Support.
ERROR	ecmp pbr refcount: can’t put the	ecmp entry not found in the software	File a ticket and contact Cumulus

Severity	Message Text	Explanation	Recommended Action
	ECMP sdkid: [int] entry.	while programming entries for PBR.	Support.
ERROR	SDK ecmp_id [uint] failed: [str]	Failed to set ecmp attributes with ecmp id [uint] in Mellanox SDK, error mesg [str].	File a ticket and contact Cumulus Support.
ERROR	unexpected duplicate ECMP clone ID key	Failed to add an ECMP clone id in, clone id hash table, as a duplicate entry already exists.	File a ticket and contact Cumulus Support.
ERROR	onlink host route not suppported on non-Spectrum backend	Onlink host route not supported on non-spectrum backend.	File a ticket and contact Cumulus Support.
ERROR	onlink host route not suppported	Onlink host route not supported on	File a ticket and contact Cumulus

Severity	Message Text	Explanation	Recommended Action
	on non-Spectrum backend	non-spectrum backend.	Support.
ERROR	unexpected duplicate:	Failed to add an ECMP entry in hash table because a duplicate entry already exists.	File a ticket and contact Cumulus Support.
ERROR	cannot find vlan_if for next hop [str]	vlan interface [str] could not be found for the next hop, as the next hop programming is being done on a vlan interface.	File a ticket and contact Cumulus Support.
ERROR	invalid rif for [str]	An interface check found that routed interface [str] is invalid.	File a ticket and contact Cumulus Support.
ERROR	hal_clag_set_port_egress_ failed in backend[[int]] for	Failed to install egress mask on MLAG port.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
ERROR	hal_clag_set_ln_egress_to failed in backend[[int]] for	Failed to install egress mask on VXLAN device.	File a ticket with Cumulus Support.
ERROR	ln_key [int] anycast_ip not set	MLAG anycast IP is not set.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_port_to_prio_map_get hal port [int] returned [str]	ASIC egress queue map configuration read failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_port_to_prio_map_set logical port 0x%x returned [str]	ASIC egress queue map configuration write failed.	File a ticket with Cumulus Support.
WARNING	hal_mlx_priority_source_trust_get HAL port [int] logical port 0x%x returned [str]	ASIC priority source trust configuration read failed.	File a ticket with Cumulus Support.
WARNING	hal_mlx_priority_source_trust_set HAL port [int] logical port 0x%x returned [str]	ASIC priority source trust configuration write failed.	File a ticket with Cumulus Support.
WARNING	hal_mlx_rewrite_enable_get HAL port	ASIC priority rewrite	File a ticket with Cumulus

Severity	Message Text	Explanation	Recommended Action
	[int] logical port 0x%x returned [str]	enable configuration read failed.	Support.
WARNING	hal_mlx_rewrite_enable_set HAL port [int] logical port 0x%x returned [str]	ASIC priority rewrite enable configuration write failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_port_tc_shared_ hal port [int] returned [str]	ASIC MC buffer configuration read failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_port_tc_shared_ hal port [int] returned [str]	ASIC MC buffer configuration write failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_port_tc_sp_egress hal port [int] returned [str]	ASIC egress queue map configuration write failed.	File a ticket with Cumulus Support.
WARNING	buffer pool [int] size 0 is invalid: this pool was not created	Invalid buffer pool size.	Check back end QoS configuration file.
WARNING	Pool configuration mode [int]	Invalid parameter.	Check back end QoS configuration

Severity	Message Text	Explanation	Recommended Action
	not recognized: defaulting to buffer units		file.
WARNING	sx_api_cos_shared_buff_pool_set for sw pool id [int] to size [int] (mode [int]) failed: [str],	ASIC buffer pool configuration write failed.	File a ticket with Cumulus Support.
WARNING	Hardware buffer pool index [int] too large (max is [int])	Too many buffer pools for ASIC limit.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_port_prio_buff_map_get failed for MLX port [int]: [str]	Priority to buffer map read failed.	File a ticket with Cumulus Support.
WARNING	switch priority [int] greater than max value [int]	Invalid switch priority.	Check QoS configuration file.
WARNING	sx_api_cos_port_prio_buff_map_set failed for MLX port [int]: [str]	ASIC buffer configuration write failed.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
WARNING	reserved buffer type [int] not recognized	Invalid buffer type.	File a ticket with Cumulus Support.
WARNING	cos ID [int] larger than maximum switch priority value [int]	Invalid internal switch priority value.	File a ticket with Cumulus Support.
WARNING	profile element index [int] too large for array size [int]: [int] map entries, priority field [int]	Priority profle index is too large for the array.	File a ticket with Cumulus Support.
WARNING	[str] failed for MLX port 0x%x, buffer count [int]: [str]	ASIC packet buffer configuration write failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_shared_buff_pool_get failed, cannot get pool size or mode : [str]	ASIC buffer pool configuration read failed.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
WARNING	pool [int] mode [int] not recognized	Invalid buffer pool mode.	File a ticket with Cumulus Support.
WARNING	MLX logical port 0x%x: cos ID [int] larger than maximum switch priority value [int]	Invalid switch priority value.	File a ticket with Cumulus Support.
WARNING	unlimited egress buffer for flow controlled switch priority [int]: unicast config may not match multicast config on some ports	Possible buffer configuration conflict.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_shared_buff_pool_get failed, cannot report pool configurations: [str]	ASIC buffer pool configuration read failed.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
WARNING	sx_api_cos_pools_buffer_get, pool count == 0, failed: [str]	Buffer pool configuration read failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_pools_buffer_get, pool count == [uint] failed: [str]	Buffer pool configuration read failed.	File a ticket with Cumulus Support.
WARNING	_pool_buffer_list_get failed: [str]	Buffer pool configuration read failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_shared_buff_pool_set failed, cannot report pool configurations: [str]	ASIC packet buffer configuration write failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_port_prio_get port [int] (0x%x) returned [str]	ASIC L2 priority source map get operation failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_port_buff_type_set failed for HAL port [int]/MLX port [int]: [str]	ASIC packet buffer configuration write failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_port_shared_buff_type_set failed for HAL	ASIC packet buffer	File a ticket with Cumulus

Severity	Message Text	Explanation	Recommended Action
	port [int]/MLX port [int]: [str]	configuration write failed.	Support.
WARNING	sx_api_cos_port_pcpdei_to_prio_set port [int] logical port 0x%x returned [str]	ASIC L2 priority source map set operation failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_port_dscp_to_prio_get port [int] (0x%x) returned [str]	ASIC L3 priority source map get operation failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_port_dscp_to_prio_set port [int] returned [str]	ASIC L3 priority source map set operation failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_port_prio_to_pcpdei_rewrite_set hal port [int] element count [int]: returned [str]	ASIC L2 priority remark map set operation failed.	File a ticket with Cumulus Support.
WARNING	switch priority [int] color [int] pcp [int] deci	failed L2 priority remark map.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
	[int]		
WARNING	sx_api_cos_port_prio_cfg_set hal port [int] logical_port [int] element count [int]: returned [str]	ASIC_3 dscp_rewrite_set priority remark map set operation failed.	File a ticket with Cumulus Support.
WARNING	switch priority [int] color [int] dscp [int]	failed L3 priority remark map.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_port_ets_element_get logical port 0x%x returned [str]	ASIC scheduler configuration read failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_port_ets_element_set (destroy) logical port 0x%x returned [str]	ASIC scheduler configuration write failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_port_ets_element_get hal port [int] returned [str]	ASIC scheduler configuration read failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_port_ets_element_set logical port 0x%x level [int] index	ASIC scheduler configuration write failed.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
	[int] returned [str]		
WARNING	sx_api_cos_port_ets_element_get hal port [int] returned [str]	ASIC scheduler configuration read failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_port_ets_element_set (destroy) hal port [int] returned [str]	ASIC scheduler configuration write failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_port_ets_element_set hal port [int] level [int] index [int] returned [str]	ASIC scheduler configuration write failed.	File a ticket with Cumulus Support.
WARNING	sx_api_port_pfc_enable_set hal port [int] returned [str]	ASIC priority flow control configuration failed.	File a ticket with Cumulus Support.
WARNING	switch priority [int] is not supported for MLX unit	Internal switch priority not supported.	Check QoS configuration file.
WARNING	sx_api_cos_redecn_general_param_get returned [str]	ASIC ECN configuration failed.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
WARNING	hal_mlx_ecn_red_set HAL port [int] min_threshold_bytes [int] is less than minimum size, using [int] bytes	invalid parameter.	Check QoS configuration file.
WARNING	hal_mlx_ecn_red_set HAL port [int] max_threshold_bytes [int] is greater than maximum size, using [int] bytes	invalid parameter.	Check QoS configuration file.
WARNING	sx_api_cos_redecn_profile_set returned [str]	ASIC ECN configuration failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_redecn_port_enable_set returned [str]	ASIC ECN configuration failed.	File a ticket with Cumulus Support.
WARNING	sx_api_cos_redecn_profile_to_tc_bind_set for hal port [int] flow type [int] returned [str]	ASIC ECN configuration failed.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
WARNING	hal_sh_datapath_file_read: egress port MC buffer percent [perc] reduced to 100.0	Invalid egress port MC buffer value.	Check back end QoS configuration file.
WARNING	priority group PG ID [int] is larger than the PG ID mask size [int]	Invalid priority group ID value configured.	Check back end QoS configuration file.
WARNING	No priority group ID found for lossless traffic	No priority group ID found for lossless traffic.	File a ticket with Cumulus Support.
WARNING	_queue_info_set: port_q_count_get failed for hal port [int]	Could not find port queue limits.	File a ticket with Cumulus Support.
WARNING	hal_sh_datapath_packet_buffer_set: [str]	Packet buffer config failed.	Check for detailed log messages.
WARNING	unable to set FEC parameters	Invalid operation for the current	Disable auto-negotiation on the port.

Severity	Message Text	Explanation	Recommended Action
	while autoneg is enabled	port configuration.	
WARNING	ethtool settings nwords too large: [int]	Invalid parameter: using default.	File a ticket with Cumulus Support.
WARNING	_port_group_priority_ma arg is NULL	Invalid parameter.	File a ticket with Cumulus Support.
WARNING	_port_group_config_g _port_group_find failed on [str] [str]	Port group not found or created.	File a ticket with Cumulus Support.
WARNING	_port_group_set_get: [str] port set not found	Port group port set not found.	Check QoS configuration file.
WARNING	_port_pause_config: config_port_pause failed: [str]	ASIC port pause configuration failed.	File a ticket with Cumulus Support.
WARNING	_priority_flow_contro hal_port_pfc_set failed on hal port [int]: [str]	ASIC priority flow control configuration failed.	File a ticket with Cumulus Support.
WARNING	_config_port_packet_bu [str]	ASIC packet buffer config	File a ticket with Cumulus

Severity	Message Text	Explanation	Recommended Action
		failed.	Support.
WARNING	_switch_priority_config: hal port [int]: [str]	switch_priority_config hal port [int]: [str]	File a ticket with Cumulus Support.
WARNING	_config_port_packet_buffers: [str]	ASIC packet buffer config failed.	File a ticket with Cumulus Support.
WARNING	_priority_map_config: priority map direction [int] is larger then max value HAL_DATAPATH_PRIORITY_DIRECTION_MAX	invalid parameter.	File a ticket with Cumulus Support.
WARNING	_priority_map_config: packet priority field [int] not supported	invalid packet priority field(s).	Check QoS configuration file.
WARNING	_hash_config: route_ecmp_max_paths_set failed: [str]	ASIC ECMP configuration failed.	File a ticket with Cumulus Support.
WARNING	_hash_config: hash config failed: [str]	ASIC symmetric hash configuration failed.	File a ticket with Cumulus Support.
WARNING	_hash_config:	ASIC ECMP	File a ticket

Severity	Message Text	Explanation	Recommended Action
	ecmp hash seed config faild: [str]	hash configuration failed.	with Cumulus Support.
WARNING	_hash_config: hash config failed: [str]	ASIC resilient hash configuration failed.	File a ticket with Cumulus Support.
WARNING	_mpls_config: mpls enable config failed: [str]	ASIC MPLS configuration failed.	File a ticket with Cumulus Support.
WARNING	_ecn_red_config: hal_datapath_ecn_red_ failed on hal port [int]: [str]	ASIC ECN/RED set configuration failed.	File a ticket with Cumulus Support.
WARNING	_port_attribute_mark flow control configuration conflict on hal port [int]:	Flow control configuration conflict.	Check QoS configuration file.
WARNING	hal_datapath_forwarding_ forwarding table profile path was NULL	Memory profile_get: allocation failed.	File a ticket with Cumulus Support.
WARNING	hal_datapath_forwarding_ sfs_config_get	Missing profile_get: forwarding	Check QoS configuration

Severity	Message Text	Explanation	Recommended Action
	failed for [str]	table profile configuration.	file.
WARNING	hal_datapath_init: packet priority source mapping configuration failed	Packet priority source map configuration failed.	Check for detailed log messages.
WARNING	hal_datapath_init: packet priority remark configuration failed	Packet priority remark map configuration failed.	Check for detailed log messages.
WARNING	_config_value_read: sfs path is null	Invalid parameter.	File a ticket with Cumulus Support.
WARNING	_config_value_read: sfs_config_get [str] failed	Configuration parameter not found.	File a ticket with Cumulus Support.
WARNING	_config_value_read: sfs_config_get [str] returned NULL configuration	Configuration value not found.	File a ticket with Cumulus Support.
WARNING	_cos_show_node_create: sfs_add failed	switchd fuse node create	File a ticket with Cumulus

Severity	Message Text	Explanation	Recommended Action
	for CoS node	failed.	Support.
WARNING	_priority_map_get: remark list has more than one packet priority value: configuring the first value	Ignoring surplus remark values.	Check QoS configuration file.
WARNING	_priority_map_list_get: [str]	ASIC priority profile create failed.	File a ticket with Cumulus Support.
WARNING	_switch_priority_config_ scheduling algorithm [str] not recognized	Invalid values_get: scheduling algorithm.	Check QoS configuration file.
WARNING	hal_list_get: list type [int] is not supported	Invalid list type.	File a ticket with Cumulus Support.
WARNING	Couldn't read a random number [int] setting seed to [uint]	No ECMP hash seed found in file.	Check file or accept default random seed.
WARNING	Couldn't read a random	No ECMP hash seed1	Check file or accept

Severity	Message Text	Explanation	Recommended Action
	number [int] setting seed1 to [uint]	found in file.	default random seed.
WARNING	Neighbor entry is not IPv4 or v6: [int]!	Netlink neighbor object has invalid family.	File a ticket with Cumulus Support.
WARNING	Neighbor entry had unexpected flags [int]	Netlink neighbor object has unsupported flags.	File a ticket with Cumulus Support.
WARNING	[str]: route table mode [int] not supported	Invalid route table ASIC mode.	File a ticket with Cumulus Support.
WARNING	[str]: host table mode [int] not supported	Invalid neighbor table ASIC mode.	File a ticket with Cumulus Support.
WARNING	Route [[str]] is not IPv4 or v6 or MPLS, family: [int]	Netlink route object has invalid family.	File a ticket with Cumulus Support.
WARNING	Route [[str]] has unexpected flags: [int]	Netlink route object has unsupported flags.	File a ticket with Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
WARNING	Route [[str]] has unexpected type: [int]	Netlink route object has unsupported type.	File a ticket with Cumulus Support.
WARNING	Route [[str]] has unexpected tos: [int]	Netlink route object has unsupported TOS value.	File a ticket with Cumulus Support.
WARNING	Route [[str]] has non-NULL src.	Netlink route object has non-NULL source.	File a ticket with Cumulus Support.
WARNING	Route [[str]] has non-zero iif: [int]	Netlink route object has non-NULL interface.	File a ticket with Cumulus Support.
WARNING	[int] routes exceeded [int] ecmp NHs, and were truncated.	Routes exceeded per-route nexthop limit.	Modify route next-hop configuration.
WARNING	[int] routes reverted to non-ECMP due to NH table	Total number of nexthops did not fit in hardware table.	Modify route next-hop configuration.
WARNING	[str]: route	Hardware	File a ticket

Severity	Message Text	Explanation	Recommended Action
	table mode [int] not supported	route table is set to an incorrect mode.	with Cumulus Support.
WARNING	[str]: host table mode [int] not supported	Hardware host table is set to an incorrect mode.	File a ticket with Cumulus Support.
WARNING	vpn_id 0x%x for ln_type [uint] ln_key [uint] tunnel_id 0x%x invalid local_ip [str]	Local IP for a tunnel is invalid.	File a ticket with Cumulus Support.
WARNING	Error removing isolated port [int] from [int]. Error: [str]	Removing isolated VPN port from the SDK failed.	File a ticket with Cumulus Support.
WARNING	Error adding isolated port [int] to [int]. Error: [str]	Adding isolated VPN port to the SDK failed.	File a ticket with Cumulus Support.
WARNING	Error removing	Moving isolated VPN	File a ticket with Cumulus

Severity	Message Text	Explanation	Recommended Action
	isolated port [int] from [int].	port from the SDK failed.	Support.
WARNING	[str]: failed to push port settings to hal. err = [int]	table_id could not be set for a port.	File a ticket with Cumulus Support.
WARNING	[str] not found in grp [str], bridge [int]	A port not found in the given group for the specific bridge.	File a ticket with Cumulus Support.
WARNING	grp [str] not found in bridge [int]	During deletion, an MDB group was not found for a specific bridge.	File a ticket with Cumulus Support.
WARNING	lid 0x%x cannot be both SPAN source	ACL: SPAN source and target cannot be the same.	Remove the rule.
WARNING	CPU not supported as mirror port	ACL: Mirror target port cannot be CPU.	Remove these SPAN rules.

