



Cumulus RMP 3.0.0

User Guide

Table of Contents

Welcome to Cumulus Networks	4
Quick Start Guide	5
Contents	5
What's New in Cumulus RMP 3.0.0	5
Open Source Contributions	5
Prerequisites	6
Supported Hardware	6
Setting up a Cumulus RMP Switch	6
Upgrading Cumulus RMP	6
Configuring Cumulus RMP	6
Configuring Switch Ports	9
Configuring a Loopback Interface	11
System Management	13
Setting Date and Time	13
Authentication, Authorization, and Accounting	16
Managing Application Daemons	31
Configuring switchd	35
Installation, Upgrading and Package Management	38
Configuring and Managing Network Interfaces	64
Contents	64
Commands	64
Man Pages	65
Configuration Files	65
Basic Commands	65
ifupdown2 Interface Classes	66
Configuring a Loopback Interface	67
ifupdown2 Interface Dependencies	69
Configuring IP Addresses	73
Specifying User Commands	74
Sourcing Interface File Snippets	75
Using Globs for Port Lists	75
Using Templates	76
Adding Descriptions to Interfaces	77
Caveats and Errata	77
Useful Links	78
Layer 1 and Switch Port Attributes	78
Configuring DHCP Relays	84

Layer 1 and Layer 2 Features	90
Spanning Tree and Rapid Spanning Tree	90
Link Layer Discovery Protocol	106
Prescriptive Topology Manager - PTM	112
Bonding - Link Aggregation	123
Ethernet Bridging - VLANs	126
 Routing	 157
Contents	157
Commands	157
Configuring Static Routing	157
Useful Links	158
 Monitoring and Troubleshooting	 159
Contents	159
Commands	159
Using the Serial Console	159
Diagnostics Using cl-support	160
Sending Log Files to a syslog Server	162
Next Steps	164
Single User Mode - Boot Recovery	165
Resource Diagnostics Using cl-resource-query	166
Monitoring System Hardware	167
Understanding and Decoding the cl-support Output File	172
Troubleshooting Network Interfaces	188
Network Troubleshooting	198
SNMP Monitoring	216
 Index	 242

Welcome to Cumulus Networks

We are transforming networking with Cumulus Linux, the industry's first, full-featured Linux operating system for networking hardware. Cumulus RMP is a network operating system solution that enables out-of-band management use cases. It provides an open platform for customers and system integrators to use as is or build rack management applications on top.

Cumulus RMP shares the same architecture, foundation and user experience with Cumulus Linux. The feature sets are also customized to the needs of out-of-band management.



Cumulus® RMP™ Rack Management Platform

This documentation is current as of June 7, 2016 for version 3.0.0. Please visit the [Cumulus Networks Web site](#) for the most up to date documentation.

Read the [release notes](#) for new features and known issues in this release.

- [Release Notes for Cumulus RMP 3.0.0](#) (see page 5)
- [Quick Start Guide](#) (see page 5)
- [Installation, Upgrading and Package Management](#)
- [System Management and Diagnostics](#) (see page 13)
- [Configuring and Managing Network Interfaces](#)
- [Layer 2 Features](#) (see page 90)
- [Routing](#) (see page 157)

Quick Start Guide

This chapter helps you get up and running with Cumulus RMP quickly and easily.

Contents

(Click to expand)

- [Contents \(see page 5\)](#)
- [What's New in Cumulus RMP 3.0.0 \(see page 5\)](#)
- [Open Source Contributions \(see page 5\)](#)
- [Prerequisites \(see page 6\)](#)
- [Supported Hardware \(see page 6\)](#)
- [Setting up a Cumulus RMP Switch \(see page 6\)](#)
- [Upgrading Cumulus RMP \(see page 6\)](#)
- [Configuring Cumulus RMP \(see page 6\)](#)
 - [Login Credentials \(see page 7\)](#)
 - [Serial Console Management \(see page 7\)](#)
 - [Wired Ethernet Management \(see page 7\)](#)
 - [In-Band Ethernet Management \(see page 7\)](#)
 - [Configuring the Hostname and Time Zone \(see page 8\)](#)
 - [Testing Cable Connectivity \(see page 8\)](#)
- [Configuring Switch Ports \(see page 9\)](#)
 - [Layer 2 Port Configuration \(see page 9\)](#)
 - [Layer 3 Port Configuration \(see page 10\)](#)
- [Configuring a Loopback Interface \(see page 11\)](#)

What's New in Cumulus RMP 3.0.0

Cumulus RMP 3.0.0 supports these [hardware platforms](#). The [release notes](#) contain information about the new features and known issues in this release.

Open Source Contributions

Cumulus Networks has forked various software projects, like CFEngine, Netdev and some Puppet Labs packages in order to implement various Cumulus RMP features. The forked code resides in the Cumulus Networks [GitHub repository](#).

Cumulus Networks developed and released as open source some new applications as well.

The list of open source projects is on the [open source software](#) page.

Prerequisites

Prior intermediate Linux knowledge is assumed for this guide. You should 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, you should use a Linux environment like [Cygwin](#) as your command line tool for interacting with Cumulus RMP.



If you're a networking engineer but are unfamiliar with Linux concepts, use [this reference guide](#) to see examples of the Cumulus RMP CLI and configuration options, and their equivalent Cisco Nexus 3000 NX-OS commands and settings for comparison. You can also [watch a series of short videos](#) introducing you to Linux in general and some Cumulus Linux-specific concepts in particular.

Supported Hardware

You can find the most up to date list of supported switches [here](#). Use this page to confirm that your switch model is supported by Cumulus Networks. The page is updated regularly, listing products by port configuration, manufacturer, and SKU part number.

Setting up a Cumulus RMP Switch

Setting up a Cumulus RMP switch is simple and straightforward. It involves:

1. Racking the switch and connecting it to power.
2. Cabling all the ports.
3. Logging in and changing the default password.
4. Configuring switch ports and a loopback interface, if needed.

This quick start guide walks you through the steps necessary for getting your Cumulus RMP switch up and running after you remove it from the box.

Upgrading Cumulus RMP

If you already have Cumulus RMP installed on your switch and are upgrading to a maintenance release (X.Y.Z, like 2.5.7) from an earlier release in the same major and minor release family **only** (like 2.5.4 to 2.5.7), you can use various methods, including `apt-get`, to upgrade to the new version instead. See [Upgrading Cumulus RMP \(see page 38\)](#) for details.

Configuring Cumulus RMP

When bringing up Cumulus RMP 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 should be located on the side of the switch or on the box in which the unit was shipped.

Login Credentials

The default installation includes one system account, *root*, with full system privileges, and one user account, *cumulus*, with sudo privileges. The *root* account password is set to null by default (which prohibits login), while the *cumulus* account is configured with this default password:

```
CumulusLinux!
```

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



For best security, you should change the default password (using the `passwd` command) before you configure Cumulus RMP on the switch.

All accounts except root are permitted remote SSH login; sudo may be used to grant a non-root account root-level access. Commands which change the system configuration require this elevated level of access.

For more information about sudo, read [Using sudo to Delegate Privileges](#) (see page 20).

Serial Console Management

Users are encouraged to perform management and configuration over the network, either in band or out of band. Use of the serial console is fully supported; however, many customers prefer the convenience of network-based management.

Typically, switches will 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 RMP contain a number of dedicated Ethernet management ports, the first of which is named *eth0*. These interfaces are geared specifically for out-of-band management use. The management interface uses DHCPv4 for addressing by default. While it is generally recommended to **not** assign an address to *eth0*, you can set a static IP address in the `/etc/network/interfaces` file:

```
auto eth0
iface eth0
    address 192.0.2.42/24
    gateway 192.0.2.1
```

In-Band Ethernet Management

All traffic that goes to the RMP switch via an interface called *vlan.1* is marked for in-band management. DHCP is enabled on this interface by default, and you can confirm the IP address at the command line. However, if you want to set a static IP address, change the configuration for *vlan.1* in `/etc/network/interfaces`:

```
auto vlan.1
iface vlan.1
    address 10.0.1.1/24
    gateway 10.0.2.1
```

Configuring the Hostname and Time Zone

To change the hostname, modify the `/etc/hostname` and `/etc/hosts` files with the desired hostname and reboot the switch. First, edit `/etc/hostname`:

```
cumulus@switch:~$ sudo vi /etc/hostname
```

Then replace the 127.0.1.1 IP address in `/etc/hosts` with the new hostname:

```
cumulus@switch:~$ sudo vi /etc/hosts
```

Reboot the switch:

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

To update the time zone, update the `/etc/timezone` file with the [correct timezone](#), run `dpkg-reconfigure --frontend noninteractive tzdata`, then reboot the switch:

```
cumulus@switch:~$ sudo vi /etc/timezone
cumulus@switch:~$ sudo dpkg-reconfigure --frontend noninteractive tzdata
cumulus@switch:~$ sudo reboot
```



It is possible to change the hostname without a reboot via a script available on [Cumulus Networks GitHub site](#).

Testing Cable Connectivity

By default, all data plane ports and the management interface are enabled.

To test cable connectivity, administratively enable a port using `ip link set <interface> up`:


```
cumulus@switch:~$ sudo ip link set swp1 up
```

To view link status, use `ip link show`. The following examples show the output of a port in "admin down", "down" and "up" mode, respectively:

```
# Administratively Down
swp1: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast state DOWN mode
DEFAULT qlen 1000

# Administratively Up but Layer 2 protocol is Down
swp1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast state
DOWN mode DEFAULT qlen 500

# Administratively Up, Layer 2 protocol is Up
swp1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
mode DEFAULT qlen 500
```

Configuring Switch Ports

Layer 2 Port Configuration

By default, all the front panel ports (swp1 through swp52) are members of a bridge called *vlan*, as seen in `/etc/network/interfaces` below. The `glob` keyword is used to put the complete range of ports into the bridge:

```
# The loopback network interface
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet dhcp

auto vlan
iface vlan
    bridge-vlan-aware yes
    # needs to scale to large port count
    bridge-ports glob swp1-52
    bridge-stp on

auto vlan.1
# update with v6 configuration
iface vlan.1 inet dhcp
```

If you modify the configuration at all, you need to activate or apply the configuration to the kernel:

```
# First, check for typos:
cumulus@switch:~$ sudo ifquery -a

# Then activate the change if no errors are found:
cumulus@switch:~$ sudo ifup -a
```

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
```



A script is available to generate a configuration that [places all physical ports in a single bridge](#).

Layer 3 Port Configuration

To configure a front panel port or bridge interface as a Layer 3 port, edit the `/etc/network/interfaces` file.

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

```
auto swp1
iface swp1
    address 10.1.1.1/30
```

To add an IP address to a bridge interface, remove the DHCP statement from the existing interface and include the address under the `iface` configuration in `/etc/network/interfaces`:

```
auto vlan.1
iface vlan.1
    address 10.2.2.1/24
```

To activate or apply the configuration to the kernel:

```
# First check for typos:
cumulus@switch:~$ sudo ifquery -a

# Then activate the change if no errors are found:
cumulus@switch:~$ sudo ifup -a
```

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

```
vlan.1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
link/ether 00:02:00:00:00:28 brd ff:ff:ff:ff:ff:ff
inet 10.2.2.1/24 scope global br0

swp1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
link/ether 44:38:39:00:6e:fe brd ff:ff:ff:ff:ff:ff
inet 10.1.1.1/30 scope global swp1
```

Configuring a Loopback Interface

Cumulus RMP has a loopback preconfigured in `/etc/network/interfaces`. 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.

To see the status of the loopback interface (`lo`), use the `ip addr show lo` command:

```
cumulus@switch:~$ ip addr show lo
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
```

Note that the loopback is up and is assigned an IP address of 127.0.0.1.

To add an IP address to a loopback interface, add it directly under the `iface lo inet loopback` definition in `/etc/network/interfaces`:

```
auto lo
iface lo inet loopback
    address 10.1.1.1
```



If an IP address is configured without a mask, as shown above, the IP address becomes a /32. So, in the above case, 10.1.1.1 is actually 10.1.1.1/32.

Multiple loopback addresses can be configured by adding additional `address` lines:

```
auto lo
iface lo inet loopback
    address 10.1.1.1
    address 172.16.2.1/24
```

System Management

Setting Date and Time

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

Contents

(Click to expand)

- [Contents \(see page 13\)](#)
- [Commands \(see page 13\)](#)
- [Setting the Time Zone \(see page 13\)](#)
- [Setting the Date and Time \(see page 14\)](#)
- [Setting Time Using NTP \(see page 15\)](#)
- [Specifying the NTP Source Interface \(see page 16\)](#)
- [Configuration Files \(see page 16\)](#)
- [Useful Links \(see page 16\)](#)

Commands

- `date`
- `dpkg-reconfigure tzdata`
- `hwclock`
- `ntpd (daemon)`
- `ntpq`

Setting the Time Zone

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

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

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

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

Then navigate the menus to enable the time zone you want. The following example selects the *US/Pacific* time zone:

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

Configuring tzdata
-----

Please select the geographic area in which you live. Subsequent
configuration
questions will narrow this down by presenting a list of cities, representing
the time zones in which they are located.

    1. Africa          4. Australia  7. Atlantic  10. Pacific  13. Etc
    2. America         5. Arctic    8. Europe   11. SystemV
    3. Antarctica      6. Asia      9. Indian   12. US
Geographic area: 12

Please select the city or region corresponding to your time zone.

    1. Alaska      4. Central  7. Indiana-Starke  10. Pacific
    2. Aleutian   5. Eastern  8. Michigan        11. Pacific-New
    3. Arizona    6. Hawaii  9. Mountain        12. Samoa
Time zone: 10

Current default time zone: 'US/Pacific'
Local time is now:      Mon Jun 17 09:27:45 PDT 2013.
Universal Time is now:  Mon Jun 17 16:27:45 UTC 2013.
```

For more info see the Debian [System Administrator's Manual – Time](#).

Setting 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
```



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 if you need 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)` if you need more information.

You can find a good overview of the software and hardware clocks in the Debian [System Administrator's Manual – Time](#), specifically the section [Setting and showing hardware clock](#).

Setting Time Using NTP

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

By default, `/etc/ntp.conf` contains some default time servers. Edit `/etc/ntp.conf` to add or update time server information. See `man ntp.conf(5)` for details on configuring `ntpd` using `ntp.conf`.

To set the initial date and time via NTP before starting the `ntpd` daemon, use `ntpd -q` (This is same as `ntpdate`, which is to be retired and not available).



`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:

```

cumulus@switch:~$ ntpq -p      remote      refid      st t when poll
reach  delay  offset  jitter
=====
==
*level1f.cs.unc. .PPS.          1 u  225 1024  377   92.505   -1.296
1.139
+ip.tcp.lv       193.11.166.8    2 u   29 1024  377  192.701    2.424
1.227
-host-86.3.217.2 131.107.13.100  2 u 1024 1024  367  240.622   11.250
7.785
+li290-38.member 128.138.141.172 2 u  553 1024  377   38.944   -0.810
1.139

```

Specifying the NTP Source Interface

You can change the source interface that NTP uses if you want to use something other than the default of eth0. Edit `ntp.conf` and edit the entry under the **# Specify interfaces** comment:

```

# Specify interfaces
interface listen bridge10

```

Configuration Files

- `/etc/default/ntp` — `ntpd` `init.d` configuration variables
- `/etc/ntp.conf` — default NTP configuration file
- `/etc/init.d/ntp` — `ntpd` `init` script

Useful Links

- Debian System Administrator's Manual – Time
- <http://www.ntp.org>
- http://en.wikipedia.org/wiki/Network_Time_Protocol
- <http://wiki.debian.org/NTP>

Authentication, Authorization, and Accounting

- SSH for Remote Access (see page 17)
- User Accounts (see page 18)
- Using `sudo` to Delegate Privileges (see page 20)

- [PAM and NSS \(see page 26\)](#)

SSH for Remote Access

You use **SSH** to securely access a Cumulus RMP switch remotely.

Contents

(Click to expand)

- [Contents \(see page 17\)](#)
- [Access Using Passkey \(Basic Setup\) \(see page 17\)](#)
 - [Completely Passwordless System \(see page 18\)](#)
- [Useful Links \(see page 18\)](#)

Access Using Passkey (Basic Setup)

Cumulus RMP uses the openSSH package to provide SSH functionality. The standard mechanisms of generating passwordless access just applies. The example below has the cumulus user on a machine called management-station connecting to a switch called *cumulus-switch1*.

First, on management-station, generate the SSH keys:

```
cumulus@management-station:~$ 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:
8c:47:6e:00:fb:13:b5:07:b4:1e:9d:f4:49:0a:77:a9 cumulus@management-
station
The key's randomart image is:
+--[ RSA 2048 ]-----+
|      .  . = o o .    |
|      o . O * . .    |
|      . o = = . o     |
|      . O o E         |
|      + S             |
|      +               |
|                      |
|                      |
|                      |
+-----+

```

Next, append the public key in `~/.ssh/id_rsa.pub` into `~/.ssh/authorized_keys` in the target user's home directory:

```
cumulus@management-station:~$ scp ~/.ssh/id_rsa.pub cumulus@cumulus-switch1:~/.ssh/authorized_keys
Enter passphrase for key '/home/cumulus/.ssh/id_rsa':
id_rsa.pub
```



Remember, you cannot use the root account to SSH to a switch in Cumulus RMP.

Completely Passwordless System

When generating the passphrase and its associated keys, as in the first step above, do not enter a passphrase. Follow all the other instructions.

Useful Links

- <http://www.debian-administration.org/articles/152>

User Accounts

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

The *cumulus* account:

- Default password is *CumulusLinux!*
- Is a user account in the *sudo* group with sudo privileges
- User can log in to the system via all the usual channels like console and SSH (see page 17)

The *root* account:

- Default password is 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 forth

For best security, you should change the default password (using the `passwd` command) before you configure Cumulus RMP on the switch.

You can add more user accounts as needed. Like the *cumulus* account, these accounts must use `sudo` to execute privileged commands (see page 20), so be sure to include them in the *sudo* group.

To access the switch without any password requires booting into a [single shell/user mode](#) (see page 165).

Enabling Remote Access for the root User

As mentioned above, the root user does not have a password set for it, and it cannot log in to a switch via SSH. This default account behavior is consistent with Debian. In order to connect to a switch using the root account, you can do one of two things for the account:

- Generate an SSH key

- Set a password

Generating an SSH Key for the root Account

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

```
$ ls -al ~/.ssh/
```

The key is named something like `id_dsa.pub`, `id_rsa.pub` or `id_ecdsa.pub`.

2. If a key doesn't exist, generate a new one by first creating the RSA key pair:

```
$ ssh-keygen -t rsa
```

3. A prompt appears, asking you to *Enter file in which to save the key (/home/root/.ssh/id_rsa):*. Press Enter to use the root user's home directory, or else provide a different destination.
4. You are prompted to *Enter 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 `/home/root/.ssh/id_rsa.pub`. The private key (identification) is now located in `/home/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
```

Setting the root User Password

1. Run:
cumulus@switch:~\$ sudo passwd root

2. Change the `/etc/ssh/sshd_config` file's `PermitRootLogin` setting from *without-password* to *yes*

```
cumulus@switch:~$ sudo vi /etc/ssh/sshd_config

...

# Authentication:
LoginGraceTime 120
PermitRootLogin yes
StrictModes yes

...
```

3. Restart the `ssh` service:

```
cumulus@switch:~$ sudo systemctl reload ssh
```

Using sudo to Delegate Privileges

By default, Cumulus RMP 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.

Contents

(Click to expand)

- [Contents \(see page 20\)](#)
- [Commands \(see page 20\)](#)
- [Using sudo \(see page 21\)](#)
- [sudoers Examples \(see page 21\)](#)
- [Configuration Files \(see page 26\)](#)
- [Useful Links \(see page 26\)](#)

Commands

- `sudo`
- `visudo`

Using sudo

`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.



Use `visudo` only to edit the `sudoers` file; do not use another editor like `vi` or `emacs`. See `man visudo(8)` for details.

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)`).

Any command can be run 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:

cumulus@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 info	<code>ethtool -m swp1</code>	<code>%noc ALL=(ALL) NOPASSWD: /sbin/ethtool</code>
Monitoring	System diagnostics	<code>cl-support</code>	<code>%noc ALL=(ALL) NOPASSWD: /usr /cumulus/bin/cl-support</code>
Monitoring	Routing diagnostics	<code>cl-resource-query</code>	<code>%noc ALL=(ALL) NOPASSWD: /usr /cumulus/bin/cl-resource-query</code>
Image management	Install images	<code>onie-select http://lab /install.bin</code>	<code>%noc ALL=(ALL) NOPASSWD: /usr /cumulus/bin/cl-img-install</code>
Package management	Any apt-get command	<code>apt-get update or apt-get install</code>	<code>%noc ALL=(ALL) NOPASSWD: /usr /bin/apt-get</code>
Package management	Just apt-get update	<code>apt-get update</code>	<code>%noc ALL=(ALL) NOPASSWD: /usr /bin/apt-get update</code>
Package management	Install packages		

Category	Privilege	Example Command	sudoers Entry
		<pre>apt-get install mtr-tiny</pre>	<pre>%noc ALL=(ALL) NOPASSWD:/usr /bin/apt-get install *</pre>
Package management	Upgrading	<pre>apt-get upgrade</pre>	<pre>%noc ALL=(ALL) NOPASSWD:/usr /bin/apt-get upgrade</pre>
L1 + 2 features	Any LLDP command	<pre>lldpcli show neighbors / configure</pre>	<pre>%noc ALL=(ALL) NOPASSWD:/usr /sbin/lldpcli</pre>
L1 + 2 features	Just show neighbors	<pre>lldpcli show neighbors</pre>	<pre>%noc ALL=(ALL) NOPASSWD:/usr /sbin/lldpcli show neighbours*</pre>
Interfaces	Modify any interface	<pre>ip link set dev swp1 {up down}</pre>	<pre>%noc ALL=(ALL) NOPASSWD: /sbin/ip link set *</pre>
Interfaces	Up any interface	<pre>ifup swp1</pre>	<pre>%noc ALL=(ALL) NOPASSWD: /sbin/ifup</pre>
Interfaces	Down any interface	<pre>ifdown swp1</pre>	<pre>%noc ALL=(ALL) NOPASSWD: /sbin/ifdown</pre>

Category	Privilege	Example Command	sudoers Entry
Interfaces	Up/down only swp2	<pre>ifup swp2 / ifdown swp2</pre>	<pre>%noc ALL=(ALL) NOPASSWD: /sbin/ifup swp2,/sbin /ifdown swp2</pre>
Interfaces	Any IP address chg	<pre>ip addr {add del} 192.0.2.1/30 dev swp1</pre>	<pre>%noc ALL=(ALL) NOPASSWD: /sbin/ip addr *</pre>
Interfaces	Only set IP address	<pre>ip addr add 192.0.2.1/30 dev swp1</pre>	<pre>%noc ALL=(ALL) NOPASSWD: /sbin/ip addr add *</pre>
Ethernet bridging	Any bridge command	<pre>brctl addbr br0 / brctl delif br0 swp1</pre>	<pre>%noc ALL=(ALL) NOPASSWD: /sbin/brctl</pre>
Ethernet bridging	Add bridges and ints	<pre>brctl addbr br0 / brctl addif br0 swp1</pre>	<pre>%noc ALL=(ALL) NOPASSWD: /sbin/brctl addbr *,/sbin /brctl addif *</pre>
Spanning tree	Set STP properties	<pre>mstpctl setmaxage br2 20</pre>	<pre>%noc ALL=(ALL) NOPASSWD: /sbin/mstpctl</pre>

Category	Privilege	Example Command	sudoers Entry
Troubleshooting	Restart switchd	<pre>systemctl restart switchd. service</pre>	<pre>%noc ALL=(ALL) NOPASSWD:/usr /sbin/service switchd *</pre>
Troubleshooting	Restart any service	<pre>systemctl cron switchd.service</pre>	<pre>%noc ALL=(ALL) NOPASSWD:/usr /sbin/service</pre>
Troubleshooting	Packet capture	<pre>tcpdump</pre>	<pre>%noc ALL=(ALL) NOPASSWD:/usr /sbin/tcpdump</pre>
L3	Add static routes	<pre>ip route add 10.2.0.0/16 via 10.0.0.1</pre>	<pre>%noc ALL=(ALL) NOPASSWD:/bin /ip route add *</pre>
L3	Delete static routes	<pre>ip route del 10.2.0.0/16 via 10.0.0.1</pre>	<pre>%noc ALL=(ALL) NOPASSWD:/bin /ip route del *</pre>
L3	Any static route chg	<pre>ip route *</pre>	<pre>%noc ALL=(ALL) NOPASSWD:/bin /ip route *</pre>
L3	Any iproute command	<pre>ip *</pre>	

Category	Privilege	Example Command	sudoers Entry
			<pre>%noc ALL=(ALL) NOPASSWD: /bin /ip</pre>

Configuration Files

- /etc/sudoers - default security policy
- /etc/sudoers.d/ - default security policy

Useful Links

- [sudo](#)
- [Adding Yourself to sudoers](#)

LDAP Authentication and Authorization

Cumulus RMP uses Pluggable Authentication Modules (PAM) and Name Switch Service (NSS) for user authentication.

NSS provides the lookup and mapping of users, while PAM provides login handling, authentication and session setup.

PAMs can be used with protocols like LDAP to provide user authentication for numerous services on a network.

Contents

(Click to expand)

- [Contents \(see page 26\)](#)
- [Configuring LDAP \(see page 27\)](#)
- [Installing libnss-Idapd \(see page 27\)](#)
- [Configuring nslcd.conf \(see page 27\)](#)
- [Troubleshooting LDAP Authentication \(see page 28\)](#)
 - [Using LDAP Debug Mode \(see page 28\)](#)
 - [Common Problems \(see page 29\)](#)
- [Configuring LDAP Authorization \(see page 30\)](#)
- [A Longer Example \(see page 31\)](#)
- [References \(see page 31\)](#)

Configuring LDAP

There are 3 common ways of configuring LDAP authentication on Linux:

- `libnss-ldap`
- `libnss-ldapd`
- `libnss-sss`

This chapter covers using `libnss-ldapd` only. From internal testing, this library worked best with Cumulus RMP and was the easiest to configure, automate and troubleshoot.

Installing libnss-ldapd



The `libnss-ldapd` and `ldap-utils` packages are not available in the Cumulus Networks repository. You must install them from the Debian repository. You need to configure the switch to reference the Debian repository. To do so, edit the `/etc/apt/sources.list` file and adding the following line:

```
deb http://ftp.us.debian.org/debian/ wheezy main
```

Then run `apt-get update` to sync with the Debian repo.

Once the Debian repository is referenced, install `libnss-ldapd`. Run:

```
cumulus@switch:~$ sudo apt-get install libnss-ldapd ldap-utils
```

This brings up an interactive prompt asking questions about the LDAP URI, base domain name and so on. To pre-fill these details, run `apt-get install debconf-utils` and populate `debconf-set-selections` with the appropriate answers. Run `debconf-show <pkg>` to check the settings.

Here is an [example of how to prefill questions using debconf-set-selections](#).



For nested group support, `libnss-ldapd` must be version 0.9 or higher. For Cumulus RMP, you can get this from the [jessie-backports](#) repo.

Configuring nslcd.conf

`/etc/nslcd.conf` is the main configuration file that needs to be changed after the package is installed. The `nslcd.conf` [man page](#) details all the available configuration options.

Here is an [example configuration](#) using Cumulus RMP.

Troubleshooting LDAP Authentication

By default, password and group information is cached by the `nscd` daemon. It is recommended when setting up LDAP authentication for the first time, to turn off this service using `systemctl stop nscd.service`.

Stop the `nsld` service 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
```

Using LDAP Debug Mode

Once you enable debug mode, run the following command to test LDAP queries:

```
cumulus@switch:~$ sudo 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: [8elf29] DEBUG: connection from pid=11766 uid=0 gid=0
nslcd: [8elf29] <passwd(all)> DEBUG: myldap_search(base="dc=cumulusnetworks,
dc=com", filter="(objectClass=posixAccount)")
nslcd: [8elf29] <passwd(all)> DEBUG: ldap_result(): uid=acc,ou=people,
dc=cumulusnetworks,dc=com
nslcd: [8elf29] <passwd(all)> (re)loading /etc/nsswitch.conf
nslcd: [8elf29] <passwd(all)> DEBUG: ldap_result(): ... 152 more results
nslcd: [8elf29] <passwd(all)> DEBUG: ldap_result(): end of results (162
total)
```

In the output above, `<passwd(all)>` indicates that the entire directory structure will be queried.

A specific user can be queried using the command:

```
cumulus@switch:~$ sudo getent passwd cumulus
```

You can replace `cumulus` with any username on the switch. The following debug output indicates that user `cumulus` 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="cumulus"> DEBUG: myldap_search(base="
dc=cumulusnetworks,dc=com", filter="(&(objectClass=posixAccount)
(uid=cumulus))")
nslcd: [8b4567] <passwd="cumulus"> DEBUG: ldap_initialize
(ldap://<ip_address>)
nslcd: [8b4567] <passwd="cumulus"> DEBUG: ldap_set_rebind_proc()
nslcd: [8b4567] <passwd="cumulus"> DEBUG: ldap_set_option
(LDAP_OPT_PROTOCOL_VERSION,3)
nslcd: [8b4567] <passwd="cumulus"> DEBUG: ldap_set_option(LDAP_OPT_DEREF,0)
nslcd: [8b4567] <passwd="cumulus"> DEBUG: ldap_set_option
(LDAP_OPT_TIMELIMIT,0)
nslcd: [8b4567] <passwd="cumulus"> DEBUG: ldap_set_option(LDAP_OPT_TIMEOUT,
0)
nslcd: [8b4567] <passwd="cumulus"> DEBUG: ldap_set_option
(LDAP_OPT_NETWORK_TIMEOUT,0)
nslcd: [8b4567] <passwd="cumulus"> DEBUG: ldap_set_option
(LDAP_OPT_REFERRALS,LDAP_OPT_ON)
nslcd: [8b4567] <passwd="cumulus"> DEBUG: ldap_set_option(LDAP_OPT_RESTART,
LDAP_OPT_ON)
nslcd: [8b4567] <passwd="cumulus"> DEBUG: ldap_simple_bind_s(NULL,NULL)
(uri="ldap://<ip_address>")
nslcd: [8b4567] <passwd="cumulus"> DEBUG: ldap_result(): end of results (0
total)

```

Notice how the `<passwd="cumulus">` shows that the specific *cumulus* user was queried.

Common Problems

- `nslcd` cannot read the SSL certificate. `nslcd` will report a “Permission denied” error in the debug during server connection negotiation. The sniffer trace output will show only a TCP handshake and then a TCP FIN from the switch. Check the permission on each directory in the path of the root SSL certificate. Ensure that it is readable by the `nslcd` user.
- The FQDN on the LDAP URI does not match the SSL FQDN exactly.
- The search filter returns wrong results. Check for typos in the search filter. Use `ldapsearch` to test your filter. For example:

```

In $HOME/.ldaprc configure basic ldapsearch parameters
-----
URI: ldaps://myadserver.rtp.example.test
BASE ou=support,dc=rtp,dc=example,dc=test
TLS_CACERT /etc/ssl/certs/rtp-example-ca.crt
-----

# ldapsearch \
-D 'CN=cumulus admin,CN=Users,DC=rtp,DC=example,DC=test' \
-w 'lQ2w3e4r!' \
"(&(ObjectClass=user) \
(memberOf=cn=cumuluslnxadm,ou=groups,ou=support,dc=rtp,
dc=example, dc=test))"

```

- When a local user exists, the order of the output of `/etc/nsswitch` needs to be updated to query LDAP before the local user database. For example, the configuration below ensures that LDAP is queried before the local database; the default is the opposite.

```

# /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:          ldap compat

```

Configuring LDAP Authorization

In the `/etc/nslcd.conf` file, the "filter" keyword defines an LDAP search filter. Use this search filter to only show the users and or groups one desires. In the example below, only users in the `cumuluslnxadm` group are shown in the passed database:

```

# This filter says to get all users who are part of the cumuluslnxadm group.
filter passwd (&(Objectclass=user)(!(objectClass=computer))
(memberOf=cn=cumuluslnxadm,ou=groups,ou=support,dc=rtp,dc=example,dc=test))

```

A Longer Example

A longer, more complete example for configuring LDAP is available on our [knowledge base](#).