Severity	Message Text	Explanation	Recommended Action
WARNING	table [str] [str] chain [str] L2 header field match not supported with IPv6 key	ACL: Specified match unsupported.	Remove the rule.
WARNING	table [str] [str] chain [str] IP TTL not supported with MAC+IPv4 key	ACL: Unsupported match.	Remove the rule.
WARNING	table [str] [str] chain [str] requires hardware IPv6 rule format but platform does not support MAC+IPv6 key combination	ACL: Unsupported match.	Remove the rule.
WARNING	table [str] [str] chain [str] requires hardware	ACL: Unsupported match.	Remove the rule.

Severity	Message Text	Explanation	Recommended Action
	IPv4 rule but platform does not support IPv4 key with		
WARNING	logical network type not supported	ACL: Unsupported interface type in internal VXLAN rules.	File a ticket with Cumulus Support.
WARNING	Detected excessive moves of mac address [str] on bridge [str], last seen on [str] and [str].	L2: Too many MAC moves seen.	Check network topology for loops or intrusion.
WARNING	Memory allocation failed	Memory exhausted.	File a ticket with Cumulus Support.
WARNING	Can’t open configuration file [str]: [int]	Failed to read configuration file in SFS.	File a ticket with Cumulus Support.
WARNING	tx failed with count [int], start %p	Failed to transfer packets in NIC.	File a ticket with Cumulus Support.
WARNING	Detected	Moved MAC	

Severity	Message Text	Explanation	Recommended Action
	excessive moves of mac address [str] on bridge [str],	addresses over threshold.	
WARNING	Unsupported command [str]	Wrong FEC command.	File a ticket with Cumulus Support.
WARNING	genl_talk returned error for ifindex [int] ([str])	Failed to read cached settings in PORT.	File a ticket with Cumulus Support.
WARNING	new tag_state [str] mismatches with [str] for [str] int_vlan [uint]	The new configured tag state [str] mismatches with the old tag state [str] for the internal VLAN [id].	File a ticket and contact Cumulus Support.
WARNING	verbosity level for SDK module [uint] not present	incorrect verbosity level for the Mellanox SDK module is being configured.	File a ticket and contact Cumulus Support.

Severity	Message Text	Explanation	Recommended Action
		This is an internal error.	
WARNING	legacy SX2 nexthop route type [uint] not handled.	for Legacy Mellanox SX2 chip, next hop of route type [uint] not handled.	File a ticket and contact Cumulus Support.
WARNING	hash_table_delete of clone parent from id_ht %p failed.	Failed to delete an ECMP clone id in, clone id hash table, as a entry does not exists.	File a ticket and contact Cumulus Support.
WARNING	[str]: no parent for [str]	Missing parent interface.	File a ticket with Cumulus Support.
WARNING	[str]: no parent for [str]	Missing parent interface.	File a ticket with Cumulus Support.

FRRouting Log Message Reference

The following table lists the HIGH severity ERROR log messages generated by FRRouting. These messages appear in `/var/log/frr/frr.log`.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
Babel	HIGH	16777217	BABEL Memory Errors	Babel has failed to allocate memory. The system is about to run out of memory.	Find the process that is causing memory shortages and remediate that process. Restart FRR.
Babel	HIGH	16777218	BABEL Packet Error	Babel has detected a packet encode/ decode problem.	Collect the relevant log files and report the issue for troubleshooting.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
Babel	HIGH	16777219	BABEL Configuration Error	Babel has detected a configuration error of some sort.	Ensure that the configuration is correct.
Babel	HIGH	16777220	BABEL Route Error	Babel has detected a routing error and is in an inconsistent state.	Gather data to report the issue for troubleshooting. Restart FRR.
BGP	HIGH	33554433	BGP attribute flag is incorrect	BGP attribute flag is set to the wrong value (Optional/ Transitive/ Partial).	Determine the soure of the attribute and determine why the attribute flag has been set

Category	Severity	Message #	Message Text	Explanation	Recommended Action
					incorrectly.
BGP	HIGH	33554434	BGP attribute length is incorrect	BGP attribute length is incorrect.	Determine the soure of the attribute and determine why the attribute length has been set incorrectly.
BGP	HIGH	33554435	BGP attribute origin value invalid	BGP attribute origin value is invalid.	Determine the soure of the attribute and determine why the origin attribute has been set incorrectly.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
BGP	HIGH	33554436	BGP as path is invalid	BGP AS path has been malformed.	Determine the soure of the update and determine why the AS path has been set incorrectly.
BGP	HIGH	33554437	BGP as path first as is invalid	BGP update has invalid first AS in AS path.	Determine the soure of the update and determine why the AS path first AS value has been set incorrectly.
BGP	HIGH	33554439	BGP	BGP	Determine

Category	Severity	Message #	Message Text	Explanation	Recommended Action
			PMSI tunnel attribute type is invalid	update has invalid type for PMSI tunnel.	the soure of the update and determine why the PMSI tunnel attribute type has been set incorrectly.
BGP	HIGH	33554440	BGP PMSI tunnel attribute length is invalid	BGP update has invalid length for PMSI tunnel.	Determine the soure of the update and determine why the PMSI tunnel attribute length has been set

Category	Severity	Message #	Message Text	Explanation	Recommended Action
					incorrectly.
BGP	HIGH	33554442	BGP peergroup operated on in error	BGP operating on peer-group instead of peers included.	Ensure the configuration doesn't contain peer-groups contained within peer-groups.
BGP	HIGH	33554443	BGP failed to delete peer structure	BGP was unable to delete the peer structure when the address-family was removed.	Determine if all expected peers are removed and restart FRR if not. This is most likely a bug.
BGP	HIGH	33554444	BGP failed to get table	BGP unable to get chunk	Ensure there is adequate memory

Category	Severity	Message #	Message Text	Explanation	Recommended Action
			chunk memory	memory for table manager.	on the device to support the table requirements.
BGP	HIGH	33554445	BGP received MACIP with invalid IP addr len	BGP received MACIP with invalid IP address length from Zebra.	Verify the MACIP entries inserted in Zebra are correct. This is most likely a bug.
BGP	HIGH	33554446	BGP received invalid label manager message	BGP received an invalid label manager message from the label manager.	Label manager sent an invalid message to BGP for the wrong protocol instance. This is

Category	Severity	Message #	Message Text	Explanation	Recommended Action
					most likely a bug.
BGP	HIGH	33554447	BGP unable to allocate memory for JSON output	BGP attempted to generate JSON output and was unable to allocate the memory required.	Ensure that the device has adequate memory to support the required functions.
BGP	HIGH	33554448	BGP update had attributes too long to send	BGP attempted to send an update but the attributes were too long to fit.	This is most likely a bug. If the problem persists, report it for troubleshooting.
BGP	HIGH	33554449	BGP update	BGP attempted	This is most

Category	Severity	Message #	Message Text	Explanation	Recommended Action
			group creation failed	to create an update group but was unable to do so.	likely a bug. If the problem persists, report it for troubleshooting.
BGP	HIGH	33554450	BGP error creating update packet	BGP attempted to create an update packet but was unable to do so.	This is most likely a bug. If the problem persists, report it for troubleshooting.
BGP	HIGH	33554451	BGP error receiving open packet	BGP received an open from a peer that was invalid.	Determine the sending peer and correct its invalid open

Category	Severity	Message #	Message Text	Explanation	Recommended Action
					packet.
BGP	HIGH	33554452	BGP error sending to peer	BGP attempted to respond to open from a peer and failed.	BGP attempted to respond to an open and could not send the packet. Check the local IP address for the source.
BGP	HIGH	33554453	BGP error receiving from peer	BGP received an update from a peer but the status was incorrect.	This is most likely a bug. If the problem persists, report it for troubleshooting.
BGP	HIGH	33554454	BGP	BGP	Determine

Category	Severity	Message #	Message Text	Explanation	Recommended Action
			error receiving update packet	received an invalid update packet.	the source of the update and resolve the invalid update being sent.
BGP	HIGH	33554455	BGP error due to capability not enabled	BGP attempted a function that did not have the capability enabled.	Enable the capability if this functionality is desired.
BGP	HIGH	33554456	BGP error receiving notify message	BGP unable to process the notification message.	BGP notify received while in a stopped state. If the problem persists,

Category	Severity	Message #	Message Text	Explanation	Recommended Action
					report it for troubleshooting.
BGP	HIGH	33554457	BGP error receiving keepalive packet	BGP unable to process a keepalive packet.	BGP keepalive received while in a stopped state. If the problem persists, report it for troubleshooting.
BGP	HIGH	33554458	BGP error receiving route refresh message	BGP unable to process route refresh message.	BGP route refresh received while in a stopped state. If the problem persists, report it for troubleshooting.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
BGP	HIGH	33554459	BGP error capability message	BGP unable to process received capability.	BGP capability message received while in a stopped state. If the problem persists, report it for troubleshooting.
BGP	HIGH	33554460	BGP error with nexthopo update	BGP unable to process nexthop update.	BGP received the nexthop update but the nexthop is not reachable in this BGP instance. Report the problem for troubleshooting.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
BGP	HIGH	33554461	Failure to apply label	BGP attempted to attempted to apply a label but could not do so.	This is most likely a bug. If the problem persists, report it for troubleshooting.
BGP	HIGH	33554462	Multipath specified is invalid	BGP was started with an invalid ECMP/ multipath value.	Correct the ECMP/ multipath value supplied when starting the BGP daemon.
BGP	HIGH	33554463	Failure to process a packet	BGP attempted to process a received packet but could	This is most likely a bug. If the problem persists, report it for

Category	Severity	Message #	Message Text	Explanation	Recommended Action
				not do so.	troubleshooting.
BGP	HIGH	33554464	Failure to connect to peer	BGP attempted to send open to a peer but couldn't connect.	This is most likely a bug. If the problem persists, report it for troubleshooting.
BGP	HIGH	33554465	BGP FSM issue	BGP neighbor transition problem.	This is most likely a bug. If the problem persists, report it for troubleshooting.
BGP	HIGH	33554466	BGP VNI creation issue	BGP could not create a new VNI.	This is most likely a bug. If the problem persists, report

Category	Severity	Message #	Message Text	Explanation	Recommended Action
					it for troubleshooting.
BGP	HIGH	33554467	BGP default instance missing	BGP could not find default instance.	Define a default instance of BGP since some feature requires its existence.
BGP	HIGH	33554468	BGP remote VTEP invalid	BGP remote VTEP is invalid and cannot be used.	Correct the remote VTEP configuration or resolve the source of the problem.
BGP	HIGH	33554469	BGP ES route error	BGP ES route incorrect as it learned both	Correct the configuration or address it so

Category	Severity	Message #	Message Text	Explanation	Recommended Action
				local and remote routes.	that same route is not learned both local and remote.
BGP	HIGH	33554470	BGP EVPN route delete error	BGP attempted to delete an EVPN route and failed.	This is most likely a bug. If the problem persists, report it for troubleshooting.
BGP	HIGH	33554471	BGP EVPN install/ uninstall error	BGP attempted to install or uninstall an EVPN prefix and failed.	This is most likely a bug. If the problem persists, report it for troubleshooting.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
BGP	HIGH	33554472	BGP EVPN route received with invalid contents	BGP received an EVPN route with invalid contents.	Determine the source of the EVPN route and resolve whatever is causing the invalid content.
BGP	HIGH	33554473	BGP EVPN route create error	BGP attempted to create an EVPN route and failed.	This is most likely a bug. If the problem persists, report it for troubleshooting.
BGP	HIGH	33554474	BGP EVPN ES entry create error	BGP attempted to create an EVPN ES	This is most likely a bug. If the problem persists,

Category	Severity	Message #	Message Text	Explanation	Recommended Action
				entry and failed.	report it for troubleshooting.
BGP	HIGH	33554475	BGP config multi-instance issue	BGP configuration attempting multiple instances without enabling the feature.	Correct the configuration so that BGP multiple-instance is enabled if desired.
BGP	HIGH	33554476	BGP AS configuration issue	BGP configuration attempted for a different AS than is currently configured.	Correct the configuration so that the correct BGP AS number is used.
BGP	HIGH	33554477	BGP EVPN AS and process name mismatch	BGP configuration has AS and process name mismatch.	Correct the configuration so that the BGP AS number

Category	Severity	Message #	Message Text	Explanation	Recommended Action
					and instance name are consistent.
BGP	HIGH	33554478	BGP Flowspec packet processing error	The BGP flowspec subsystem has detected an error in the sending or receiving of a packet.	Gather log files from both sides of the peering relationship and report the issue for troubleshooting.
BGP	HIGH	33554479	BGP Flowspec Installation/ removal Error	The BGP flowspec subsystem has detected that there was a failure for installation/	Gather log files from the router and report the issue for troubleshooting. Restart

Category	Severity	Message #	Message Text	Explanation	Recommended Action
				removal/ modification of Flowspec from the dataplane.	FRR.
EIGRP	HIGH	50331649	EIGRP Packet Error	EIGRP has a packet that does not correctly decode or encode.	Gather log files from both sides of the neighbor relationship and report the issue for troubleshooting.
EIGRP	HIGH	50331650	EIGRP Configuration Error	EIGRP has detected a configuration error.	Correct the configuration issue. If it still persists, report the issue for

Category	Severity	Message #	Message Text	Explanation	Recommended Action
					troubleshooting.
General	HIGH	100663297	Failure to raise or lower privileges	FRR attempted to raise or lower its privileges and was unable to do so.	Ensure that you are running FRR as the frr user and that the user has sufficient privileges to properly access root privileges.
General	HIGH	100663298	VRF Failure on Start	Upon startup, FRR failed to properly initialize and start up the VRF	Ensure that there is sufficient memory to start processes, then restart FRR.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
				subsystem.	
General	HIGH	100663299	Socket Error	When attempting to access a socket, a system error occured and FRR was unable to properly complete the request.	Ensure that there are sufficient system resources available and ensure that the frr user has sufficient permisions to work.
General	HIGH	100663303	System Call Error	FRR has detected an error from using a vital system call and has	Ensure permissions are correct for FRR users and groups. Additionally, check that

Category	Severity	Message #	Message Text	Explanation	Recommended Action
				probably already exited.	sufficient system resources are available.
General	HIGH	100663304	VTY Subsystem Error	FRR has detected a problem with the specified configuration file.	Ensure the configuration file exists and has the correct permissions for operations. Additionally, ensure that all config lines are correct as well.
General	HIGH	100663305	SNMP Subsystem Error	FRR has detected a problem with the	Examine the callback message and ensure SNMP

Category	Severity	Message #	Message Text	Explanation	Recommended Action
				SNMP library it uses. A callback from this subsystem has indicated some error.	is properly set up and working.
General	HIGH	100663306	Interface Subsystem Error	FRR has detected a problem with interface data from the kernel as it deviates from what we would expect to happen	Open an issue with all relevant log files and restart FRR.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
				via normal netlink messaging.	
General	HIGH	100663307	NameSpace Subsystem Error	FRR has detected a problem with namespace data from the kernel as it deviates from what we would expect to happen via normal kernel messaging.	Open an issue with all relevant log files and restart FRR.
General	HIGH	4043309068	A necessary work	A necessary work	Notify a developer.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
			queue does not exist.	queue does not exist.	
General	HIGH	100663308	Developmental Escape Error	FRR has detected an issue where new development has not properly updated all code paths.	Open an issue with all relevant log files.
General	HIGH	100663309	ZMQ Subsystem Error	FRR has detected an issue with the ZeroMQ subsystem and ZeroMQ is not working properly	Open an issue with all relevant log files and restart FRR.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
				now.	
General	HIGH	100663310	Feature or system unavailable	FRR was not compiled with support for a particular feature or it is not available on the current platform.	Recompile FRR with the feature enabled or find out what platforms support the feature.
General	HIGH	4043309071	IRDP message length mismatch	The length encoded in the IP TLV does not match the length of the packet received.	Notify a developer.
General	HIGH	4043309073	Dataplane	Installation	Check

Category	Severity	Message #	Message Text	Explanation	Recommended Action
			installation failure	of routes to the underlying dataplane failed.	all configuration parameters for correctness.
General	HIGH	4043309075	Netlink backend not available	FRR was not compiled with support for Netlink. Any operations that require Netlink will fail.	Recompile FRR with Netlink or install a package that supports this feature.
General	HIGH	4043309076	Protocol Buffers backend not available	FRR was not compiled with support for protocol buffers. Any operations	Recompile FRR with protobuf support or install a package that supports this

Category	Severity	Message #	Message Text	Explanation	Recommended Action
				that require protobuf will fail.	feature.
General	HIGH	4043309087	Cannot set receive buffer size	The socket receive buffer size could not be set in the kernel.	Ignore this error.
General	HIGH	4043309089	Receive buffer overrun	The kernel's buffer for a socket has been overrun, rendering the socket invalid.	Zebra will restart itself. Notify a developer if this issue shows up frequently.
General	HIGH	4043309091	Received unexpected response from	Received unexpected response from	Notify a developer.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
			kernel	the kernel via Netlink.	
General	HIGH	4043309094	String could not be parsed as IP prefix	There was an attempt to parse a string as an IPv4 or IPv6 prefix, but the string could not be parsed and this operation failed.	Notify a developer.
General	HIGH	268435457	WATCHFRR Connection Error	WATCHFRR has detected a connectivity issue with one of	Ensure that FRR is still running. If it isn't, report

Category	Severity	Message #	Message Text	Explanation	Recommended Action
				the FRR daemons.	the issue for troubleshooting.
ISIS	HIGH	67108865	ISIS Packet Error	ISIS has detected an error with a packet from a peer.	Gather log information and report the issue for troubleshooting. Restart FRR.
ISIS	HIGH	67108866	ISIS Configuration Error	ISIS has detected an error within the configuration for the router.	Ensure configuration is correct.
OSPF	HIGH	134217729	Failure to process a packet	OSPF attempted to process a received	This is most likely a bug. If the problem

Category	Severity	Message #	Message Text	Explanation	Recommended Action
				packet but could not do so.	persists, report it for troubleshooting.
OSPF	HIGH	134217730	Failure to process Router LSA	OSPF attempted to process a router LSA, but there was an advertising ID mismtach with the link ID.	Check the OSPF network configuration for any configuration issue. If the problem persists, report it for troubleshooting.
OSPF	HIGH	134217731	OSPF Domain Corruption	OSPF attempted to process a router LSA, but there was an	Check OSPF network database for a corrupted LSA. If the problem persists,

Category	Severity	Message #	Message Text	Explanation	Recommended Action
				advertising ID mismtach with the link ID.	shut down the OSPF domain and report the problem for troubleshooting.
OSPF	HIGH	134217732	OSPF Initialization failure	OSPF failed to initialized the OSPF default instance.	Ensure there is adequate memory on the device. If the problem persists, report it for troubleshooting.
OSPF	HIGH	134217733	OSPF SR Invalid DB	OSPF segment routing database is invalid.	This is most likely a bug. If the problem persists, report

Category	Severity	Message #	Message Text	Explanation	Recommended Action
					it for troubleshooting.
OSPF	HIGH	134217734	OSPF SR hash node creation failed	OSPF segment routing node creation failed.	This is most likely a bug. If the problem persists, report it for troubleshooting.
OSPF	HIGH	134217735	OSPF SR Invalid lsa id	OSPF segment routing invalid LSA ID.	Restart the OSPF instance. If the problem persists, report it for troubleshooting.
OSPF	HIGH	134217736	OSPF SR Invalid Algorithm	OSPF segment routing invalid algorithm.	This is most likely a bug. If the problem persists, report

Category	Severity	Message #	Message Text	Explanation	Recommended Action
					it for troubleshooting.
PIM	HIGH	184549377	PIM MSDP Packet Error	PIM has received a packet from a peer that does not correctly decode.	Check the MSDP peer and ensure it is correctly working.
PIM	HIGH	184549378	PIM Configuration Error	PIM has detected a configuration error.	Ensure the configuration is correct and apply the correct configuration.
RIP	HIGH	201326593	RIP Packet Error	RIP has detected a packet encode/ decode issue.	Gather log files from both sides and open a

Category	Severity	Message #	Message Text	Explanation	Recommended Action
					Issue
Zebra	HIGH	4043309057	Error reading response from label manager	Zebra could not read the ZAPI header from the label manager.	Wait for the error to resolve on its own. If it does not resolve, restart Zebra.
Zebra	HIGH	4043309058	Label manager could not find ZAPI client	Zebra was unable to find a ZAPI client matching the given protocol and instance number.	Ensure that clients that use the label manager are properly configured and running.
Zebra	HIGH	4043309059	Zebra could not relay label	Zebra found the client and	Ensure that clients that use the

Category	Severity	Message #	Message Text	Explanation	Recommended Action
			manager response	instance to relay the label manager response or request, but was unable to do so, possibly because the connection was closed.	label manager are properly configured and running.
Zebra	HIGH	100663300	ZAPI Error	A version mismatch has been detected between Zebra and a client protocol.	Two different versions of FRR have been installed and the install is not properly set up. Completely

Category	Severity	Message #	Message Text	Explanation	Recommended Action
					stop FRR, remove it from the system and reinstall. Typically, only developers should see this issue.
Zebra	HIGH	4043309061	Mismatch between ZAPI instance and encoded message instance	While relaying a request to the external label manager, Zebra noticed that the instance number encoded in the message did not	Notify a developer.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
				match the client instance number.	
Zebra	HIGH	100663301	ZAPI Error	The ZAPI subsystem has detected an encoding issue between Zebra and a client protocol.	Restart FRR.
Zebra	HIGH	100663302	ZAPI Error	The ZAPI subsystem has detected a socket error between Zebra and a client.	Restart FRR.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
Zebra	HIGH	4043309064	Zebra label manager used all available labels	Zebra is unable to assign additional label chunks because it has exhausted its assigned label range.	Make the label range bigger and restart Zebra.
Zebra	HIGH	4043309065	Daemon mismatch when releasing label chunks	Zebra noticed a mismatch between a label chunk and a protocol daemon number or instance when releasing unused label	Ignore this error.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
				chunks.	
Zebra	HIGH	4043309066	Zebra did not free any label chunks	Zebra's chunk cleanup procedure ran but no label chunks were released.	Ignore this error.
Zebra	HIGH	4043309067	Dataplane returned invalid status code	The underlying dataplane responded to a Zebra message or other interaction with an unrecognized unknown or invalid status code.	Notify a developer.
Zebra	HIGH	4043309069	Failed to add	A client requested	Notify a

Category	Severity	Message #	Message Text	Explanation	Recommended Action
			FEC for MPLS client	a label binding for a new FEC but Zebra was unable to add the FEC to its internal table.	developer.
Zebra	HIGH	4043309070	Failed to remove FEC for MPLS client	Zebra was unable to find and remove an FEC in its internal table.	Notify a developer.
Zebra	HIGH	4043309072	Attempted to perform nexthop update for	Zebra attempted to perform a nexthop	Notify a developer.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
			unknown address family	update for unknown address family.	
Zebra	HIGH	4043309074	Zebra table lookup failed	Zebra attempted to look up a table for a particular address family and a subsequent address family but didn't find anything.	If you entered a command to trigger this error, make sure you entered the arguments correctly. Check your configuration file for any potential errors. If these look correct, notify a developer.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
Zebra	HIGH	4043309077	Table manager used all available IDs	Zebra's table manager used up all IDs available to it and can't assign any more.	Reconfigure Zebra with a larger range of table IDs.
Zebra	HIGH	4043309078	Daemon mismatch when releasing table chunks	Zebra noticed a mismatch between a table ID chunk and a protocol daemon number instance when releasing unused table chunks.	Ignore this error.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
Zebra	HIGH	4043309079	Zebra did not free any table chunks	Zebra's table chunk cleanup procedure ran but no table chunks were released.	Ignore this error.
Zebra	HIGH	4043309080	Address family specifier unrecognized	Zebra attempted to process information from somewhere that included an address family specifier but did not recognize the provided specifier.	Ensure that your configuration is correct. If it is, notify a developer.
Zebra	HIGH	4043309081	Incorrect	Zebra's	Notify

Category	Severity	Message #	Message Text	Explanation	Recommended Action
			protocol for table manager client	table manager only accepts connections from daemons managing dynamic routing protocols, but received a connection attempt from a daemon that does not meet this criterion.	a developer.
Zebra	HIGH	4043309082	Mismatch between message and client protocol and/or instance	Zebra detected a mismatch between a client's protocol	Notify a developer.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
				and/or instance numbers versus those stored in a message transiting its socket.	
Zebra	HIGH	4043309083	Label manager unable to assign label chunk	Zebra's label manager was unable to assign a label chunk to client.	Ensure that Zebra has a sufficient label range available and that there is not a range collision.
Zebra	HIGH	4043309084	Label request from unidentified client	Zebra's label manager received a label request	Notify a developer.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
				from an unidentified client.	
Zebra	HIGH	4043309085	Table manager unable to assign table chunk	Zebra's table manager was unable to assign a table chunk to a client.	Ensure that Zebra has sufficient table ID range available and that there is not a range collision.
Zebra	HIGH	4043309086	Table request from unidentified client	Zebra's table manager received a table request from an unidentified client.	Notify a developer.
Zebra	HIGH	4043309088	Unknown Netlink message type	Zebra received a Netlink	Verify that you are running

Category	Severity	Message #	Message Text	Explanation	Recommended Action
				message with an unrecognized type field.	the latest version of FRR to ensure kernel compatibility. If the problem persists, notify a developer.
Zebra	HIGH	4043309090	Netlink message length mismatch	Zebra received a Netlink message with incorrect length fields.	Notify a developer.
Zebra	HIGH	4043309092	Bad sequence number in Netlink message	Zebra received a Netlink message with a bad sequence number.	Notify a developer.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
Zebra	HIGH	4043309093	Multipath number was out of valid range	The multipath number specified to Zebra must be in the appropriate range.	Provide a multipath number that is within its accepted range.
Zebra	HIGH	4043309095	Failed to add MAC address to interface	Zebra attempted to assign a MAC address to a VXLAN interface but failed.	Notify a developer.
Zebra	HIGH	4043309096	Failed to delete VNI	Zebra attempted to delete a VNI entry and failed.	Notify a developer.

Category	Severity	Message #	Message Text	Explanation	Recommended Action
Zebra	HIGH	4043309097	Adding remote VTEP failed	Zebra attempted to add a remote VTEP and failed.	Notify a developer.
Zebra	HIGH	4043309098	Adding VNI failed	Zebra attempted to add a VNI hash to an interface and failed.	Notify a developer.

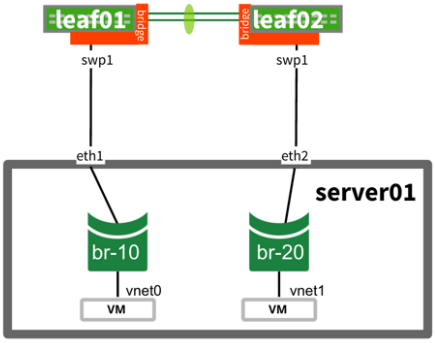
Network Solutions

This section discusses the various architectures and strategies available with Cumulus Linux, provides demos to help you build virtual simulations of production networks and validate configurations, and describes different solutions, such as the Cumulus Hyperconverged Solution (HCS) and RDMA over Converged Ethernet (RoCE).

Data Center Host to ToR Architecture

This chapter discusses the various architectures and strategies available from the top of rack (ToR) switches all the way down to the server hosts.

Layer 2 - Traditional Spanning Tree - Single Attached

Example	Summary
	Bond and Etherchannel are not configured on host to multiple switches (bonds can still occur but only to one switch at a time), so leaf01 and leaf02 see two different MAC addresses.

Benefits	Considerations
• Established technology: Interoperability with other vendors, easy configuration, a lot of documentation from multiple vendors and the industry	• The load balancing mechanism on the host can cause problems. If there is only host pinning to each NIC, there are no problems, but if you have a bond, you need to

Benefits	Considerations
• Ability to use spanning tree commands: PortAdminEdge and BPDU guard • Layer 2 reachability to all VMs	look at an MLAG solution. • No active-active host links. Some operating systems allow HA (NIC failover), but this still does not utilize all the bandwidth. VMs use one NIC, not two.