References

- <https://wiki.debian.org/LDAP/PAM>
- <https://raw.githubusercontent.com/arthurdejong/nss-pam-ldapd/master/nslcd.conf>
- <http://backports.debian.org/Instructions/>

Managing Application Daemons

You manage application daemons in Cumulus RMP in the following ways:

- Identifying active listener ports
- Identifying daemons currently active or stopped
- Identifying boot time state of a specific daemon
- Disabling or enabling a specific daemon

Using *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.

```
systemctl start | stop | restart | status | reload | enable | disable |
reenable SERVICENAME.service
```

For example to restart networking, run the command:

```
systemctl restart networking.service
```



Unlike the *service* command in Debian Wheezy, the service name is written **after** the *systemctl* subcommand, not before it.

Understanding the *systemctl* Subcommands

systemctl has a number of subcommands that perform a specific operation on a given daemon.

- **status**: Returns the status of the specified daemon.
- **start**: Starts the daemon.
- **stop**: Stops the daemon.

- **restart:** Stops, then starts the daemon, all the while maintaining state. So if there are dependent services or services that mark the restarted service as *Required*, the other services also get restarted. For example, running `systemctl restart zebra` restarts any of the routing protocol daemons that are enabled and running, such as `bgpd` or `ospfd`.
- **reload:** Reloads a daemon's configuration.
- **enable:** Enables the daemon 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 daemon, but does not stop it unless you use the `systemctl stop SERVICENAME.service` command or reboot the switch. A disabled daemon can still be started or stopped.
- **reenable:** Disables, then enables a daemon. 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 daemons.

Ensuring 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 for this are stored in the service script. The settings are `StartLimitInterval` (which defaults to 10 seconds) and `StartBurstLimit` (which defaults to 5 attempts), but many services override these defaults, sometimes with much longer times. `switchd.service`, for example, sets `StartLimitInterval=10m` and `StartBurstLimit=3`, which means if you restart `switchd` more than 3 times in 10 minutes, it will not start.

When the restart fails for this reason, a message similar to the following appears:

```
Job for switchd.service failed. See 'systemctl status switchd.service' and
'journalctl -xn' for details.
```

And `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.

Contents

(Click to expand)

- [Using systemd and the systemctl Command \(see page 31\)](#)
 - [Understanding the systemctl Subcommands \(see page 31\)](#)
 - [Ensuring a Service Starts after Multiple Restarts \(see page 32\)](#)
- [Contents \(see page 32\)](#)

- Identifying Active Listener Ports for IPv4 and IPv6 (see page 33)
- Identifying Daemons Currently Active or Stopped (see page 33)
- Identifying Boot Time State of a Specific Daemon (see page 34)

Identifying Active Listener Ports for IPv4 and IPv6

You can identify the active listener ports under both IPv4 and IPv6 using the `lsof` command:

```
cumulus@switch:~$ sudo lsof -Pnl +M -i4
COMMAND PID USER FD TYPE DEVICE SIZE/OFF NODE NAME
ntpd 1882 104 16u IPv4 3954 0t0 UDP *:123
ntpd 1882 104 18u IPv4 3963 0t0 UDP 127.0.0.1:123
ntpd 1882 104 19u IPv4 3964 0t0 UDP 192.168.8.37:123
snmpd 1987 105 8u IPv4 5423 0t0 UDP *:161
zebra 1993 103 10u IPv4 5151 0t0 TCP 127.0.0.1:2601 (LISTEN)
sshd 2496 0 3u IPv4 5809 0t0 TCP *:22 (LISTEN)
jdoo 2622 0 6u IPv4 6132 0t0 TCP 127.0.0.1:2812 (LISTEN)
sshd 31700 0 3r IPv4 187630 0t0 TCP 192.168.8.37:22->192.168.8.3:50386
(ESTABLISHED)

cumulus@switch:~$ sudo lsof -Pnl +M -i6
COMMAND PID USER FD TYPE DEVICE SIZE/OFF NODE NAME
ntpd 1882 104 17u IPv6 3955 0t0 UDP *:123
ntpd 1882 104 20u IPv6 3965 0t0 UDP [::1]:123
ntpd 1882 104 21u IPv6 3966 0t0 UDP [fe80::7272:cfff:fe96:6639]:123
sshd 2496 0 4u IPv6 5811 0t0 TCP *:22 (LISTEN)
```

Identifying Daemons Currently Active or Stopped

To determine which daemons are currently active or stopped, run `cl-service-summary`:

```
cumulus@switch:~$ sudo cl-service-summary
Service cron          enabled    active
Service ssh           enabled    active
Service syslog        enabled    active
Service arp_refresh   enabled    active
Service clagd         enabled    active
Service lldpd         enabled    active
Service mstpd         enabled    active
Service poed          enabled    inactive
Service portwd        enabled    inactive
Service ptmd          enabled    active
Service pwmd          enabled    active
```

Service smond	enabled	active
Service switchd	enabled	active
Service vxrd	disabled	inactive
Service vxsnd	disabled	inactive
Service bgpd	disabled	inactive
Service isisd	disabled	inactive
Service ospf6d	disabled	inactive
Service ospfd	disabled	inactive
Service rdnbrd	disabled	inactive
Service ripd	disabled	inactive
Service ripngd	disabled	inactive
Service zebra	disabled	inactive

Another way to get this information is to use the `systemctl status` command, then pipe the results to `grep`, using the `-` or `+` operators:

```
cumulus@switch:~$ sudo systemctl status | grep +
[ ? ] aclinit
[ + ] arp_refresh
[ + ] auditd
...

cumulus@switch:~$ sudo systemctl status | grep -
[ - ] isc-dhcp-server
[ - ] openvswitch-vtep
[ - ] ptmd
...
```

Identifying Boot Time State of a Specific Daemon

The `ls` command can provide the boot time state of a daemon. A file link with a name starting with **S** identifies a boot-time-enabled daemon. A file link with a name starting with **K** identifies a disabled daemon.

```
cumulus@switch:~/etc:~$ sudo ls -l rc*.d | grep <daemon name>
```

For example:

```
cumulus@switch:~/etc$ sudo ls -l rc*.d | grep snmpd
lrwxrwxrwx 1 root root 15 Apr 4 2014 K02snmpd -> ../init.d/snmpd
lrwxrwxrwx 1 root root 15 Apr 4 2014 K02snmpd -> ../init.d/snmpd
lrwxrwxrwx 1 root root 15 Apr 4 2014 S01snmpd -> ../init.d/snmpd
```

```
lrwxrwxrwx 1 root root 15 Apr 4 2014 S01snmpd -> ../init.d/snmpd
lrwxrwxrwx 1 root root 15 Apr 4 2014 S01snmpd -> ../init.d/snmpd
lrwxrwxrwx 1 root root 15 Apr 4 2014 S01snmpd -> ../init.d/snmpd
lrwxrwxrwx 1 root root 15 Apr 4 2014 K02snmpd -> ../init.d/snmpd
```

Configuring switchd

switchd is the daemon at the heart of Cumulus RMP. It communicates between the switch and Cumulus RMP, and all the applications running on Cumulus RMP.

The switchd configuration is stored in `/etc/cumulus/switchd.conf`.

Contents

(Click to expand)

- [Contents \(see page 35\)](#)
- [The switchd File System \(see page 35\)](#)
- [Configuring switchd Parameters \(see page 37\)](#)
- [Restarting switchd \(see page 37\)](#)
- [Commands \(see page 38\)](#)
- [Configuration Files \(see page 38\)](#)

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. You can see the contents by parsing the switchd tree; run `tree /cumulus/switchd`. The output below is for a switch with one switch port configured:

```
cumulus@cumulus:~# sudo tree /cumulus/switchd/
/cumulus/switchd/
|-- config
|   |-- acl
|   |   |-- non_atomic_update_mode
|   |   `-- optimize_hw
|   |-- arp
|   |   `-- next_hops
|   |-- buf_util
|   |   |-- measure_interval
|   |   `-- poll_interval
|   |-- coalesce
|   |   |-- reducer
|   |   `-- timeout
|   |-- disable_internal_restart
```

```

| | -- ignore_non_swps
| | -- interface
| | | -- swp1
| | |   |-- storm_control
| | |     |-- broadcast
| | |     |-- multicast
| | |     |-- unknown_unicast
| | -- logging
| | -- route
| | | -- host_max_percent
| | | -- max_routes
| | |   |-- table
| | |-- stats
| | |   |-- poll_interval
|-- ctrl
| | -- acl
| | -- hal
| | |   |-- resync
| | -- logger
| | -- netlink
| | |   |-- resync
| | -- resync
| | |-- sample
| | |   |-- ulog_channel
|-- run
| | |-- route_info
| | |   |-- ecmp_nh
| | |     |-- count
| | |     |-- max
| | |     |-- max_per_route
| | |   |-- host
| | |     |-- count
| | |     |-- count_v4
| | |     |-- count_v6
| | |     |-- max
| | |   |-- mac
| | |     |-- count
| | |     |-- max
| | |   |-- route
| | |     |-- count_0
| | |     |-- count_1
| | |     |-- count_total
| | |     |-- count_v4
| | |     |-- count_v6
| | |     |-- mask_limit

```

```
|          |-- max_0
|          |-- max_1
|          `-- max_total
|-- version
```

Configuring switchd Parameters

You can use `cl-cfg` to configure many `switchd` parameters at runtime (like ACLs, interfaces, and route table utilization), which minimizes disruption to your running switch. However, some options are read only and cannot be configured at runtime.

For example, to see data related to routes, run:

```
cumulus@cumulus:~$ sudo cl-cfg -a switchd | grep route
route.table = 254
route.max_routes = 32768
route.host_max_percent = 50
cumulus@cumulus:~$
```

To modify the configuration, run `cl-cfg -w`. For example, to set the buffer utilization measurement interval to 1 minute, run:

```
cumulus@cumulus:~$ sudo cl-cfg -w switchd buf_util.measure_interval=1
```

To verify that the value changed, use `grep`:

```
cumulus@cumulus:~# cl-cfg -a switchd | grep buf
buf_util.poll_interval = 0
buf_util.measure_interval = 1
```



You can get some of this information by running `cl-resource-query`; though you cannot update the `switchd` configuration with it.

Restarting switchd

Whenever you modify your network configuration (typically changing any `*.conf` file, like `/etc/cumulus/datapath/traffic.conf`), you must restart `switchd` for the changes 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 (that is, edit `/etc/network/interfaces`).



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

Commands

- `cl-cfg`

Configuration Files

- `/etc/cumulus/switchd.conf`

Installation, Upgrading and Package Management

A Cumulus RMP switch can have only one image of the operating system installed. This section discusses installing new and updating existing Cumulus RMP disk images, and configuring those images with additional applications (via packages) if desired.

A Cumulus RMP switch comes pre-installed with the operating system.

Zero touch provisioning is a way to quickly deploy and configure new switches in a large-scale environment.

- [Managing Cumulus RMP Disk Images \(see page 38\)](#)
- [Adding and Updating Packages \(see page 46\)](#)
- [Zero Touch Provisioning \(see page 52\)](#)

Managing Cumulus RMP Disk Images

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

Cumulus RMP comes preinstalled on your switch. However there may be instances where you need to perform a full image installation. Before you install Cumulus RMP, the switch can be in two different states:

- The switch already has Cumulus RMP installed on it, so you only need to [upgrade it \(see page \)](#).
- The switch has no image on it (so the switch is only running **ONIE**) or you desire or require a clean installation. In which case, you would install Cumulus RMP in one of the following ways, using:
 - [DHCP/a Web server with DHCP options \(see page 39\)](#)
 - [DHCP/a Web server without DHCP options \(see page 40\)](#)
 - [A Web server with no DHCP \(see page 40\)](#)
 - [FTP or TFTP without a Web server \(see page 41\)](#)
 - [Local file installation \(see page 41\)](#)
 - [USB \(see page 42\)](#)



ONIE is an open source project, equivalent to PXE on servers, that allows installation of network operating systems (NOS) on bare metal switches.

Unlike Cumulus Linux, there is no license to install on a Cumulus RMP switch.

Understanding these Examples

The sections in this chapter are ordered from the most repeatable to the least repeatable methods. For instance, DHCP can scale to hundreds of switch installs with zero manual input, compared to something like USB installs. Installing via USB is fine for a single switch here and there but is not scalable.

You can name your Cumulus RMP installer binary using any of the [ONIE naming schemes mentioned here](#).

Contents

- [Understanding these Examples \(see page 39\)](#)
- [Installing via a DHCP/Web Server Method with DHCP Options \(see page 39\)](#)
- [Installing via a DHCP/Web Server Method without DHCP Options \(see page 40\)](#)
- [Installing via a Web Server with no DHCP \(see page 40\)](#)
- [Installing via FTP or TFTP without a Web Server \(see page 41\)](#)
- [Installing via a Local File \(see page 41\)](#)
- [Installing via USB \(see page 42\)](#)
 - [Preparing for USB Installation \(see page 42\)](#)
- [Upgrading Cumulus RMP \(see page 45\)](#)
- [Useful Links \(see page 46\)](#)

Installing via a DHCP/Web Server Method with DHCP Options

Installing Cumulus RMP in this manner is as simple as setting up a DHCP/Web server on your laptop and connecting the eth0 management port of the switch to your laptop.

Once you connect the cable, the installation proceeds as follows:

1. The bare metal switch boots up and asks for an 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 RMP binary, installs and reboots.
4. Success! You are now running Cumulus RMP.



The most common method is for you to send DHCP option 114 with the entire URL to the Web server (this could be the same system). However, there are many other ways to use DHCP even if you don't have full control over DHCP. [See the ONIE user guide](#) for help.

Here's 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 www-server = "http://172.0.24.14/onie-installer-[PLATFORM]";  
}
```

Here's 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]"
```

Don't have a Web server? There is a [free Apache example](#) you can utilize.

Installing via a DHCP/Web Server Method without DHCP Options

If you have a laptop on same network and the switch can pull DHCP from the corporate network, but you cannot modify DHCP options (maybe it's controlled by another team), do the following:

1. Place the Cumulus RMP binary in a directory on the Web server.
2. Run the `onie-nos-install` command manually, since DHCP options can't be modified:

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

Installing via a Web Server with no DHCP

Use the following method if your laptop is on the same network as the switch eth0 interface but no DHCP server is available.

One thing to note is ONIE is in [discovery mode](#), so if you are setting a static IPv4 address for the eth0 management port, you need to disable discovery mode or else ONIE may get confused.

1. To disable discovery mode, run:

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

or, on older ONIE versions if that command isn't supported:


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

2. Assign a static address to eth0 via ONIE (using `ip addr add`):

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

3. Place the Cumulus RMP installer image in a directory on your Web server.
4. Run the `onie-nos-install` command manually since there are no DHCP options:

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

Installing via FTP or TFTP without a Web Server

1. Set up DHCP or static addressing for eth0, as in the examples above.
2. If you are utilizing static addressing, disable ONIE discovery mode.
3. Place the Cumulus RMP installer image into a TFTP or FTP directory.
4. If you are not utilizing 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
```

Installing via a Local File

1. Set up DHCP or static addressing for eth0, as in the examples above.
2. If you are utilizing static addressing, disable ONIE discovery mode.
3. Use `scp` to copy the Cumulus RMP binary to the switch.
Note: Windows users can use [WinScp](#).
4. Run the following command:

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

Installing via USB

Follow the steps below to conduct a full installation of Cumulus RMP. This wipes out all pre-existing configuration files that may be present on the switch.



Make sure to back up any important configuration files that you may need to restore the configuration of your switch after the installation finishes.

Preparing for USB Installation

1. Download the Cumulus RMP image from the [Cumulus Downloads page](#).
2. Prepare your flash drive by formatting in one of the supported formats: FAT32, vFAT or EXT2.

Optional: Preparing a USB Drive inside Cumulus RMP



It is possible that you could severely damage your system with the following utilities, so please use caution when performing the actions below!

- a. Insert your flash drive into the USB port on the switch running Cumulus RMP and log in to the switch.
- b. Determine and note which device your flash drive can be found at using output from `cat /proc/partitions` and `sudo fdisk -l [device]`. For example, `sudo fdisk -l /dev/sdb`. These instructions assume your USB drive is the `/dev/sdb` device, which is typical. Make sure to modify the commands below to use the proper device for your USB drive.
- c. Create a new partition table on the device:

```
sudo parted /dev/sdb mklabel msdos
```



The `parted` utility should already be installed. However, if it is not, install it with:
`sudo apt-get install parted`

- d. Create a new partition on the device:

```
sudo parted /dev/sdb -a optimal mkpart primary 0% 100%
```

- e. Format the partition to your filesystem of choice using ONE of the examples below:

```
sudo mkfs.ext2 /dev/sdb1
sudo mkfs.msos -F 32 /dev/sdb1
sudo mkfs.vfat /dev/sdb1
```



To use `mkfs.msdos` or `mkfs.vfat`, you need to install the `dosfstools` package from the [Debian software repositories](#) (step 3 here shows you how to add repositories from Debian), as they are not included by default.

- f. To continue installing Cumulus RMP, mount the USB drive in order to move files to it.

```
sudo mkdir /mnt/usb
sudo mount /dev/sdb1 /mnt/usb
```

3. Copy the image file over to the flash drive and rename the image file to `onie-installer_x86-64`.



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 may still be present. Make sure to remove the file extension otherwise ONIE will not be able to detect the file!

4. Insert the USB stick into the switch, then 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 Cumulus RMP, connect to the console and reboot the switch into the ONIE environment with the `sudo onie-select -i` command, followed by `sudo reboot`. Then skip to step 8 below.
- If the switch is already online in ONIE, use the `reboot` command.



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

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

```

GNU GRUB  version 1.99-27+deb7u2

+-----+
|Cumulus Linux 2.5.3a-3b46bef-201509041633-build - slot 1|
|Cumulus Linux 2.5.3a-3b46bef-201509041633-build - slot 1 (recovery mode)|
|Cumulus Linux 2.5.3a-3b46bef-201509041633-build - slot 2|
|Cumulus Linux 2.5.3a-3b46bef-201509041633-build - slot 2 (recovery mode)|
|ONIE|
+-----+

Use the ^ and v keys to select which entry is highlighted.
Press enter to boot the selected OS, 'e' to edit the commands
before booting or 'c' for a command-line.

```

7. Cumulus RMP 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*.

```

GNU GRUB  version 2.02~beta2+e4a1fe391

+-----+
|*ONIE: Install OS|
| ONIE: Rescue|
| ONIE: Uninstall OS|
| ONIE: Update ONIE|
| ONIE: Embed ONIE|
+-----+

Use the ^ and v keys to select which entry is highlighted.
Press enter to boot the selected OS, 'e' to edit the commands
before booting or 'c' for a command-line.

```

8. At this point, the USB drive should be automatically recognized and mounted. The image file should be located and automatic installation of Cumulus RMP should begin. Here is some sample output:

```

ONIE: OS Install Mode  ...

Version : penguin_arctica-2014.05.05-6919d98-201410171013
Build  Date: 2014-10-17T10:13+0800
Info: Mounting kernel filesystems... done.
Info: Mounting LABEL=ONIE-BOOT on /mnt/onie-boot  ...
initializing eth0...
scsi 6:0:0:0: Direct-Access  SanDisk Cruzer Facet 1.26 PQ: 0
ANSI: 6
sd 6:0:0:0: [sdb] 31266816 512-byte logical blocks: (16.0 GB/14.9
GiB)
sd 6:0:0:0: [sdb] Write Protect is off
sd 6:0:0:0: [sdb] Write cache: disabled, read cache: enabled,
doesn't support DPO or FUA

```

```
sd 6:0:0:0: [sdb] Attached SCSI disk

<...snip...>

ONIE: Executing installer: file://dev/sdb1/onie-installer-x86_64
Verifying image checksum ... OK.
Preparing image archive ... OK.
Dumping image info...
Control File Contents
=====
Description: Cumulus Linux
OS-Release: 3.0.0-3b46bef-201509041633-build
Architecture: amd64
Date: Fri, 04 Sep 2015 17:10:30 -0700
Installer-Version: 1.2
Platforms: accton_as5712_54x accton_as6712_32x
mlx_sx1400_i73612 dell_s6000_s1220 dell_s4000_c2338
dell_s3000_c2338 cel_redstone_xp cel_smallstone_xp cel_pebble
quanta_panther quanta_ly8_rangeley quanta_ly6_rangeley
quanta_ly9_rangeley
Homepage: http://www.cumulusnetworks.com/
```

9. After installation completes, the switch automatically reboots into the newly installed instance of Cumulus RMP.
10. Determine and note at which device your flash drive can be found by using output from `cat /proc/partitions` and `sudo fdisk -l [device]`. For example, `sudo fdisk -l /dev/sdb`.



These instructions assume your USB drive is the `/dev/sdb` device, which is typical if the USB stick was inserted after the machine was already booted. However, if the USB stick was plugged in during the boot process, it is possible the device could be `/dev/sda`. Make sure to modify the commands below to use the proper device for your USB drive!

11. Create a mount point to mount the USB drive to:

```
sudo mkdir /mnt/mountpoint
```

12. Mount the USB drive to the newly created mount point:

```
sudo mount /dev/sdb1 /mnt/mountpoint
```

Upgrading Cumulus RMP

If you already have Cumulus RMP installed on your switch and you are upgrading to an X.Y.Z release, like 2.5.7 from an earlier release in the same major and minor release family **only** (like 2.5.4 to 2.5.7), you can use `apt-get` to upgrade to the new version. (If are upgrading to a major (X.0) or minor (X.Y) release, you must do a full image install, as described above.)

To upgrade to a maintenance (X.Y.Z) release using `apt-get`:

1. Run `apt-get update`.
2. Run `apt-get dist-upgrade`.
3. Reboot the switch.

Useful Links

- [Open Network Install Environment \(ONIE\) Home Page](#)

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.

Contents

(Click to expand)

- [Contents \(see page 46\)](#)
- [Commands \(see page 46\)](#)
- [Updating the Package Cache \(see page 46\)](#)
- [Listing Available Packages \(see page 48\)](#)
- [Adding a Package \(see page 49\)](#)
- [Listing Installed Packages \(see page 50\)](#)
- [Upgrading to Newer Versions of Installed Packages \(see page 50\)](#)
 - [Upgrading a Single Package \(see page 50\)](#)
 - [Upgrading All Packages \(see page 51\)](#)
- [Adding Packages from Another Repository \(see page 51\)](#)
- [Configuration Files \(see page 52\)](#)
- [Useful Links \(see page 52\)](#)

Commands

- `apt-get`
- `apt-cache`
- `dpkg`

Updating the Package Cache

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

```
cumulus@switch:~$ sudo apt-get update
Get:1 http://repo3.cumulusnetworks.com CumulusRMP-3 InRelease [7,624 B]
```

```
Get:2 http://repo3.cumulusnetworks.com CumulusRMP-3-security-updates
InRelease [7,555 B]
Get:3 http://repo3.cumulusnetworks.com CumulusRMP-3-updates InRelease
[7,660 B]
Get:4 http://repo3.cumulusnetworks.com CumulusRMP-3/cumulus Sources [20 B]
Get:5 http://repo3.cumulusnetworks.com CumulusRMP-3/upstream Sources [20 B]
Get:6 http://repo3.cumulusnetworks.com CumulusRMP-3/cumulus amd64 Packages
[38.4 kB]
Get:7 http://repo3.cumulusnetworks.com CumulusRMP-3/upstream amd64 Packages
[445 kB]
Get:8 http://repo3.cumulusnetworks.com CumulusRMP-3-security-updates
/cumulus Sources [20 B]
Get:9 http://repo3.cumulusnetworks.com CumulusRMP-3-security-updates
/upstream Sources [11.8 kB]
Get:10 http://repo3.cumulusnetworks.com CumulusRMP-3-security-updates
/cumulus amd64 Packages [20 B]
Get:11 http://repo3.cumulusnetworks.com CumulusRMP-3-security-updates
/upstream amd64 Packages [8,941 B]
Get:12 http://repo3.cumulusnetworks.com CumulusRMP-3-updates/cumulus
Sources [20 B]
Get:13 http://repo3.cumulusnetworks.com CumulusRMP-3-updates/upstream
Sources [776 B]
Get:14 http://repo3.cumulusnetworks.com CumulusRMP-3-updates/cumulus amd64
Packages [38.4 kB]
Get:15 http://repo3.cumulusnetworks.com CumulusRMP-3-updates/upstream amd64
Packages [444 kB]
Ign http://repo3.cumulusnetworks.com CumulusRMP-3/cumulus Translation-en_US
Ign http://repo3.cumulusnetworks.com CumulusRMP-3/cumulus Translation-en
Ign http://repo3.cumulusnetworks.com CumulusRMP-3/upstream Translation-en_US
Ign http://repo3.cumulusnetworks.com CumulusRMP-3/upstream Translation-en
Ign http://repo3.cumulusnetworks.com CumulusRMP-3-security-updates/cumulus
Translation-en_US
Ign http://repo3.cumulusnetworks.com CumulusRMP-3-security-updates/cumulus
Translation-en
Ign http://repo3.cumulusnetworks.com CumulusRMP-3-security-updates/upstream
Translation-en_US
Ign http://repo3.cumulusnetworks.com CumulusRMP-3-security-updates/upstream
Translation-en
Ign http://repo3.cumulusnetworks.com CumulusRMP-3-updates/cumulus
Translation-en_US
Ign http://repo3.cumulusnetworks.com CumulusRMP-3-updates/cumulus
Translation-en
Ign http://repo3.cumulusnetworks.com CumulusRMP-3-updates/upstream
Translation-en_US
Ign http://repo3.cumulusnetworks.com CumulusRMP-3-updates/upstream
```

```
Translation-en
Fetched 1,011 kB in 1s (797 kB/s)
Reading package lists... Done
```

Listing Available Packages

Once the cache is populated, use `apt-cache` to search the cache to find the packages you are interested in or to get information about an available package. Here are examples of the `search` and `show` sub-commands:

```
cumulus@switch:~$ apt-cache search tcp
fakeroot - tool for simulating superuser privileges
libwrap0 - Wietse Venema's TCP wrappers library
libwrap0-dev - Wietse Venema's TCP wrappers library, development files
netbase - Basic TCP/IP networking system
nmap - The Network Mapper
openbsd-inetd - OpenBSD Internet Superserver
openssh-client - secure shell (SSH) client, for secure access to remote
machines
openssh-server - secure shell (SSH) server, for secure access from remote
machines
rsyslog - reliable system and kernel logging daemon
socat - multipurpose relay for bidirectional data transfer
tcpd - Wietse Venema's TCP wrapper utilities
tcpdump - command-line network traffic analyzer
tcpreplay - Tool to replay saved tcpdump files at arbitrary speeds
tcpstat - network interface statistics reporting tool
tcptrace - Tool for analyzing tcpdump output
tcpxtract - extracts files from network traffic based on file signatures

cumulus@switch:~$ apt-cache show tcpreplay
Package: tcpreplay
Version: 3.4.3-2+wheezy1
Architecture: powerpc
Maintainer: Noël Köthe <noel@debian.org>
Installed-Size: 984
Depends: libc6 (>= 2.7), libpcap0.8 (>= 0.9.8)
Homepage: http://tcpreplay.synfin.net/
Priority: optional
Section: net
Filename: pool/main/t/tcpreplay/tcpreplay_3.4.3-2+wheezy1_powerpc.deb
Size: 435904
SHA256: 03dc29057cb608d2ddf08207aedef18d47988ed6c23db0af69d30746768a639ae
SHA1: 8eelb9b02dacd0c48a474844f4466eb54c7e1568
```



```
MD5sum: cf20bec7282ef77a091e79372a29fe1e
Description: Tool to replay saved tcpdump files at arbitrary speeds
Tcpreplay is aimed at testing the performance of a NIDS by
replaying real background network traffic in which to hide
attacks. Tcpreplay allows you to control the speed at which the
traffic is replayed, and can replay arbitrary tcpdump traces. Unlike
programmatically-generated artificial traffic which doesn't
exercise the application/protocol inspection that a NIDS performs,
and doesn't reproduce the real-world anomalies that appear on
production networks (asymmetric routes, traffic bursts/lulls,
fragmentation, retransmissions, etc.), tcpreplay allows for exact
replication of real traffic seen on real networks.
cumulus@switch:~$
```



The search commands look for the search terms not only in the package name but in other parts of the package information. Consequently, it will match on more packages than you would expect.

Adding a Package

In order to add a new package, first ensure the package is not already installed in the system:

```
cumulus@switch:~$ dpkg -l | grep {name of package}
```

If the package is installed already, ensure it's the version you need. If it's an older version, then update the package from the Cumulus RMP repository:

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

If the package is not already on the system, add it by running `apt-get install`. This retrieves the package from the Cumulus RMP repository and installs it on your system together with any other packages that this package might depend on.

For example, the following adds the package `tcpreplay` to the system:

```
cumulus@switch:~$ sudo 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.
Get:1 https://repo.cumulusnetworks.com/ CumulusRMP-2.5.3/main tcpreplay
powerpc 3.4.3-2+wheezy1 [436 kB]
Fetched 436 kB in 0s (1501 kB/s)
Selecting previously unselected package tcpreplay.
(Reading database ... 15930 files and directories currently installed.)
Unpacking tcpreplay (from .../tcpreplay_3.4.3-2+wheezy1_powerpc.deb) ...
Processing triggers for man-db ...
Setting up tcpreplay (3.4.3-2+wheezy1) ...
cumulus@switch:~$

```

Listing Installed Packages

The APT cache contains information about all the packages available on the repository. To see which packages are actually installed on your system, use `dpkg`. The following example lists all the packages on the system that have "tcp" in their package names:

```

cumulus@switch:~$ dpkg -l \*tcp\*
Desired=Unknown/Install/Remove/Purge/Hold
| Status=Not/Inst/Conf-files/Unpacked/halF-conf/Half-inst/trig-aWait/Trig-
pend
| / Err?=(none)/Reinst-required (Status,Err: uppercase=bad)
||/ Name          Version          Architecture Description
+++-----+-----+-----+-----+
=====
ii  tcpd             7.6.q-24         powerpc        Wietse Venema's TCP wrapper
utili
ii  tcpdump          4.3.0-1          powerpc        command-line network traffic
anal
ii  tcpreplay        3.4.3-2+whee     powerpc        Tool to replay saved tcpdump
file
cumulus@switch:~$

```

Upgrading to Newer Versions of Installed Packages

Upgrading a Single Package

A single package can be upgraded by simply installing that package again with `apt-get install`. You should perform an update first so that the APT cache is populated with the latest information about the packages.

To see if a package needs to be upgraded, use `apt-cache show <pkgname>` to show the latest version number of the package. Use `dpkg -l <pkgname>` to show the version number of the installed package.

Upgrading All Packages

You can update all packages on the system with `apt-get update`. This upgrades all installed versions with their latest versions but will not install any new packages.

Adding Packages from Another Repository

As shipped, Cumulus RMP searches the Cumulus RMP 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.



For several packages, Cumulus Networks has added features or made bug fixes and these packages must not be replaced with versions from other repositories. Cumulus RMP has been configured to ensure that the packages from the Cumulus RMP repository are always preferred over packages from other repositories.

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



Packages not part of the Cumulus RMP repository have generally not been tested, and may not be supported by Cumulus RMP support.

Installing packages outside of the Cumulus RMP repository requires the use of `apt-get`, but, depending on the package, `easy-install` and other commands can also be used.

To install a new package, please complete the following steps:

1. First, ensure package is not already installed in the system. Use the `dpkg` command:

```
cumulus@switch:~$ dpkg -l | grep {name of package}
```

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

```
cumulus@switch:~$ sudo apt-get update
cumulus@switch:~$ sudo apt-get install {name of package}
```

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

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

Otherwise, the repository may be listed in `/etc/apt/sources.list` but is commented out, as can be the case with the testing repository:

```
#deb http://repo.cumulusnetworks.com CumulusRMP-VERSION  
testing
```

To uncomment the repository, remove the `#` at the start of the line, then save the file:

```
deb http://repo.cumulusnetworks.com CumulusRMP-VERSION testing
```

4. Run `apt-get update` then install the package:

```
cumulus@switch:~$ sudo apt-get update  
cumulus@switch:~$ sudo apt-get install {name of package}
```

Configuration Files

- `/etc/apt/apt.conf`
- `/etc/apt/preferences`
- `/etc/apt/sources.list`

Useful Links

- [Debian GNU/Linux FAQ, Ch 8 Package management tools](#)
- [man pages for apt-get, dpkg, sources.list, apt_preferences](#)

Zero Touch Provisioning

Zero touch provisioning (ZTP) enables network devices to be quickly deployed in large-scale environments. Data center engineers only need to rack and stack the switch, then connect it to the management network — or alternatively, insert a USB stick and boot the switch. From here, the provisioning process can start automatically and deploy a configuration.

The provisioning framework allows for a one-time, user-provided script to be executed. This script can be used to add the switch to a configuration management (CM) platform such as [Puppet](#), [Chef](#), [CFEngine](#), or even a custom, home-grown tool.

In addition, you can use the `autoprovision` command in Cumulus RMP to manually invoke your provisioning script.

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

- Via a local file
- Using a USB drive inserted into the switch (ZTP-USB)
- Via DHCP

Each method is discussed in greater detail below.

Contents

- [Commands \(see page 53\)](#)
- [Zero Touch Provisioning Using a Local File \(see page 53\)](#)
- [Zero Touch Provisioning Using USB \(ZTP-USB\) \(see page 54\)](#)
- [Zero Touch Provisioning over DHCP \(see page 54\)](#)
 - [Triggering ZTP over DHCP \(see page 54\)](#)
 - [Configuring The DHCP Server \(see page 55\)](#)
 - [Detailed Look at HTTP Headers \(see page 55\)](#)
- [Writing ZTP Scripts \(see page 56\)](#)
 - [Example ZTP Scripts \(see page 56\)](#)
- [Testing and Debugging ZTP Scripts \(see page 58\)](#)
- [Manually Using the ztp Command \(see page 62\)](#)
- [Notes \(see page 63\)](#)
- [Configuration Files \(see page 63\)](#)

Commands

- [ztp](#)

Zero Touch Provisioning Using 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:

```
/mnt/usb/cumulus-ztp-amd64-cel_pebble-rUNKNOWN
/mnt/usb/cumulus-ztp-amd64-cel_pebble
/mnt/usb/cumulus-ztp-cel_pebble
/mnt/usb/cumulus-ztp-amd64
/mnt/usb/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.

Zero Touch Provisioning Using USB (ZTP-USB)



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

If the `ztp` process did 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 RMP 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 stick **before** you power up the switch.

At minimum, the script should:

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

Follow these steps to perform zero touch provisioning using USB:

1. Copy the Cumulus RMP [installation image](#) (see [page 42](#)) to the USB stick.
2. The `ztp` process searches the root filesystem of the newly mounted device 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 script's contents are parsed to ensure it contains the `CUMULUS-AUTOPROVISIONING` flag (see [example scripts](#) (see [page 56](#))).

Zero Touch Provisioning over DHCP

If the `ztp` process did not discover a local/ONIE script or applicable USB drive, it checks DHCP every 10 seconds for up to 5 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 via 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 zero touch provisioning process over DHCP follows these steps:

1. The first time you boot Cumulus RMP, `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 zero touch provisioning process itself will start.
4. The zero touch provisioning process requests the contents of the script from the URL, sending additional [HTTP headers](#) (see [page 55](#)) containing details about the switch.
5. The script's contents are parsed to ensure it contains the `CUMULUS-AUTOPROVISIONING` flag (see [example scripts](#) (see [page 56](#))).
6. If provisioning is necessary, then the script executes locally on the switch with root privileges.
7. The return code of the script gets examined. If it is 0, then the provisioning state is marked as complete in the autoprovisioning configuration file.

Triggering ZTP over DHCP

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

- Booting the switch
- Plugging a cable into or unplugging it from the eth0 port
- Disconnecting then reconnecting the switch's power cord

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

Configuring The DHCP Server

During the DHCP process over eth0, Cumulus RMP will request 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 would look 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, the hostname of the switch can be specified via 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 dcl-tor-sw1 { hardware ethernet 44:38:39:00:1a:6b; fixed-
address 192.168.0.101; option host-name "dcl-tor-sw1"; }
}
```

Detailed Look at 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		3.0.0-5c6829a-2013
09251712-final		
CUMULUS-LICENSE-INSTALLED	Either 0 or 1	0
CUMULUS-MANUFACTURER		odm
CUMULUS-PRODUCTNAME		switch_model

CUMULUS-SERIAL	XYZ123004
CUMULUS-VERSION	3.0.0
CUMULUS-PROV-COUNT	0
CUMULUS-PROV-MAX	32

Writing ZTP Scripts



Remember to include the following line in any of the supported scripts which are expected to be run via the autoprovisioning framework.

```
# CUMULUS-AUTOPROVISIONING
```

This line is required somewhere in the script file in order for execution to occur.

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

The script can be written in any language currently supported by Cumulus RMP, 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.

Example ZTP Scripts

The following script install Cumulus RMP from USB and applies a configuration:

```
#!/bin/bash
function error() {
    echo -e "\e[0;33mERROR: The Zero Touch Provisioning 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

trap error ERR

#Add Debian Repositories
```



```
echo "deb http://http.us.debian.org/debian jessie main" >> /etc/apt
/sources.list
echo "deb http://security.debian.org/ jessie/updates main" >> /etc/apt
/sources.list

#Update Package Cache
apt-get update -y

#Install netshow diagnostics commands
apt-get install -y netshow htop nmap

#Load interface config from usb
cp /mnt/usb/interfaces /etc/network/interfaces

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

#Reload interfaces to apply loaded config
ifreload -a

#Output state of interfaces
netshow interface

# CUMULUS-AUTOPROVISIONING
exit 0
```

Here is a simple script to install puppet:

```
#!/bin/bash
function error() {
    echo -e "\e[0;33mERROR: The Zero Touch Provisioning script failed
while running the command $BASH_COMMAND at line $BASH_LINENO.\e[0m" >&
    2
    exit 1
}
trap error ERR
apt-get update -y
apt-get upgrade -y
apt-get install puppet -y
sed -i /etc/default/puppet -e 's/START=no/START=yes/'
sed -i /etc/puppet/puppet.conf -e 's/\[main\]/\[main\]
\npluginsync=true/'
systemctl restart puppet.service
# CUMULUS-AUTOPROVISIONING
exit 0
```

This script illustrates how to specify an internal APT mirror and puppet master:

```
#!/bin/bash
function error() {
    echo -e "\e[0;33mERROR: The Zero Touch Provisioning script failed
while running the command $BASH_COMMAND at line $BASH_LINENO.\e[0m" >&
    exit 1
}
trap error ERR
sed -i /etc/apt/sources.list -e 's/repo.cumulusnetworks.com/labrepo.
mycompany.com/'
apt-get update -y
apt-get upgrade -y
apt-get install puppet -y
sed -i /etc/default/puppet -e 's/START=no/START=yes/'
sed -i /etc/puppet/puppet.conf -e 's/\[main\]/\[main\]
\npluginsync=true/'
sed -i /etc/puppet/puppet.conf -e 's/\[main\]/\[main\]
\nserver=labpuppet.mycompany.com/'
systemctl restart puppet.service
# CUMULUS-AUTOPROVISIONING
exit 0
```

Now `puppet` can take over management of the switch, configuration authentication, changing the default root password, and setting up interfaces and routing protocols.

Several ZTP example scripts are available in the [Cumulus GitHub repository](#).

Testing and Debugging 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-s6000-01 (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
```

You can also run `ztp -s` to get more information about the current state of ZTP.

ZTP INFO:

```
State          enabled
Version        1.0
Result         Script Failure
Date           Tue May 10 22:42:09 2016 UTC
Method         ZTP DHCP
URL            http://192.0.2.1/demo.sh
```

If ZTP ran when the switch booted 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 RMP 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-s6000-01 ztp[400]: ztp [400]: ZTP DHCP: Looking f
or ZTP Script provided by DHCP
May 11 16:38:45 dell-s6000-01 ztp[400]: ztp [400]: Attempting to
provision via ZTP DHCP from http://192.0.2.1/demo.sh
May 11 16:38:45 dell-s6000-01 ztp[400]: ztp [400]: ZTP DHCP: URL
response code 200
May 11 16:38:45 dell-s6000-01 ztp[400]: ztp [400]: ZTP DHCP: Found
Marker CUMULUS-AUTOPROVISIONING
May 11 16:38:45 dell-s6000-01 ztp[400]: ztp [400]: ZTP DHCP:
Executing http://192.0.2.1/demo.sh
May 11 16:38:45 dell-s6000-01 ztp[400]: ztp [400]: ZTP DHCP: Payload
returned code 1
May 11 16:38:45 dell-s6000-01 ztp[400]: ztp [400]: Script returned
failure
May 11 16:38:45 dell-s6000-01 systemd[1]: ztp.service: main process
exited, code=exited, status=1/FAILURE
May 11 16:38:45 dell-s6000-01 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:
State 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_s6000_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_s6000_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-s6000-01 ztp[400]: ztp [400]: ZTP DHCP: Looking f
or ZTP Script provided by DHCP
May 11 16:38:45 dell-s6000-01 ztp[400]: ztp [400]: Attempting to
provision via ZTP DHCP from http://192.0.2.1/demo.sh
May 11 16:38:45 dell-s6000-01 ztp[400]: ztp [400]: ZTP DHCP: URL
response code 200
May 11 16:38:45 dell-s6000-01 ztp[400]: ztp [400]: ZTP DHCP: Found
Marker CUMULUS-AUTOPROVISIONING
May 11 16:38:45 dell-s6000-01 ztp[400]: ztp [400]: ZTP DHCP:
Executing http://192.0.2.1/demo.sh
May 11 16:38:45 dell-s6000-01 ztp[400]: ztp [400]: ZTP DHCP: Payload
returned code 1
May 11 16:38:45 dell-s6000-01 ztp[400]: ztp [400]: Script returned
failure
May 11 16:38:45 dell-s6000-01 systemd[1]: ztp.service: main process
exited, code=exited, status=1/FAILURE
May 11 16:38:45 dell-s6000-01 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
/ztpt/ztpt_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/ztpt/cumulus-ztp-x86_64-
dell_s6000_sl220-rUNKNOWN
2016-05-11T16:37:45.486394+00:00 cumulus ztp [400]: ZTP LOCAL:
Waterfall search for /var/lib/cumulus/ztpt/cumulus-ztp-x86_64-
dell_s6000_sl220
2016-05-11T16:37:45.488385+00:00 cumulus ztp [400]: ZTP LOCAL:
Waterfall search for /var/lib/cumulus/ztpt/cumulus-ztp-x86_64-dell
2016-05-11T16:37:45.489665+00:00 cumulus ztp [400]: ZTP LOCAL:
Waterfall search for /var/lib/cumulus/ztpt/cumulus-ztp-x86_64
2016-05-11T16:37:45.490854+00:00 cumulus ztp [400]: ZTP LOCAL:
Waterfall search for /var/lib/cumulus/ztpt/cumulus-ztp
2016-05-11T16:37:45.492296+00:00 cumulus ztp [400]: ZTP USB: Looking f
or 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-s6000-01 (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
cumulus@switch:~$ sudo ztp -s
State      enabled
Version    1.0
Result     Script Failure
Date       Tue May 10 22:44:17 2016 UTC
Method     ZTP Manual
URL        http://192.0.2.1/demo.sh
```

Manually Using the ztp Command

To enable zero touch provisioning, use the `-e` option:

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



Enabling `ztp` means that `ztp` will try to occur the next time the switch boots. However, if ZTP already occurred 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 bootstrap:

- Password changes
- Users and groups changes
- Packages changes
- Interfaces changes

When the switch is booted for the very first time, ZTP records the state of some 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 will compare 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 zero touch provisioning, 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 /mnt/usb/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 Thu May 5 16:49:33 2016 UTC
Method Switch manually configured
URL None
```

Notes

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

Configuration Files

- `/var/lib/cumulus/autopvision.conf`

Configuring and Managing Network Interfaces

`ifupdown` is the network interface manager for Cumulus RMP. Cumulus RMP uses an updated version of this tool, `ifupdown2`.

For more information on network interfaces, see [Layer 1 and Switch Port Attributes](#) (see page 78).



By default, `ifupdown` is quiet; use the verbose option `-v` when you want to know what is going on when bringing an interface down or up.

Contents

(Click to expand)

- [Contents](#) (see page 64)
- [Commands](#) (see page 64)
- [Man Pages](#) (see page 65)
- [Configuration Files](#) (see page 65)
- [Basic Commands](#) (see page 65)
- [ifupdown2 Interface Classes](#) (see page 66)
 - [Bringing All auto Interfaces Up or Down](#) (see page 67)
- [Configuring a Loopback Interface](#) (see page 67)
- [ifupdown2 Interface Dependencies](#) (see page 69)
 - [ifup Handling of Upper \(Parent\) Interfaces](#) (see page 72)
- [Configuring IP Addresses](#) (see page 73)
 - [Purging Existing IP Addresses on an Interface](#) (see page 74)
- [Specifying User Commands](#) (see page 74)
- [Sourcing Interface File Snippets](#) (see page 75)
- [Using Globs for Port Lists](#) (see page 75)
- [Using Templates](#) (see page 76)
- [Adding Descriptions to Interfaces](#) (see page 77)
- [Caveats and Errata](#) (see page 77)
- [Useful Links](#) (see page 78)

Commands

- [ifdown](#)

- ifquery
- ifreload
- ifup
- mako-render

Man Pages

- man ifdown(8)
- man ifquery(8)
- man ifreload
- man ifup(8)
- man ifupdown-addons-interfaces(5)
- man interfaces(5)

Configuration Files

- /etc/network/interfaces

Basic Commands

To bring up an interface or apply changes to an existing interface, run:

```
cumulus@switch:~$ sudo ifup <ifname>
```

To bring down a single interface, run:

```
cumulus@switch:~$ sudo ifdown <ifname>
```

Runtime Configuration (Advanced)



A runtime configuration is non-persistent, which means the configuration you create here does not persist after you reboot the switch.

To administratively bring an interface up or down, run:

```
cumulus@switch:~$ sudo ip link set dev swp1 {up|down}
```



If you specified *manual* as the address family, you must bring up that interface manually using `ifconfig`. For example, if you configured a bridge like this:

```
auto bridge01
iface bridge01 inet manual
```

You can only bring it up by running `ifconfig bridge01 up`.



`ifdown` 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.

To see the link and administrative state, use the `ip link show` command:

```
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
```

In this example, `swp1` is administratively UP and the physical link is UP (LOWER_UP flag). More information on interface administrative state and physical state can be found in [this knowledge base article](#).

ifupdown2 Interface Classes

`ifupdown2` provides for the grouping of interfaces into separate classes, where a class is simply a user-defined label used to group interfaces that share a common function (like uplink, downlink or compute). You specify classes in `/etc/network/interfaces`.

The most common class users are familiar with 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 make up 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, or still use the `-a` options since these interfaces are also in the `auto` class:

```
cumulus@switch:~$ sudo ifup --allow=uplinks
cumulus@switch:~$ sudo ifreload -a
```

Bringing All auto Interfaces Up or Down

You can easily bring up or down all interfaces marked with the common `auto` class in `/etc/network/interfaces`. Use the `-a` option. For further details, see individual man pages for `ifup(8)`, `ifdown(8)`, `ifreload(8)`.

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, which is equivalent to running `ifdown` then `ifup`, the one difference being that `ifreload` skips any configurations that didn't change):

```
cumulus@switch:~$ sudo ifreload -a
```

Configuring a Loopback Interface

Cumulus RMP has a loopback preconfigured in `/etc/network/interfaces`. 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`.



The loopback interface `lo` must always be specified in `/etc/network/interfaces` and must always be up.

ifupdown Behavior with Child Interfaces

By default, `ifupdown` recognizes and uses any interface present on the system — whether a VLAN, bond or physical interface — that is listed as a dependent of an interface. You are not required to list them in the `interfaces` file unless they need a specific configuration, for [MTU](#), [link speed](#), and so forth (see [page 78](#)). And if you need to delete a child interface, you should delete all references to that interface from the `interfaces` file.

For this example, `swp1` and `swp2` below 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	Without Child Interfaces Defined
<pre> 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 </pre>	<pre> auto bridge iface bridge bridge-vlan-aware yes bridge-ports swp1 swp2 bridge-vids 1-100 bridge-pvid 1 bridge-stp on </pre>

Bridge in Traditional Mode - Example

For this example, `swp1.100` and `swp2.100` below 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	Without Child Interfaces Defined
<pre> 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 </pre>	<pre> auto br-100 iface br-100 address 10.0.12.2/2 4 address 2001:dad: beef::3/64 bridge-ports swp1.1 00 swp2.100 bridge-stp on </pre>

For more information on the bridge in traditional mode vs the bridge in VLAN-aware mode, please read [this knowledge base article](#).

ifupdown2 Interface Dependencies

`ifupdown2` understands interface dependency relationships. When `ifup` and `ifdown` are run with all interfaces, they always run with all interfaces in dependency order. When run with the interface list on the command line, the default behavior is to not run with dependents. But 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-dependends` option. `--with-dependends` 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
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4

auto bond2
iface bond2
    address 100.0.0.5/16
    bond-slaves swp31 swp32
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4

auto br2001
iface br2001
    address 12.0.1.3/24
    bridge-ports bond1.2001 bond2.2001
    bridge-stp on
```

Using `ifup --with-dependends br2001` 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-dependends br2001
```

Similarly, specifying `ifdown --with-depends br2001` 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
```



As mentioned earlier, `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 terms of the above example, `ifdown br2001` deletes `br2001`.

To guide you through which interfaces will be brought down and up, use the `--print-dependency` option to get the list of dependents.

Use `ifquery --print-dependency=list -a` to get the dependency list of 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, use:

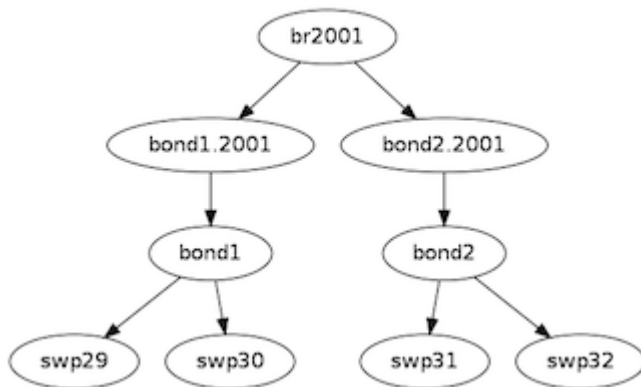
```
cumulus@switch:~$ sudo ifquery --print-dependency=list br2001
br2001 : ['bond1.2001', 'bond2.2001']
bond1.2001 : ['bond1']
bond2.2001 : ['bond2']
bond1 : ['swp29', 'swp30']
bond2 : ['swp31', 'swp32']
swp29 : None
swp30 : None
swp31 : None
```

```
swp32 : None
```

To print the dependency information of an interface in dot format:

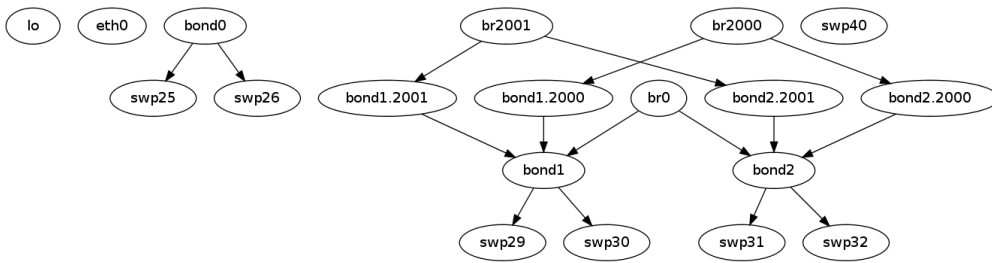
```
cumulus@switch:~$ sudo ifquery --print-dependency=dot br2001
/* Generated by GvGen v.0.9 (http://software.inl.fr/trac/wiki/GvGen)
*/
digraph G {
    compound=true;
    node1 [label="br2001"];
    node2 [label="bond1.2001"];
    node3 [label="bond2.2001"];
    node4 [label="bond1"];
    node5 [label="bond2"];
    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:

```
cumulus@switch:~$ sudo ifquery --print-dependency=dot -a >interfaces_all.dot
```



ifup Handling of Upper (Parent) Interfaces

When you run `ifup` on a logical interface (like a bridge, bond or VLAN interface), if the `ifup` resulted in the creation of the logical interface, by default it implicitly tries to execute on the interface's upper (or parent) interfaces as well. This helps in most cases, especially when a bond is brought down and up, as in the example below. This section describes the behavior of bringing up the upper interfaces.

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`.

As you can see above, implicitly bringing up the upper interface helps, but 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 `/etc/network/ifupdown2/ifupdown2.conf`.

With `skip_upperifaces=1`, you will 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`.



Although specifying a subinterface like `swp1.100` and then running `ifup swp1.100` will also result in the automatic creation of the `swp1` interface in the kernel, Cumulus Networks recommends you specify the parent interface `swp1` as well. A parent interface is one where any physical layer configuration can reside, such as `link-speed 1000` or `link-duplex full`.

It's important to note that if you only create `swp1.100` and not `swp1`, then you cannot run `ifup swp1` since you did not specify it.

Configuring IP Addresses

In `/etc/network/interfaces`, list all IP addresses as shown below under the `iface` section (see `man interfaces` for more information):

```
auto swp1
iface swp1
    address 12.0.0.1/30
    address 12.0.0.2/30
```

The address method and address family are not mandatory. They default to `inet/inet6` and `static` by default, but `inet/inet6` **must** be specified if you need to specify `dhcp` or `loopback`:

```
auto lo
iface lo inet loopback
```

You can specify both IPv4 and IPv6 addresses in the same `iface` stanza:

```
auto swp1
iface swp1
    address 192.0.2.1/30
    address 192.0.2.2/30
    address 2001:DB8::1/126
```

Runtime Configuration (Advanced)



A runtime configuration is non-persistent, which means the configuration you create here does not persist after you reboot the switch.

To make non-persistent changes to interfaces at runtime, use `ip addr add`:

```
cumulus@switch:~$ sudo ip addr add 192.0.2.1/30 dev swp1
cumulus@switch:~$ sudo ip addr add 2001:DB8::1/126 dev swp1
```

To remove an addresses from an interface, use `ip addr del`:

```
cumulus@switch:~$ sudo ip addr del 192.0.2.1/30 dev swp1
cumulus@switch:~$ sudo ip addr del 2001:DB8::1/126 dev swp1
```

See `man ip` for more details on the options available to manage and query interfaces.

To show the assigned address on an interface, use `ip addr show`:

```
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
```

Purging 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 including the `address-purge` setting in the interface's configuration. For example, add the following to the interface configuration in `/etc/network/interfaces`:

```
auto swp1
iface swp1
    address-purge no
```



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 [Caveats and Errata \(see page 77\)](#) section below for some cautions about using multiple `iface` stanzas for the same interface.

Specifying User Commands

You can specify additional user commands in the `interfaces` file. As shown in the example below, 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:

```
auto swp1
iface swp1
    address 12.0.0.1/30
    up /sbin/foo bar
```

Any valid command can be hooked in the sequencing of bringing an interface up or down, although commands should be limited in scope to network-related commands associated with the particular interface.

For example, it wouldn't make sense to install some Debian package on `ifup` of `swp1`, even though that is technically possible. See `man interfaces` for more details.

Sourcing Interface File Snippets

Sourcing interface files helps organize and manage the `interfaces(5)` file. For example:

```
cumulus@switch:~$ 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:~$ 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
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
```

Using Globs for Port Lists

Some modules support globs to define port lists (that is, a range of ports). You can use the `glob` keyword to specify bridge ports and bond slaves:

```
auto br0
iface br0
    bridge-ports glob swp1-6.100
```

```
auto br1
iface br1
    bridge-ports glob swp7-9.100 swp11.100 glob swp15-18.100
```

Using Templates

ifupdown2 supports [Mako-style templates](#). The Mako template engine is run over the `interfaces` file before parsing.

Use the template to declare cookie-cutter bridges in the `interfaces` file:

```
%for v in [11,12]:
auto vlan${v}
iface vlan${v}
    address 10.20.${v}.3/24
    bridge-ports glob swp19-20.${v}
    bridge-stp on
%endfor
```

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
```



Regarding 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.



You can test your template and confirm it evaluates correctly by running `mako-render /etc/network/interfaces`.



For more examples of configuring Mako templates, read this [knowledge base article](#).

Adding Descriptions to Interfaces

You can add descriptions to the interfaces configured in `/etc/network/interfaces` by using the `alias` keyword. For example:

```
auto swp1
iface swp1
    alias swp1 hypervisor_port_1
```

You can query interface descriptions by running `ip link show`. The alias appears on the `alias` line:

```
cumulus@switch$ ip link show swp1
3: swp1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast
state DOWN mode DEFAULT qlen 500
    link/ether aa:aa:aa:aa:aa:bc brd ff:ff:ff:ff:ff:ff
    alias hypervisor_port_1
```

Interface descriptions also appear in the [SNMP OID \(see page 170\) IF-MIB::ifAlias](#).

Caveats and Errata

While `ifupdown2` supports the inclusion of multiple `iface` stanzas for the same interface, Cumulus Networks recommends you use a single `iface` stanza for each interface, if possible.

There are cases where you must specify more than one `iface` stanza for the same interface. For example, the configuration for a single interface can come from many places, like a template or a sourced file.

If you do specify multiple `iface` stanzas for the same interface, make sure the stanzas do not specify the same interface attributes. Otherwise, unexpected behavior can result.

For example, `swp1` is configured in two places:

```
cumulus@switch:~$ cat /etc/network/interfaces

source /etc/interfaces.d/speed_settings

auto swp1
iface swp1
    address 10.0.14.2/24
```

As well as `/etc/interfaces.d/speed_settings`

```
cumulus@switch:~$ cat /etc/interfaces.d/speed_settings
```

```
auto swp1
iface swp1
    link-speed 1000
    link-duplex full
```

ifupdown2 correctly parses a configuration like this because the same attributes are not specified in multiple `iface` stanzas.

And, as stated in the note above, you cannot purge existing addresses on interfaces with multiple `iface` stanzas.

Useful Links

- <http://wiki.debian.org/NetworkConfiguration>
- <http://www.linuxfoundation.org/collaborate/workgroups/networking/bonding>
- <http://www.linuxfoundation.org/collaborate/workgroups/networking/bridge>
- <http://www.linuxfoundation.org/collaborate/workgroups/networking/vlan>

Layer 1 and Switch Port Attributes

This chapter discusses the various network interfaces on a switch running Cumulus RMP.

Contents

(Click to expand)

- Contents (see page 78)
- Commands (see page 78)
- Man Pages (see page 79)
- Configuration Files (see page 79)
- Interface Types (see page 79)
- Settings (see page 79)
 - Port Speed and Duplexing (see page 79)
 - Auto-negotiation (see page 81)
 - MTU (see page 81)
- Verification and Troubleshooting Commands (see page 83)
 - Statistics (see page 83)
 - Querying SFP Port Information (see page 84)
- Useful Links (see page 84)

Commands

- `ethtool`
- `ip`

Man Pages

- man ethtool
- man interfaces
- man ip
- man ip addr
- man ip link

Configuration Files

- /etc/network/interfaces

Interface Types

Cumulus RMP exposes network interfaces for several types of physical and logical devices:

- lo, network loopback device
- ethN, switch management port(s), for out of band management only
- swpN, switch front panel ports
- (optional) brN, bridges (IEEE 802.1Q VLANs)
- (optional) bondN, bonds (IEEE 802.3ad link aggregation trunks, or port channels)

Settings

You can set the MTU, speed, duplex and auto-negotiation settings under a physical or logical interface stanza:

```
auto swp1
iface swp1
    address 10.1.1.1/24
    mtu 9000
    link-speed 10000
    link-duplex full
    link-autoneg off
```

To load the updated configuration, run the `ifreload -a` command:

```
cumulus@switch:~$ sudo ifreload -a
```

Port Speed and Duplexing

Cumulus RMP supports both half- and **full-duplex** configurations. Supported port speeds include 1G and 10G. Set the speeds in terms of Mbps, where the setting for 1G is 1000 and 10G is 10000.

You can create a persistent configuration for port speeds in `/etc/network/interfaces`. Add the appropriate lines for each switch port stanza. For example:

```
auto swp1
iface swp1
    address 10.1.1.1/24
    link-speed 10000
    link-duplex full
```



If you specify the port speed in `/etc/network/interfaces`, you must also specify the duplex mode setting along with it; otherwise, `ethtool` defaults to half duplex.

You can also configure these settings at run time, using `ethtool`.

Runtime Configuration (Advanced)



A runtime configuration is non-persistent, which means the configuration you create here does not persist after you reboot the switch.

You can use `ethtool` to configure duplexing and the speed for your switch ports. You must specify both port speed and duplexing in the `ethtool` command; auto-negotiation is optional. The following examples use `swp1`.

- To set the port speed to 1G, run:

```
ethtool -s swp1 speed 1000 duplex full
```

- To set the port speed to 10G, run:

```
ethtool -s swp1 speed 10000 duplex full
```

- To enable duplexing, run:

```
ethtool -s swp1 speed 10000 duplex full|half
```

Port Speed Limitations

Ports can be configured to one speed less than their maximum speed.

Switch port Type	Lowest Configurable Speed
1G	100 Mb
10G	1 Gigabit (1000 Mb)

Auto-negotiation

You can enable or disable **auto-negotiation** (that is, set it *on* or *off*) on a switch port.

```
auto swp1
iface swp1
    link-autoneg off
```

Runtime Configuration (Advanced)



A runtime configuration is non-persistent, which means the configuration you create here does not persist after you reboot the switch.

You can use `ethtool` to configure auto-negotiation for your switch ports. The following example use `swp1`:

- To enable or disable auto-negotiation, run:

```
ethtool -s swp1 speed 10000 duplex full autoneg on|off
```

MTU

Interface MTU applies to the management port, front panel port, bridge, VLAN subinterfaces and bonds.

```
auto swp1
iface swp1
    mtu 9000
```

Runtime Configuration (Advanced)



A runtime configuration is non-persistent, which means the configuration you create here does not persist after you reboot the switch.

To set swp1 to Jumbo Frame MTU=9000, use `ip link set`:

```
cumulus@switch:~$ sudo ip link set dev swp1 mtu 9000
cumulus@switch:~$ ip link show dev swp1
3: swp1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9000 qdisc
pfifo_fast state UP mode DEFAULT qlen 500
    link/ether 44:38:39:00:03:c1 brd ff:ff:ff:ff:ff:ff
```



You must take care to ensure there are no MTU mismatches in the conversation path. MTU mismatches will result in dropped or truncated packets, degrading or blocking network performance.

When you are configuring MTU for a bridge, its MTU setting is the lowest MTU setting of any interface that is a member of that bridge (that is, every interface specified in `bridge-ports` in the bridge configuration in the `interfaces` file). Consider this bridge configuration:

```
auto br0
iface br0
    bridge-ports bond1 bond2 bond3 bond4 peer5
    bridge-vlan-aware yes
    bridge-vids 100-110
    bridge-stp on
```

In order for br0 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.