Active-Active Mode	Active-Passive Mode	L2 to L3 Demarcation
None (not possible with traditional spanning tree)	VRR	• ToR layer (recommended) • Spine layer • Core/edge/exit You can configure VRR on a pair of switches at any level in the network. However, the higher up the network, the larger the layer 2 domain becomes. The benefit is layer 2 reachability. The drawback is that the layer 2 domain is more difficult to troubleshoot, does

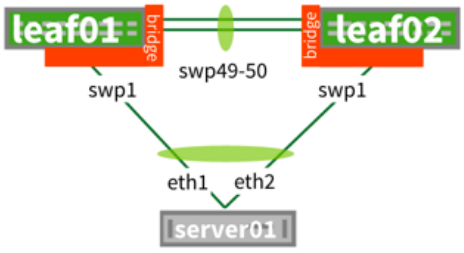
Active-Active Mode	Active-Passive Mode	L2 to L3 Demarcation
		not scale as well, and the pair of switches running VRR needs to carry the entire MAC address table of everything below it in the network. Cumulus Professional Services recommends minimizing the layer 2 domain as much as possible. For more information, see this presentation.

Example Configuration

leaf01 Ubuntu host

```
auto bridge
iface bridge
  bridge-vlan-aware yes
  bridge-ports swp1 peerlink
  bridge-vids 1-2000
  bridge-stp on

auto bridge.10
iface bridge.10
  address 10.1.10.2/24

auto peerlink
iface peerlink
    bond-slaves glob swp49-50

auto swp1
iface swp1
  mstpctl-portadminedge yes
  mstpctl-bpduguard yes
```

Layer 2 - MLAG

Example	Summary
	MLAG (multi-chassis link aggregation) uses both uplinks at the same time. VRR enables both spines to act as gateways simultaneously for HA (high availability) and active-active mode (both are used at the same time).

Benefits	Considerations
100% of links utilized	• More complicated (more moving parts) • More configuration • No interoperability between vendors • ISL (inter-switch link) required

Active-Active Mode	Active-Passive Mode	L2 to L3 Demarcation	More Information
VRR	None	• ToR layer (recommended) • Spine layer • Core/ edge/exit	• Can be done with either the traditional or VLAN-

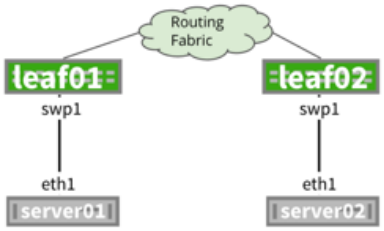
Active-Active Mode	Active-Passive Mode	L2 to L3 Demarcation	More Information
			aware bridge driver depending on overall STP needs. • There are a few different solutions including Cisco VPC and Arista MLAG, but none of them interoperate and are very vendor specific. • Cumulus Networks Layer 2 HA validated design guide.

Example Configuration

leaf01 Ubuntu host

```
auto bridge
iface bridge
  bridge-vlan-aware yes
  bridge-ports host-01 peerlink
  bridge-vids 1-2000
  bridge-stp on

auto bridge.10
iface bridge.10
  address 172.16.1.2/24
  address-virtual 44:38:39:00:00:10 172.16.1.1/24

auto peerlink
iface peerlink
    bond-slaves glob swp49-50

auto peerlink.4094
iface peerlink.4094
    address 169.254.1.2
    clagd-enable yes
    clagd-peer-ip 169.254.1.2
    clagd-system-mac 44:38:39:FF:40:94

auto host-01
iface host-01
  bond-slaves swp1
  clag-id 1
  {bond-defaults removed for brevity}
```

Layer 3 - Single-attached Hosts

Example	Summary
	The server (physical host) has only has one link to one ToR switch.

Benefits	Considerations
• Relatively simple network configuration • No STP • No MLAG • No layer 2 loops • No crosslink between leafs • Greater route scaling and flexibility	• No redundancy for ToR, upgrades can cause downtime. • There is often no software to support application layer redundancy.

FHR (First Hop Redundancy)	More Information
No redundancy for ToR, uses single ToR as gateway.	For additional bandwidth, links between host and leaf can be bonded.

Example Configuration

leaf01 leaf02 Ubuntu host1 Ubuntu host2

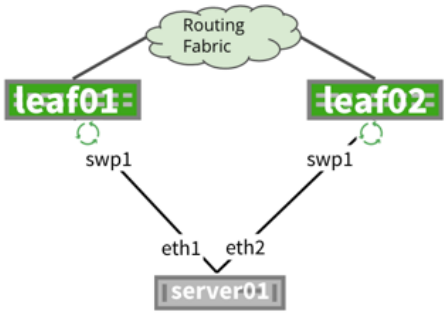
`/etc/network/interfaces` file

```
auto swp1
iface swp1
  address 172.16.1.1/30
```

`/etc/frr/frr.conf` file

```
router ospf
  router-id 10.0.0.11
interface swp1
  ip ospf area 0
```

Layer 3 - Redistribute Neighbor

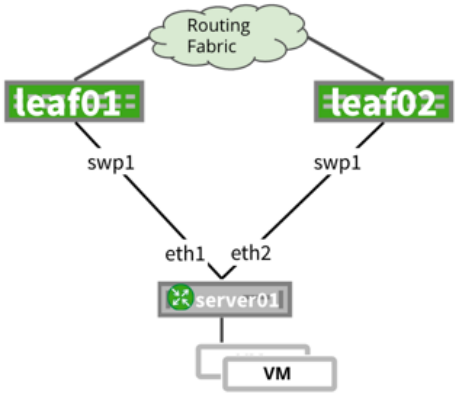
Example	Summary
	The Redistribute neighbor daemon grabs ARP entries dynamically and uses the redistribute table for FRRouting to take these dynamic entries and redistribute them into the fabric.

Benefits	Considerations
Configuration in FRRouting is simple (route map plus redistribute table)	• Silent hosts do not receive traffic (depending on ARP). • IPv4 only. • If two VMs are on the same layer 2 domain, they can learn about each other directly instead of using the gateway, which causes problems (such as VM migration or getting the network routed). Put hosts on /32 (no other layer 2 adjacency). • VM moves do not trigger a route withdrawal from the original leaf (four hour timeout). • Clearing ARP impacts routing.

Benefits	Considerations
	• No layer 2 adjacency between servers without VXLAN.

FHR (First Hop Redundancy)	More Information
• Equal cost route installed on server, host, or hypervisor to both ToRs to load balance evenly. • For host/VM/container mobility, use the same default route on all hosts (such as x.x.x.1) but do not distribute or advertise the .1 on the ToR into the fabric. This allows the VM to use the same gateway no matter to which pair of leafs it is cabled.	Cumulus Networks blog post introducing redistribute neighbor

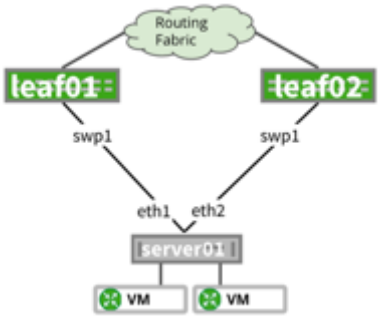
Layer 3 - Routing on the Host

Example	Summary
	Routing on the host means there is a routing application (such as FRRouting, either on the bare metal host (no VMs or containers) or the hypervisor (for example, Ubuntu with KVM). This is highly recommended by the Professional Services team.

Benefits	Considerations
• No requirement for MLAG • No spanning tree or layer 2 domain • No loops • You can use three or more ToRs instead of the usual two • Host and VM mobility • You can use traffic engineering to migrate traffic from one ToR to another when upgrading both hardware and software	• The hypervisor or host OS might not support a routing application like FRRouting and requires a virtual router on the hypervisor. • No layer 2 adjacency between servers without VXLAN.

FHR (First Hop Redundancy)	More Information
• The first hop is still the ToR, just like redistribute neighbor • A default route can be advertised by all leaf/ToRs for dynamic ECMP paths	• Installing the FRRouting Package on an Ubuntu Server • Configure FRRouting

Layer 3 - Routing on the VM

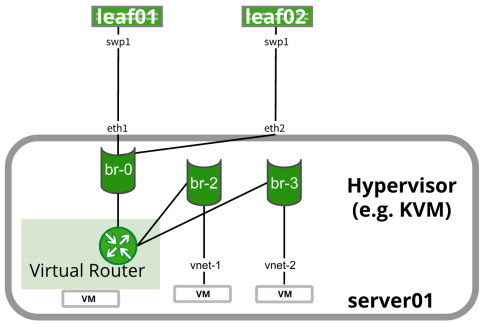
Example	Summary
	Instead of routing on the hypervisor, each virtual machine uses its own routing stack.

Benefits	Considerations
In addition to routing on host: • The hypervisor/base OS does not need to be able to do routing • VMs can be authenticated into routing fabric	• All VMs must be capable of routing. • You need to take scale considerations into an account; instead of one routing process, there are as

Benefits	Considerations
	many as there are VMs. • No layer 2 adjacency between servers without VXLAN.

FHR (First Hop Redundancy)	More Information
• The first hop is still the ToR, just like redistribute neighbor • You can use multiple ToRs (two or more)	• Installing the FRRouting Package on an Ubuntu Server • Configure FRRouting

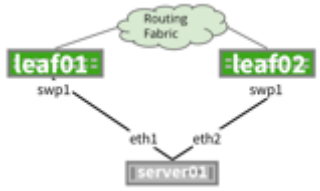
Layer 3 - Virtual Router

Example	Summary
	Virtual router (vRouter) runs as a VM on the hypervisor or host and sends routes to the ToR using BGP or OSPF.

Benefits	Considerations
In addition to routing on a host: • Multi-tenancy can work, where multiple customers share the same racks • The base OS does not need to be routing capable	• ECMP might not work correctly (load balancing to multiple ToRs); the Linux kernel in older versions is not capable of ECMP per flow (it does it per packet). • No layer 2 adjacency between servers without VXLAN.

FHR (First Hop Redundancy)	More Information
• The gateway is the vRouter, which has two routes out (two ToRs) • You can use multiple vRouters	• Installing the FRRouting Package on an Ubuntu Server • Configure FRRouting

Layer 3 - Anycast with Manual Redistribution

Example	Summary
	In contrast to routing on the host (preferred), this method allows you to route **to** the host. The ToRs are the gateway, as with redistribute neighbor, except because there is no daemon running, you must manually configure the networks

Example	Summary
	under the routing process. There is a potential to black hole unless you run a script to remove the routes when the host no longer responds.

Benefits	Considerations
• Most benefits of routing **on** the host • No requirement for host to run routing • No requirement for redistribute neighbor	• Removing a subnet from one ToR and re-adding it to another (network statements from your router process) is a manual process. • Network team and server team have to be in sync, or the server team controls the ToR, or automation is used used whenever VM migration occurs. • When using VMs or containers it is very easy to black hole traffic, as the leafs continue to advertise prefixes even when the VM is down. • No layer 2 adjacency between servers without VXLAN.

FHR (First Hop Redundancy)
The gateways are the ToRs, exactly like redistribute neighbor with an equal cost route installed.

Example Configuration

leaf01 leaf02 Ubuntu host

`/etc/network/interfaces file`

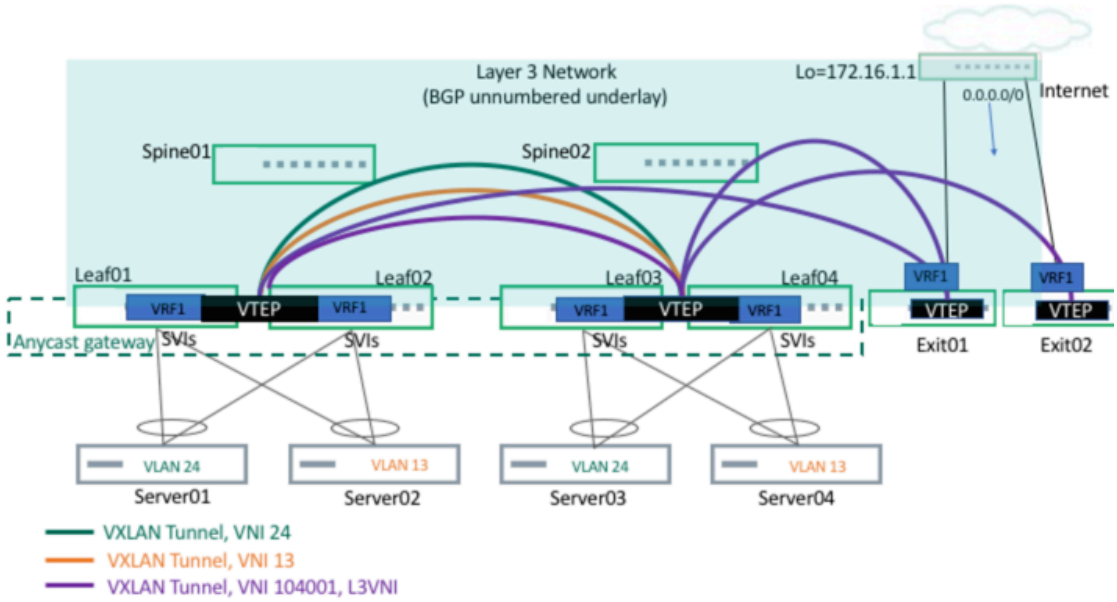
```
auto swp1
iface swp1
  address 172.16.1.1/30
```

`/etc/frr/frr.conf` file

```
router ospf
  router-id 10.0.0.11
interface swp1
  ip ospf area 0
```

Layer 3 - EVPN with Symmetric VXLAN Routing

Symmetric VXLAN routing is configured directly on the ToR, using EVPN for both VLAN and VXLAN bridging as well as VXLAN and external routing.



Each server is configured on a VLAN, with a total of two VLANs for the setup. MLAG is also set up between servers and the leafs. Each leaf is configured with an anycast gateway and the servers default gateways are pointing towards the corresponding leaf switch IP gateway address. Two tenant VNIs (corresponding to two VLANs/VXLANs) are bridged to corresponding VLANs.

Benefits	Considerations
• Layer 2 domain is reduced to the pair of ToRs • Aggregation layer is all layer 3 (VLANs do not have to exist on spine switches) • Greater route scaling and flexibility • High availability	Needs MLAG (with the same considerations as the MLAG section above).

Active-Active Mode	Active-Passive Mode	Demarcation	More Information
VRR	None	ToR layer	• Cumulus Networks EVPN with symmetric routing demo on GitHub • Ethernet Virtual Private Network - EVPN • VXLAN Routing

Example /etc/network/interfaces File Configuration

leaf01 Leaf02 Server01 Server02

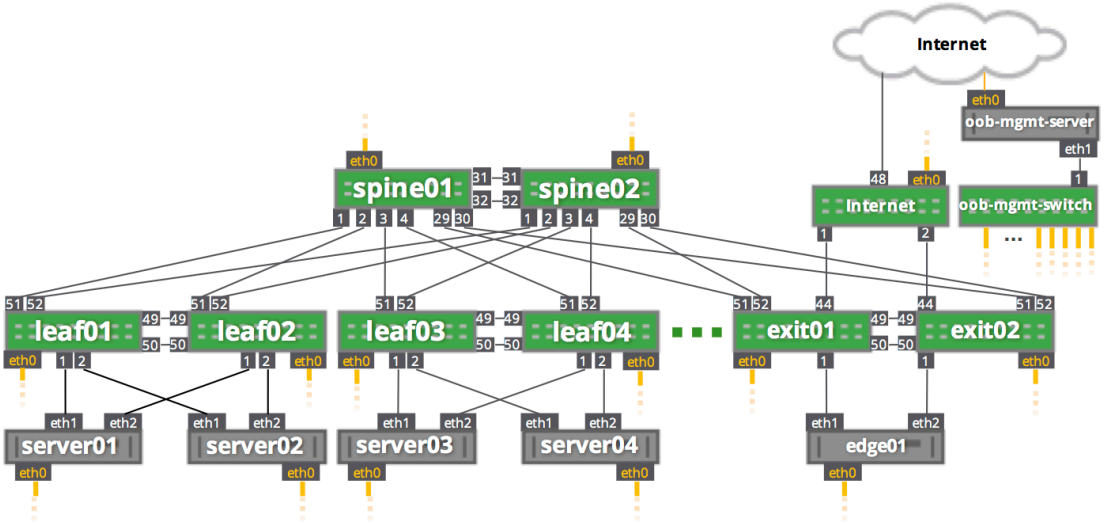
```
# Loopback interface
auto lo
iface lo inet loopback
  address 10.0.0.11/32
  clagd-vxlan-anycast-ip 10.0.0.112
  alias loopback interface

# Management interface
 auto eth0
 iface eth0 inet dhcp
    vrf mgmt

auto mgmt
iface mgmt
    address 127.0.0.1/8
    address ::1/128
    vrf-table auto

# Port to Server01
auto swp1
iface swp1
  alias to Server01
  # This is required for Vagrant only
  post-up ip link set swp1 promisc on

# Port to Server02
auto swp2
iface swp2
  alias to Server02
```

Cumulus Networks Services Demos

The Services team demos provide a virtual environment built using either VirtualBox or `libvirt` using Vagrant to manage the VMs. This environment utilizes the reference topology shown below. Vagrant and Cumulus VX can be used together to build virtual simulations of production networks to validate configurations, develop automation code and simulate failure scenarios.