```
auto peer5
iface peer5
    bond-slaves swp3 swp4
    bond-mode 802.3ad
    bond-miimon 100
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit_hash_policy layer3+4
    mtu 9000
```

To show MTU, use `ip link show`:

```
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
```

Verification and Troubleshooting Commands

Statistics

High-level interface statistics are available with the `ip -s link` command:

```
cumulus@switch:~$ ip -s 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
    RX: bytes  packets  errors  dropped  overrun  mcast
        21780      242      0        0         0       242
    TX: bytes  packets  errors  dropped  carrier  collsns
        114554     11325    0        0         0        0
```

Low-level interface statistics are available with `ethtool`:

```
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
    SoftInDrops: 0
    SoftInFrameErrors: 0
    HwIfOutDiscards: 0
```

```
HwIfOutErrors: 0
HwIfOutQDrops: 0
HwIfOutNonQDrops: 0
SoftOutErrors: 0
SoftOutDrops: 0
SoftOutTxFifoFull: 0
HwIfOutQLen: 0
```

Querying SFP Port Information

You can verify SFP settings using `ethtool -m`. The following example shows the output for 1G and 10G modules:

```
cumulus@switch:~# sudo ethtool -m | egrep '(swp|RXPow :|TXPow :|EthernetComplianceCode)'
```



```
swp1: SFP detected
      EthernetComplianceCodes : 1000BASE-LX
      RXPow : -10.4479dBm
      TXPow : 18.0409dBm
swp3: SFP detected
      10GEthernetComplianceCode : 10G Base-LR
      RXPow : -3.2532dBm
      TXPow : -2.0817dBm
```

Useful Links

- <http://wiki.debian.org/NetworkConfiguration>
- <http://www.linuxfoundation.org/collaborate/workgroups/networking/vlan>
- <http://www.linuxfoundation.org/collaborate/workgroups/networking/bridge>
- <http://www.linuxfoundation.org/collaborate/workgroups/networking/bonding>

Configuring DHCP Relays

You can configure an interface so it can make DHCP relay requests for IPv4 and IPv6.

To run DHCP for both IPv4 and IPv6, you need to initiate the DHCP relay and DHCP server twice: once for IPv4 using the `-4` option, and once for IPv6 using the `-6` option. Following are the configurations on the host, leaf and DHCP server using the following topology:

```
Server
(10.0.100.2)
|
|
(10.0.100.1)
```

```

Leaf
(10.0.1.1)
|
|
Host

```

Contents

- [Configuring the Relays \(see page 85\)](#)

Configuring the Relays

Here is the host configuration:

```

/etc/network/interfaces
auto eth1
iface eth1 inet dhcp

auto eth1
iface eth1 inet6 dhcp

```

Here is the leaf configuration:

```

cumulus@switch:~$ /usr/sbin/dhcrelay -4 -i swp1 10.0.100.2 -i swp51
cumulus@switch:~$ /usr/sbin/dhcrelay -6 -l swp1 -u 2001:db8:100::2%swp51

```

You have to run two independent instances of `dhcrelay`. The `dhcrelay` feature is part of the `isc-dhcp-relay` services. The format of this command is below:

```

cumulus@switch:~$ /usr/sbin/dhcrelay -4 -i <interface_facing_host>
<ip_address_dhcp_server> -i <interface_facing_dhcp_server>
cumulus@switch:~$ /usr/sbin/dhcrelay -6 -l <interface_facing_host> -u
<ip_address_dhcp_server>%<interface_facing_dhcp_server>

```

See the `man dhcrelay` for more information.

The `systemd` launch scripts can be edited so that the `dhcrelay` service starts with the switch.

The launch scripts for `systemd` are located in `/lib/systemd/system`. Edit or create two files as described below:

```
cumulus@leaf01:/lib/systemd/system$ cat dhcrelay.service
[Unit]
Description=DHCPv4 Relay Agent Daemon
Documentation=man:dhcrelay(8)
After=network-online.target networking.service syslog.service

[Service]
Type=simple
EnvironmentFile=-/etc/default/isc-dhcp-relay
# Here, we are expecting the INTF_CMD to contain
# the -i for each interface specified,
ExecStart=/usr/sbin/dhcrelay -d -q $INTF_CMD $SERVERS $OPTIONS

[Install]
WantedBy=multi-user.target

cumulus@leaf01:/lib/systemd/system$ cat /etc/default/isc-dhcp-relay
SERVERS="10.0.100.2"

INTF_CMD="-i swp1 -i swp51"
```

```
cumulus@leaf01:/lib/systemd/system$ cat dhcrelay6.service
[Unit]
Description=DHCPv6 Relay Agent Daemon
Documentation=man:dhcrelay(8)
After=network-online.target networking.service syslog.service

[Service]
Type=simple
EnvironmentFile=-/etc/default/isc-dhcp-relay6
ExecStart=/usr/sbin/dhcrelay -6 -d -q $INTF_CMD $SERVERS $OPTIONS

[Install]
WantedBy=multi-user.target

cumulus@leaf01:/lib/systemd/system$ cat /etc/default/isc-dhcp-relay6
SERVERS=" -u 2001:db8:100::2%swp51"

INTF_CMD="-l swp1"
```

Here is the DHCP server configuration:

```
/usr/sbin/dhcpd -6 -cf /etc/dhcp/dhcpd6.conf swp1
/usr/sbin/dhcpd -4 -cf /etc/dhcp/dhcpd.conf swp1
```

The configuration files for the two DHCP servers need to have two pools:

- Pool 1: Subnet overlaps interfaces
- Pool 2: Subnet that includes the addresses

Here are the sample configurations:

```
/etc/dhcp/dhcpd.conf
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;
}
```

```
/etc/dhcp/dhcpd6.conf
subnet6 2001:db8:100::/64 {
}
subnet6 2001:db8:1::/64 {
    range6 2001:db8:1::100 2001:db8:1::200;
}
```

Just as you did with the DHCP relay scripts, the `systemd` initialization scripts can be used to launch the DHCP server at start. Here are sample configurations:

```
cumulus@spine01:/lib/systemd/system$ cat dhcpd.service
[Unit]
Description=DHCPv4 Server Daemon
Documentation=man:dhcpd(8) man:dhcpd.conf(5)
After=network-online.target networking.service syslog.service

[Service]
Type=simple
EnvironmentFile=-/etc/default/isc-dhcp-server
ExecStart=/usr/sbin/dhcpd -f -q $DHCPD_CONF $DHCPD_PID $INTERFACES $OPTIONS

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

```
cumulus@spine01:/lib/systemd/system$ cat /etc/default/isc-dhcp-server
DHCPD_CONF="-cf /etc/dhcp/dhcpd.conf"
```

```
INTERFACES="swp1"
```

```
cumulus@spine01:/lib/systemd/system$ 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;
}
```

```
cumulus@spine01:/lib/systemd/system$ cat dhcpd6.service
[Unit]
Description=DHCPv6 Server Daemon
Documentation=man:dhcpd(8) man:dhcpd.conf(5)
After=network-online.target networking.service syslog.service
```

```
[Service]
Type=simple
EnvironmentFile=-/etc/default/isc-dhcp-server6
ExecStart=/usr/sbin/dhcpd -f -q -6 $DHCPD_CONF $DHCPD_PID $INTERFACES
$OPTIONS
```

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

```
cumulus@spine01:/lib/systemd/system$ cat /etc/default/isc-dhcp-server6
DHCPD_CONF="-cf /etc/dhcp/dhcpd6.conf"
```

```
INTERFACES="swp1"
```

```
cumulus@spine01:/lib/systemd/system$ 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;
}
```

Layer 1 and Layer 2 Features

Spanning Tree and Rapid Spanning Tree

Spanning tree protocol (STP) is always recommended in layer 2 topologies, as it prevents bridge loops and broadcast radiation on a bridged network.

`mstpd` is a daemon that implements IEEE802.1D 2004 and IEEE802.1Q 2011. Currently, STP is disabled by default on the bridge in Cumulus RMP.

To enable STP, configure `brctl stp <bridge> on`.



The STP modes Cumulus RMP supports vary depending upon which [bridge driver mode](#) (see [page 126](#)) is in use. For a bridge configured in *traditional* mode, STP, RSTP, PVST and PVRST are supported; with the default set to PVRST. *VLAN-aware* (see [page 147](#)) bridges only operate in RSTP mode.

If a bridge running RSTP (802.1w) receives a common STP (802.1D) BPDU, it will automatically fall back to 802.1D operation.

You can configure `mstpd` to be in common STP mode only, by setting `setforcevers` to `STP`.

Contents

(Click to expand)

- [Contents](#) (see page 90)
- [Commands](#) (see page 91)
- [PVST/PVRST](#) (see page 91)
- [Creating a Bridge and Configuring STP](#) (see page 91)
- [Configuring Spanning Tree Parameters](#) (see page 93)
 - [Understanding Spanning Tree Parameters](#) (see page 94)
- [Bridge Assurance](#) (see page 101)
- [BPDU Guard](#) (see page 101)
 - [Configuring BPDU Guard](#) (see page 102)
 - [Recovering a Port Disabled by BPDU Guard](#) (see page 102)
- [BPDU Filter](#) (see page 104)
- [Storm Control](#) (see page 105)
- [Configuration Files](#) (see page 105)
- [Man Pages](#) (see page 105)
- [Useful Links](#) (see page 105)
- [Caveats and Errata](#) (see page 105)

Commands

- brctl
- mstpctl

mstpctl is a utility to configure STP. mstpctl is started by default on bootup. mstpctl logs and errors are located in `/var/log/daemon.log`.

PVST/PVRST

Per VLAN Spanning Tree (PVST) creates a spanning tree instance for a bridge. Rapid PVST (PVRST) supports RSTP enhancements for each spanning tree instance. You must create a bridge corresponding to the untagged native/access VLAN, and all the physical switch ports must be part of the same VLAN. When connected to a switch that has a native VLAN configuration, the native VLAN **must** be configured to be VLAN 1 only.

Cumulus RMP supports the RSTP/PVRST/PVST modes of STP natively when the bridge is configured in [traditional mode](#) (see page 126).

Creating a Bridge and Configuring STP

To create a bridge, configure the bridge stanza under `/etc/network/interfaces`. More information on configuring bridges [can be found here](#) (see page 126). To enable STP on the bridge, include the keyword `bridge-stp on`.

```
auto br2
iface br2
    bridge-ports swp1.101 swp4.101 swp5.101
    bridge-stp on
```

To enable the bridge, run `ifreload -a`:

```
cumulus@switch:~$ sudo ifreload -a
```

Runtime Configuration (Advanced)

You use `brctl` to create the bridge, add bridge ports in the bridge and configure STP on the bridge. `mstpctl` is used only when an admin needs to change the default configuration parameters for STP:

```
cumulus@switch:~$ sudo brctl addbr br2

cumulus@switch:~$ sudo brctl addif br2 swp1.101 swp4.101 swp5.101
```

```
cumulus@switch:~$ sudo brctl stp br2 on

cumulus@switch:~$ sudo ifconfig br2 up
```

To get the bridge state, use:

```
cumulus@switch:~$ sudo brctl show
bridge name      bridge id                STP enabled    interfaces
br2              8000.001401010100       yes            swp1.101
                                               swp4.101
                                               swp5.101
```

To get the mstpd bridge state, use:

```
cumulus@switch:~$ sudo mstpctl showbridge br2
br2 CIST info
  enabled          yes
  bridge id        F.000.00:14:01:01:01:00
  designated root  F.000.00:14:01:01:01:00
  regional root    F.000.00:14:01:01:01:00
  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           200
  force protocol version rstp
  time since topology change 90843s
  topology change count      4
  topology change            no
  topology change port       swp4.101
  last topology change port   swp5.101
```

To get the mstpd bridge port state, use:

```
cumulus@switch:~$ sudo mstpctl showport br2
E swp1.101 8.001 forw F.000.00:14:01:01:01:00 F.000.00:14:01:01:01:00
8.001 Desg
  swp4.101 8.002 forw F.000.00:14:01:01:01:00 F.000.00:14:01:01:01:00
8.002 Desg
  E swp5.101 8.003 forw F.000.00:14:01:01:01:00 F.000.00:14:01:01:01:00
```

8.003 Desg

```
cumulus@switch:~$ sudo mstpctl showportdetail br2 swp1.101
br2:swp1.101 CIST info
enabled          yes          role          Designated
port id          8.001          state         forwarding
external port cost 2000          admin external cost 0
internal port cost 2000          admin internal cost 0
designated root   F.000.00:14:01:01:01:00 dsgn external cost 0
dsgn regional root F.000.00:14:01:01:01:00 dsgn internal cost 0
designated bridge F.000.00:14:01:01:01:00 designated port 8.001
admin edge port  no          auto edge port yes
oper edge port  yes          topology change ack no
point-to-point  yes          admin point-to-point auto
restricted role  no          restricted TCN  no
port hello time 2          disputed       no
bpdu guard port no          bpdu guard error no
network port     no          BA inconsistent no
Num TX BPDU      45772          Num TX TCN      4
Num RX BPDU      0          Num RX TCN      0
Num Transition FWD 2          Num Transition BLK 2
```

Configuring Spanning Tree Parameters

The persistent configuration for a bridge is set in `/etc/network/interfaces`. The configuration below shows every possible option configured. There is no requirement to configure any of these options:

```
auto br2
iface br2 inet static
    bridge-ports swp1 swp2 swp3
    bridge-stp on
    mstpctl-maxage 20
    mstpctl-fdelay 15
    mstpctl-maxhops 20
    mstpctl-txholdcount 6
    mstpctl-forcevers rstp
    mstpctl-treeprio 32768
    mstpctl-hello 2
    mstpctl-portpathcost swp1=0 swp2=0
    mstpctl-portadmededge swp1=no swp2=no
    mstpctl-portautoedge swp1=yes swp2=yes
    mstpctl-portp2p swp1=no swp2=no
    mstpctl-portrestrrole swp1=no swp2=no
    mstpctl-bpduguard swp1=no swp2=no
```

```
mstpctl-portrestrtcn swp1=no swp2=no
mstpctl-portnetwork swp1=no
mstpctl-treeportprio swp3=128
```

Understanding Spanning Tree Parameters

The spanning tree parameters are defined in the IEEE 802.1D, 802.1Q specifications and in the table below. While configuring spanning tree in a persistent configuration, as described above, is the preferred method, you can also use `mstpctl (8)` to configure spanning tree protocol parameters at runtime.




A runtime configuration is non-persistent, which means the configuration you create here does not persist after you reboot the switch.


The `mstp` daemon is an open source project that some network engineers may be unfamiliar with. For example, many incumbent vendors use the keyword `portfast` to describe a port that is automatically set to forwarding when the port is brought up. The `mstp` equivalent is `mstpctl-portadmedge`. For more comparison [please read this knowledge base article](#).

Examples are included below:

Parameter	Description
maxage	<p>Sets the bridge's <i>maximum age</i> to <code><max_age></code> seconds. The default is 20.</p> <p>The maximum age must meet the condition $2 * (\text{Bridge Forward Delay} - 1 \text{ second}) \geq \text{Bridge Max Age}$.</p> <p>To set this parameter persistently, configure it under the bridge stanza:</p> <pre>mstpctl-maxage 24</pre> <p>To set this parameter at runtime, use:</p> <pre>mstpctl setmaxage <bridge> <max_age></pre> <pre>cumulus@switch:~\$ sudo mstpctl setmaxage br2 24</pre>
ageing	<p>Sets the Ethernet (MAC) address <i>ageing time</i> in <code><time></code> seconds for the bridge when the running version is STP, but not RSTP/MSTP. The default is 300.</p> <p>To set this parameter persistently, configure it under the bridge stanza:</p> <pre>mstpctl-ageing 240</pre> <p>To set this parameter at runtime, use:</p>

Parameter	Description
	<pre>mstpctl setageing <bridge> <time></pre> <pre>cumulus@switch:~\$ sudo mstpctl setageing br2 240</pre>
fdelay	<p>Sets the bridge's <i>bridge forward delay</i> to <time> seconds. The default is 15.</p> <p>The bridge forward delay must meet the condition $2 * (\text{Bridge Forward Delay} - 1 \text{ second}) \geq \text{Bridge Max Age}$.</p> <p>To set this parameter persistently, configure it under the bridge stanza:</p> <pre>mstpctl-fdelay 15</pre> <p>To set this parameter at runtime, use:</p> <pre>mstpctl setfdelay <bridge> <time></pre> <pre>cumulus@switch:~\$ sudo mstpctl setfdelay br2 15</pre>
maxhops	<p>Sets the bridge's <i>maximum hops</i> to <max_hops>. The default is 20.</p> <p>To set this parameter persistently, configure it under the bridge stanza:</p> <pre>mstpctl-maxhops 24</pre> <p>To set this parameter at runtime, use:</p> <pre>mstpctl setmaxhops <bridge> <max_hops></pre> <pre>cumulus@switch:~\$ sudo mstpctl setmaxhops br2 24</pre>
txholdcount	<p>Sets the bridge's <i>bridge transmit hold count</i> to <tx_hold_count>. The default is 6.</p> <p>To set this parameter persistently, configure it under the bridge stanza:</p>

Parameter	Description
	<pre>mstpctl-txholdcount 6</pre> <p>To set this parameter at runtime, use:</p> <pre>mstpctl settxholdcount <bridge> <tx_hold_count></pre> <pre>cumulus@switch:~\$ sudo mstpctl settxholdcount br2 5</pre>
forcevers	<p>Sets the bridge's <i>force STP version</i> to either RSTP/STP. MSTP is not supported currently. The default is <i>RSTP</i>.</p> <p>To set this parameter persistently, configure it under the bridge stanza:</p> <pre>mstpctl-forcevers rstp</pre> <p>To set this parameter at runtime, use:</p> <pre>mstpctl setforcevers <bridge> {mstp rstp stp}</pre> <pre>cumulus@switch:~\$ sudo mstpctl setforcevers br2 rstp</pre>
treeprio	<p>Sets the bridge's <i>tree priority</i> to <priority> for an MSTI instance. The priority value is a number between 0 and 65535 and must be a multiple of 4096. The bridge with the lowest priority is elected the <i>root bridge</i>. The default is 32768.</p> <div style="border: 1px solid red; padding: 5px; margin: 10px 0;">  For <code>msti</code>, only 0 is supported currently. </div> <p>To set this parameter persistently, configure it under the bridge stanza:</p> <pre>mstpctl-treeprio 8192</pre> <p>To set this parameter at runtime, use:</p> <pre>mstpctl settreeprio <bridge> <mstid> <priority></pre>

Parameter	Description
	<pre>cumulus@switch:~\$ sudo mstpctl settreeprio br2 0 8192</pre>
treeportprio	<p>Sets the <i>priority</i> of port <code><port></code> to <code><priority></code> for the MSTI instance. The priority value is a number between 0 and 240 and must be a multiple of 16. The default is 128.</p> <div>  For <code>msti</code>, only 0 is supported currently. </div> <p>To set this parameter persistently, configure it under the bridge stanza:</p> <pre>mstpctl-treeportprio swp4.101 64</pre> <p>To set this parameter at runtime, use:</p> <pre>mstpctl settreeportprio <bridge> <port> <mstid> <priority></pre> <pre>cumulus@switch:~\$ sudo mstpctl settreeportprio br2 swp4.101 0 64</pre>
hello	<p>Sets the bridge's <i>bridge hello time</i> to <code><time></code> seconds. The default is 2.</p> <p>To set this parameter persistently, configure it under the bridge stanza:</p> <pre>mstpctl-hello 20</pre> <p>To set this parameter at runtime, use:</p> <pre>mstpctl sethello <bridge> <time></pre> <pre>cumulus@switch:~\$ sudo mstpctl sethello br2 20</pre>
portpathcost	<p>Sets the <i>port cost</i> of the port <code><port></code> in bridge <code><bridge></code> to <code><cost></code>. The default is 0. <code>mstp</code>d supports only long mode; that is, 32 bits for the path cost.</p> <p>To set this parameter persistently, configure it under the bridge stanza:</p>

Parameter	Description
	<pre>mstpctl-portpathcost swp1.101=10</pre> <p>To set this parameter at runtime, use:</p> <pre>mstpctl setportpathcost <bridge> <port> <cost></pre> <pre>cumulus@switch:~\$ sudo mstpctl setportpathcost br2 swp1.101 10</pre>
portadmedge	<p>Enables/disables the <i>initial edge state</i> of the port <port> in bridge <bridge>. The default is <i>no</i>.</p> <p>To set this parameter persistently, configure it under the bridge stanza:</p> <pre>mstpctl-portadmedge swp1.101=yes</pre> <p>To set this parameter at runtime, use:</p> <pre>mstpctl setportadmedge <bridge> <port> {yes no}</pre> <pre>cumulus@switch:~\$ sudo mstpctl setportadmedge br2 swp1.101 yes</pre>
portautoedge	<p>Enables/disables the <i>auto transition</i> to/from the edge state of the port <port> in bridge <bridge>. The default is <i>yes</i>.</p> <p>To set this parameter persistently, configure it under the bridge stanza:</p> <pre>mstpctl-portautoedge swp1.101=no</pre> <p>To set this parameter at runtime, use:</p> <pre>mstpctl setportautoedge <bridge> <port> {yes no}</pre>

Parameter	Description
	<pre>cumulus@switch:~\$ sudo mstpctl setportautoedge br2 swp1.101 no</pre>
portp2p	<p>Enables/disables the <i>point-to-point detection mode</i> of the port <port> in bridge <bridge>. The default is <i>auto</i>.</p> <p>To set this parameter persistently, configure it under the bridge stanza:</p> <pre>mstpctl-portp2p swp1.101=no</pre> <p>To set this parameter at runtime, use:</p> <pre>mstpctl setportp2p <bridge> <port> {yes no auto}</pre> <pre>cumulus@switch:~\$ sudo mstpctl setportp2p br2 swp1.101 no</pre>
portrestrrole	<p>Enables/disables the ability of the port <port> in bridge <bridge> to take the <i>root role</i>. The default is <i>no</i>.</p> <p>To set this parameter persistently, configure it under the bridge stanza:</p> <pre>mstpctl-portrestrrole swp1.101=no</pre> <p>To set this parameter at runtime, use:</p> <pre>mstpctl setportrestrrole <bridge> <port> {yes no}</pre> <pre>cumulus@switch:~\$ sudo mstpctl setportrestrrole br2 swp1.101 yes</pre>
portrestrtcn	<p>Enables/disables the ability of the port <port> in bridge <bridge> to propagate <i>received topology change notifications</i>. The default is <i>no</i>.</p> <p>To set this parameter persistently, configure it under the bridge stanza:</p>

Parameter	Description
	<pre>mstpctl-portrestrtcn swp1.101=yes</pre> <p>To set this parameter at runtime, use:</p> <pre>mstpctl setportrestrtcn <bridge> <port> {yes no}</pre> <pre>cumulus@switch:~\$ sudo mstpctl setportrestrtcn br2 swp1.101 yes</pre>
portnetwork	<p>Enables/disables the <i>bridge assurance capability</i> for a network port <port> in bridge <bridge>. The default is <i>no</i>.</p> <p>To set this parameter persistently, configure it under the bridge stanza:</p> <pre>mstpctl-portnetwork swp4.101=yes</pre> <p>To set this parameter at runtime, use:</p> <pre>mstpctl setportnetwork <bridge> <port> {yes no}</pre> <pre>cumulus@switch:~\$ sudo mstpctl setportnetwork br2 swp4.101 yes</pre>
bpduguard	<p>Enables/disables the <i>BPDU guard configuration</i> of the port <port> in bridge <bridge>. The default is <i>no</i>.</p> <p>To set this parameter persistently, configure it under the bridge stanza:</p> <pre>mstpctl-bpduguard swp1=no</pre> <p>To set this parameter at runtime, use:</p> <pre>mstpctl setbpduguard <bridge> <port> {yes no}</pre>

Parameter	Description
	<pre>cumulus@switch:~\$ sudo mstpctl setbpduguard br2 swp1.101 yes</pre>
portbpdufilter	<p>Enables/disables the <i>BPDU filter</i> functionality for a port <port> in bridge <bridge>. The default is <i>no</i>.</p> <p>To set this parameter persistently, configure it under the bridge stanza:</p> <pre>mstpctl-bpdufilter swp4.101=yes</pre> <p>To set this parameter at runtime, use:</p> <pre>mstpctl setportbpdufilter <bridge> <port> {yes no}</pre> <pre>cumulus@switch:~\$ sudo mstpctl setportbpdufilter br2 swp4.101 yes</pre>

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 (in error), you can enable bridge assurance on the port by enabling port type network. The port would be in a bridge assurance inconsistent state until a BPDU is received from the peer. You need to configure the port type network on both the ends of the link:

```
cumulus@switch:~$ sudo mstpctl setportnetwork br1007 swp1.1007 yes

cumulus@switch:~$ sudo mstpctl showportdetail br1007 swp1.1007 | grep
network
network port          yes                      BA inconsistent      yes

cumulus@switch:~$ sudo grep -in assurance /var/log/daemon.log | grep mstp
1365:Jun 25 18:03:17 mstpd: br1007:swp1.1007 Bridge assurance inconsistent
```

BPDU Guard

To protect the spanning tree topology from unauthorized switches affecting the forwarding path, you can configure *BPDU guard* (Bridge Protocol Data Unit). One very common example is when someone hooks up a new switch to an access port off of a leaf switch. If this new switch is configured with a low priority, it could become the new root switch and affect the forwarding path for the entire Layer 2 topology.

Configuring BPDU Guard

You configure BPDU guard under the bridge stanza in `/etc/network/interfaces`:

```
auto br2
iface br2 inet static
    bridge-ports swp1 swp2 swp3 swp4 swp5 swp6
    bridge-stp on
    mstpctl-bpduguard swp1=yes swp2=yes swp3=yes swp4=yes
```

To load the new configuration, run `ifreload -a`:

```
cumulus@switch:~$ sudo ifreload -a
```

Non-Persistent Configuration

You can also configure BPDU guard on an individual port using a runtime configuration.

Runtime Configuration (Advanced)



A runtime configuration is non-persistent, which means the configuration you create here does not persist after you reboot the switch.

```
cumulus@switch:~$ sudo mstpctl setbpduguard br2 swp1 yes
cumulus@switch:~$ sudo mstpctl setbpduguard br2 swp2 yes
cumulus@switch:~$ sudo mstpctl setbpduguard br2 swp3 yes
cumulus@switch:~$ sudo mstpctl setbpduguard br2 swp4 yes
```

Recovering a Port Disabled by BPDU Guard

If a BPDU is received on the port, STP will bring down the port and log 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, run `mstpctl showportdetail <bridge name>`:

```
cumulus@switch:~$ sudo mstpctl showportdetail br2 swp1 | grep guard
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 un-shut or bring up the port with `sudo ifup [port]`, as shown in the example below:



Bringing up the disabled port does not fix the problem if the configuration on the connected end-station has not been rectified.

```
cumulus@leaf2$ mstpctl showportdetail bridge bond0
bridge:bond0 CIST info
  enabled          no          role
Disabled
  port id          8.001       state
discarding
  external port cost 305          admin external cost 0
  internal port cost 305          admin internal cost 0
  designated root    8.000.6C:64:1A:00:4F:9C dsgrn external cost 0
  dsgrn regional root 8.000.6C:64:1A:00:4F:9C dsgrn internal cost 0
  designated bridge  8.000.6C:64:1A:00:4F:9C designated port    8.00
1
  admin edge port    no          auto edge port      yes
  oper edge port     no          topology change ack no
  point-to-point     yes         admin point-to-point auto
  restricted role     no          restricted TCN       no
  port hello time    10          disputed            no
  bpdu guard port    yes         bpdu guard error     yes
  network port       no          BA inconsistent      no
  Num TX BPDU        3           Num TX TCN           2
  Num RX BPDU        488         Num RX TCN           2
  Num Transition FWD 1           Num Transition BLK    2
  bpdufilter port    no
  clag ISL           no          clag ISL Oper UP     no
  clag role          unknown      clag dual conn mac   0:0:
0:0:0:0
  clag remote portID F.FFF        clag system mac      0:0:
0:0:0:0
```

```
cumulus@leaf2$ sudo ifup bond0
```

```
cumulus@leaf2$ mstpctl showportdetail bridge bond0
bridge:bond0 CIST info
  enabled          yes          role          Root
  port id          8.001       state
forwarding
  external port cost 305          admin external cost 0
  internal port cost 305          admin internal cost 0
  designated root    8.000.6C:64:1A:00:4F:9C dsgrn external cost 0
  dsgrn regional root 8.000.6C:64:1A:00:4F:9C dsgrn internal cost 0
```

```

1 designated bridge 8.000.6C:64:1A:00:4F:9C designated port 8.00
admin edge port no auto edge port yes
oper edge port no topology change ack no
point-to-point yes admin point-to-point auto
restricted role no restricted TCN no
port hello time 2 disputed no
bpdu guard port no bpdu guard error no
network port no BA inconsistent no
Num TX BPDU 3 Num TX TCN 2
Num RX BPDU 43 Num RX TCN 1
Num Transition FWD 1 Num Transition BLK 0
bpdufilter port no
clag ISL no clag ISL Oper UP no
clag role unknown clag dual conn mac 0:0:
0:0:0:0
clag remote portID F.FFF clag system mac 0:0:
0:0:0:0

```

BPDU Filter

You can enable `bpdufilter` on a switch port, which filters BPDUs in both directions. This effectively disables STP on the port.

To enable it, add the following to `/etc/network/interfaces` under the `bridge port iface` section example:

```

auto br100
iface br100
    bridge-ports swp1.100 swp2.100
    mstpctl-portbpdufilter swp1=yes swp2=yes

```

To load the new configuration from `/etc/network/interfaces`, run `ifreload -a`:

```
cumulus@switch:~$ sudo ifreload -a
```

For more information, see `man(5) ifupdown-addons-interfaces`.

Runtime Configuration (Advanced)



A runtime configuration is non-persistent, which means the configuration you create here does not persist after you reboot the switch.

To enable BPDU filter at runtime, run `mstpctl`:


```
cumulus@switch:~$ sudo mstpctl setportbpdufilter br100 swp1.100=yes swp2.100=yes
```

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.

You configure storm control in `/etc/cumulus/switchd.conf`. The configuration persists across reboots and restarting `switchd`. If you change the storm control configuration in this file after rebooting the switch, you must **restart `switchd`** to activate the new configuration.

You configure storm control for each physical port. For example, to enable broadcast and multicast storm control at 400 packets per second (pps) and 3000 pps, respectively, for `swp1`, edit `/etc/cumulus/switchd.conf` and configure it as follows:

```
# Storm Control setting on a port, in pps, 0 means disable
interface.swp1.storm_control.broadcast = 400
interface.swp1.storm_control.multicast = 3000
```

Configuration Files

- `/etc/network/interfaces`

Man Pages

- `brctl(8)`
- `bridge-utils-interfaces(5)`
- `ifupdown-addons-interfaces(5)`
- `mstpctl(8)`
- `mstpctl-utils-interfaces(5)`

Useful Links

The source code for `mstpd/mstpctl` was written by [Vitalii Demianets](#) and is hosted at the sourceforge URL below.

- <https://sourceforge.net/projects/mstpd/>
- http://en.wikipedia.org/wiki/Spanning_Tree_Protocol

Caveats and Errata

- MSTP is not supported currently. However, interoperability with MSTP networks can be accomplished using PVRSTP or PVSTP.

Link Layer Discovery Protocol

The `lldpd` daemon implements the IEEE802.1AB (Link Layer Discovery Protocol, or LLDP) standard. LLDP allows you to know which ports are neighbors of a given port. By default, `lldpd` runs as a daemon and is started at system boot. `lldpd` command line arguments are placed in `/etc/default/lldpd`. `lldpd` configuration options are placed in `/etc/lldpd.conf` or under `/etc/lldpd.d/`.

For more details on the command line arguments and config options, please see `man lldpd(8)`.

`lldpd` supports CDP (Cisco Discovery Protocol, v1 and v2). `lldpd` logs by default into `/var/log/daemon.log` with an `lldpd` prefix.

`lldpcli` is the CLI tool to query the `lldpd` daemon for neighbors, statistics and other running configuration information. See `man lldpcli(8)` for details.

Contents

(Click to expand)

- [Contents \(see page 106\)](#)
- [Commands \(see page 106\)](#)
- [Man Pages \(see page 106\)](#)
- [Configuring LLDP \(see page 106\)](#)
- [Example lldpcli Commands \(see page 107\)](#)
- [Enabling the SNMP Subagent in LLDP \(see page 111\)](#)
- [Configuration Files \(see page 111\)](#)
- [Useful Links \(see page 111\)](#)
- [Caveats and Errata \(see page 111\)](#)

Commands

- `lldpd` (daemon)
- `lldpcli` (interactive CLI)

Man Pages

- `man lldpd`
- `man lldpcli`

Configuring 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-blacklist "eth0"
```

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 see all neighbors on all ports/interfaces:

```
cumulus@switch:~$ sudo lldpcli show neighbors
-----
LLDP neighbors:
-----
Interface:      eth0, via: CDPv1, RID: 72, Time: 0 day, 00:33:40
Chassis:
  ChassisID:    local test-server-1
  SysName:      test-server-1
  SysDescr:     Linux running on
Linux 3.2.2+ #1 SMP Mon Jun 10 16:21:22 PDT 2013 ppc
  MgmtIP:       192.0.2.72
  Capability:   Router, on
Port:
  PortID:       ifname eth1
-----
Interface:      swp1, via: CDPv1, RID: 87, Time: 0 day, 00:36:27
nChassis:
  ChassisID:    local T1
  SysName:      T1
  SysDescr:     Linux running on
Cumulus RMP
  MgmtIP:       192.0.2.15
```

```
Capability: Router, on
Port:
  PortID: ifname swp1
  PortDescr: swp1
-----
... and more (output truncated to fit this doc)
```

To see neighbors on specific ports:

```
cumulus@switch:~$ sudo lldpcli show neighbors ports swp1,swp2
-----
Interface: swp1, via: CDPv1, RID: 87, Time: 0 day, 00:36:27
Chassis:
  ChassisID: local T1
  SysName: T1
  SysDescr: Linux running on
Cumulus RMP
  MgmtIP: 192.0.2.15
  Capability: Router, on
Port:
  PortID: ifname swp1
  PortDescr: swp1
-----
Interface: swp2, via: CDPv1, RID: 123, Time: 0 day, 00:36:27
Chassis:
  ChassisID: local T2
  SysName: T2
  SysDescr: Linux running on
Cumulus RMP
  MgmtIP: 192.0.2.15
  Capability: Router, on
Port:
  PortID: ifname swp1
  PortDescr: swp1
```

To see 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
-----

... and more (output truncated to fit this document)

```

To see 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 see the `lldpd` running configuration:

```
cumulus@switch:~$ sudo lldpcli show running-configuration
```

Global configuration:

Configuration:

```
Transmit delay: 1
Transmit hold: 4
Receive mode: no
Pattern for management addresses: (none)
Interface pattern: (none)
Interface pattern for chassis ID: (none)
Override description with: (none)
Override platform with: (none)
Advertise version: yes
Disable LLDP-MED inventory: yes
LLDP-MED fast start mechanism: yes
LLDP-MED fast start interval: 1
```

Runtime Configuration (Advanced)



A runtime configuration does not persist when you reboot the switch — all changes are lost.

To configure active interfaces:

```
lldpcli configure system interface pattern "swp*"
```

To configure inactive interfaces:

```
lldpcli configure system interface pattern-blacklist "eth0"
```



The active interface list always overrides the inactive interface list.

To reset any interface list to none:

```
lldpcli configure system interface pattern-blacklist ""
```

Enabling 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
# Uncomment to start SNMP subagent and enable CD
P, SONMP and EDP protocol
#DAEMON_ARGS="-x -c -s -e"

# Enable CDP by default
DAEMON_ARGS="-c"
DAEMON_ARGS="-x"
```

Configuration Files

- `/etc/lldpd.conf`
- `/etc/lldpd.d`
- `/etc/default/lldpd`

Useful Links

- <http://vincentbernat.github.io/lldpd/>
- http://en.wikipedia.org/wiki/Link_Layer_Discovery_Protocol

Caveats and Errata

- Annex E (and hence Annex D) of IEEE802.1AB (lldp) is not supported.

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)`.

Contents

(Click to expand)

- [Contents \(see page 112\)](#)
- [Supported Features \(see page 112\)](#)
- [Configuring PTM \(see page 113\)](#)
- [Basic Topology Example \(see page 113\)](#)
- [Advanced PTM Configuration \(see page 114\)](#)
 - [Scripts \(see page 114\)](#)
 - [Configuration Parameters \(see page 114\)](#)
- [Bidirectional Forwarding Detection \(BFD\) \(see page 118\)](#)
 - [Configuring BFD \(see page 118\)](#)
 - [Echo Function \(see page 118\)](#)
- [Scripts \(see page 119\)](#)
- [Using ptmd Service Commands \(see page 120\)](#)
- [Using ptmctl Commands \(see page 120\)](#)
 - [ptmctl Examples \(see page 120\)](#)
 - [ptmctl Error Outputs \(see page 122\)](#)
- [Configuration Files \(see page 123\)](#)
- [Useful Links \(see page 123\)](#)

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, like `swp1` or `eth0`, are currently supported. Cumulus RMP does not support specifying virtual interfaces like bonds or subinterfaces like `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 RMP, [see below](#) (see page 118) and see `man ptmd(8)`.
- 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](#) (see page 113).

Configuring PTM

`ptmd` verifies the physical network topology against a DOT-specified network graph file, `/etc/ptm.d/topology.dot`. This file must be present or else `ptmd` will not start. You can specify an alternate file using the `-c` option.



This file must be present or else `ptmd` will not start. You can specify an alternate file using the `-c` option.

PTM also 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, then it is a PASS, else it is a FAIL.

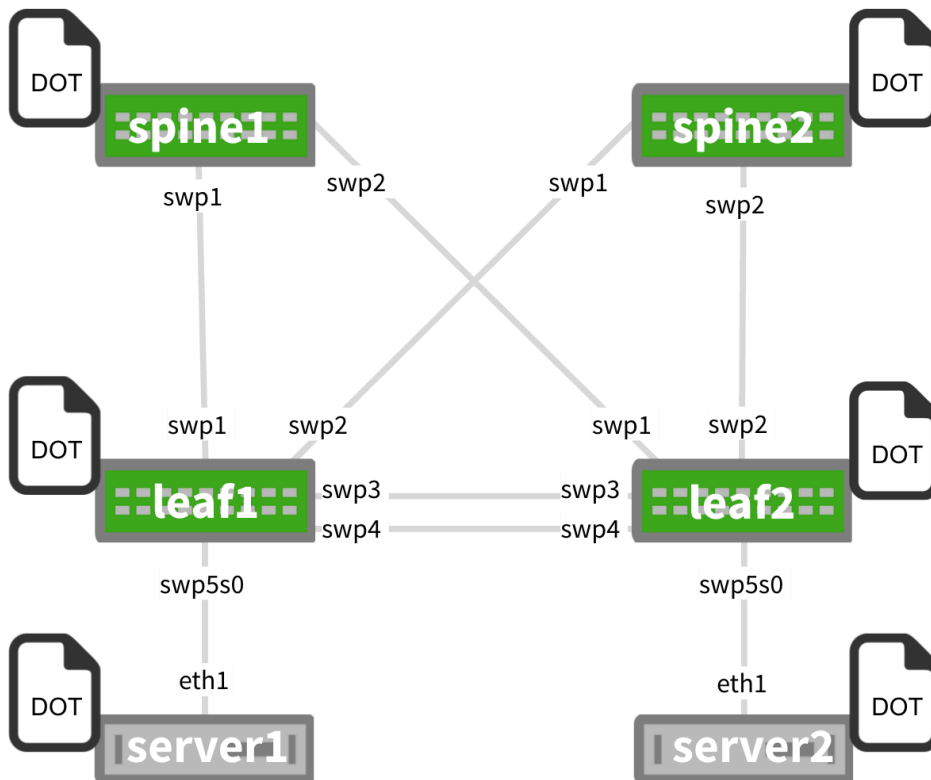


PTM performs its LLDP neighbor check using the PortID ifname TLV information. Previously, it used the PortID port description TLV information.

Basic Topology Example

This is a basic example DOT file and its corresponding topology diagram. You should use the same `topology.dot` file on all switches, and don't 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";
}
```



Advanced PTM Configuration

PTM allows for more advanced configuration of the topology file using parameters you specify in the topology file.

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, running `if-topo-pass` when an LLDP or BFD check passes and running `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](#) section below (see page).

You should 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 whether PTM should use 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 will ignore the FQDN and only look for `switch04`, since that is the host name of the switch it's running on:



It's a good idea to always wrap the hostname in double quotes, like "www.example.com". Otherwise, `ptmd` can fail if you specify a fully-qualified domain name as the hostname and do not wrap it in double quotes.

```
graph G {
    hostnametype="hostname"
    BFD="upMinTx=150,requiredMinRx=250"
    "cumulus":swp44 -- "switch04.cumulusnetworks.com":swp20
    "cumulus":swp46 -- "switch04.cumulusnetworks.com":swp22
}
```

However, in this next example, PTM will compare using the FQDN and look for `switch05.cumulusnetworks.com`, which is the FQDN of the switch it's running on:

```
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`, which specifies a custom parameter tuple for BFD.
- `lldptmpl`, which 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 while BFD1 and BFD2 are template for BFD parameters.

Supported BFD and LLDP Parameters

`ptmd` supports the following BFD parameters:

- `upMinTx`: the minimum transmit interval, which defaults to 300ms, specified in milliseconds.

- `requiredMinRx`: the minimum interval between received BFD packets, which defaults to 300ms, specified in milliseconds.
- `detectMult`: the detect multiplier, which defaults to 3, and can be any non-zero value.
- `afi`: 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"]
}
```



When you specify `match_hostname=fqdn`, `ptmd` will match the entire FQDN, like `cumulus-2.domain.com` in the example below. If you do not specify anything for `match_hostname`, `ptmd` will match based on hostname only, like `cumulus-3` below, and ignore 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.



BFD requires an IP address for any interface for 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.



You cannot specify BFD multihop sessions in the `topology.dot` file since you cannot specify the source and destination IP address pairs in that file.

Configuring BFD

You configure BFD by specifying the configuration in the `topology.dot` file. However, the topology file has some limitations:

- The `topology.dot` file supports creating 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 will not be created.

Echo Function

Cumulus RMP supports the *echo function* for IPv4 single hops only, and with the a synchronous operating mode only (Cumulus RMP does not support demand mode).

You use the echo function primarily to test the forwarding path on a remote system. To enable the echo function, set `echoSupport` to 1 in the topology file.

Once 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 of milliseconds.

You can use more aggressive detection times for echo packets since the round-trip time is reduced because they are accessing the forwarding path. You configure the detection interval by setting the `echoMinRx` parameter in the topology file to a non-zero value of milliseconds; the minimum setting is 50 milliseconds. Once 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 RMP are 8 bytes long and have the following format:

0	1	2	3
Version	Length	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.

Transmitting and Receiving 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.

Using 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:** 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:** The minimum interval between transmitting BFD control packets when the echo packets are being exchanged.

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, running `if-topo-pass` when an LLDP or BFD check passes and running `if-topo-fails` when the check fails. The scripts receive an argument string that is the result of the `ptmctl` command, described in ptmd Commands below.

You should modify these default scripts as needed.

Using ptmd Service Commands

PTM sends client notifications in CSV format.

`cumulus@switch:~$ sudo systemctl start|restart|force-reload ptmd.service`: Starts or restarts the `ptmd` service. The `topology.dot` file must be present in order for the service to start.

`cumulus@switch:~$ sudo systemctl reload ptmd.service`: Instructs `ptmd` to read the `topology.dot` file again without restarting, applying the new configuration to the running state.

`cumulus@switch:~$ sudo systemctl stop ptmd.service`: Stops the `ptmd` service.

`cumulus@switch:~$ sudo systemctl status ptmd.service`: Retrieves the current running state of `ptmd`.

Using ptmctl Commands

`ptmctl` is a client of `ptmd`; it retrieves the daemon's operational state. It connects to `ptmd` over a Unix socket and listens for notifications. `ptmctl` parses the CSV notifications sent by `ptmd`.

See `man ptmctl` for more information.

ptmctl Examples

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      BFD      BFD      det_mult tx_timeout rx_timeout
echo_tx_timeout echo_rx_timeout max_hop_cnt
      status nbr      nbr      on      upd
Type  state  peer  DownDiag
```



```

-----
swp45 pass h1:swp1 h1:swp1 h1 swp1 swp1 IfName 5m: 5s N
/A N/A N/A N/A N/A N/A N/A
/A N/A N/A N/A
swp46 fail h2:swp1 h2:swp1 h2 swp1 swp1 IfName 5m: 5s N
/A N/A N/A N/A N/A N/A N/A
/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` options (results are for an IPv6-connected peer):

```

cumulus@switch:~$ sudo ptmctl -b -d

```

```

port peer          state local type      diag det  tx_timeout
rx_timeout echo      echo      max      rx_ctrl tx_ctrl rx_echo
tx_echo

mult              tx_timeout rx_timeout
hop_cnt
-----
-----
-----
swp1 fe80::202:ff:fe00:1 Up      N/A    singlehop N/A    3    300
900      0          0          N/A    187172 185986 0
0
swp1 3101:abc:bcad::2 Up      N/A    singlehop N/A    3    300
900      0          0          N/A    501    533    0
0

```

ptmctl Error Outputs

If there are errors in the topology file or there isn't a session, PTM will return 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
```

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.

Configuration Files

- `/etc/ptm.d/topology.dot`
- `/etc/ptm.d/if-topo-pass`
- `/etc/ptm.d/if-topo-fail`

Useful Links

- [Bidirectional Forwarding Detection \(BFD\)](#)
- [Graphviz](#)
- [LLDP on Wikipedia](#)
- [PTMd GitHub repo](#)

Bonding - Link Aggregation

Linux bonding provides a method for aggregating multiple network interfaces (the slaves) into a single logical bonded interface (the bond). Cumulus RMP bonding supports the IEEE 802.3ad link aggregation mode. Link aggregation allows one or more links to be aggregated together to form a *link aggregation group* (LAG), such that a media access control (MAC) client can treat the link aggregation group as if it were a single link. The benefits of link aggregation are:

- Linear scaling of bandwidth as links are added to LAG
- Load balancing
- Failover protection

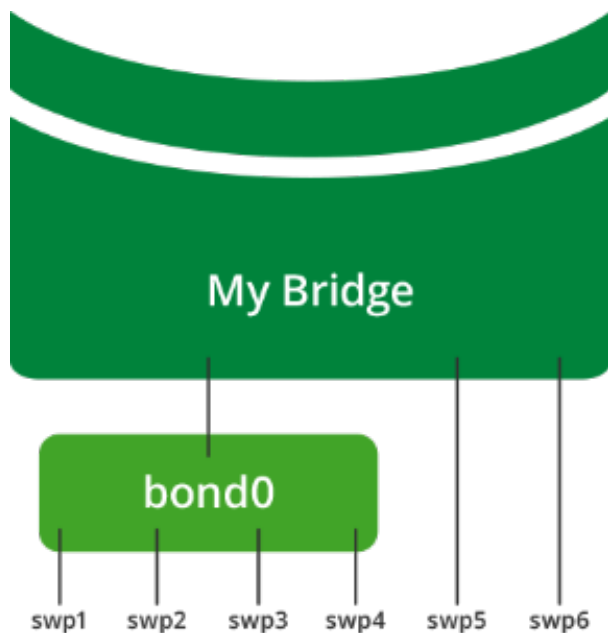
The Cumulus RMP LAG control protocol is LACP version 1.

Contents

(Click to expand)

- [Contents \(see page 123\)](#)
- [Example: Bonding 4 Slaves \(see page 124\)](#)
- [Hash Distribution \(see page 126\)](#)
- [Configuration Files \(see page 126\)](#)
- [Useful Links \(see page 126\)](#)
- [Caveats and Errata \(see page 126\)](#)

Example: Bonding 4 Slaves



In this example, front panel port interfaces swp1-swp4 are slaves in bond0 (swp5 and swp6 are not part of bond0).

The name of the bond is arbitrary as long as it follows Linux interface naming guidelines, and is unique within the switch. The only bonding mode supported in Cumulus RMP is 802.3ad. There are several 802.3ad settings that can be applied to each bond.

All of the following settings except for `bond-slaves` are set to the recommended defaults and should only be added to a configuration in `/etc/network/interfaces` if you plan to use a different setting.

- `bond-slaves`: The list of slaves in bond.
- `bond-mode`: Is set to *802.3ad* by default and **must not** be changed.
- `bond-miimon`: How often the link state of each slave is inspected for link failures. It *100*, the recommended value.
- `bond-use-carrier`: How to determine link state. It defaults to *1*.
- `bond-xmit-hash-policy`: Hash method used to select the slave for a given packet; it defaults to *layer3+4* and **must not** be changed.
- `bond-lacp-rate`: Rate to ask link partner to transmit LACP control packets. It defaults to *1*.
- `bond-min-links`: Specifies the minimum number of links that must be active before asserting carrier on the bond. It defaults to *1*, but a value greater than 1 is useful if higher level services need to ensure a minimum of aggregate bandwidth before putting the bond in service. Keeping `bond-min-links` set to *1* indicates the bond must have at least one active member for bond to assert carrier. If the number of active members drops below the `bond-min-links` setting, the bond will appear 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 will become *link-up*.

See Useful Links below for more details on settings.

To configure the bond, edit `/etc/network/interfaces` and add a stanza for bond0:

```
auto bond0
iface bond0
    address 10.0.0.1/30
    bond-slaves swp1 swp2 swp3 swp4
```

However, if you are intending that the bond become part of a bridge, you don't need to specify an IP address. The configuration would look like this:

```
auto bond0
iface bond0
    bond-slaves glob swp1-4
```

See `man interfaces` for more information on `/etc/network/interfaces`.

When networking is started on switch, `bond0` is created as MASTER and interfaces `swp1-sw4` come up in SLAVE mode, as seen in the `ip link show` command:

```
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
```

And

```
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
```



All slave interfaces within a bond will have the same MAC address as the bond. Typically, the first slave added to the bond donates its MAC address for the bond. The other slaves' MAC addresses are set to the bond MAC address. The bond MAC address is used as source MAC address for all traffic leaving the bond, and provides a single destination MAC address to address traffic to the bond.

Hash Distribution

Egress traffic through a bond is distributed to a slave based on a packet hash calculation. This distribution provides load balancing over the slaves. The hash calculation uses packet header data to pick 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. Traffic for a given conversation flow will always hash to the same slave. Many flows will be distributed over all the slaves to load balance the total traffic. In a failover event, the hash calculation is adjusted to steer traffic over available slaves.

Configuration Files

- `/etc/network/interfaces`

Useful Links

- <http://www.linuxfoundation.org/collaborate/workgroups/networking/bonding>
- 802.3ad (Accessible writeup)
- Link aggregation from Wikipedia

Caveats and Errata

- An interface cannot belong to multiple bonds.
- Slave ports within a bond should all be set to the same speed/duplex, and should match the link partner's slave ports.
- A bond cannot enslave VLAN subinterfaces. A bond can have subinterfaces, but not the other way around.

Ethernet Bridging - VLANs

Ethernet bridges provide a means for hosts to communicate at layer 2. Bridge members can be individual physical interfaces, bonds or logical interfaces that traverse an 802.1Q VLAN trunk.

Cumulus RMP has two modes for configuring bridges: *VLAN-aware* (see page 147) and *traditional*. The bridge driver in Cumulus RMP is capable of VLAN filtering, which allows for configurations that are similar to incumbent network devices. While Cumulus RMP supports Ethernet bridges in traditional mode Cumulus Networks recommends using *VLAN-aware* mode.

For a comparison of traditional and VLAN-aware modes, read [this knowledge base article](#).



You can configure both VLAN-aware and traditional mode bridges on the same network in Cumulus RMP; however you should not have more than one VLAN-aware bridge on a given switch.

Contents

(Click to expand)

- Contents (see page 127)
 - Configuration Files (see page 127)
 - Commands (see page 127)
 - Creating a Bridge between Physical Interfaces (see page 127)
 - Creating the Bridge and Adding Interfaces (see page 128)
 - Showing and Verifying the Bridge Configuration (see page 129)
 - Bridge Interface MAC Address and MTU (see page 130)
 - Examining MAC Addresses (see page 130)
 - Multiple Bridges (see page 131)
 - Configuring an SVI (Switch VLAN Interface) (see page 133)
 - Showing and Verifying the Bridge Configuration (see page 135)
 - Using Trunks in Traditional Bridging Mode (see page 136)
 - Trunk Example (see page 137)
 - Showing and Verifying the Trunk (see page 138)
 - Additional Examples (see page 138)
 - Configuration Files (see page 138)
 - Useful Links (see page 139)
 - Caveats and Errata (see page 139)

Configuration Files

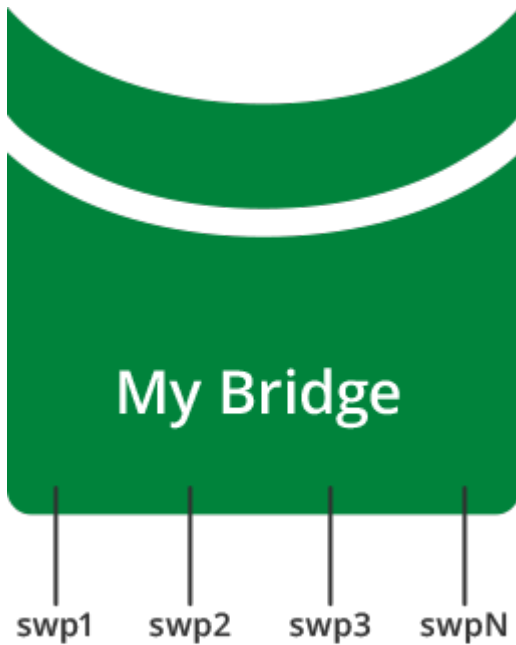
- /etc/network/interfaces

Commands

- brctl
- bridge
- ip addr
- ip link

Creating a Bridge between Physical Interfaces

The basic use of bridging is to connect all of the physical and logical interfaces in the system into a single layer 2 domain.



Creating the Bridge and Adding Interfaces

You statically manage bridge configurations in `/etc/network/interfaces`. The following configuration snippet details an example bridge used throughout this chapter, explicitly enabling [spanning tree](#) (see page 90) and setting the bridge MAC address ageing timer. First, create a bridge with a descriptive name of 15 characters or fewer. Then add the logical interfaces (bond0) and physical interfaces (swp5, swp6) to assign to that bridge.

```
auto my_bridge
iface my_bridge
    bridge-ports bond0 swp5 swp6
    bridge-ageing 150
    bridge-stp on
```

Keyword	Explanation
bridge-ports	List of logical and physical ports belonging to the logical bridge.
bridge-ageing	Maximum amount of time before a MAC addresses learned on the bridge expires from the bridge MAC cache. The default value is 300 seconds.
bridge-stp	Enables spanning tree protocol on this bridge. The default spanning tree mode is Per VLAN Rapid Spanning Tree Protocol (PVRST). For more information on spanning-tree configurations see the configuration section: Spanning Tree and Rapid Spanning Tree (see page 90).

To bring up the bridge my_bridge, use the `ifreload` command:

```
cumulus@switch:~$ sudo ifreload -a
```

Runtime Configuration (Advanced)



A runtime configuration is non-persistent, which means the configuration you create here does not persist after you reboot the switch.

To create the bridge and interfaces on the bridge, run:

```
cumulus@switch:~$ sudo brctl addbr my_bridge

cumulus@switch:~$ sudo brctl addif my_bridge bond0 swp5 swp6

cumulus@switch:~$ sudo brctl show
bridge name      bridge id        STP enabled      interfaces
my_bridge        8000.44383900129b  yes              bond0
                                                           swp5
                                                           swp6
```

```
cumulus@switch:~$ sudo ip link set up dev my_bridge
```

```
cumulus@switch:~$ sudo ip link set up dev bond0
```

```
cumulus@switch:~$ sudo for I in {5..6}; do ip link set up dev swp$I; done
```

Showing and Verifying the Bridge Configuration

```
cumulus@switch:~$ ip link show my_bridge
56: my_bridge: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue
state UP mode DEFAULT
    link/ether 44:38:39:00:12:9b brd ff:ff:ff:ff:ff:ff
```



Do not try to bridge the management port, eth0, with any switch ports (like swp0, swp1, and so forth). For example, if you created a bridge with eth0 and swp1, it will **not** work.

Using netshow to Display Bridge Information

`netshow` is a Cumulus RMP tool for retrieving information about your network configuration.

```
cumulus@switch$ netshow interface bridge
```

	Name	Speed	Mtu	Mode	Summary
UP	my_bridge	N/A	1500	Bridge/L2	Untagged: bond0, swp5-6 Root Port: bond0 VlanID: Untagged

Bridge Interface MAC Address and MTU

A 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. The bridge's MAC address is inherited from the first interface that is added to the bridge as a member. The bridge MAC address remains unchanged until the member interface is removed from the bridge, at which point the bridge will inherit from the next member interface, if any. The bridge can also be assigned an IP address, as discussed below.

Examining MAC Addresses

A bridge forwards frames by looking up the destination MAC address. A bridge learns the source MAC address of a frame when the frame enters the bridge on an interface. After the MAC address is learned, the bridge maintains an age for the MAC entry in the bridge table. The age is refreshed when a frame is seen again with the same source MAC address. When a MAC is not seen for greater than the MAC ageing time, the MAC address is deleted from the bridge table.

The following shows the MAC address table of the example bridge. Notice that the `is local?` column indicates if the MAC address is the interface's own MAC address (`is local` is `yes`), or if it is learned on the interface from a packet's source MAC (where `is local` is `no`):

```
cumulus@switch:~$ sudo brctl showmacs my_bridge
```

port	name	mac addr	is local?	ageing timer
swp4		06:90:70:22:a6:2e	no	19.47
swp1		12:12:36:43:6f:9d	no	40.50
bond0		2a:95:22:94:d1:f0	no	1.98
swp1		44:38:39:00:12:9b	yes	0.00
swp2		44:38:39:00:12:9c	yes	0.00
swp3		44:38:39:00:12:9d	yes	0.00
swp4		44:38:39:00:12:9e	yes	0.00
bond0		44:38:39:00:12:9f	yes	0.00
swp2		90:e2:ba:2c:b1:94	no	12.84
swp2		a2:84:fe:fc:bf:cd	no	9.43

You can use the `bridge fdb` command to display the MAC address table as well:

```
cumulus@switch$ bridge fdb show
70:72:cf:9d:4e:36 dev swp2 VLAN 0 master bridge-A permanent
70:72:cf:9d:4e:35 dev swp1 VLAN 0 master bridge-A permanent
70:72:cf:9d:4e:38 dev swp4 VLAN 0 master bridge-B permanent
70:72:cf:9d:4e:37 dev swp3 VLAN 0 master bridge-B permanent
```

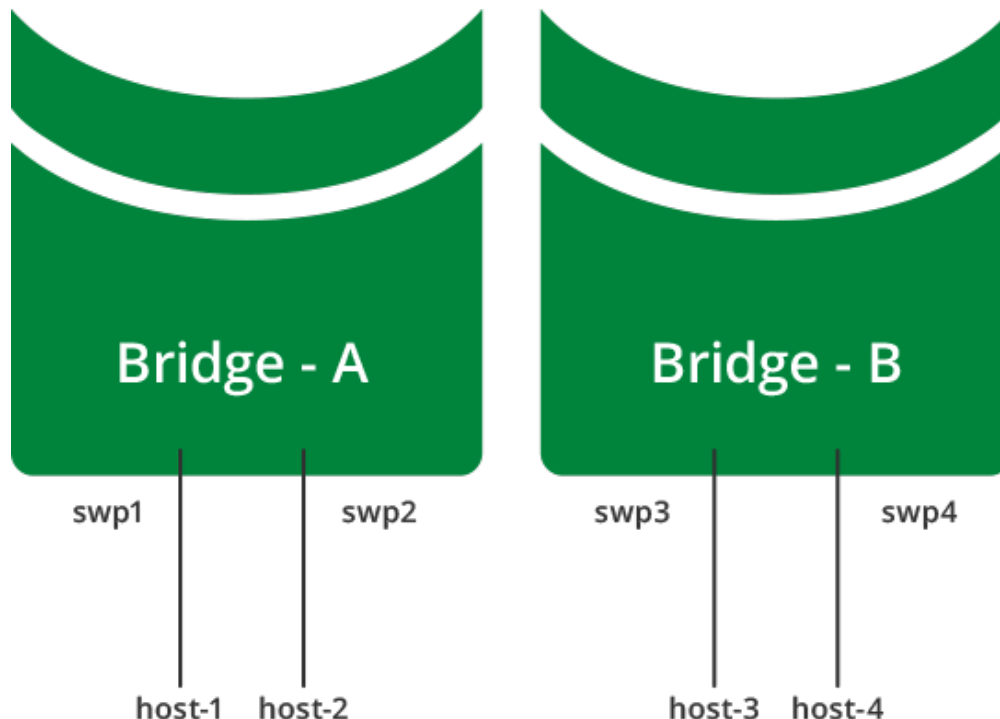


You can clear a MAC address from the table using the `bridge fdb` command:

```
cumulus@switch:~$ sudo bridge fdb del 90:e2:ba:2c:b1:94 dev swp2
```

Multiple Bridges

Sometimes it is useful to logically divide a switch into multiple layer 2 domains, so that hosts in one domain can communicate with other hosts in the same domain but not in other domains. You can achieve this by configuring multiple bridges and putting different sets of interfaces in the different bridges. In the following example, host-1 and host-2 are connected to the same bridge (bridge-A), while host-3 and host-4 are connected to another bridge (bridge-B). host-1 and host-2 can communicate with each other, so can host-3 and host-4, but host-1 and host-2 cannot communicate with host-3 and host-4.