Reference Topology

The Cumulus Linux *reference topology* includes cabling (in DOT format for dual use with PTM), MAC addressing, IP addressing, switches and servers. This topology is blessed by our Professional Services Team to fit a majority of designs seen in the field.



IP and MAC Addressing

Hostname	eth0 IP	eth0 MAC	Interface Count
oob-mgmt-server	192.168.0.254	any	
oob-mgmt-switch	192.168.0.1	any	Cumulus RMP
leaf01	192.168.0.11	A0:00:00:00:00:11	48x10g w/ 6x40g uplink
leaf02	192.168.0.12	A0:00:00:00:00:12	48x10g w/ 6x40g uplink
leaf03	192.168.0.13	A0:00:00:00:00:13	48x10g w/ 6x40g uplink
leaf04	192.168.0.14	A0:00:00:00:00:14	48x10g w/ 6x40g uplink
spine01	192.168.0.21	A0:00:00:00:00:21	32x40g
spine02	192.168.0.22	A0:00:00:00:00:22	32x40g
server01	192.168.0.31	A0:00:00:00:00:31	10g NICs
server02	192.168.0.32	A0:00:00:00:00:32	10g NICs
server03	192.168.0.33	A0:00:00:00:00:33	10g NICs
server04	192.168.0.34	A0:00:00:00:00:34	10g NICs
exit01	192.168.0.41	A0:00:00:00:00:41	48x10g w/ 6x40g uplink

Hostname	eth0 IP	eth0 MAC	Interface Count
			(exit leaf)
exit02	192.168.0.42	A0:00:00:00:00:42	48x10g w/ 6x40g uplink (exit leaf)
edge01	192.168.0.51	A0:00:00:00:00:51	10g NICs (customer edge device, firewall, load balancer, etc.)
internet	192.168.0.253	any	(represents internet provider edge device)

Build the Topology

Virtual Appliance

You can build out the reference topology in hardware or using Cumulus VX. The Cumulus Reference Topology using Vagrant is essentially the reference topology built out inside Vagrant with VirtualBox or KVM. The installation and setup instructions for bringing up the entire reference topology on a laptop or server are on the cldemo-vagrant GitHub repo.

Hardware

Any switch from the hardware compatibility list is compatible with the topology as long as you follow the interface count from the table above. Of course, in your own production environment, you don't have to use exactly the same devices and cabling as outlined above.

Demos

You can find an up to date list of all the demos in the cldemo-vagrant GitHub repository, which is available to anyone free of charge.

Campus Deployments

Cumulus Linux includes a number of features that you can use to deploy in a campus setting. These features include:

- 802.1X Interfaces
- Inter-subnet Routing
- Power over Ethernet
- TDR - time domain reflectometer
- Voice VLAN

These features work in conjunction with the following core Cumulus Linux functionality to provide a complete campus deployment:

- Automation/zero touch provisioning
- Half duplex mode
- Layer 2, including LLDP and MLAG
- Layer 3, including OSPF and BGP
- QoS

Related Information

For a deep dive into campus architecture, read the campus architecture solution guide.

Docker on Cumulus Linux

Cumulus Linux can be used to run the Docker container platform. You can install Docker Engine directly on a Cumulus Linux switch and run Docker containers natively on the switch.

To set up Docker on Cumulus Linux, run the following commands **as root**.

1. Install the authentication key for Docker:

```
root@switch:~# curl -fsSL https://download.docker.com/linux/
debian/gpg | apt-key add -
```

2. Configure the repositories for Docker:

```
root@switch:~# echo "deb [arch=amd64]
https://download.docker.com/linux/debian buster stable" >/etc/
apt/sources.list.d/docker.list
```

3. Install the Docker package:

```
root@switch:~# apt update
root@switch:~# apt install -y docker-ce
```

4. Configure Docker to minimize impact on the system's firewall and forwarding configuration:

```
root@switch:~# cat >/etc/docker/daemon.json <<EOD
{
     "iptables": false,
     "ip-forward": false,
     "ip-masq": false
}
EOD
```

5. Configure Docker to run in the management VRF:

```
root@switch:~# cp /lib/systemd/system/docker.service /lib/
systemd/system/docker@.service
root@switch:~# sed -i -re '
     /^Requires=docker.socket$/ d;
     /^ExecStart\>/ s/-H fd:\/\/ //
' /lib/systemd/system/docker@.service


root@switch:~# echo "docker" >>/etc/vrf/systemd.conf
root@switch:~# systemctl daemon-reload
root@switch:~# systemctl mask docker.socket
```

```
root@switch:~# systemctl disable --now docker.service
root@switch:~# systemctl enable --now docker@mgmt
```

6. Test your installation by running the `hello-world` container:

```
root@switch:~# docker run hello-world
```

> **NOTE**
>
> Be mindful of the types of applications you want to run in containers on a Cumulus Linux switch. Depending on the configuration of the container, DHCP servers, custom scripts, and other lightweight services run well. However, VPN, NAT and encryption-type services are CPU-intensive and might lead to undesirable effects on critical applications. Resource-intensive services are not supported.

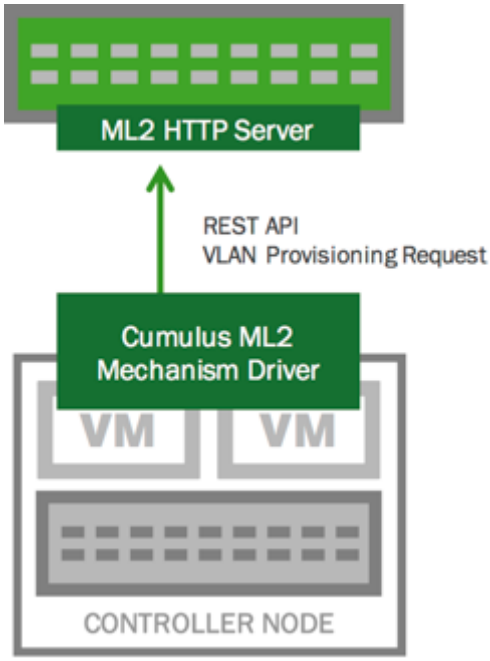
OpenStack Neutron ML2 and Cumulus Linux

The Modular Layer 2 (ML2) plugin is a framework that allows OpenStack Networking to use a variety of non-vendor-specific layer 2 networking technologies. The ML2 framework simplifies adding support for new layer 2 networking technologies and enables dynamic provisioning of VLAN/VXLAN on switches in an OpenStack environment instead of manually provisioning layer 2 connectivity for each VM.

The plugin supports configuration caching. The cached configuration is replayed back to the Cumulus Linux switch from the Cumulus ML2 mechanism driver when a switch or process restart is detected.

To deploy OpenStack ML2 in a network with Cumulus Linux switches, you need the following:

- A REST API, which is installed with Cumulus Linux.
- The Modular Layer 2 (ML2) mechanism driver for OpenStack, which you install on the OpenStack Neutron controller node. The driver is available as a Python package from upstream.
- The OpenStack Queens release.



Configure the REST API

1. Configure the relevant settings in the `/etc/restapi.conf` file:

```
[ML2]
#local_bind = 10.40.10.122
#service_node = 10.40.10.1


# Add the list of inter switch links that
# need to have the vlan included on it by default
# Not needed if doing Hierarchical port binding
#trunk_interfaces = uplink
```

2. Restart the REST API service for the configuration changes to take effect:

```
cumulus@switch:~$ sudo systemctl restart restserver
```

Additional REST API calls have been added to support bridge configuration using the bridge name instead of network ID.

Install and Configure the ML2 Driver

1. Install the ML2 mechanism driver on your Neutron host, which is available upstream:

```
root@neutron:~# git clone https://github.com/CumulusNetworks/
networking-cumulus.git
root@neutron:~# cd networking-cumulus
root@neutron:~# python setup.py install
root@neutron:~# neutron-db-manage upgrade head
```

2. Configure the host to use the ML2 driver:

```
root@neutron:~# openstack-config --set /etc/neutron/plugins/
ml2/ml2_conf.ini mechanism_drivers linuxbridge,cumulus
```

3. List the Cumulus Linux switches to configure. Edit the `/etc/neutron/plugins/ml2/ml2_conf.ini` file and add the IP addresses of the Cumulus Linux switches to the `switches` line. For example:

```
[ml2_cumulus]
switches="192.168.10.10,192.168.20.20"
```

The ML2 mechanism driver includes the following parameters, which you can configure in the `/etc/neutron/plugins/ml2/ml2_conf.ini` file.

Parameter	Description
`switches`	The list of Cumulus Linux switches connected to the Neutron host. Specify a list of IP addresses.
`scheme`	The scheme for the base URL for the ML2 API. For example, HTTP.
`protocol_port`	The protocol port for the bast URL for the ML2 API. The default value is *8000*.
`sync_time`	A periodic time interval for polling the Cumulus Linux switch. The default value is *30* seconds.
`spf_enable`	Enables and disables SPF for the

Parameter	Description
	bridge. The default value is *False*.
`new_bridge`	Enables and disables VLAN-aware bridge mode for the bridge configuration. The default value is *False*, so a traditional mode bridge is created.

OpenStack with Cumulus in the Cloud

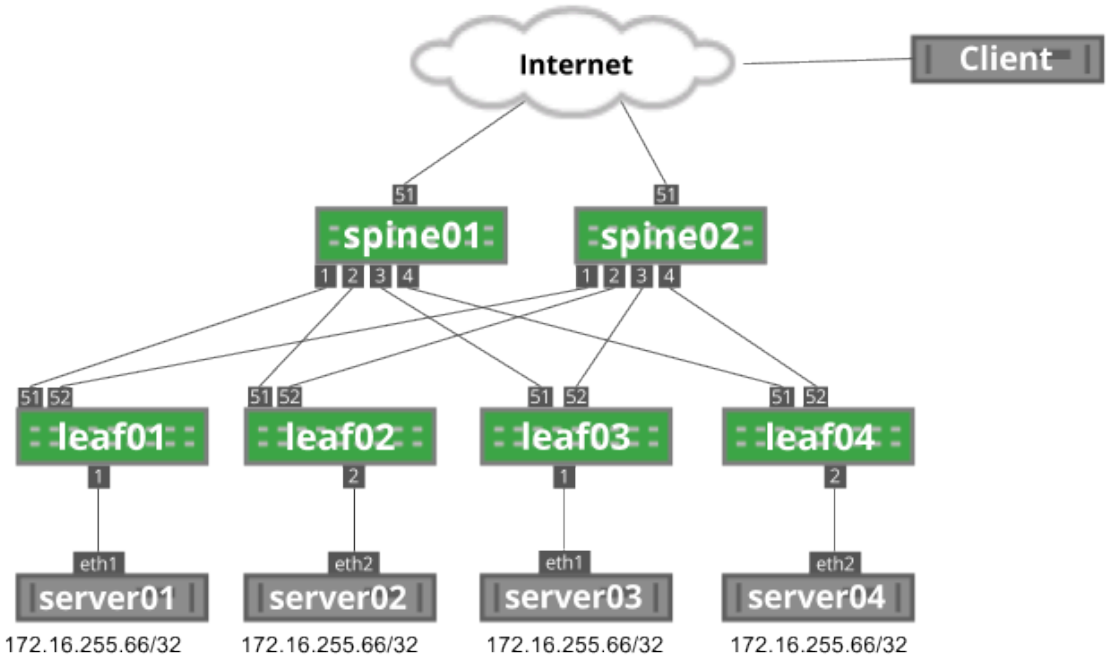
OpenStack Neutron is available as a preconfigured option with Cumulus in the Cloud. Add the ML2 driver, described above.

Anycast Design Guide

Routing on the Host enables you to run OSPF or BGP directly on server hosts. This can enable a network architecture known as *anycast*, where many servers can provide the same service without needing layer 2 extensions or load balancer appliances.

Anycast is not a new protocol or protocol implementation and does not require any additional network configuration. Anycast leverages the equal cost multipath (ECMP) capabilities inherent in layer 3 networks to provide stateless load sharing services.

The following image depicts an example anycast network. Each server is advertising the 172.16.255.66/32 anycast IP address.

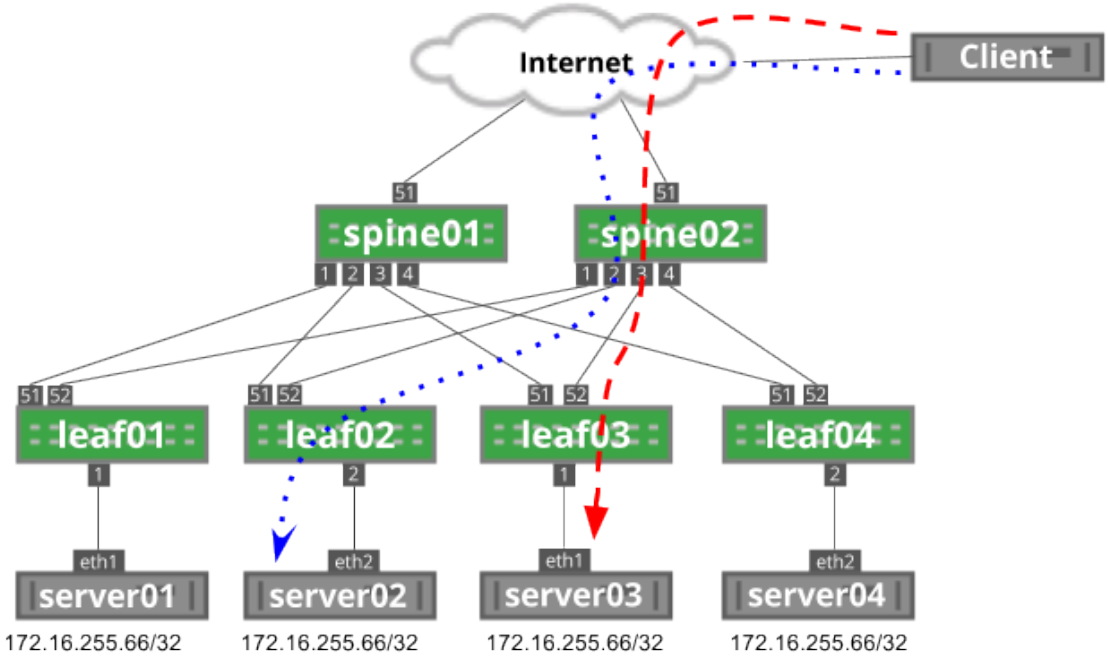


Anycast Architecture

Anycast relies on layer 3 equal cost multipath functionality to provide load sharing throughout the network. Each server announces a route for a service. As the route is propagated through the network, each network device sees the route as originating from multiple places. As an end user connects to the anycast IP, each network device performs a hardware hash of the layer 3 and layer 4 headers to determine which path to use.