To configure multiple bridges, edit `/etc/network/interfaces`:

```
auto bridge-A
iface bridge-A
    bridge-ports swp1 swp2
    bridge-stp on

auto my_bridge
iface my_bridge
    bridge-ports swp3 swp4
    bridge-stp on
```

To bring up the bridges bridge-A and bridge-B, use the `ifreload` command:

```
cumulus@switch:~$ sudo ifreload -a
```

Runtime Configuration (Advanced)



A runtime configuration is non-persistent, which means the configuration you create here does not persist after you reboot the switch.

```
cumulus@switch:~$ sudo brctl addbr bridge-A

cumulus@switch:~$ sudo brctl addif bridge-A swp1 swp2

cumulus@switch:~$ sudo brctl addbr bridge-B

cumulus@switch:~$ sudo brctl addif bridge-B swp3 swp4

cumulus@switch:~$ sudo for I in {1..4}; do ip link set up dev swp$I; done

cumulus@switch:~$ sudo ip link set up dev bridge-A

cumulus@switch:~$ sudo ip link set up dev bridge-B

cumulus@switch:~$ sudo brctl show
bridge name      bridge id        STP enabled      interfaces
bridge-A         8000.44383900129b  yes              swp1
                  8000.44383900129b  yes              swp2
bridge-B         8000.44383900129d  yes              swp3
                  8000.44383900129d  yes              swp4
```

```
cumulus@switch$ ip link show bridge-A
97: bridge-A: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue
state UP mode DEFAULT
    link/ether 70:72:cf:9d:4e:35 brd ff:ff:ff:ff:ff:ff
cumulus@switch$ ip link show bridge-B
98: bridge-B: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue
state UP mode DEFAULT
    link/ether 70:72:cf:9d:4e:37 brd ff:ff:ff:ff:ff:ff
```

Using netshow to Display the Bridges

netshow is a Cumulus RMP tool for retrieving information about your network configuration.

```
cumulus@switch$ netshow interface bridge
```

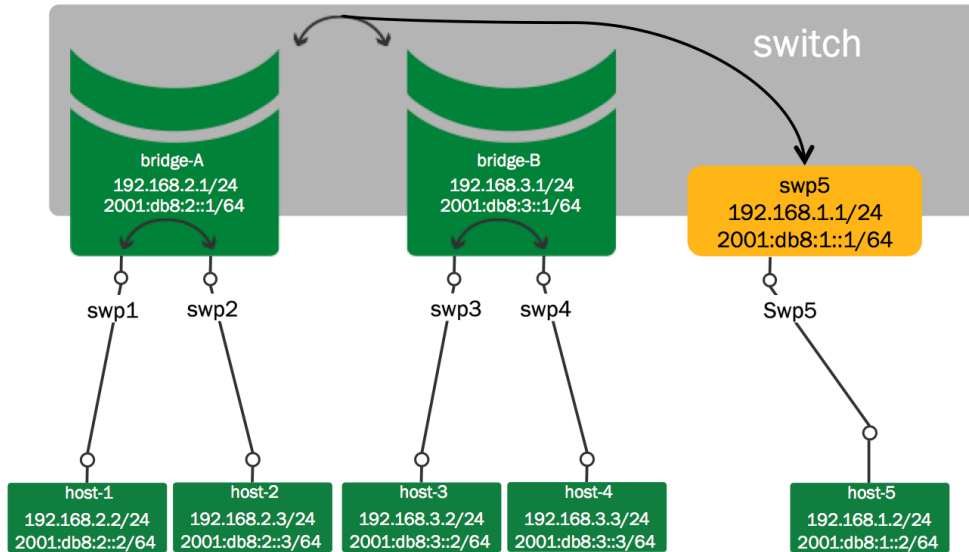
	Name	Speed	Mtu	Mode	Summary
UP	bridge-A	N/A	1500	Bridge/L2	Untagged: swp1-2 Root Port: swp2 VlanID: Untagged
UP	bridge-B	N/A	1500	Bridge/L2	Untagged: swp3-4 Root Port: swp3 VlanID: Untagged

Configuring an SVI (Switch VLAN Interface)

A bridge creates a layer 2 forwarding domain for hosts to communicate. A bridge can be assigned an IP address — typically of the same subnet as the hosts that are members of the bridge — and participate in routing topologies. This enables hosts within a bridge to communicate with other hosts outside the bridge through layer 3 routing.



When an interface is added to a bridge, it ceases to function as a router interface, and the IP address on the interface, if any, becomes reachable.



The configuration for the two bridges example looks like the following:

```
auto swp5
iface swp5
    address 192.168.1.2/24
    address 2001:DB8:1::2/64
auto bridge-A
iface bridge-A
    address 192.168.2.1/24
    address 2001:DB8:2::1/64
    bridge-ports swp1 swp2
    bridge-stp on
auto bridge-B
iface bridge-B
    address 192.168.3.1/24
    address 2001:DB8:3::1/64
    bridge-ports swp3 swp4
    bridge-stp on
```

To bring up swp5 and bridges bridge-A and bridge-B, use the `ifreload` command:

```
cumulus@switch:~$ sudo ifreload -a
```

Showing and Verifying the Bridge Configuration

```
cumulus@switch$ ip addr show bridge-A
106: bridge-A: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue
state UP
    link/ether 70:72:cf:9d:4e:35 brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.1/24 scope global bridge-A
    inet6 2001:db8:2::1/64 scope global
    valid_lft forever preferred_lft forever
    inet6 fe80::7272:cfff:fe9d:4e35/64 scope link
    valid_lft forever preferred_lft forever
```

```
cumulus@switch$ ip addr show bridge-B
107: bridge-B: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue
state UP
    link/ether 70:72:cf:9d:4e:37 brd ff:ff:ff:ff:ff:ff
    inet 192.168.3.1/24 scope global bridge-B
    inet6 2001:db8:3::1/64 scope global
    valid_lft forever preferred_lft forever
    inet6 fe80::7272:cfff:fe9d:4e37/64 scope link
    valid_lft forever preferred_lft forever
```

To see all the routes on the switch use the `ip route show` command:

```
cumulus@switch$ ip route show
192.168.1.0/24 dev swp5 proto kernel scope link src 192.168.1.2 dead
192.168.2.0/24 dev bridge-A proto kernel scope link src 192.168.2.1
192.168.3.0/24 dev bridge-B proto kernel scope link src 192.168.3.1
```

Runtime Configuration (Advanced)



A runtime configuration is non-persistent, which means the configuration you create here does not persist after you reboot the switch.

To add an IP address to a bridge:

```
cumulus@switch:~$ sudo ip addr add 192.0.2.101/24 dev bridge-A

cumulus@switch:~$ sudo ip addr add 192.0.2.102/24 dev bridge-B
```

Using netshow to Display the SVI

netshow is a Cumulus RMP tool for retrieving information about your network configuration.

```
cumulus@switch$ netshow interface bridge
```

	Name	Speed	Mtu	Mode	Summary
UP	bridge-A	N/A	1500	Bridge/L3	IP: 192.168.2.1/24, 2001:db8:2::1/64 Untagged: swp1-2 Root Port: swp2 VlanID: Untagged
UP	bridge-B	N/A	1500	Bridge/L3	IP: 192.168.3.1/24, 2001:db8:3::1/64 Untagged: swp3-4 Root Port: swp3 VlanID: Untagged

Using Trunks in Traditional Bridging Mode

The [IEEE 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.

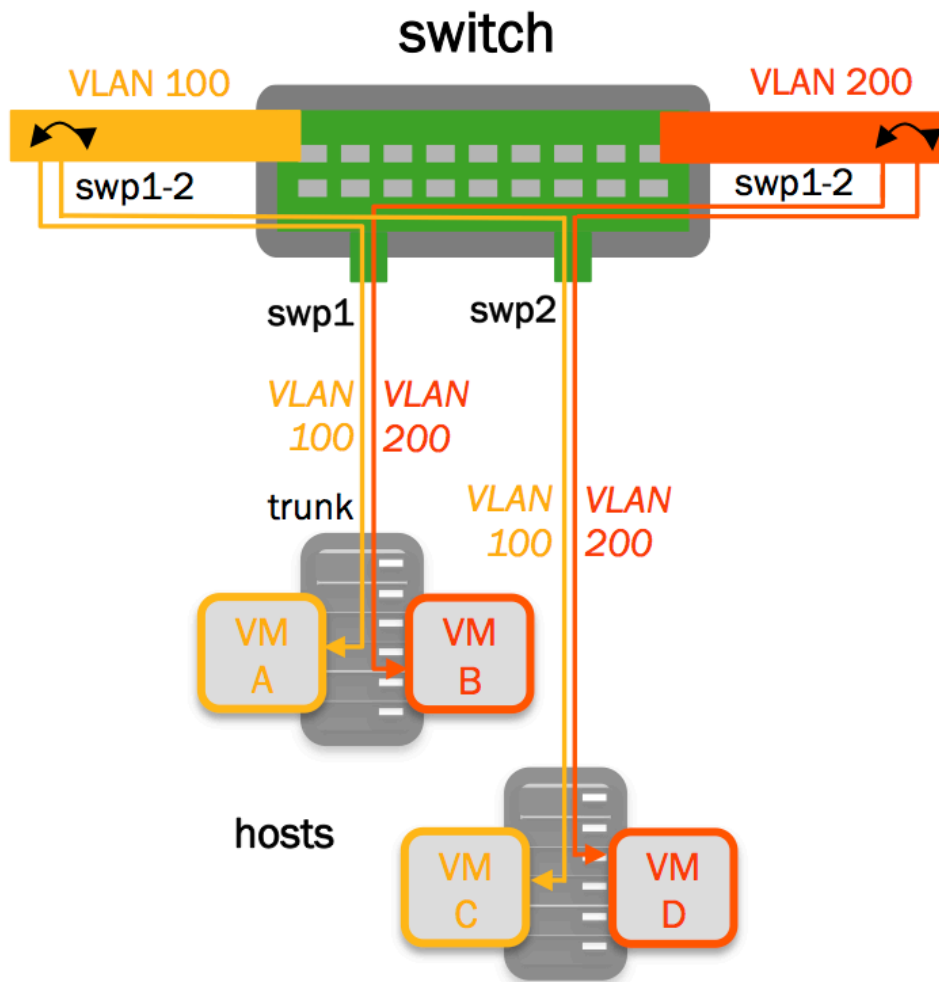
802.1Q also identifies an *untagged* frame as belonging to the *native* VLAN (most network devices default their native VLAN to 1). The concept of native, non-native, tagged or untagged has generated confusion due to mixed terminology and vendor-specific implementations. Some clarification is in order:

- 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.

! 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



Configure the following in `/etc/network/interfaces`:

```
auto br-VLAN100
iface br-VLAN100
    bridge-ports swp1.100 swp2.100
    bridge-stp on
auto br-VLAN200
```

```

iface br-VLAN200
  bridge-ports swp1.200 swp2.200
  bridge-stp on

```

To bring up br-VLAN100 and br-VLAN200, use the `ifreload` command:

```
cumulus@switch:~$ sudo ifreload -a
```

Showing and Verifying the Trunk

```

cumulus@en-sw2$ brctl show
bridge name bridge id STP enabled interfaces
br-VLAN100 8000.7072cf9d4e35 no swp1.100
           swp2.100
br-VLAN200 8000.7072cf9d4e35 no swp1.200
           swp2.200

```

Using `netshow` to Display the Trunk

`netshow` is a Cumulus RMP tool for retrieving information about your network configuration.

```

cumulus@switch$ netshow interface bridge

```

	Name	Speed	Mtu	Mode	Summary
UP	br-VLAN100	N/A	1500	Bridge/L2	Tagged: swp1-2 STP: rootSwitch(32768) VlanID: 100
UP	br-VLAN200	N/A	1500	Bridge/L2	Tagged: swp1-2 STP: rootSwitch(32768) VlanID: 200

Additional Examples

You can find additional examples of VLAN tagging in [this chapter](#) (see page 139).

Configuration Files

- `/etc/network/interfaces`
- `/etc/network/interfaces.d/`
- `/etc/network/if-down.d/`

- [/etc/network/if-post-down.d/](#)
- [/etc/network/if-pre-up.d/](#)
- [/etc/network/if-up.d/](#)

Useful Links

- <http://www.linuxfoundation.org/collaborate/workgroups/networking/bridge>
- <http://www.linuxfoundation.org/collaborate/workgroups/networking/vlan>
- <http://www.linuxjournal.com/article/8172>

Caveats and Errata

- The same bridge cannot contain multiple subinterfaces of the **same** port as members. Attempting to apply such a configuration will result in an error.

VLAN Tagging

This article shows two examples of VLAN tagging (see page), one basic and one more advanced. They both demonstrate the streamlined interface configuration from `ifupdown2`. For more information, see [Configuring and Managing Network Interfaces](#) (see page 64).

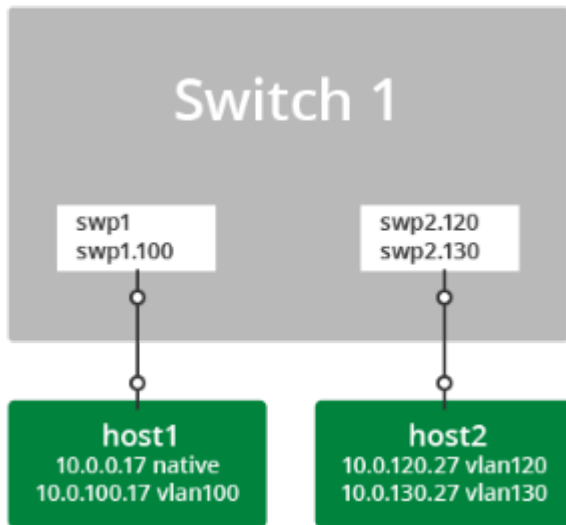
Contents

(Click to expand)

- [Contents](#) (see page 139)
- [VLAN Tagging, a Basic Example](#) (see page 139)
 - [Persistent Configuration](#) (see page 140)
- [VLAN Tagging, an Advanced Example](#) (see page 140)
 - [Persistent Configuration](#) (see page 141)
 - [VLAN Translation](#) (see page 146)

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*.

Persistent Configuration

To configure the above example persistently, configure `/etc/network/interfaces` like this:

```
# 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

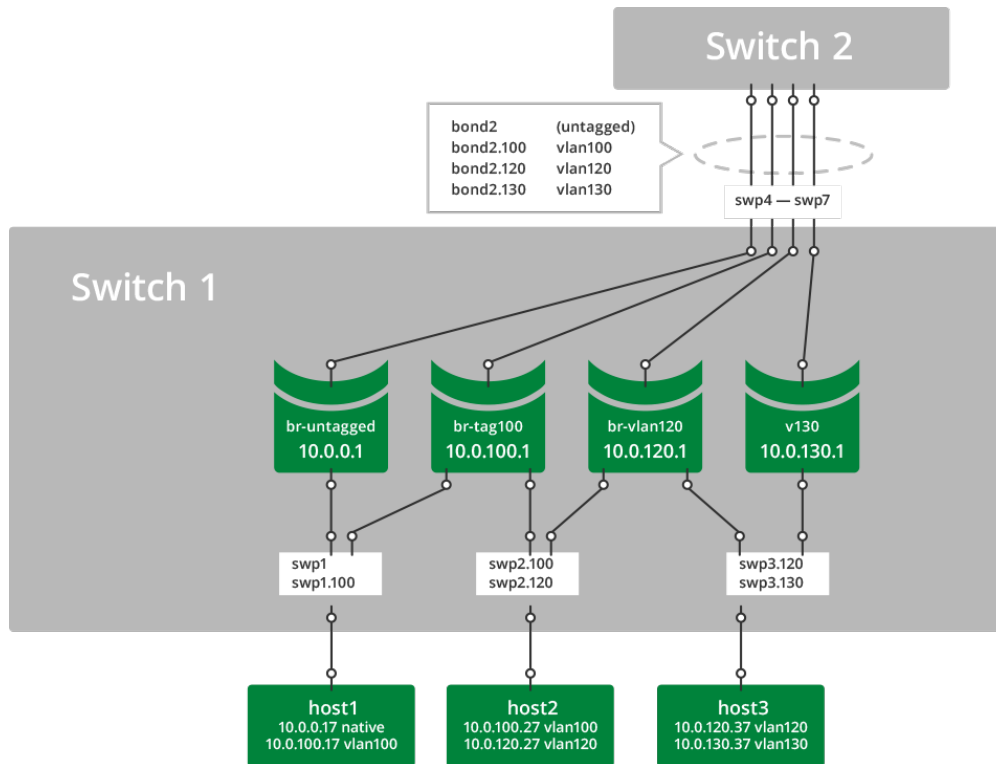
auto swp2
iface swp2

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 RMP 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).

Persistent Configuration

From `/etc/network/interfaces`:

```
# 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
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
```

```

bond-xmit-hash-policy layer3+4

# 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 vl30
iface vl30
    address 10.0.130.1/24
    bridge-ports swp2.130 swp3.130 bond2.130
    bridge-stp on

# - - - - -

```

To verify:

```

cumulus@switch:~$ sudo mstpcpl 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 mstpcctl 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



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 RMP 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:

```
cumulus@switch:~$ sudo brctl addbr br_mix

cumulus@switch:~$ sudo ip link add link swp10 name swp10.100 type vlan id
100
cumulus@switch:~$ sudo ip link add link swp11 name swp11.200 type vlan id
200

cumulus@switch:~$ sudo brctl addif br_mix swp10.100 swp11.200
can't add swp11.200 to bridge br_mix: Invalid argument
```

In some cases, it may be useful to relax this restriction. For example, two servers may 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.



A bridge in [VLAN-aware mode \(see page 147\)](#) 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`:

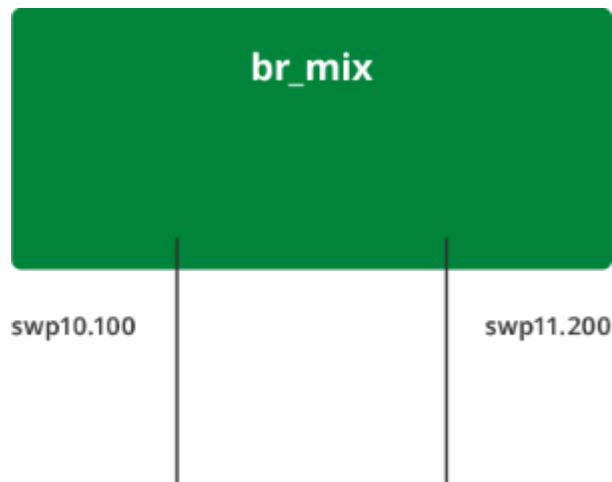
```
cumulus@switch:~$ echo net.bridge.bridge-allow-multiple-vlans = 1 | sudo
tee /etc/sysctl.d/multiple_vlans.conf
net.bridge.bridge-allow-multiple-vlans = 1
cumulus@switch:~$ sudo sysctl -p /etc/sysctl.d/multiple_vlans.conf
net.bridge.bridge-allow-multiple-vlans = 1
```

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.

Once the `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
```



VLAN-aware Bridge Mode for Large-scale Layer 2 Environments

The Cumulus RMP 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 RMP implements a configuration model for large-scale L2 environments, with **one single instance** of [Spanning Tree \(see page 90\)](#). Each physical bridge member port is configured with the list of allowed VLANs as well as its port VLAN ID (either PVID or native VLAN — see below). 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.



You can configure both VLAN-aware and traditional mode bridges on the same network in Cumulus RMP; however you should not have more than one VLAN-aware bridge on a given switch.

Contents

(Click to expand)

- [Defining VLAN-aware Bridge Attributes \(see page 148\)](#)
- [Basic Trunking \(see page 149\)](#)
- [VLAN Filtering/VLAN Pruning \(see page 150\)](#)
- [Untagged/Access Ports \(see page 150\)](#)
 - [Dropping Untagged Frames \(see page 151\)](#)
- [VLAN Layer 3 Addressing/Switch Virtual Interfaces and other VLAN Attributes \(see page 152\)](#)
- [Using the glob Keyword to Configure Multiple Ports in a Range \(see page 152\)](#)
- [Example Configuration with Access Ports and Pruned VLANs \(see page 153\)](#)
- [Example Configuration with Bonds \(see page 154\)](#)
- [Converting a Traditional Bridge to VLAN-aware or Vice Versa \(see page 155\)](#)
- [Caveats and Errata \(see page 156\)](#)

Defining VLAN-aware Bridge Attributes

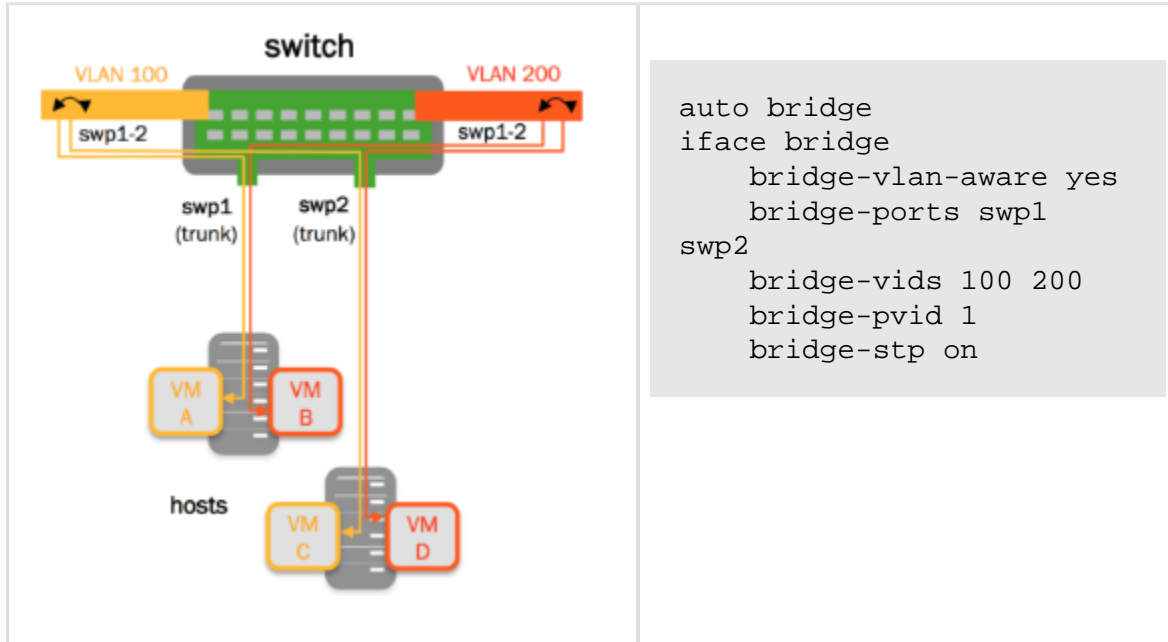
To configure a VLAN-aware bridge, include the `bridge-vlan-aware` attribute, setting it to `yes`. Name the bridge `bridge` to help ensure it is the only VLAN-aware bridge on the switch. The following attributes are useful for configuring VLAN-aware bridges:

- `bridge-vlan-aware`: Set to `yes` to indicate that the bridge is in VLAN-aware mode.
- `bridge-pvid`: A PVID is the bridge's *Primary VLAN Identifier*. The PVID defaults to 1; specifying the PVID identifies that VLAN as the native VLAN.
- `bridge-vids`: A VID is the *VLAN Identifier*, which declares the VLANs associated with this bridge.
- `bridge-access`: Declares the physical switch port as an *access port*. Access ports ignore all tagged packets; put all untagged packets into the `bridge-pvid`.
- `bridge-allow-untagged`: When set to `no`, it drops any untagged frames for a given switch port.

For a definitive list of bridge attributes, run `ifquery --syntax-help` and look for the entries under **bridge**, **bridgevlan** and **mstpctl**.

Basic Trunking

A basic configuration for a VLAN-aware bridge configured for STP that contains two switch ports looks like this:



The above configuration actually includes 3 VLANs: the tagged VLANs 100 and 200 and the untagged (native) VLAN of 1.



The `bridge-pvid 1` is implied by default. You do not have to specify `bridge-pvid`. And while it does not hurt 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-vlan-aware yes
    bridge-ports
swp1 swp2
    bridge-vids
1 100 200
    bridge-stp on
  
```

```

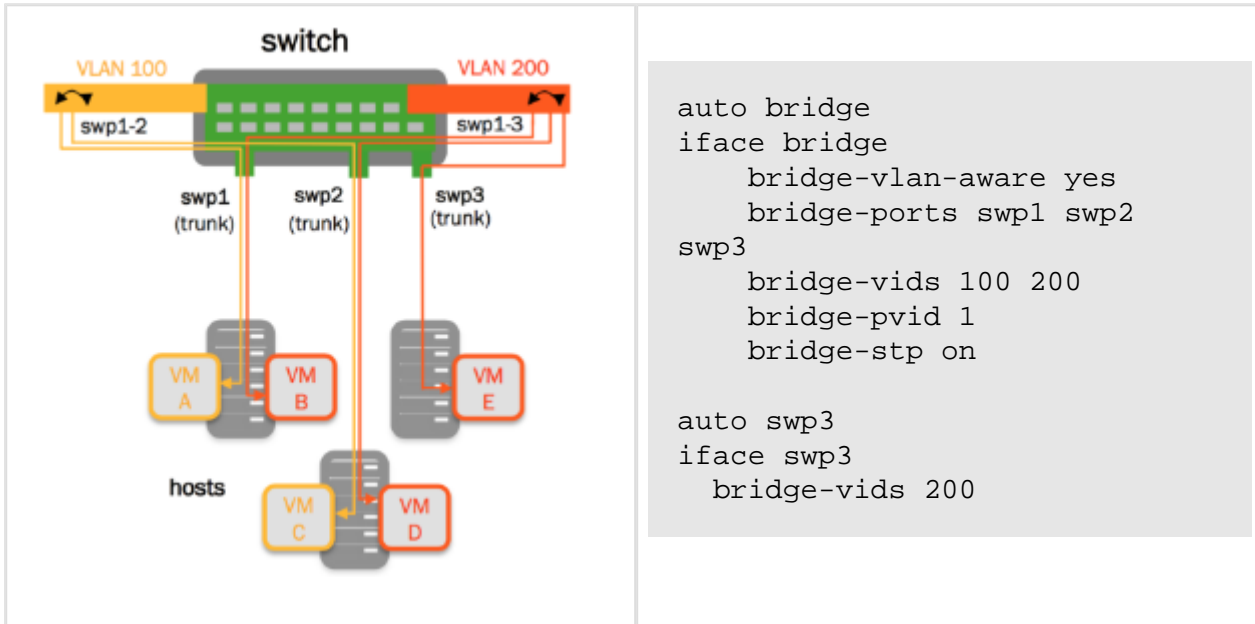
auto bridge
iface bridge
    bridge-vlan-aware yes
    bridge-ports
swp1 swp2
    bridge-vids
1 100 200
    bridge-pvid 1
    bridge-stp on
  
```

```

auto bridge
iface bridge
    bridge-vlan-aware yes
    bridge-ports
swp1 swp2
    bridge-vids
100 200
    bridge-stp on
  
```

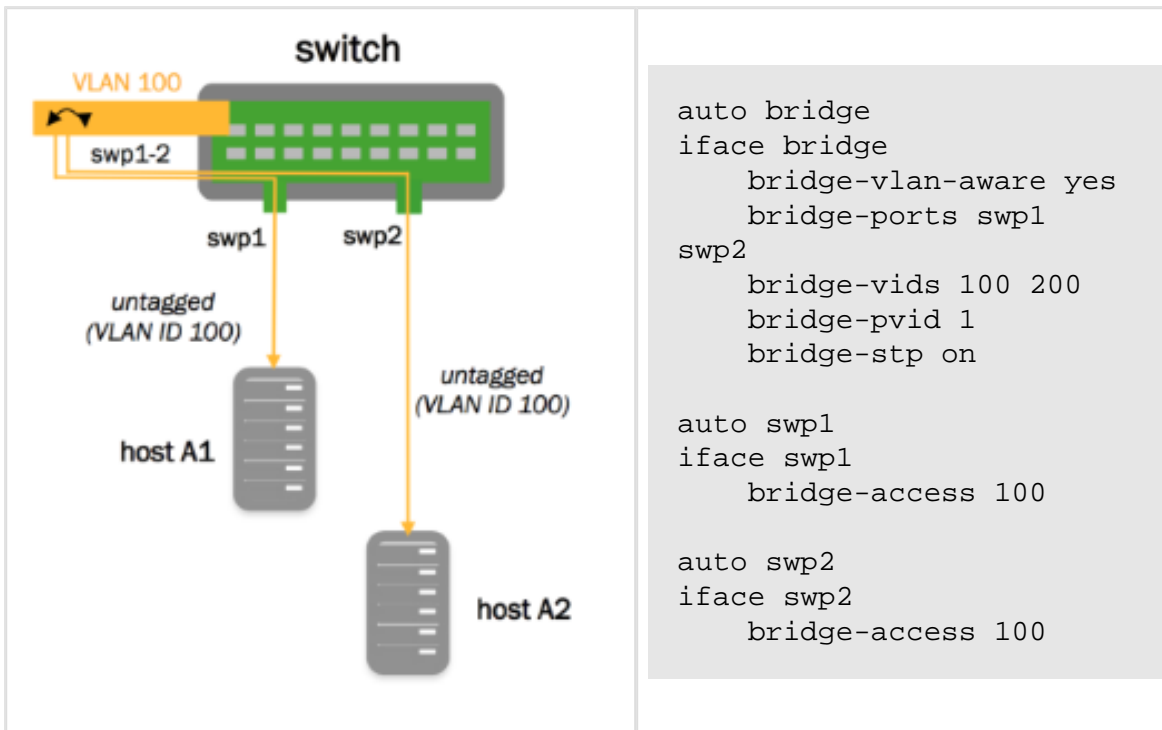
VLAN Filtering/VLAN Pruning

By default, the bridge port inherits the bridge VLANs. A port's configuration can override the bridge VLANs. Do this by specifying port-specific VLANs using the `bridge-vids` attribute.



Untagged/Access Ports

As described above, access ports ignore all tagged packets. In the configuration below, swp1 and swp2 are configured as access ports. All untagged traffic goes to the specified VLAN, which is VLAN 100 in the example below.



Dropping Untagged Frames

With VLAN-aware bridge mode, it's possible to configure a switch port so it drops any untagged frames. To do this, add `bridge-allow-untagged no` under the switch port stanza in `/etc/network/interfaces`. This leaves the bridge port without a PVID and drops untagged packets.

Consider the following example bridge:

```
auto bridge
iface bridge
    bridge-vlan-aware yes
    bridge-ports swp1 swp9
    bridge-vids 2-100
    bridge-pvid 101
    bridge-stp on
```

Here is the VLAN membership for that configuration:

```
cumulus@switch$ bridge -c vlan show
portvlan ids
swp1 101 PVID Egress Untagged
    2-100

swp9 101 PVID Egress Untagged
    2-100

bridge 101
```

To configure swp9 to drop untagged frames, add `bridge-allow-untagged no`:

```
auto swp9
iface swp9
    bridge-allow-untagged no
```

When you check VLAN membership for that port, it shows that there is **no** untagged VLAN.

```
cumulus@switch$ bridge -c vlan show
portvlan ids
swp1 101 PVID Egress Untagged
    2-100
```

```
swp9 2-100
```

```
bridge 101
```

VLAN Layer 3 Addressing/Switch Virtual Interfaces and other VLAN Attributes

When configuring the VLAN attributes for the bridge, put the attributes in a separate stanza for each VLAN interface: <bridge>.<vlanid>. If you are configuring the SVI for the native VLAN, you must declare the native VLAN in its own stanza and specify its IP address. Specifying the IP address in the bridge stanza itself returns an error.

```
auto bridge.100
iface bridge.100
    address 192.168.10.1/24
    address 2001:db8::1/32
    hwaddress 44:38:39:ff:00:00

# l2 attributes
auto bridge.100
vlan bridge.100
    bridge-igmp-querier-src 172.16.101.1
```



The *vlan* object type in the l2 attributes section above is used to specify layer 2 VLAN attributes only. Currently, the only supported layer 2 VLAN attribute is `bridge-igmp-querier-src`.

However, if your switch is configured for multicast routing, then 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.

You can specify a range of VLANs as well. For example:

```
auto bridge.[1-2000]
vlan bridge.[1-2000]
    ATTRIBUTE VALUE
```

Using the glob Keyword to Configure Multiple Ports in a Range

The `glob` keyword referenced in the `bridge-ports` attribute indicates that swp1 through swp52 are part of the bridge, which is a short cut that saves you from enumerating each port individually:

```
auto bridge
iface bridge
```



```
bridge-vlan-aware yes
bridge-ports glob swp1-52
bridge-stp on
bridge-vids 310 700 707 712 850 910
```

Example Configuration with Access Ports and Pruned VLANs

The following example contains an access port and a switch port that is *pruned*; that is, it only sends and receives 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-vlan-aware yes
    bridge-ports glob swp1-52
    bridge-stp on
    bridge-vids 310 700 707 712 850 910

auto swp1
iface swp1
    mstpctl-portadminedge yes
    mstpctl-bpduguard yes
    bridge-access 310

# 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-portadminedge yes
    mstpctl-bpduguard 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-portpathcost 10
    mstpctl-portnetwork yes

# 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-portpathcost 0
    mstpctl-portnetwork yes
```

Example Configuration with Bonds

This configuration 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 4094
auto bridge.4094
iface bridge.4094
    address 11.100.1.252/24

# l2 attributes for vlan 4094
auto bridge.4094
vlan bridge.4094
    bridge-igmp-querier-src 172.16.101.1

#
# vlan-aware bridge
#
auto bridge
iface bridge
    bridge-vlan-aware yes
    bridge-ports uplink1 peerlink downlink glob swp2-20
    bridge-stp on

# svi peerlink vlan
auto peerlink.4094
iface peerlink.4094
    address 192.168.10.1/30
    broadcast 192.168.10.3
```

Converting a Traditional Bridge to VLAN-aware or Vice Versa


You cannot automatically convert a traditional bridge to/from a VLAN-aware bridge simply by changing the configuration in the `/etc/network/interfaces` file. If you need to change the mode for a bridge, do the following:

1. Delete the traditional mode bridge from the configuration and bring down all its member switch port interfaces.
2. Create a new VLAN-aware bridge, as described above.
3. Bring up the bridge.

These steps assume you are converting a traditional mode bridge to a VLAN-aware one. To do the opposite, delete the VLAN-aware bridge in step 1, and create a new traditional mode bridge in step 2.

Caveats and Errata

- **STP:** Because [Spanning Tree and Rapid Spanning Tree \(see page 90\)](#) (STP) are enabled on a per-bridge basis, VLAN-aware mode essentially supports a single instance of STP across all VLANs. A common practice when using a single STP instance for all VLANs is to define all every VLAN on each switch in the spanning tree instance. `mstpd` continues to be the user space protocol daemon, and Cumulus RMP supports RSTP.
- **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 RMP reserves a range of 700 VLANs by default; this range is 3300-3999. In case any of your user-defined VLANs conflict with the default reserved range, you can modify the range, as long as the new range is a contiguous set of VLANs with IDs anywhere between 2 and 4094, and the minimum size of the range is 300 VLANs:
 1. Edit `/etc/cumulus/switchd.conf`, uncomment `resv_vlan_range` and specify the new range.
 2. Restart `switchd` (`sudo systemctl restart switchd.service`) for the new range to take effect.



While restarting `switchd`, all running ports will flap and forwarding will be interrupted.
- **VLAN translation:** A bridge in VLAN-aware mode cannot have VLAN translation enabled for it; only bridges configured in [traditional mode \(see page 126\)](#) can utilize VLAN translation.

Routing

This chapter discusses routing on switches running Cumulus RMP.

Contents

(Click to expand)

- [Contents \(see page 157\)](#)
- [Commands \(see page 157\)](#)
- [Configuring Static Routing \(see page 157\)](#)
 - [Persistently Adding a Static Route \(see page 157\)](#)
- [Useful Links \(see page 158\)](#)

Commands

- [ip route](#)

Configuring Static Routing

The `ip route` command allows manipulating the kernel routing table directly from the Linux shell. See `man ip(8)` for details.

To display the routing table:

```
cumulus@switch:~$ ip route show
default via 10.0.1.2 dev eth0
default via 10.42.0.1 dev eth0
10.42.0.0/22 dev eth0 proto kernel scope link src 10.42.0.65
10.168.2.0/24 dev swp2 proto kernel scope link src 10.168.2.1
10.168.26.0/24 dev swp26 proto kernel scope link src 10.168.26.1
```

Persistently Adding a Static Route

A static route can be persistently added by adding up `ip route add ..` into `/etc/network/interfaces`. For example:

```
cumulus@switch:~$ 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), ifup(8)
#
```

```
# Please see /usr/share/doc/python-ifupdown2/examples/ for examples
#
#

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet dhcp

auto swp2
iface swp2 inet static
    address 10.168.2.1/24
    up ip route add 203.0.113.0/24 via 10.168.2.2

auto swp26
iface swp26 inet static
    address 10.168.26.1/24
```

Configuration Files

- /etc/network/interfaces

Useful Links

- <http://linux-ip.net/html/tools-ip-route.html>

Monitoring and Troubleshooting

This chapter introduces monitoring and troubleshooting Cumulus RMP.

Contents

(Click to expand)

- [Contents \(see page 159\)](#)
- [Commands \(see page 159\)](#)
- [Using the Serial Console \(see page 159\)](#)
 - [Configuring the Serial Console \(see page 159\)](#)
- [Diagnostics Using cl-support \(see page 160\)](#)
- [Sending Log Files to a syslog Server \(see page 162\)](#)
- [Next Steps \(see page 164\)](#)

Commands

- [cl-support](#)

Using the Serial Console

The serial console can be a useful tool for debugging issues, especially when you find yourself rebooting the switch often or if you don't have a reliable network connection.

The default serial console baud rate is 115200, which is the baud rate [ONIE](#) uses.

Configuring the Serial Console

On x86 switches, you configure serial console baud rate by editing `grub`.



Incorrect configuration settings in `grub` can cause the switch to be inaccessible via the console. Grub changes should be carefully reviewed before implementation.

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 `/etc/default/grub`. 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 your switch's 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.

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.

You should run `cl-support` before you submit a support request to Cumulus Networks as this file helps in the investigation of issues:

```
cumulus@switch:~$ sudo cl-support -h
Usage: cl-support [-h] [-s] [-t] [-v] [reason]...

Args:
[reason]: Optional reason to give for invoking cl-support.
          Saved into tarball's cmdline.args file.

Options:
-h: Print this usage statement
-s: Security sensitive collection
-t: User filename tag
-v: Verbose
-e MODULES: Enable modules. Comma separated module list (run with -e help
```



```
for module names)
-d MODULES: Disable modules. Comma separated module list (run with -d help
for module names)
```

Example output:

```
cumulus@switch:~$ ls /var/support
cl_support_20130806_032720.tar.xz
```

The directory structure is compressed using LZMA2 compression and can be extracted using the `unxz` command:

```
cumulus@switch:~$ cd /var/support
cumulus@switch:~$ sudo unxz cl_support_20130729_140040.tar.xz
cumulus@switch:~$ sudo tar xf cl_support_20130729_140040.tar
cumulus@switch:~$ ls -l cl_support_20130729_140040/

-rwxr-xr-x  1 root root 7724 Jul 29 14:00 cl-support
-rw-r--r--  1 root root  52 Jul 29 14:00 cmdline.args
drwxr-xr-x  2 root root 4096 Jul 29 14:00 core
drwxr-xr-x 64 root root 4096 Jul 29 13:51 etc
drwxr-xr-x  4 root root 4096 Jul 29 14:00 proc
drwxr-xr-x  2 root root 4096 Jul 29 14:01 support
drwxr-xr-x  3 root root 4096 Jul 29 14:00 sys
drwxr-xr-x  3 root root 4096 Aug  8 15:22 var
```

The directory contains the following elements:

Directory	Description
core	Contains the core files on the switch, including those generated from <code>switchd</code> .
etc	Is a replica of the switch's <code>/etc</code> directory. <code>/etc</code> contains all the general Linux configuration files, as well as configurations for the system's network interfaces and other packages.
log	Is a replica of the switch's <code>/var/log</code> directory. Most Cumulus RMP log files are located in this directory. Notable log files include <code>switchd.log</code> and <code>daemon.log</code> log files, and <code>syslog</code> . For more information, read this knowledge base article .
proc	Is a replica of the switch's <code>/proc</code> directory. In Linux, <code>/proc</code> contains runtime system information (like system memory, devices mounted, and hardware configuration). These files are not actual files but the current state of the system.

Directory	Description
support	Is a set of files containing further system information, which is obtained by <code>cl-support</code> running commands such as <code>ps -aux</code> , <code>netstat -i</code> , and so forth — even the routing tables.

`cl-support`, when untarred, contains a `cmdline.args` file. This file indicates what reason triggered it. When contacting Cumulus Networks technical support, please attach the `cl-support` file if possible. For more information about `cl-support`, please read [Understanding and Decoding the cl-support Output File](#) (see page 172).



If you have issues extracting the script with the `tar` command, like an error saying the file does not look like tar archive, try using the `unxz` command first:

```
cumulus@switch:~$ sudo unxz cl_support_20130729_140040.tar.xz
```

You can save a lot of disk space and perhaps some time if you do not run `unxz` on the tar file.

Sending Log Files to a syslog Server

All logging on Cumulus RMP 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; here's an example:

```
2015-08-14T18:21:43.337804+00:00 cumulus switchd[3629]:
switchd.c:1409 switchd version 1.0-cl2.5+5
```

Local logging: Most logs within Cumulus RMP are sent to files in the `/var/log` directory. Most relevant information is placed within the `/var/log/syslog` file. For more information on specific log files, see [Troubleshooting Log Files](#).

Export logging: To send `syslog` files to an external `syslog` server, add a rule specifying to copy all messages (*.*) to the IP address and switch port of your `syslog` server in the `rsyslog` configuration files as described below.

In the following example, `192.168.1.2` is the remote `syslog` server and `514` is the port number. For UDP-based syslog, use a single @ before the IP address: `@192.168.1.2:514`. For TCP-based syslog, use two @@ before the IP address: `@@192.168.1.2:514`.

1. Create a file called something like `/etc/rsyslog.d/90-remotesyslog.conf`. Make sure it starts with a number lower than 99 so that it executes before `99-syslog.conf`. Add content like the following:

```
## Copy all messages to the remote syslog server at 192.168.1.2 port
514
*. *                                @192.168.1.2:514
```

2. Restart `rsyslog`.

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



All Cumulus RMP 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`.



In the case of the `switchd` rules file, the file must be numbered lower than 25. For example, `13-switchd-remote.conf`.

If you need to send other log files (e.g. `switchd` logs) to a `syslog` server, configure a new file in `/etc/rsyslog.d`, as described above, and add lines similar to the following lines:

```
## Logging switchd messages to remote syslog server
$ModLoad imfile
$InputFileName /var/log/switchd.log
$InputFileStateFile logfile-log
$InputFileTag switchd:
$InputFileSeverity info
$InputFileFacility local7
$InputFilePollInterval 5
$InputRunFileMonitor

if $programname == 'switchd' then @192.168.1.2:514
```

Then restart `syslog`:

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

In the above configuration, each setting is defined as follows:

Setting	Description
\$ModLoad <i>imfile</i>	Enables the <code>rsyslog</code> module to watch file contents.
\$InputFileName	The file to be sent to the <code>syslog</code> server. In this example, you are going to send changes made to <code>/var/log/switchd.log</code> to the <code>syslog</code> server.
\$InputFileStateFile	This is used by <code>rsyslog</code> to track state of the file being monitored. This must be unique for each file being monitored.
\$InputFileTag	Defines the <code>syslog</code> tag that will precede the <code>syslog</code> messages. In this example, all logs are prefaced with <code>switchd</code> .
\$InputFileSeverity	Defines the logging severity level sent to the <code>syslog</code> server.
\$InputFileFacility	Defines the logging format. <code>local7</code> is common.
\$InputFilePollInterval	Defines how frequently in seconds <code>rsyslog</code> looks for new information in the file. Lower values provide faster updates but create slightly more load on the CPU.
\$InputRunFileMonitor	Enables the file monitor module with the configured settings.

In most cases, the settings to customize include:

Setting	Description
\$InputFileName	The file to stream to the <code>syslog</code> server.
\$InputFileStateFile	A unique name for each file being watched.
\$InputFileTag	A prefix to the log message on the server.

Finally, the `if $programname` line is what sends the log files to the `syslog` server. It follows the same syntax as the `/var/log/syslog` file, where `@` indicates UDP, `192.168.1.2` is the IP address of the `syslog` server, and `514` is the UDP port. The value `switchd` must match the value in `$InputFileTag`.

Next Steps

The links listed below discuss more specific monitoring topics.

Single User Mode - Boot Recovery

Use single user mode to assist in troubleshooting system boot issues or for password recovery.

Contents

(Click to expand)

- [Contents \(see page 165\)](#)
- [Entering Single User Mode \(see page 165\)](#)

Entering Single User Mode

1. Boot the switch, as soon as you see the GRUB menu.

```

GNU GRUB  version 2.02~beta2-22+deb8u1

+-----+
|*Cumulus RMP GNU
/Linux
| Advanced options for Cumulus RMP GNU
/Linux
|
ONIE
|
|
|
+-----+
-----+

```

2. Use the ^ and v arrow keys to select **Advanced options for Cumulus RMP GNU/Linux**. A menu similar to the following should appear:

```

GNU GRUB  version 2.02~beta2-22+deb8u1

+-----+
| Cumulus RMP GNU/Linux, with Linux 4.1.0-cl-1-

```

```

amd64 |
| Cumulus RMP GNU/Linux, with Linux 4.1.0-cl-1-amd64
(sysvinit) |
| *Cumulus RMP GNU/Linux, with Linux 4.1.0-cl-1-amd64 (recovery
mode) |

|
|

+-----+
-----+

```

3. Select **Cumulus RMP GNU/Linux, with Linux 4.1.0-cl-1-amd64 (recovery mode)**.
4. Press ctrl-x to reboot.
5. After the system reboots, set a new password.

```

# passwd
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully

```

6. Reboot the system:

```

# sync
# reboot -f
Restarting the system.

```

Resource Diagnostics Using cl-resource-query

You can use `cl-resource-query` to retrieve information about host entries, MAC entries, L2 and L3 routes, and ingress and degrees ACL counters and entries that are in use. This is especially useful because Cumulus RMP syncs 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.

In order to avoid this, routes in the hardware should be monitored and kept below the ASIC limits. For example on a Cumulus RMP system, the limits are as follows:

```

routes: 8092 <<<< if all routes are IPv6, or 16384 if all routes are IPv4
long mask routes 2048 <<<< these are routes with a mask longer than the
route mask limit
route mask limit 64

```

```
host_routes: 8192
ecmp_nhs: 16346
ecmp_nhs_per_route: 52
```

You can monitor this in Cumulus RMP with the `cl-resource-query` command.

```
cumulus@switch:~$ sudo cl-resource-query
Host entries:          1,    0% of maximum value    8192 <<<< this is
the default software-imposed limit, 50% of the hardware limit
IPv4 neighbors:        1          <<<< these are counts of the number
of valid entries in the table
IPv6 neighbors:        0
IPv4 entries:          13,    0% of maximum value    32668
IPv6 entries:          18,    0% of maximum value    16384
IPv4 Routes:           13
IPv6 Routes:           18
Total Routes:          31,    0% of maximum value    32768
MAC entries:           12,    0% of maximum value    32768
```

Monitoring System Hardware

You monitor system hardware in these ways, using:

- `decode-syseeprom`
- `sensors`
- `smond`
- [Net-SNMP](#) (see page 216)
- `watchdog`

Contents

(Click to expand)

- [Contents](#) (see page 167)
- [Commands](#) (see page 168)
- [Monitoring Hardware Using decode-syseeprom](#) (see page 168)
 - [Command Options](#) (see page 169)
 - [Related Commands](#) (see page 169)
- [Monitoring Hardware Using sensors](#) (see page 169)
 - [Command Options](#) (see page 170)
- [Monitoring Switch Hardware Using SNMP](#) (see page 170)
- [Monitoring System Units Using smond](#) (see page 170)

- Command Options (see page 171)
- Keeping the Switch Alive Using the Hardware Watchdog (see page 171)
- Configuration Files (see page 172)
- Useful Links (see page 172)

Commands

- decode-syseeprom
- dmidecode
- lshw
- sensors
- smond

Monitoring Hardware 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.

For example:

```
cumulus@switch:~$ decode-syseeprom
TlvInfo Header:
  Id String:      TlvInfo
  Version:        1
  Total Length: 159
TLV Name          Code Len Value
-----
Product Name      0x21   6 Pebble
Part Number       0x22  14 R0854-G0008-02
Serial Number     0x23  19 D2070023918PE000012
Manufacture Date  0x25  19 01/15/2015 14:30:00
Device Version    0x26   1 2
Label Revision    0x27   6 Pebble
Platform Name     0x28   6 Pebble
MAC Addresses     0x2A   2 73
Manufacturer      0x2B   9 CELESTICA
Manufacture Country 0x2C   3 CHN
Vendor Name       0x2D   9 CELESTICA
Diag Version      0x2E   5 1.0.0
Service Tag       0x2F   2 LB
Vendor Extension  0xFD   1 0x62
Base MAC Address  0x24   6 44:38:39:00:89:DD
ONIE Version      0x29  13 2014.11.0.0.2
CRC-32            0xFE   4 0x19AFD83A
(checksum valid)
```


Command Options

Usage: /usr/cumulus/bin/decode-syseeprom [-a][-r][-s [args]][-t]

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.
-s	Sets the EEPROM content if the EEPROM is writable. args can be supplied in 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 will 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.
-t TARGET	Selects the target EEPROM (board , psu2 , psu1) for the read or write operation; default is board .
-e, --serial	Prints the device serial number.
-m	Prints the base MAC address for management interfaces.

Related Commands

You can also use the `dmidecode` command to retrieve hardware configuration information that's been populated in the BIOS.

You can use `apt-get` to install the `lshw` program on the switch, which also retrieves hardware configuration information.

Monitoring Hardware Using sensors

The `sensors` command provides a method for monitoring the health of your switch hardware, such as power, temperature and fan speeds. This command executes `lm-sensors`.

For example:

```
cumulus@switch:~$ sensors
coretemp-isa-0000
Adapter: ISA adapter
Core 0:          +32.0 C  (high = +110.0 C, crit = +110.0 C)
Core 2:          +35.0 C  (high = +110.0 C, crit = +110.0 C)
```

```
lm75-i2c-0-4a
Adapter: SMBus I801 adapter at e000
temp1:      +27.0 C  (high = +60.0 C, hyst = +25.0 C)
```

Command Options

Usage: `sensors` [OPTION]... [CHIP]...

Option	Description
-c, --config-file	Specify a config file; use - after -c to read the config file from <code>stdin</code> ; by default, <code>sensors</code> references the configuration file in <code>/etc/sensors.d/</code> .
-s, --set	Executes set statements in the config file (root only); <code>sensors -s</code> 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 <code>sensors.conf</code> .

If [CHIP] is not specified in the command, all chip info will be 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-*`

Monitoring Switch Hardware Using SNMP

You can read about Net-SNMP in [this chapter](#) (see page 216).

Monitoring System Units Using smond

The `smond` daemon monitors these system units: power, board, temp, fan and volt. It updates their corresponding LEDs, and logs the change in the state. Changes in system unit state are detected via the `cp1d` 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 `smonct1` to display sensor information for the various system units:

```
cumulus@switch:~$ 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
```

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`.

Keeping the Switch Alive Using the Hardware Watchdog

Cumulus RMP includes a simplified version of the `wd_keepalive(8)` daemon from the standard Debian package `watchdog`. `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 QuantaMesh BMS T1048-LB9 switches only; you must enable the watchdog on all other switch platforms. When enabled, it starts when you boot the switch, before `switchd` starts.

To enable the hardware watchdog, edit the `/etc/watchdog.d/<your_platform>` file and set `run_watchdog` to `1`:

```
run_watchdog=1
```

To disable the watchdog, edit the `/etc/watchdog.d/<your_platform>` file and set `run_watchdog` to 0 :

```
run_watchdog=0
```

Then stop the daemon:

```
cumulus@switch:~$ sudo systemctl stop wd_keepalive.service
```

You can modify the settings for the watchdog — like the timeout setting and scheduler priority — in its configuration file, `/etc/watchdog.conf`.

Configuration Files

- `/etc/cumulus/switchd.conf`
- `/etc/cumulus/sysledcontrol.conf`
- `/etc/sensors.d/<switch>.conf` - sensor configuration file (do **not** edit it!)
- `/etc/watchdog.conf`

Useful Links

- <http://packages.debian.org/search?keywords=lshw>
- <http://lm-sensors.org>
- Net-SNMP tutorials

Understanding and Decoding the cl-support Output File

The `cl-support` command generates a tar archive of useful information for troubleshooting that can be auto-generated or manually created. To manually create it, run the `cl-support` command. The `cl-support` file is automatically generated when:

- There is a **core file dump** of any application (not specific to Cumulus RMP, but something all Linux distributions support)
- Memory usage surpasses 90% of the total system memory (memory usage > 90% for 1 cycle)
- The **loadavg** over 15 minutes has on average greater than 2 (loadavg (15min) > 2)

The Cumulus Networks support team may request you submit the output from `cl-support` to help with the investigation of issues you might experience with Cumulus RMP.

```
cumulus@switch:~$ sudo cl-support -h
Usage: cl-support [-h] [reason]...
```

```
Args:
[reason]: Optional reason to give for invoking cl-support.
          Saved into tarball's reason.txt file.
Options:
-h: Print this usage statement
```

Example output:

```
cumulus@switch:~$ ls /var/support
cl_support__switch_20141204_203833
```

(Click to expand)

- The `cl-support` command generates a tar archive of useful information for troubleshooting that can be auto-generated or manually created. To manually create it, run the `cl-support` command. The `cl-support` file is automatically generated when: (see page 172)
- Understanding the File Naming Scheme (see page 173)
- Decoding the Output (see page 173)

Understanding the File Naming Scheme

The `cl-support` command generates a file under `/var/support` with the following naming scheme. The following example describes the file called `cl_support__switch_20141204_203833.tar.xz`.

cl_support	switch	20141204	203833
This is always prepended to the <code>tar.gz</code> output.	This is the hostname of the switch where <code>cl-support</code> was executed.	The date in year, month, day; so 20141204 is December, 4th, 2014.	The time in hours, minutes, seconds; so 203833 is 20, 38, 33 (20:38:33) or the equivalent to 8:38:33 PM.

Decoding the Output

Decoding a `cl_support` file is a simple process performed using the `tar` command. The following example illustrates extracting the `cl_support` file:

```
tar -xf cl_support__switch_20141204_203834.tar.xz
```

The `-xf` options are defined here:

Option	Description
-x	Extracts to disk from the archive.

Option	Description
-f	Reads the archive from the specified file.

```
cumulus@switch:~$ ls -l cl_support__switch_20141204_203834/
```

```
-rwxr-xr-x  1 root root 7724 Jul 29 14:00 cl-support
-rw-r--r--  1 root root   52 Jul 29 14:00 cmdline.args
drwxr-xr-x  2 root root 4096 Jul 29 14:00 core
drwxr-xr-x 64 root root 4096 Jul 29 13:51 etc
drwxr-xr-x  4 root root 4096 Jul 29 14:00 proc
drwxr-xr-x  2 root root 4096 Jul 29 14:01 support
drwxr-xr-x  3 root root 4096 Jul 29 14:00 sys
drwxr-xr-x  3 root root 4096 Aug  8 15:22 var
```

The `cl_support` file, when untarred, contains a `reason.txt` file. This file indicates what reason triggered the event. When contacting Cumulus Networks technical support, please attach the `cl-support` file if possible.

The directory contains the following elements:

Directory	Description
cl-support	This is a copy of the <code>cl-support</code> script that generated the <code>cl_support</code> file. It is copied so Cumulus Networks knows exactly which files were included and which weren't. This helps to fix future <code>cl-support</code> requests in the future.
core	Contains the core files generated from the Cumulus RMP HAL (hardware abstraction layer) process, <code>switchd</code> .
etc	<code>etc</code> is the core system configuration directory. <code>cl-support</code> replicates the switch's <code>/etc</code> directory. <code>/etc</code> contains all the general Linux configuration files, as well as configurations for the system's network interfaces, <code>quagga</code> , <code>monit</code> , and other packages.
var/log	<code>/var</code> is the "variable" subdirectory, where programs record runtime information. System logging, user tracking, caches and other files that system programs create and monitor go into <code>/var</code> . <code>cl-support</code> includes only the <code>log</code> subdirectory of the <code>var</code> system-level directory and replicates the switch's <code>/var/log</code> directory. Most Cumulus RMP log files are located in this directory. Notable log files include <code>switchd.log</code> and <code>daemon.log</code> log files, and <code>syslog</code> . For more information, read this knowledge base article .
proc	<code>proc</code> (short for processes) provides system statistics through a directory-and-file interface. In Linux, <code>/proc</code> contains runtime system information (like system memory, devices mounted, and hardware configuration). <code>cl-support</code> simply replicates the switch's <code>/proc</code> directory to determine the current state of the system.

Directory	Description
support	<code>support</code> is not a replica of the Linux file system like the other folders listed above. Instead, it is a set of files containing the output of commands from the command line. Examples include the output of <code>ps -aux</code> , <code>netstat -i</code> , and so forth — even the routing tables are included.

Here is more information on the file structure:

- [Troubleshooting the etc Directory \(see page 177\)](#) — In terms of sheer numbers of files, `/etc` contains the largest number of files to send to Cumulus Networks by far. However, log files could be significantly larger in file size.
- [Troubleshooting Log Files \(see page 175\)](#) — This guide highlights the most important log files to look at. Keep in mind, `c1-support` includes all of the log files.
- [Troubleshooting the support Directory \(see page 188\)](#) — This is an explanation of the `support` directory included in the `c1-support` output.

Troubleshooting Log Files

The only real unique entity for logging on Cumulus RMP compared to any other Linux distribution is `switchd.log`, which logs the HAL (hardware abstraction layer) from hardware like the Broadcom ASIC.

[This guide on NixCraft](#) is amazing for understanding how `/var/log` works. The green highlighted rows below are the most important logs and usually looked at first when debugging.

Log	Description	Why is this important?
<code>/var/log/alternatives.log</code>	Information from the update-alternatives are logged into this log file.	
<code>/var/log/apt</code>	Information the <code>apt</code> utility can send logs here; for example, from <code>apt-get install</code> and <code>apt-get remove</code> .	
<code>/var/log/audit/</code>	Contains log information stored by the Linux audit daemon, <code>auditd</code> .	
<code>/var/log/auth.log</code>	Authentication logs.	
<code>/var/log/boot.log</code>	Contains information that is logged when the system boots.	
<code>/var/log/btmp</code>	This file contains information about failed login attempts. Use the <code>last</code> command to view the <code>btmp</code> file. For example:	

Log	Description	Why is this important?
	<pre>last -f /var/log/btmp more</pre>	
/var/log/daemon.log	Contains information logged by the various background daemons that run on the system.	
/var/log/dmesg	Contains kernel ring buffer information. When the system boots up, it prints number of messages on the screen that display information about the hardware devices that the kernel detects during boot process. These messages are available in the kernel ring buffer and whenever a new message arrives, the old message gets overwritten. You can also view the content of this file using the <code>dmesg</code> command.	<code>dmesg</code> is one of the few places to determine hardware errors.
/var/log/dpkg.log	Contains information that is logged when a package is installed or removed using the <code>dpkg</code> command.	
/var/log/faillog	Contains failed user login attempts. Use the <code>faillog</code> command to display the contents of this file.	
/var/log/fsck/*	The <code>fsck</code> utility is used to check and optionally repair one or more Linux filesystems.	
/var/log/mail.log	Mail server logs.	
/var/log/messages	General messages and system related information.	
/var/log/monit.log	<code>monit</code> is a utility for managing and monitoring processes, files, directories and filesystems on a Unix system.	
/var/log/news/*	The <code>news</code> command keeps you informed of news concerning the system.	
/var/log/ntpstats	Logs for network configuration protocol.	
/var/log/kern.log	Kernel logs.	

Log	Description	Why is this important?
/var/log /switchd. log/	The HAL log for Cumulus RMP.	This is specific to Cumulus RMP. Any <code>switchd</code> crashes are logged here.
/var/log /syslog	The main system log, which logs everything except auth-related messages.	The primary log; it's easiest to <code>grep</code> this file to see what occurred during a problem.
/var/log /wtmp	Login records file.	
/var/log /yum.log	<code>apt</code> command log file.	

Troubleshooting the `etc` Directory

The `c1-support` (see page 172) script replicates the `/etc` directory.

Files that `c1-support` deliberately excludes are:

File	Description
/etc/nologin	<code>nologin</code> prevents unprivileged users from logging into the system.
/etc /alternatives	<code>update-alternatives</code> creates, removes, maintains and displays information about the symbolic links comprising the Debian alternatives system.

This is the alphabetical of the output from running `ls -l` on the `/etc` directory structure created by `c1-support`. The green highlighted rows are the ones Cumulus Networks finds most important when troubleshooting problems.

File	Description	Why is this important?
adduser.conf	The file <code>/etc/adduser.conf</code> contains defaults for the programs <code>adduser</code> , <code>addgroup</code> , <code>deluser</code> , and <code>delgroup</code> .	

File	Description	Why is this important?
adjtime	Corrects the time to synchronize the system clock .	
apt	apt (Advanced Package Tool) is the command-line tool for handling packages . This folder contains all the configurations.	apt interactions or unsupported apps can affect machine performance.
audit	The directory that contains <code>auditd.conf</code> , which is the file that controls the configuration of the audit remote logging subsystem .	
audit	The directory that contains the <code>/etc/audit/auditd.conf</code> , which contains configuration information specific to the audit daemon .	
bash.bashrc	Bash is an sh-compatible command language interpreter that executes commands read from standard input or from a file.	
bash_completion	This points to <code>/usr/share/bash-completion/bash_completion</code> .	
bash_completion.d	This folder contains app-specific code for Bash completion on Cumulus RMP, such as <code>mstpctl</code> .	
bcm.d	Broadcom-specific ASIC file structure (hardware interaction). If there are questions contact the Cumulus Networks Support team. This is unique to Cumulus RMP.	
bindresvport.blacklist	This file contains a list of port numbers between 600 and 1024, which should not be used by <code>bindresvport</code> .	
ca-certificates	The folder for <code>ca-certificates</code> . It is empty by default on Cumulus RMP; see below for more information.	
ca-certificates.conf	Each lines list the pathname of activated CA certificates under <code>/usr/share/ca-certificates</code> .	
calendar	The system-wide default calendar file .	
chef	This is an example of something that is not included by default. In this instance, <code>cl-support</code> included the chef folder for some reason.	This is not installed by default, but this tool could have been installed or

File	Description	Why is this important?
		configured incorrectly, which is why it's included in the <code>c1-support</code> output.
<code>cron.d</code>	<code>cron</code> is a daemon that executes scheduled commands.	
<code>cron.daily</code>	See above.	
<code>cron.hourly</code>	See above.	
<code>cron.monthly</code>	See above.	
<code>cron.weekly</code>	See above.	
<code>crontab</code>	See above.	
<code>cumulus</code>	<p>This directory contains the following:</p> <ul style="list-style-type: none"> • ACL information, stored in the <code>acl</code> directory. • <code>switchd</code> configuration file, <code>switchd.conf</code>. • <code>qos</code>, which is under the <code>datapath</code> directory. • The routing protocol process priority, <code>nice.conf</code>. • The breakout cable configuration, under <code>ports.conf</code>. 	This folder is specific to Cumulus RMP and does not exist on other Linux platforms. For example, while you can configure <code>iptables</code> , to hardware accelerate rules into the hardware you need to use <code>c1-acltool</code> and have the rules under the <code>/etc/cumulus/acl/policy.d/<filename.rules></code>
<code>debconf.conf</code>	Debconf is a configuration system for Debian packages.	
<code>debian_version</code>	The complete Debian version string.	
<code>debsums-ignore</code>	<code>debsums</code> verifies installed package files against their MD5 checksums. This file identifies the packages to ignore.	
<code>default</code>	This folder contains files with configurable flags for many different applications (most installed by default or added manually). For example, <code>/etc/default/networking</code> has a flag for <code>EXCLUDE_INTERFACES=</code> , which is set to nothing by default, but a user could change it to something like <code>swp3</code> .	
<code>deluser.conf</code>		

File	Description	Why is this important?
	The file <code>/etc/deluser.conf</code> contains defaults for the programs <code>deluser</code> and <code>delgroup</code> .	
<code>dhcp</code>	This directory contains DHCP-specific information .	
<code>dpkg</code>	The package manager for Debian.	
<code>e2fsck.conf</code>	The configuration file for e2fsck . It controls the default behavior of <code>e2fsck</code> while it checks <code>ext2</code> , <code>ext3</code> or <code>ext4</code> filesystems.	
<code>environment</code>	Utilized by <code>pam_env</code> for setting and unsetting environment variables.	
<code>ethertypes</code>	This file can be used to show readable characters instead of hexadecimal numbers for the protocols. For example, <code>0x0800</code> will be represented by <code>IPv4</code> .	
<code>fstab</code>	Static information about the filesystems .	
<code>fstab.d</code>	The directory that can contain additional <code>fstab</code> information; it is empty by default.	
<code>fw_env.config</code>	Configuration file utilized by <code>U-Boot</code> .	
<code>gai.conf</code>	Configuration file for sorting the return information from getaddrinfo .	
<code>groff</code>	The directory containing information for <code>groffer</code> , an application used for displaying Unix man pages .	
<code>group</code>	The <code>/etc/group</code> file is a text file that defines the groups on the system.	
<code>group-</code>	Backup for the <code>/etc/group</code> file.	
<code>gshadow</code>	<code>/etc/gshadow</code> contains the shadowed information for group accounts .	
<code>gshadow-</code>	Backup for the <code>/etc/gshadow</code> file.	
<code>host.conf</code>	Resolver configuration file , which contains options like <code>multi</code> that determines whether <code>/etc/hosts</code> will respond with multiple entries for DNS names.	