Every packet in a flow from an end user has the same source and destination IP address as well as source and destination port numbers. The hash performed by the network devices results in the same answer for every packet, ensuring all packets in a flow are sent to the same destination.

In the following image, the client initiates two flows: the blue, dotted flow and the red dashed flow. Each flow has the same source IP address (the client's IP address), destination IP address (172.16.255.66) and same destination port (depending on the service; for example, DNS is port 53). Each flow has a unique source port generated by the client.



In this example, each flow hashes to different servers based on this source port, which you can see when you run `ip route show` to the destination IP address:

```
cumulus@spine02$ ip route show 172.16.255.66
172.16.255.66  proto zebra  metric 20
    nexthop via 169.254.64.0  dev swp1 weight 1
    nexthop via 169.254.64.2  dev swp2 weight 1
    nexthop via 169.254.64.2  dev swp3 weight 1
    nexthop via 169.254.64.0  dev swp4 weight 1
```

On a Cumulus Linux switch, you can see the hardware hash with the `cl-ecmpcalc` command. In Figure 2, two flows originate from a remote user

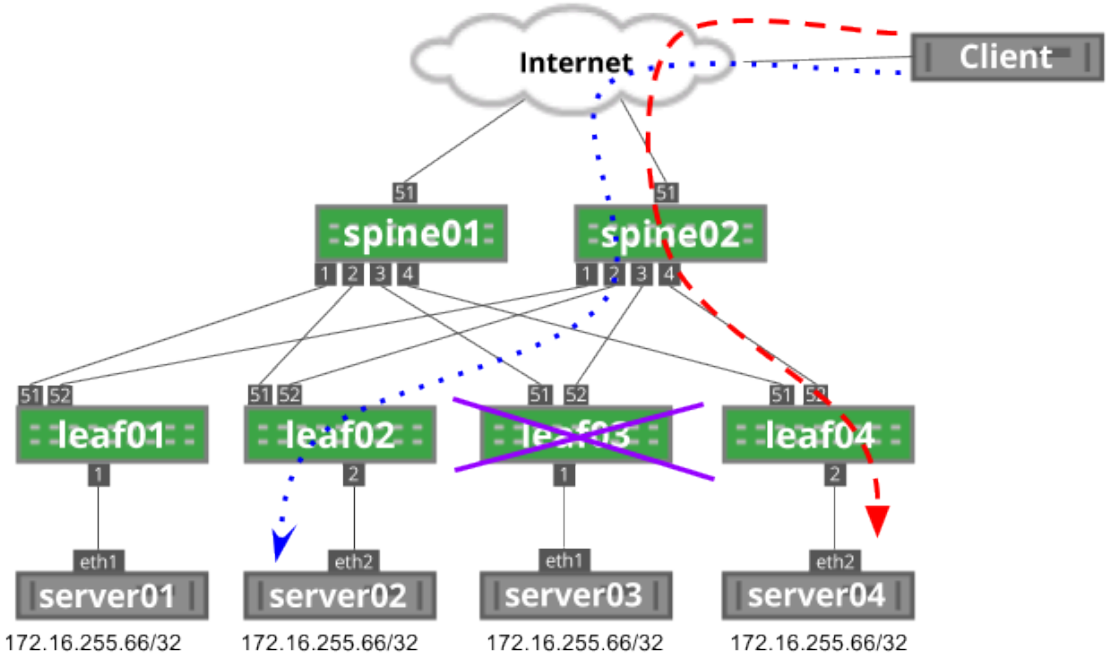
destined to the anycast IP address. Each session has a different source port. Using the `cl-ecmpcalc` command, you can see that the sessions were hashed to different egress ports.

```
cumulus@spine02$ sudo cl-ecmpcalc -p udp -s 10.2.0.100 --sport
32700 -d 172.31.255.66 --dport 53 -i swp51
ecmpcalc: will query hardware
swp2

cumulus@spine02$ sudo cl-ecmpcalc -p udp -s 10.2.0.100 --sport
31884 -d 172.31.255.66 --dport 53 -i swp51
ecmpcalc: will query hardware
swp3
```

Anycast with TCP and UDP

A key component to the functionality and cost effective nature of anycast is that the network does not maintain state for flows. Every packet is handled individually through the routing table, saving memory and resources that would be required to track individual flows, similar to the functionality of a load balancing appliance.

As previously described, every packet in a flow hashes to the same next hop. However, if that next hop is no longer valid, the traffic flows to another anycast next hop instead. For example, in the image below, if leaf03 fails, traffic flows to a different anycast address; in this case, server04:



For stateless applications that rely on UDP, like DNS, this does not present a problem. However, for stateful applications that rely on TCP, like HTTP, this breaks any existing traffic flows, such as a file download. If the TCP three-way handshake was established on server03, after the failure, server04 would have no connection built and would send a TCP reset message back to the client, restarting the session.

This is not to say that it is not possible to use TCP-based applications for anycast. However, TCP applications in an anycast environment should have short-lived flows (measured in seconds or less) to reduce the impact of network changes or failures.

Resilient Hashing

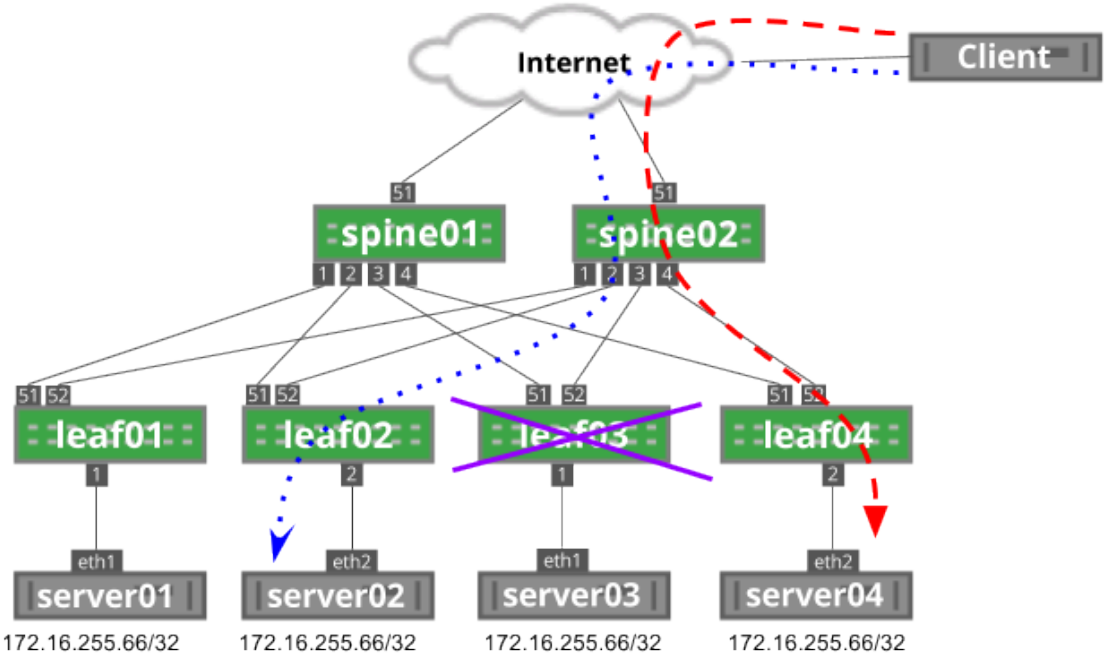
Resilient hashing provides a method to prevent failures from impacting the

hash result of unrelated flows. However, resilient hashing does not prevent rehashing when new next hops are added.

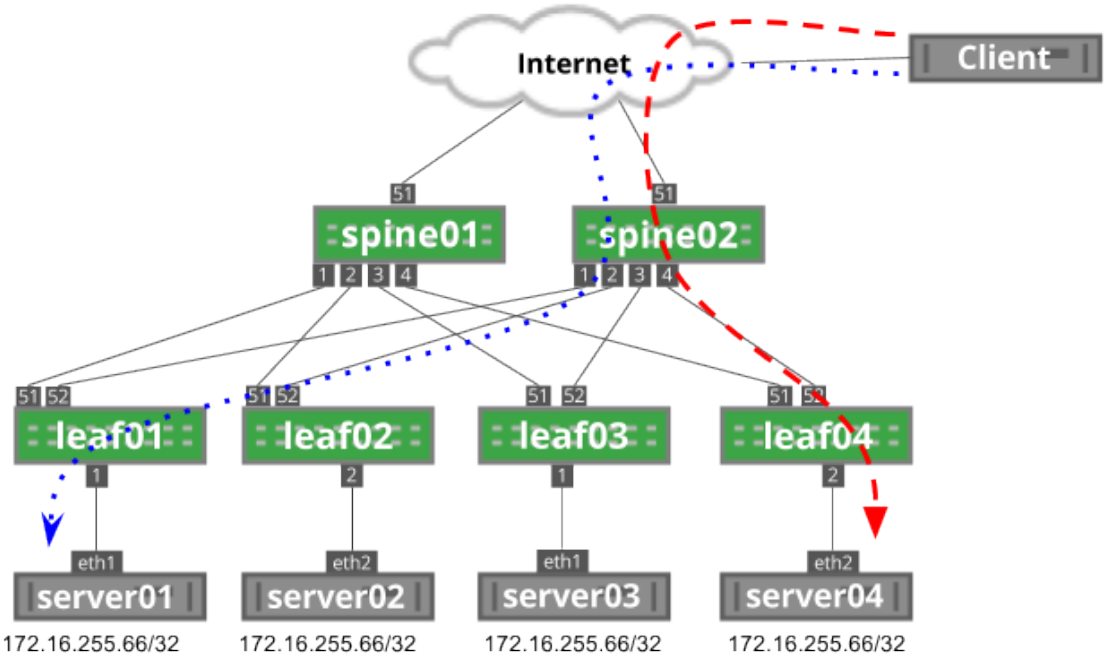
As previously mentioned, the hardware hashing function determines which path gets used for a given flow. The simplified version of that hash is the combination of protocol, source IP address, destination IP address, source layer 4 port and destination layer 4 port. The full hashing function includes not only these fields but also the list of possible layer 3 next hop addresses. The hash result is passed through a *modulo* of the number of next hop addresses. If the number of next hop addresses changes, through either addition or subtraction of the next hops, this changes the hash result for all traffic, including flows that have already established.

Continuing with the example in Figure 3, leaf03 is in a failed state, so traffic is hashing to server04. This is a result of the hash considering three possible next hop IPs (leaf01, leaf02, leaf04). When leaf03 is brought back online, the number of possible next hop IPs grows to four. This changes the modulo value that is part of the hashing function, which may result in traffic being sent to a different server, even if previously unaffected by the change.

As you can see below, leaf03 is in a failed state. The blue dotted flow uses leaf02 to reach server02.

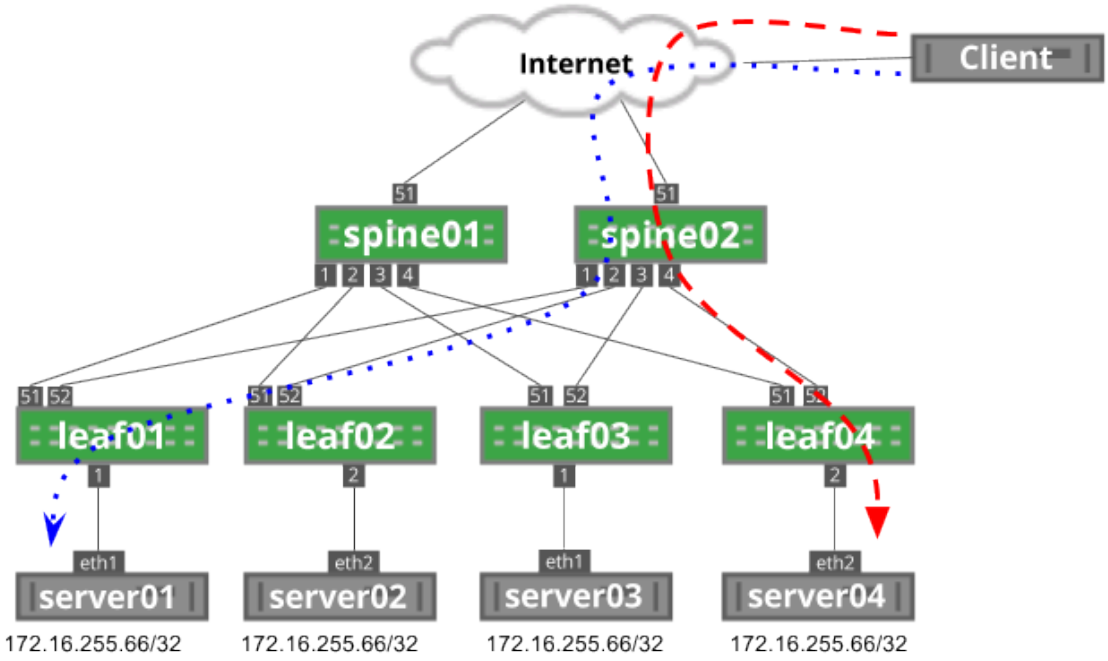


As leaf03 is brought back into service, the hashing function on spine02 changes, impacting the blue dotted flow:

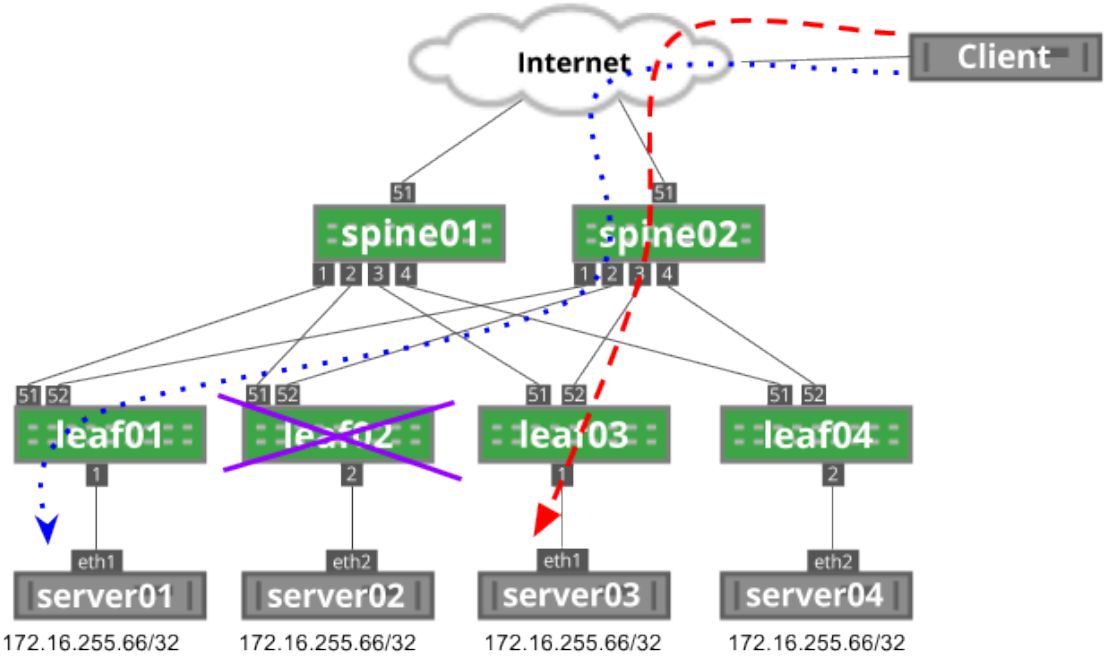


Just as the addition of a device can impact unrelated traffic, the removal of

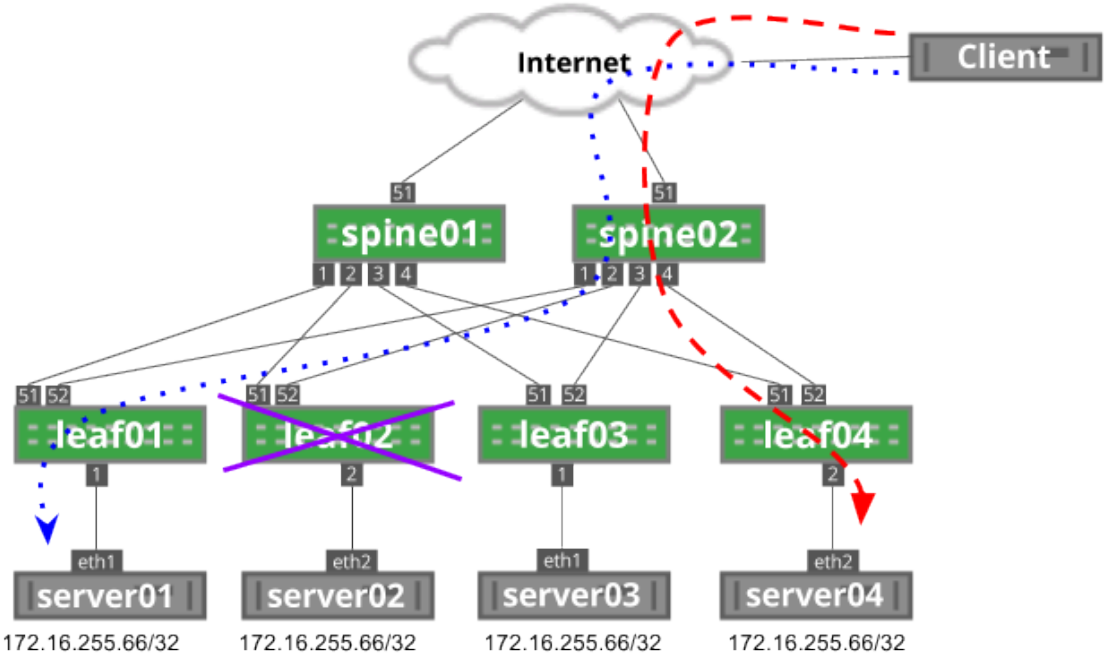
a device can also impact unrelated traffic, since again, the modulo of the hash function is changed. You can see this below, where the blue dotted flow goes through leaf01 and the red dashed line goes through leaf04.



Now, leaf02 has failed. As a result, the modulo on spine02 has changed from four possible next hops to only three next hops. In this example, the red dashed line has rehashed to leaf03:



To help solve this issue, resilient hashing can prevent traffic flows from shifting on unrelated failure scenarios. With resilient hashing enabled, the failure of leaf02 does not impact both existing flows, since they do not currently flow through leaf02:



Although resilient hashing can prevent rehashing on next hop failure, it cannot prevent rehashing on next hop addition.

You can read more information on resilient hashing in the ECMP chapter.

Applications for Anycast

As previously mentioned, UDP-based applications are great candidates for anycast architectures, such as NTP or DNS.

When considering applications to be deployed in an anycast scenario, the first two questions to answer are:

- Whether the application relies on TCP for proper sequencing of data.
- Whether the application relies on more than one session as part of the application.

Applications with Multiple Connections

The network has no knowledge of any sessions or relationships between different sessions for the same application. This affects protocols that rely on more than one TCP or UDP connection to function properly - one example being FTP.

FTP data transfers require two connections: one for control and one for the file transfer. These two connections are independent, with their own TCP ports. Consider the scenario where an FTP server was deployed in an anycast architecture. When the secondary data connection is initiated, the traffic is destined initially to the same FTP server IP address, but the network hashes this traffic as a new, unique flow because the ports are different. This may result in the new session ending up on a new server. The new server would only accept that data connection if the FTP server application was capable of robust information sharing, as it has no history of the original request in the control session.

Initiating Traffic vs. Receiving Traffic

It is also important to understand that an outbound TCP session should never be initiated over an anycast IP address, as traffic that originates from an anycast IP address may not return to the same anycast server after the network hash. Contrast this with inbound sessions, where the network hash is the same for all packets in a flow, so the inbound traffic will hash to the same anycast server.

TCP and Anycast

TCP-based applications can be used with anycast, with the following recommendations:

- TCP sessions are short lived.
- The impact of a failed session or TCP reset does not impact the application. For example, a web page refresh is acceptable.
- There is application-level session management that is completely independent of the TCP session.
- A redirection middleware layer handles incorrectly hashed flows.

TCP applications that have longer-lived flows should not be used as anycast services. For example:

- FTP or other large file transfers.
- Transactions that must be completed and journaled. For example, financial transactions.
- Streaming media without application-level automated recovery.

It should be noted that anycast TCP is possible and has been implemented by a number of organizations, one notable example being LinkedIn.

Conclusion

Anycast can provide a low cost, highly scalable implementation for services. However, the limitations inherent in network-based ECMP makes anycast challenging to integrate with some applications. An anycast

architecture is best suited for stateless applications or applications that are able to share session state at the application layer.

RDMA over Converged Ethernet - RoCE

RDMA over Converged Ethernet (RoCE) provides the ability to write to compute or storage elements using remote direct memory access (RDMA) over an Ethernet network instead of using host CPUs. RoCE relies on congestion control and lossless Ethernet to operate. Cumulus Linux supports features that can enable lossless Ethernet for RoCE environments.

> **NOTE**
>
> While Cumulus Linux can support RoCE environments, the hosts send and receive the RoCE packets.

RoCE helps you obtain a *converged network*, where all services run over the Ethernet infrastructure, including Infiniband apps.

There are two versions of RoCE, which run at separate layers of the stack:

- RoCEv1, which runs at the link layer and cannot be run over a routed network. Therefore, it requires the priority flow control (PFC) to be enabled.
- RoCEv2, which runs over layer 3. Because it is a routed solution, consider using explicit congestion notification (ECN) with RoCEv2 as ECN bits are communicated end-to-end across a routed network.

Enable RDMA over Converged Ethernet with PFC

RoCEv1 uses the Infiniband (IB) Protocol over converged Ethernet. The IB global route header rides directly on top of the Ethernet header. The lossless Ethernet layer handles congestion hop by hop.

To learn the Cumulus Linux settings you need to configure support for RoCEv1; see the example configuration in the PFC section of the Buffer and Queue Management chapter.

On switches with Spectrum ASICs, you can use NCLU to configure RoCE with PFC:

```
cumulus@switch:~$ net add interface swp1 storage-optimized pfc
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following configuration in the `/etc/cumulus/datapath/traffic.conf` file. They configure PFC on cos 3 and ECN on cos 3 in the `/etc/cumulus/datapath/traffic.conf` file. They also add a flow control buffer pool for lossless traffic and change the buffer limits in the `/usr/lib/python2.7/dist-packages/cumulus/__chip_config/mlx/datapath.conf` file.

```
cumulus@switch:~$ sudo cat /etc/cumulus/datapath/traffic.conf
...
ecn_red.port_group_list = [ROCE_ECN]
pfc.ROCE_PFC.port_set = swp1
pfc.ROCE_PFC.cos_list = [3]
pfc.ROCE_PFC.xoff_size = 18000
pfc.ROCE_PFC.xon_delta = 18000
pfc.ROCE_PFC.tx_enable = true
pfc.ROCE_PFC.rx_enable = true
pfc.ROCE_PFC.port_buffer_bytes = 70000
ecn_red.ROCE_ECN.port_set = swp1
ecn_red.ROCE_ECN.cos_list = [3]
ecn_red.ROCE_ECN.min_threshold_bytes = 150000
ecn_red.ROCE_ECN.max_threshold_bytes = 1500000
ecn_red.ROCE_ECN.ecn_enable = true
ecn_red.ROCE_ECN.red_enable = true
ecn_red.ROCE_ECN.probability = 100


...
```

NOTE

While link pause is another way to provide lossless ethernet, PFC is

the preferred method. PFC allows more granular control by pausing the traffic flow for a given CoS group, instead of the entire link.

Enable RDMA over Converged Ethernet with ECN

RoCEv2 requires flow control for lossless Ethernet. RoCEv2 uses the Infiniband (IB) Transport Protocol over UDP. The IB transport protocol includes an end-to-end reliable delivery mechanism and has its own sender notification mechanism.

RoCEv2 congestion management uses RFC 3168 to signal congestion experienced to the receiver. The receiver generates an RoCEv2 congestion notification packet directed to the source of the packet.

To learn the Cumulus Linux settings, you need to configure support for RoCEv2; see the example configuration in the ECN section of the Buffer and Queue Management chapter.

On switches with Spectrum ASICs, you can use NCLU to configure RoCE with ECN:

```
cumulus@switch:~$ net add interface swp1 storage-optimized
```

```
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following configuration in the `/etc/cumulus/datapath/traffic.conf` file:

```
cumulus@switch:~$ sudo cat /etc/cumulus/datapath/traffic.conf
...
ecn_red.port_group_list = [ROCE_ECN]
ecn_red.ROCE_ECN.port_set = swp1
ecn_red.ROCE_ECN.cos_list = [3]
ecn_red.ROCE_ECN.min_threshold_bytes = 150000
ecn_red.ROCE_ECN.max_threshold_bytes = 1500000
ecn_red.ROCE_ECN.ecn_enable = true
ecn_red.ROCE_ECN.red_enable = true
ecn_red.ROCE_ECN.probability = 100
...
```

The `storage-optimized` command changes the buffer limits in the `/usr/lib/python2.7/dist-packages/cumulus/__chip_config/mlx/datapath.conf` file.

It also enables drop behaviors and Random Early Detection (RED). RED identifies packets that have been added to a long egress queue. The ECN

action marks the packet and forwards it, requiring the packet to be ECT-capable. However, the drop action drops the packet, requiring the packet to **not** be ECT-capable.

Related Information

- RoCE introduction - roceinitiative.org
- RoCEv2 congestion management - community.mellanox.com
- Configuring RoCE over a DSCP-based lossless network on a switch with a Mellanox Spectrum ASIC

Cumulus Hyperconverged Solution with Nutanix

The Cumulus Hyperconverged Solution (HCS) in Cumulus Linux supports automated integration with the Nutanix Prism Management solution and the Nutanix AHV hypervisor. Cumulus HCS automatically configures ports attached to Nutanix nodes, provisions networking and manages VLANs with Nutanix Prism and Nutanix AHV.

In addition, you can augment the deployment with:

- Cumulus on a Stick for zero touch provisioning Nutanix and Cumulus HCS without any user interaction or additional equipment.
- Cumulus NetQ for network telemetry and unprecedented real-time and historic visibility into dynamic changes in both the network and virtual machines.
- Out-of-band management and IPMI access using Cumulus RMP or a generic Cumulus Linux switch, enabling the full provisioning of a zero-touch data and management network, eliminating any network deployment delays when standing up a Nutanix cluster.

Cumulus HCS has two major components:

- **Nutanix LLDP Switch Agent**. When enabled, the agent listens for directly connected Nutanix servers via LLDP and enables MLAG bonding on the relevant ports.

- **Nutanix Webhook VLAN Provisioner**. Cumulus Linux switches register

with the Nutanix CVM and wait to receive Nutanix webhooks. When a new VM is deployed on a server in the cluster, the CVM sends a message to the Cumulus Linux switch with the physical server name and relevant VLANs. The switch then dynamically provisions the configuration on the ports of the specific physical server.

Cumulus HCS periodically polls Nutanix Prism for information about VMs in the cluster. When a new VM is discovered, the service automatically identifies the physical Nutanix server hosting the VM and discovers any VLANs required for the VM. The service then automatically adds these VLANs to the default VLAN-aware bridge, the MLAG peer link and the automatically created bond to the Nutanix node. When a VM is powered off, removed or moved, and the associated VLAN has no other VMs, the VLAN is automatically removed from the bridge, peer link and dynamic bond.

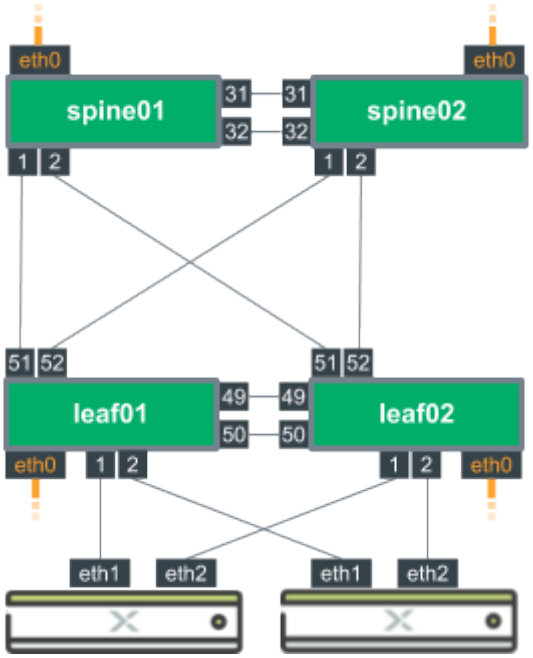
Requirements

- 2 compatible switches running Cumulus Linux
- Nutanix AOS 5.5.8 or later
- Nutanix AHV 20170830.185 or later
- LLDP enabled on Nutanix (which is the default in 5.5.8 and later)
- IP connectivity between the Cumulus Linux switches and the Nutanix controller VMs (CVMs)
- MLAG enabled on the Cumulus Linux switches

Cumulus HCS runs on any platform. However, this chapter assumes a typical Nutanix deployment with the following configuration:

- Leaf switches with 48 x 10G or 25G ports
- Four or more 40G or 100G uplinks
- Nutanix servers are attached to any of the 10G or 25G ports
- MLAG peer link is on the first two uplink ports: swp49 and swp50
- Connections to other infrastructure are on ports swp51 and above
- The eth0 management interface is configured for management VRF via DHCP
- For automatic configuration, the gateway IP addresses for all VMs, including the CVM, do not exist on the Cumulus Linux switches

The example configuration utilizes the following topology. All configuration focuses on the leaf01 and leaf02 switches. Configurations for spine01 and spine02 are not included.



Configure Cumulus HCS and Nutanix

The method you choose for configuring Cumulus HCS and Nutanix depends upon whether or not you already have Cumulus Linux installed on your switches, which are named *leaf01* and *leaf02* in the example configuration above.

- If you have bare-metal switches without Cumulus Linux installed, follow the steps below for configuring a bare-metal switch with ZTP.
- If Cumulus Linux is already installed on your switches, follow the steps below for manually configuring an existing Cumulus Linux switch.

Configure the Service with ZTP

The following steps describe how to use zero touch provisioning to install Cumulus Linux and fully configure Cumulus HCS and Nutanix on your network.

To do this, you need a Cumulus on a Stick disk image and a USB stick with at least 1GB of storage.

1. Insert the USB stick into your computer and copy the Cumulus on a Stick files onto it.

2. On the USB stick, open the `ztp_config.txt` file in a text editor and set your Nutanix username and password and the server IP address, then save and close the file.

```
# Fill in the parameters below to allow for ZTP to
# automatically configure the switch for Nutanix
#
# The username for the Nutanix API. Likely the username you
use to login to Prism. (Required)
NUTANIX_USERNAME=admin
# The password for the user. (Required)
NUTANIX_PASSWORD=nutanix/4u
# The IP address of a Nutanix CVM or the CVM anycast/cluster
IP. (Required)
NUTANIX_IP=10.1.1.123
# IP address and subnet mask of this switch in the CVM
subnet. Used to communicate to the Prism API.
SWITCH_CVM_IP=10.1.1.254/24
# If you do not want to use DHCP on the switch eth0, define
the static switch IP and mask below. (Optional)
#SWITCH_MANAGEMENT_IP=10.0.0.11/24
# If you define a static IP, what is the gateway the switch
should use for management traffic? (Optional)
#SWITCH_DEFAULT_GATEWAY=10.1.1.1
# If you have Layer 2 connections to existing infrastructure,
define them here. Separate the interfaces with a comma.
(Optional)
UPLINKS=swp51,swp52
```

```
# It is assumed that ports swp49 and swp50 will be used for
the inter-switch link. If you have other ports, define them
here. (Optional)
#PEERLINK=swp49,swp50
```

3. Place the USB stick into the Cumulus Linux switch (leaf01) and power on the switch. Cumulus Linux is automatically installed, including the license and a baseline configuration. The switch reboots multiple times during this process. Depending on your specific hardware platform, this process can take up to 20 minutes. After the installation completes, the LEDs corresponding to the ports connected to the Nutanix nodes illuminate in green.

4. When the installation completes, remove the USB stick and repeat this procedure on the other Cumulus Linux switch (leaf02).

Configure the Service Manually

If Cumulus Linux is already installed on your switches, follow the steps below to configure Cumulus Linux, Nutanix and Cumulus HCS.

1. Configure MLAG on both the leaf01 and leaf02 nodes. The `sys-mac` is a MAC address from the reserved MAC address space and must be the same on both MLAG peers. If you are deploying more than one pair of switches with MLAG, the `sys-mac` must be unique for each pair of MLAG-configured switches.

leaf01 leaf02

```
cumulus@leaf01:~$ net add interface swp49,swp50 mtu 9216
cumulus@leaf01:~$ net add clag peer sys-mac
44:38:39:FF:40:00 interface swp49,swp50 primary
cumulus@leaf01:~$ net commit
```

2. Configure the default layer 2 bridge. Add a unique IP address to each leaf in the same subnet as the CVM.

leaf01 leaf02

```
cumulus@leaf01:~$ net add bridge bridge ports peerlink
cumulus@leaf01:~$ net add bridge bridge pvid 1
cumulus@leaf01:~$ net add vlan 1 ip address 10.1.1.201/24
cumulus@leaf01:~$ net commit
```

NOTE

In both configurations the `pvid` value of *1* indicates the native VLAN ID. If you do not know the value for the native VLAN ID,

use *1*.

3. Edit the `/etc/default/cumulus-hyperconverged` file and set the Nutanix username, password and server IP address. Do this on both switches (leaf01 and leaf02). Cumulus Linux uses the settings in this file to authenticate and communicate with the Nutanix cluster.

```
cumulus@leaf02:~$ sudo nano /etc/default/cumulus-
hyperconverged


### /etc/default/cumulus-hyperconverged config file
# username for Prism (required)
USERNAME=admin
# password for Prism (required)
PASSWORD=nutanixpassword
# CVM address used by the service (required)
SERVER=10.1.1.11
# Hook server address (optional)
#HOOK_SERVER=10.0.0.0
# Hook port (optional)
#HOOK_PORT=9440
# Socket timeout (optional)
```

```
#SOCKET_TIMEOUT=10.0.0.0
# single/multi rack configuration (optional)
VXLAN_CONFIG=False
# loglevel: verbose/debug (optional)
LOGLEVEL=verbose
# periodic sync timeout (optional)
#PERIODIC_SYNC_TIMEOUT=60
```

These settings are defined below.

> **NOTE**
>
> The server IP address may be a specific Nutanix CVM address or the virtual cluster IP address.

4. Enable and start Cumulus HCS on leaf01 and leaf02.

```
cumulus@leaf01:~$ sudo systemctl enable cumulus-hyperconverged
cumulus@leaf01:~$ sudo systemctl start cumulus-hyperconverged
```

```
cumulus@leaf02:~$ sudo systemctl enable cumulus-hyperconverged
cumulus@leaf02:~$ sudo systemctl start cumulus-hyperconverged
```

5. Verify that the service is running on leaf01 and leaf02.

```
cumulus@leaf01:~$ sudo systemctl status cumulus-hyperconverged
● cumulus-hyperconverged.service - Cumulus Linux
Hyperconverged Daemon
        Loaded: loaded (/lib/systemd/system/cumulus-
hyperconverged.service; enabled)
        Active: active (running) since Mon 2019-01-07
03:36:26 UTC; 56min ago
      Main PID: 4206 (cumulus-hyperco)
        CGroup: /system.slice/cumulus-hyperconverged.service
                ├─4206 /usr/bin/python /usr/bin/cumulus-
hyperconverged
                └─6300 /usr/sbin/lldpcli -f json watch
```

```
cumulus@leaf02:~$ sudo systemctl status cumulus-hyperconverged
● cumulus-hyperconverged.service - Cumulus Linux
Hyperconverged Daemon
```

```
        Loaded: loaded (/lib/systemd/system/cumulus-
hyperconverged.service; enabled)
        Active: active (running) since Mon 2019-01-07
03:36:26 UTC; 56min ago
      Main PID: 4207 (cumulus-hyperco)
        CGroup: /system.slice/cumulus-hyperconverged.service
                ├─4207 /usr/bin/python /usr/bin/cumulus-
hyperconverged
                └─4300 /usr/sbin/lldpcli -f json watch
```

TIP

If the service fails to start, you may find more information in the service's log file. View the log with `sudo journalctl -u cumulus-hyperconverged`.

6. Enable the server-facing ports to accept inbound LLDP frames and configure jumbo MTU on both leaf01 and leaf02.

```
cumulus@leaf01:~$ net add interface swp1-48 mtu 9216
```

```
cumulus@leaf01:~$ net commit
```

```
cumulus@leaf02:~$ net add interface swp1-48 mtu 9216
cumulus@leaf02:~$ net commit
```

At this point, the service is fully configured. It may take up to 60 seconds for LLDP frames to be received to trigger Cumulus HCS.

Cumulus HCS Configuration Settings

Some of the settings you can configure include:

- HOOK_SERVER: the source IP the switch uses when communicating with the Nutanix API. By default, it follows the routing table.
- HOOK_PORT: the port on which the Nutanix CVM is running. The default is *9440*.
- SOCKET_TIMEOUT: the amount of time to wait for a timeout when attempting to communicate with the Nutanix API. The default is *10* seconds.
- VXLAN_CONFIG: when set to *TRUE*, Cumulus HCS automatically provisions VXLAN VNIs as well as VLANs.
- LOGLEVEL: describes the logging level. *Verbose* and *Debug* are acceptable values. Verbose provides information about bond and VLAN creation while Debug helps in troubleshooting by providing more information from sources like LLDP and the Nutanix webhook.

- PERIODIC_SYNC_TIMEOUT: how long before Cumulus HCS times out dynamic configurations without contacting the Nutanix API. The default is *60* seconds.

Configure Uplinks

How you configure uplinks depends upon whether you configured Cumulus HCS with ZTP or manually.

If you used ZTP, you can edit the ZTP settings file to define the uplink ports and the VLANs assigned to those uplinks.

If you manually configured the service, you need to enable the uplinks and define the associated VLANs, as shown below. You need to configure both leaf01 and leaf02.

```
cumulus@leaf01:~$ net add interface swp51-52 mtu 9216
cumulus@leaf01:~$ net add interface swp51-52 bridge vids
1-2999,4000-4094
cumulus@leaf01:~$ net add interface swp51-52 bridge pvid 1
cumulus@leaf01:~$ net commit
```

```
cumulus@leaf02:~$ net add interface swp51-52 mtu 9216
cumulus@leaf02:~$ net add interface swp51-52 bridge vids
1-2999,4000-4094
```

```
cumulus@leaf02:~$ net add interface swp51-52 bridge pvid 1
cumulus@leaf02:~$ net commit
```

TIP

In this example, all VLANs are allowed on the uplink ports. Configuring any set of VLANs is allowed. Be aware that VLANs 3000-3999 are reserved on Cumulus Linux. This example assumes the untagged or native VLAN is VLAN ID (`pvid`) *1*. Change the VLAN ID as needed.

Add Local Default Gateways

You can add one or more local default gateways on both switches to provide a redundant solution, as shown below. It does not matter whether you configured Cumulus HCS with ZTP or manually. ZTP does not add any gateway configuration.

To provide redundant gateways for the dual-attached Nutanix servers, Cumulus Linux relies on Virtual Router Redundancy (VRR). VRR enables hosts to communicate with any redundant router without reconfiguration, running dynamic routing protocols, or running router redundancy

protocols. This means that redundant routers will respond to Address Resolution Protocol (ARP) requests from hosts. Routers are configured to respond in an identical manner, but if one fails, the other redundant routers will continue to respond, leaving the hosts with the impression that nothing has changed.

Configure leaf01 Configure leaf02

```
cumulus@leaf01:~$ net add vlan 1 ip address 10.1.1.11/24
cumulus@leaf01:~$ net add vlan 1 ip address-virtual
00:00:5e:00:01:01 10.1.1.1/24
cumulus@leaf01:~$ net commit
```

The first configuration line defines the IP address assigned to each switch, which is required and must be unique. On leaf01, this IP address is *10.1.1.11/24*; on leaf02, it is *10.1.1.12/24*.

The second line defines the virtual IP address that is used as the default gateway address for any hosts in this VLAN. On both leaf01 and leaf02 this IP address is *10.1.1.1/24*. The address-virtual MAC address is assigned from a reserved pool of MAC addresses. The address must start with 00:00:05:00:01: and end with any hex value between 00 and ff. Both leaf01 and leaf02 must have the same MAC address. Outside of this switch pair, this MAC address must be unique and only be assigned to a single switch pair in your network.

Out-of-band Solutions

You can configure out-of-band management in one of two ways:

- Using Cumulus RMP, which is the recommended way.
- Running Cumulus Linux on a supported 1G non-Cumulus RMP switch.

Cumulus RMP

Cumulus RMP is a ready-to-deploy solution that enables out-of-band management for web-scale networks. With Cumulus RMP, you can directly manage and support Nutanix systems in the rack without relying on the rest of the network.

To deploy Nutanix with Cumulus RMP, connect the Nutanix 1G IPMI, 1G Shared IPMI and 1G ports to the Cumulus RMP switch. No additional configuration is required.

Cumulus RMP does not support MLAG or active/active connections across Cumulus RMP switches. Connections across more than one Cumulus RMP switch rely on traditional spanning tree protocol for redundancy.

Other Cumulus Linux 1G Switches

If you want to use a non-Cumulus RMP 1G switch that supports Cumulus Linux for out-of-band management, you must manually install the Cumulus Linux software and license and set up the baseline configuration. The default Cumulus on a Stick image has this information.

After you install the software, you can use the following command to configure all ports for a single, untagged management VLAN, including any uplinks.

```
cumulus@oob-switch:~$ net add interface swp1-52 bridge access 1
```

You can assign a management IP address to this same untagged bridge interface. Use an appropriate IP address for your infrastructure.

```
cumulus@oob-switch:~$ net add vlan 1 ip address 192.0.2.1/24
```

Apply the configuration:

```
cumulus@oob-switch:~$ net commit
```

> **NOTE**
>
> In both configurations the value of *1* indicates the native or untagged VLAN ID. If you want to use a different VLAN ID, just replace the *1* in both commands with the desired VLAN ID.

Troubleshoot Cumulus HCS

Some ways you can troubleshoot Cumulus HCS include:

- Checking that bonds are being dynamically created.
- Ensuring LLDP messages are being received.
- Verifying the Cumulus HCS configuration.

Verify Dynamic Bonds Are Being Created

Use the `net show interface bonds` command to verify that bonds are being dynamically created. The following example shows that three bonds, *bond_swp1*, *bond_swp2* and ***bond_swp3*** are created automatically, which means that Cumulus HCS is operating correctly. The name of every dynamically created bond begins with *bond_* and ends with the interface name.

```
cumulus@leaf01:~$ net show interface bonds
    Name       Speed   MTU  Mode     Summary
--  ---------  -----  ----  -------
---------------------------------
UP  bond_swp1  1G     1500  802.3ad  Bond Members: swp1(UP)
UP  bond_swp2  1G     1500  802.3ad  Bond Members: swp2(UP)
UP  bond_swp3  1G     1500  802.3ad  Bond Members: swp3(UP)
UP  peerlink   2G     1500  802.3ad  Bond Members: swp49(UP),
swp50(UP)
```

Verify LLDP Messages Are Being Received

If bonds are not being created, then LLDP messages may not be getting through. You can check for this possibility using the `net show lldp` command:

```
cumulus@leaf01:~$ net show lldp
LocalPort  Speed  Mode           RemoteHost       RemotePort
---------  -----  -------------  ---------------  ----------
swp1       1G     BondMember     NTNX-e08c61ec-A  ens3
swp2       1G     BondMember     NTNX-d618a06d-A  ens3
swp3       1G     BondMember     NTNX-4e6eac27-A  ens3
swp49      1G     BondMember     leaf02           swp49
swp50      1G     BondMember     leaf02           swp50
swp52      1G     NotConfigured  spine01          swp1
swp52      1G     NotConfigured  spine02          swp1
```

View Detailed Nutanix LLDP Information

Cumulus HCS replies on the LLDP `SysDescr` field to identify a Nutanix host. Run the `net show lldp <swp>` command to view the complete LLDP details of the Nutanix node and verify the `SysDescr` field.

```
cumulus@leaf01:~$ net show lldp swp1
```

```
-------------------------------------------------------------------------------
LLDP neighbors:
-------------------------------------------------------------------------------
Interface:    swp1, via: LLDP, RID: 37, Time: 0 day, 01:42:23
  Chassis:
    ChassisID:    mac 2c:c2:60:50:f6:8a
    SysName:      NTNX-e08c61ec-A
    SysDescr:     CentOS Linux 7 (Core) Linux
4.4.77-1.el7.nutanix.20180425.199.x86_64 #1 SMP Thu Apr 26
01:01:53 UTC 2018 x86_64
    MgmtIP:       10.1.1.10
    MgmtIP:       fe80::2ec2:60ff:fe50:f68a
    Capability:   Bridge, on
    Capability:   Router, off
    Capability:   Wlan, off
    Capability:   Station, off
  Port:
    PortID:       mac 2c:c2:60:50:f6:8a
    PortDescr:    ens3
    TTL:          120
    PMD autoneg:  supported: yes, enabled: yes
      Adv:          10Base-T, HD: yes, FD: yes
      Adv:          100Base-TX, HD: yes, FD: yes
      Adv:          1000Base-T, HD: no, FD: yes
```

```
      MAU oper type: 1000BaseTFD - Four-pair Category 5 UTP,
full duplex mode
----------------------------------------------------------------------------
```

Considerations

Reloading Cumulus HCS causes the bond interfaces to rebuild. For the stability of the Nutanix cluster, do not reload the service on both leaf switches simultaneously.