File	Description	Why is this important?
hostname	The system host name , such as leaf1, spine1, sw1.	
hosts	The static table lookup for hostnames.	
hosts.allow	The part of the host_access program for controlling a simple access control language. <code>hosts.allow=Access</code> is granted when a daemon/client pair matches an entry.	
hosts.deny	See hosts.allow above, except that access is denied when a daemon/client pair matches an entry.	
init	Default location of the system job configuration files .	
init.d	In order for a service to start when the switch boots, you should add the necessary script to the director here. The differences between <code>init</code> and <code>init.d</code> are explained well here .	
inittab	The format of the inittab file used by the <code>sysv</code> -compatible <code>init</code> process.	
inputrc	The initialization file utilized by <code>readline</code> .	
insserv	This application enables installed system init scripts ; this directory is empty by default.	
insserv.conf	Configuration file for insserv .	
insserv.conf.d	Additional directory for insserv configurations .	
iproute2	Directory containing values for the Linux command line tool <code>ip</code> .	
issue	<code>/etc/issue</code> is a text file that contains a message or system identification to be printed before the login prompt.	
issue.net	Identification file for telnet sessions .	
ld.so.cache	Contains a compiled list of candidate libraries previously found in the augmented library path.	
ld.so.conf	Used by the <code>ldconfig</code> tool, which configures dynamic linker run-time bindings .	

ld.so.conf.d	The directory that contains additional <code>ld.so.conf</code> configuration (see above).	
ldap	The directory containing the <code>ldap.conf</code> configuration file used to set the system-wide default to be applied when running LDAP clients.	
libaudit.conf	Configuration file utilized by <code>get_auditfail_action</code> .	
libnl-3	Directory for the configuration relating to the <code>libnl</code> library, which is the core library for implementing the fundamentals required to use the netlink protocol such as socket handling, message construction and parsing, and sending and receiving of data.	
lldpd.d	Directory containing configuration files whose commands are executed by <code>lldpdcli</code> at startup.	
localtime	Copy of the original data file for <code>/etc/timezone</code> .	
logcheck	Directory containing <code>logcheck.conf</code> and logfiles utilized by the <code>log_check</code> program, which scans system logs for interesting lines.	
login.defs	Shadow password suite configuration.	
logrotate.conf	Rotates, compresses and mails system logs.	
logrotate.d	Directory containing additional log rotate configurations.	
lsb-release	Shows the current version of Linux on the system. Run <code>cat /etc/lsb-release</code> for output.	This shows you the version of the operating system you are running; also compare this to the output of <code>cl-img-select</code> .
magic	Used by the <code>file</code> command to determine file type. <code>magic</code> tests check for files with data in particular fixed formats.	
magic.mime	The <code>magic</code> MIME type causes the <code>file</code> command to output MIME type strings rather than the more traditional human readable ones.	

File	Description	Why is this important?
mailcap	The <code>mailcap</code> file is read by the <code>metamail</code> program to determine how to display non-text at the local site .	
mailcap.order	The order of entries in the <code>/etc/mailcap</code> file can be altered by editing the <code>/etc/mailcap.order</code> file.	
manpath.config	The manpath configuration file is used by the manual page utilities to assess users' manpaths at run time, to indicate which manual page hierarchies (manpaths) are to be treated as system hierarchies and to assign them directories to be used for storing cat files.	
mime.types	MIME type description file for cups .	
mke2fs.conf	Configuration file for <code>mke2fs</code> , which is a program that creates an ext, ext3 or ext4 filesystem .	
modprobe.d	Configuration directory for <code>modprobe</code> , which is a utility that can add and remove modules from the Linux kernel .	
modules	The kernel modules to load at boot time .	
monit	<code>monit</code> is a utility for monitoring services on a Unix system ; this directory has configuration files beneath it.	
motd	The contents of <code>/etc/motd</code> (" message of the day ") are displayed by <code>pam_motd</code> after a successful login but just before it executes the login shell.	
mtab	The programs <code>mount</code> and <code>umount</code> maintain a list of currently mounted filesystems in the <code>/etc/mtab</code> file. If no arguments are given to <code>mount</code> , this list is printed.	
nanorc	The GNU <code>nano</code> <code>rcfile</code> .	
network	Contains the network interface configuration for <code>ifup</code> and <code>ifdown</code> .	The main configuration file is under <code>/etc/network/interfaces</code> . This is where you configure L2 and L3 information for all of your front panel ports (swp

File	Description	Why is this important?
		interfaces). Settings like MTU, link speed, IP address information, VLANs are all done here.
networks	Network name information.	
nsswitch.conf	System databases and name service switch configuration file.	
ntp.conf	NTP (network time protocol) server configuration file.	
openvswitch	The directory containing the <code>conf.db</code> file, which is used by <code>ovsdb-server</code> .	
openvswitch-vtep	Configuration files used for the VTEP daemon and <code>ovsdb-server</code> .	
opt	Host-specific configuration files for add-on applications installed in <code>/opt</code> .	
os-release	Operating system identification.	
pam.conf	The PAM (pluggable authentication module) configuration file. When a PAM-aware privilege granting application is started, it activates its attachment to the PAM-API. This activation performs a number of tasks, the most important being the reading of the configuration file(s).	
pam.d	Alternate directory to configure PAM (see above).	
passwd	User account information.	
passwd-	Backup file for <code>/etc/passwd</code> .	
perl	Perl is an available scripting language. <code>/etc/perl</code> contains configuration files specific to Perl.	
profile	<code>/etc/profile</code> is utilized by <code>sysprofile</code> , a modular centralized shell configuration.	
profile.d	The directory version of the above, which contains configuration files.	

File	Description	Why is this important?
protocols	The protocols definition file , a plain ASCII file that describes the various DARPAnet protocols that are available from the TCP/IP subsystem.	
ptm.d	The directory containing scripts that are run if PTM (see page 112) passes or fails.	Cumulus RMP-specific folder for PTM (prescriptive topology manager).
python	<code>python</code> is an available scripting language.	
python2.6	The 2.6 version of <code>python</code> .	
python2.7	The 2.7 version of <code>python</code> .	
rc.local	The <code>/etc/rc.local</code> script is used by the system administrator to execute after all the normal system services are started , at the end of the process of switching to a multiuser runlevel. You can use it to start a custom service, for example, a server that's installed in <code>/usr/local</code> . Most installations don't need <code>/etc/rc.local</code> ; it's provided for the minority of cases where it's needed .	
rc0.d	Like <code>rc.local</code> , these scripts are booted by default, but the number of the folder represents the Linux runlevel . This folder 0 represents runlevel 0 (halt the system).	
rc1.d	This is run level 1, which is single-user/minimal mode.	
rc2.d	Runlevels 2 through 5 are multiuser modes. Debian systems (such as Cumulus RMP) come with <code>id=2</code> , which indicates that the default runlevel will be 2 when the multi-user state is entered , and the scripts in <code>/etc/rc2.d/</code> will be run.	
rc3.d	See above.	
rc4.d	See above.	
rc5.d	See above.	
rc6.d	Runlevel 6 is reboot the system.	
rcS.d	S stands for <i>single</i> and is equivalent to rc1.	

File	Description	Why is this important?
resolv.conf	Resolver configuration file, which is where DNS is set (domain, nameserver and search).	You need DNS to reach the Cumulus RMP repository.
rmt	This is not a mistake. The shell script <code>/etc/rmt</code> is provided for compatibility with other Unix-like systems, some of which have utilities that expect to find (and execute) <code>rmt</code> in the <code>/etc</code> directory on remote systems.	
rpc	The <code>rpc</code> file contains human-readable names that can be used in place of RPC program numbers.	
rsyslog.conf	The <code>rsyslog.conf</code> file is the main configuration file for <code>rsyslogd</code> , which logs system messages on *nix systems.	
rsyslog.d	The directory containing additional configuration for <code>rsyslog.conf</code> (see above).	
securetty	This file lists terminals into which the root user can log in.	
security	The <code>/etc/security</code> directory contains security-related configurations files. Whereas PAM concerns itself with the methods used to authenticate any given user, the files under <code>/etc/security</code> are concerned with just what a user can or cannot do. For example, the <code>/etc/security/access.conf</code> file contains a list of which users are allowed to log in and from what host (for example, using telnet). The <code>/etc/security/limits.conf</code> file contains various system limits, such as maximum number of processes.	
selinux	NSA Security-Enhanced Linux.	
sensors.d	The directory from which the <code>sensors</code> program loads its configuration; this is unique for each hardware platform. See also Monitoring System Hardware (see page 167) .	
sensors3.conf	The <code>sensors.conf</code> file describes how <code>libsensors</code> , and thus all programs using it, should translate the raw readings from the kernel modules to real-world values.	
services		

File	Description	Why is this important?
	<code>services</code> is a plain ASCII file providing a mapping between human-readable textual names for internet services and their underlying assigned port numbers and protocol types.	
<code>shadow</code>	<code>shadow</code> is a file that contains the password information for the system's accounts and optional aging information.	
<code>shadow-</code>	The backup for the <code>/etc/shadow</code> file.	
<code>shells</code>	The pathnames of valid login shells.	
<code>skel</code>	The skeleton directory (usually <code>/etc/skel</code>) is used to copy default files and also sets a umask for the creation used by <code>pam_mkhomedir</code> .	
<code>snmp</code>	Interface functions to the <code>SNMP</code> (simple network management protocol) toolkit.	
<code>ssh</code>	The <code>ssh</code> configuration.	
<code>ssl</code>	The <code>OpenSSL</code> <code>ssl</code> library implements the Secure Sockets Layer (SSL v2/v3) and Transport Layer Security (TLS v1) protocols. This directory holds certificates and configuration.	
<code>staff-group-for-usr-local</code>	Use <code>cat</code> or <code>more</code> on this file to learn more information, see http://bugs.debian.org/299007 .	
<code>sudoers</code>	The <code>sudoers</code> policy plugin determines a user's <code>sudo</code> privileges.	
<code>sudoers.d</code>	The directory file containing additional <code>sudoers</code> configuration (see above).	
<code>sysctl.conf</code>	Configures kernel parameters at boot.	
<code>sysctl.d</code>	The directory file containing additional configuration (see above).	
<code>systemd</code>	<code>systemd</code> system and service manager.	
<code>terminfo</code>	Terminal capability database.	

File	Description	Why is this important?
timezone	If this file exists, it is read and its contents are used as the time zone name .	
ucf.conf	The update configuration file preserves user changes in configuration files.	
udev	Dynamic device management .	
ufw	Provides both a command line interface and a framework for managing a netfilter firewall .	
vim	Configuration file for command line tool vim .	
wgetrc	Configuration file for command line tool wget .	

Troubleshooting the support Directory

The `support` directory is unique in the fact that it is not a copy of the switch's filesystem. Actually, it is the output from various commands. For example:

File	Equivalent Command	Description
support /ip.addr	<pre>cumulus@switc h:~\$ ip addr show</pre>	This shows you all the interfaces (including swp front panel ports), IP address information, admin state and physical state.

Troubleshooting Network Interfaces

The following sections describe various ways you can troubleshoot `ifupdown2`.

Contents

(Click to expand)

- [Contents \(see page 188\)](#)
- [Enabling Logging for Networking \(see page 189\)](#)
- [Using ifquery to Validate and Debug Interface Configurations \(see page 189\)](#)
- [Debugging Mako Template Errors \(see page 191\)](#)
- [ifdown Cannot Find an Interface that Exists \(see page 191\)](#)

- Removing All References to a Child Interface (see page 192)
- MTU Set on a Logical Interface Fails with Error: "Numerical result out of range" (see page 193)
- Interpreting iproute2 batch Command Failures (see page 193)
- Understanding the "RTNETLINK answers: Invalid argument" Error when Adding a Port to a Bridge (see page 193)

Enabling Logging for Networking

The `/etc/default/networking` file contains two settings for logging:

- To get `ifupdown2` logs when the switch boots (stored in `syslog`)
- To enable logging when you run `systemctl [start|stop|reload] networking.service`

This file also contains an option for excluding interfaces when you boot the switch or run `systemctl start|stop|reload networking.service`. You can exclude any interface specified in `/etc/network/interfaces`. These interfaces do not come up when you boot the switch or start/stop/reload the networking service.

```
$cat /etc/default/networking
#
#
# Parameters for the /etc/init.d/networking script
#
#

# Change the below to yes if you want verbose logging to be enabled
VERBOSE="no"

# Change the below to yes if you want debug logging to be enabled
DEBUG="no"

# Change the below to yes if you want logging to go to syslog
SYSLOG="no"

# Exclude interfaces
EXCLUDE_INTERFACES=
```

Using 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 returns exit code `0` or `1` if the configuration does not match:

```
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]
```



`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
```

Debugging 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 was interrupted before it updated 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
```

Removing All References to a Child Interface

If you have a configuration with a child interface, whether it's 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 interface continues to be used by the parent 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 you comment out or simply delete bond1 from `/etc/network/interfaces`, 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
```

Interpreting 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)
```

Understanding the "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 can typically occur 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.

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.

Contents

(Click to expand)

- Contents (see page 194)
- Commands (see page 194)
- Monitoring Interfaces Using ethtool (see page 194)
 - Viewing and Clearing Interface Counters (see page 195)
- Monitoring Switch Port SFP/QSFP Using ethtool (see page 196)

Commands

- `cl-netstat`
- `ethtool`

Monitoring Interfaces Using ethtool

To check the status of an interface using `ethtool`:

```
cumulus@switch:~$ ethtool swp1
Settings for swp1:
    Supported ports: [ TP ]
    Supported link modes:   10baseT/Half 10baseT/Full
                           100baseT/Half 100baseT/Full
                           1000baseT/Full
    Supported pause frame use: Symmetric Receive-only
    Supports auto-negotiation: Yes
    Advertised link modes:  10baseT/Half 10baseT/Full
                           100baseT/Half 100baseT/Full
                           1000baseT/Half 1000baseT/Full
    Advertised pause frame use: Symmetric
    Advertised auto-negotiation: No
    Speed: 1000Mb/s
    Duplex: Full
    Port: FIBRE
    PHYAD: 0
    Transceiver: external
    Auto-negotiation: on
```

```
Current message level: 0x00000000 (0)
```

```
Link detected: yes
```

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
```

Viewing and Clearing 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      8391      0      0      0
9694      0          0          0  BMRU
lo        16436     0      1693      0      0      0
1693      0          0          0  LRU
swp1      1500      0      11914     0      8948     0
20854     0          9338      0  BMRU
swp2      1500      0      20734     0      17969     0
12033     0          13142     0  BMRU

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.
-d	Deletes saved statistics, either the <code>uid</code> or the specified tag.
-D	Deletes all saved statistics.
-j	Display in JSON format.
-l	Lists saved tags.
-r	Displays raw statistics (unmodified output of <code>cl-netstat</code>).
-t <tag name>	Saves statistics with <tag name>.
-v	Prints <code>cl-netstat</code> version and exits.

Monitoring Switch Port SFP/QSFP Using ethtool

To see hardware capabilities and measurement information on SFP or the 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 swp49
Identifier                                : 0xff (reserved or
unknown)
Optical diagnostics support              : Yes
Laser bias current                       : 130.046 mA
Laser output power                       : 6.5025 mW / 8.13 dBm
Receiver signal average optical power    : 6.5535 mW / 8.16 dBm
Module temperature                       : 0.00 degrees C / 32.00
degrees F
Module voltage                           : 6.5282 V
Alarm/warning flags implemented          : Yes
Laser bias current high alarm            : On
Laser bias current low alarm             : On
Laser bias current high warning          : On
Laser bias current low warning           : On
Laser output power high alarm            : On
Laser output power low alarm             : On
Laser output power high warning          : On
Laser output power low warning           : On
Module temperature high alarm            : On
Module temperature low alarm             : On
Module temperature high warning          : On
Module temperature low warning           : On
Module voltage high alarm                : On
Module voltage low alarm                 : On
Module voltage high warning              : On
Module voltage low warning               : On
Laser rx power high alarm                : On
Laser rx power low alarm                 : On
Laser rx power high warning              : On
Laser rx power low warning               : On
Laser bias current high alarm threshold  : 130.046 mA
Laser bias current low alarm threshold   : 130.046 mA
Laser bias current high warning threshold : 130.046 mA
Laser bias current low warning threshold : 130.046 mA
Laser output power high alarm threshold  : 6.5025 mW / 8.13 dBm
Laser output power low alarm threshold   : 6.5025 mW / 8.13 dBm
Laser output power high warning threshold : 6.5025 mW / 8.13 dBm
Laser output power low warning threshold : 6.5025 mW / 8.13 dBm
Module temperature high alarm threshold  : -1.00 degrees C / 30.20
degrees F
Module temperature low alarm threshold    : 0.00 degrees C / 32.00
degrees F
Module temperature high warning threshold : 0.00 degrees C / 32.00
```

```
degrees F
    Module temperature low warning threshold : 0.00 degrees C / 32.00
degrees F
    Module voltage high alarm threshold      : 6.5282 V
    Module voltage low alarm threshold       : 6.5282 V
    Module voltage high warning threshold    : 6.5282 V
    Module voltage low warning threshold     : 6.5282 V
    Laser rx power high alarm threshold      : 6.5535 mW / 8.16 dBm
    Laser rx power low alarm threshold       : 6.5535 mW / 8.16 dBm
    Laser rx power high warning threshold    : 6.5535 mW / 8.16 dBm
    Laser rx power low warning threshold     : 6.5535 mW / 8.16 dBm
```

Network Troubleshooting

Cumulus RMP contains a number of command line and analytical tools to help you troubleshoot issues with your network.

Contents

(Click to expand)

- [Contents \(see page 198\)](#)
- [Commands \(see page 198\)](#)
- [Checking Reachability Using ping \(see page 199\)](#)
- [Printing Route Trace Using traceroute \(see page 199\)](#)
- [Manipulating the System ARP Cache \(see page 199\)](#)
- [Generating Traffic Using mz \(see page 200\)](#)
- [Creating Counter ACL Rules \(see page 201\)](#)
- [Configuring SPAN and ERSPAN \(see page 202\)](#)
 - [Configuring SPAN for Switch Ports \(see page 202\)](#)
 - [Configuring SPAN for Bonds \(see page 206\)](#)
 - [Configuring ERSPAN \(see page 206\)](#)
 - [Selective Spanning \(see page 208\)](#)
 - [Removing SPAN Rules \(see page 210\)](#)
- [Useful Links \(see page 210\)](#)
- [Caveats and Errata \(see page 210\)](#)

Commands

- [arp](#)
- [cl-acltool](#)
- [ip](#)

- mz
- ping
- traceroute

Checking Reachability Using ping

`ping` is used 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
```

Printing 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
```

Manipulating 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
```

Generating Traffic Using mz

`mzis` 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 11.0.0.1 and destination 11.0.0.2, do the following:

```
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 - http://www.perihel.at/sec
/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=

```

Creating 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.



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

```

Configuring SPAN and ERSPAN

SPAN (Switched Port Analyzer) provides for the mirroring of all packets coming in from or going out of an interface to a local port for monitoring. This port is referred to as a mirror-to-port (MTP). The original packet is still switched, while a mirrored copy of the packet is sent to the MTP port.

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 |
-----

```

SPAN and ERSPAN are configured via `cl-acltool`. The match criteria for SPAN and ERSPAN can only be an interface; more granular match terms are not supported. The interface can be a port, a subinterface or a bond interface. Both ingress and egress interfaces can be matched.

Cumulus RMP supports a maximum of 2 SPAN destinations. Multiple rules can point to the same SPAN destination. The MTP interface can be a physical port, a subinterface, or a bond interface. The SPAN /ERSPAN action is independent of security ACL actions. If packets match both a security ACL rule and a SPAN rule, both actions will be carried out.



Always place your rules files under `/etc/cumulus/acl/policy.d/`.

Configuring SPAN for Switch Ports

This section describes how to set up, install, verify and uninstall SPAN rules. In the examples that follow, you will 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'
```



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.
```



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 were 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
```

Configuring 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'
```



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 were installed:

```
cumulus@switch:~$ sudo iptables -L -v | grep SPAN
    19  1938 SPAN          all -- any    bond0    anywhere
anywhere          dport:bond1
```

Configuring ERSPAN

This section describes how to configure ERSPAN for all packets coming in from swp1 to 12.0.0.2:

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'
```

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.
```

Verify that the ERSPAN rules were 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.



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.

Selective Spanning

SPAN/ERSPAN traffic rules can be configured to limit the traffic that is spanned, to reduce the volume of copied data.



Cumulus Linux 3.0 supports selective spanning for iptables only. ip6ables and ebttables 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



Only two unique mirror targets are supported in a given rule set.

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
```

Removing 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:~$
```

Configuration Files

- /etc/cumulus/acl/policy.conf

Useful Links

- <http://en.wikipedia.org/wiki/Ping>
- <https://en.wikipedia.org/wiki/Traceroute>
- <http://www.perihel.at/sec/mz/mzguide.html>

Caveats and Errata

- SPAN rules cannot match outgoing subinterfaces.

Using netshow to Troubleshoot Your Network Configuration

`netshow` is a tool in Cumulus RMP that quickly returns a lot of information about your network configuration. It's a tool designed by network operators for network troubleshooters since existing command line tools have too many options. `netshow` addresses this by leveraging the network troubleshooting experience from a wide group of troubleshooters and boiling it down to just a few important options. `netshow` quickly aggregates basic network information on Linux devices with numerous interfaces. `netshow` intelligently informs the administrator what network type an interface belongs to, and shows the most relevant information to a network administrator.

`netshow` can be used on any distribution of Linux, not just Cumulus RMP.

Installing netshow

Starting with Cumulus RMP 3.0.0, `netshow` is installed by default in Cumulus RMP.

Using netshow

Running `netshow` with no arguments displays all available command line arguments usable by `netshow`. (Running `netshow --help` gives you the same information.) The output looks like this:

```
cumulus@leaf1$ netshow
Usage:
    netshow system
    netshow counters [errors] [all] [-l | --legend ]
    netshow lldp [-l | --legend ]
    netshow interface [<iface>] [all] [--mac | -m ] [--oneline | -l | -l |
--legend ]
    netshow access [all] [--mac | -m ] [--oneline | -l | --legend | -l ]
    netshow bridges [all] [--mac | -m ] [--oneline | -l | --legend | -l ]
    netshow bonds [all] [--mac | -m ] [--oneline | -l | --legend ]
    netshow bondmems [all] [--mac | -m ] [--oneline | -l | -l | --legend ]
    netshow mgmt [all] [--mac | -m ] [--oneline | -l | -l | --legend ]
    netshow l2 [all] [--mac | -m ] [--oneline | -l | -l | --legend ]
    netshow l3 [all] [--mac | -m ] [--oneline | -l | -l | --legend ]
    netshow trunks [all] [--mac | -m ] [--oneline | -l | -l | --legend ]
    netshow (--version | -V)

Help:
    * default is to show interfaces only in the UP state.
    counters                summary of physical port counters.
    interface               summary info of all interfaces
    access                  summary of physical ports with l2 or l3 config
    bonds                   summary of bonds
    bondmems                summary of bond members
    bridges                 summary of ports with bridge members
    mgmt                    summary of mgmt ports
    l3                      summary of ports with an IP.
    l2                      summary of access, trunk and bridge interfaces
    phy                     summary of physical ports
    trunks                  summary of trunk interfaces
    lldp                    physical device neighbor information
    interface <iface>      list summary of a single interface
    system                  system information

Options:
    all                    show all ports include those are down or admin down
    --mac                  show interface MAC in output
    --version              netshow software version
```

```
--oneline  output each entry on one line
-l         alias for --oneline
-l         alias for --legend
--legend   print legend key explaining abbreviations
```

A Linux administrator can quickly see the few options available with the tool. One core tenet of `netshow` is for it to have a small number of command options. `netshow` is not designed to solve your network problem, but to help answer this simple question: "What is the basic network setup of my Linux device?" By helping to answer that question, a Linux administrator can spend more time troubleshooting the specific network problem instead of spending most of their time understanding the basic network state.

Originally developed for Cumulus Linux, `netshow` works on Debian-based servers and switches and Red Hat-based Linux systems.

`netshow` is designed by network operators, which has rarely occurred in the networking industry, where most command troubleshooting tools are designed by developers and are most useful in the network application development process.

Showing Interfaces

To show all available interfaces that are physically UP, run `netshow interface`:

```
cumulus@leaf1:~$ netshow interface
-----
To view the legend, rerun "netshow" cmd with the "--legend" option
-----
```

	Name	Speed	MTU	Mode	Summary
UP	eth0	1G	1500	Mgmt	IP: 192.168.0.12/24(DHCP)
UP	lo	N/A	16436	Mgmt	IP: 127.0.0.1/8, ::1/128

Whereas `netshow interface all` displays every interface regardless of state:

```
cumulus@leaf1:~$ netshow interface all
```

	Name	Speed	Mtu	Mode	Summary
UP	lo	N/A	16436	Loopback	IP: 127.0.0.1/8, ::1/128
UP	eth0	1G	1500	Mgmt	IP: 192.168.0.11/24 (DHCP)
ADMDN	swp1s0	10G(4x10)	1500	Unknwn	
ADMDN	swp1s1	10G(4x10)	1500	Unknwn	
ADMDN	swp1s2	10G(4x10)	1500	Unknwn	
ADMDN	swp1s3	10G(4x10)	1500	Unknwn	
ADMDN	swp2	40G(QSFP)	1500	Unknwn	
ADMDN	swp3	40G(QSFP)	1500	Unknwn	
ADMDN	swp4	40G(QSFP)	1500	Unknwn	

ADMDN	swp5	40G(QSFP)	1500	Unknwn
ADMDN	swp6	40G(QSFP)	1500	Unknwn
ADMDN	swp7	40G(QSFP)	1500	Unknwn
ADMDN	swp8	40G(QSFP)	1500	Unknwn
ADMDN	swp9	40G(QSFP)	1500	Unknwn
ADMDN	swp10	40G(QSFP)	1500	Unknwn
ADMDN	swp11	40G(QSFP)	1500	Unknwn
ADMDN	swp12	40G(QSFP)	1500	Unknwn
ADMDN	swp13	40G(QSFP)	1500	Unknwn
ADMDN	swp14	40G(QSFP)	1500	Unknwn
ADMDN	swp15	40G(QSFP)	1500	Unknwn
ADMDN	swp16	40G(QSFP)	1500	Unknwn
ADMDN	swp17	40G(QSFP)	1500	Unknwn
ADMDN	swp18	40G(QSFP)	1500	Unknwn
ADMDN	swp19	40G(QSFP)	1500	Unknwn
ADMDN	swp20	40G(QSFP)	1500	Unknwn
ADMDN	swp21	40G(QSFP)	1500	Unknwn
ADMDN	swp22	40G(QSFP)	1500	Unknwn
ADMDN	swp23	40G(QSFP)	1500	Unknwn
ADMDN	swp24	40G(QSFP)	1500	Unknwn
ADMDN	swp25	40G(QSFP)	1500	Unknwn
ADMDN	swp26	40G(QSFP)	1500	Unknwn
ADMDN	swp27	40G(QSFP)	1500	Unknwn
ADMDN	swp28	40G(QSFP)	1500	Unknwn
ADMDN	swp29	40G(QSFP)	1500	Unknwn
ADMDN	swp30	40G(QSFP)	1500	Unknwn
ADMDN	swp31	40G(QSFP)	1500	Unknwn
ADMDN	swp32s0	10G(4x10)	1500	Unknwn
ADMDN	swp32s1	10G(4x10)	1500	Unknwn
ADMDN	swp32s2	10G(4x10)	1500	Unknwn
ADMDN	swp32s3	10G(4x10)	1500	Unknwn

You can get information about the switch itself by running `netshow system`:

```
cumulus@leaf1:~$ netshow system

Arctica 4804IP
Cumulus Version 3.0.0~1462473422.02602ac
Build: Cumulus RMP 3.0.0~1462473422.02602ac

Chipset: Broadcom Hurricane2

Port Config: 48 x 10G-SFP+ & 4 x 40G-QSFP+
```

```
CPU: (x86_64) Intel Atom C2758 2.40GHz
```

```
Uptime: 2 days, 21:31:00
```

```
cumulus@leaf1:~$
```

Other Useful netshow Features

`netshow` uses the `python network-docopt` package. This is inspired by `docopt` and provides the ability to specify partial commands, without tab completion and running the complete option. For example:

```
netshow int RUNS netshow interface
netshow sys RUNS netshow system
```

`netshow` will eventually support interface name autocompletion. In the near future, if you run `netshow int tap123` and there is only one interface starting with `tap123`, `netshow` will autocomplete the command option with the full interface.

Contributions Welcome!

`netshow` is an open source project licensed under GPLv2. To contribute please contact Cumulus Networks through the [Cumulus Community Forum](#) or the [Netshow Linux Provider Github Repository Home](#). You can find developer documentation at netshow.readthedocs.org. The documentation is still under development.

Monitoring System Statistics and Network Traffic with sFlow

`sFlow` is a monitoring protocol that samples network packets, application operations, and system counters. `sFlow` enables you to 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. `hsflowd` is not included in the base Cumulus RMP installation. After installation, `hsflowd` will automatically start when the switch boots up.

Contents

(Click to expand)

- [Contents \(see page 214\)](#)
- [Installing hsflowd \(see page 214\)](#)
- [Configuring sFlow \(see page 215\)](#)
 - [Configuring sFlow via DNS-SD \(see page 215\)](#)
 - [Manually Configuring /etc/hsflowd.conf \(see page 215\)](#)
- [Configuring sFlow Visualization Tools \(see page 216\)](#)
- [Configuration Files \(see page 216\)](#)
- [Useful Links \(see page 216\)](#)

Installing *hsflowd*

To download and install the *hsflowd* package, use `apt-get`:

```
cumulus@switch:~$ sudo apt-get update
cumulus@switch:~$ sudo apt-get install -y hsflowd
```

Configuring *sFlow*

You can configure *hsflowd* to send to the designated collectors via two methods:

- DNS service discovery (DNS-SD)
- Manually configuring `/etc/hsflowd.conf`

Configuring *sFlow* via *DNS-SD*

With this method, you need to 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.1G=2048"
    "sampling.10G=4096"
    "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.

After the initial configuration is ready, bring up the sFlow daemon by running:

```
cumulus@switch:~$ sudo systemctl start hsflowd.service
```

No additional configuration is required in `/etc/hsflowd.conf`.

Manually Configuring `/etc/hsflowd.conf`

With this method you will set up the collectors and variables on each switch.

Edit `/etc/hsflowd.conf` and change `DNSSD = on` to `DNSSD = off`:

```
DNSSD = off
```

Then set up your collectors and sampling rates in `/etc/hsflowd.conf`:

```
# Manual Configuration (requires DNSSD=off above)
#####

# Typical configuration is to send every 30 seconds
polling = 20

sampling.1G=2048
sampling.10G=4096

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 2048 packets 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.



Some collectors require each source to transmit on a different port, others may listen on only one port. Please refer to the documentation for your collector for more information.

Configuring sFlow Visualization Tools

For information on configuring various sFlow visualization tools, read this [Help Center article](#).

Configuration Files

- `/etc/hsflowd.conf`

Useful Links

- [sFlow Collectors](#)
- [sFlow Wikipedia page](#)

SNMP Monitoring

Cumulus RMP utilizes the open source Net-SNMP agent `snmpd`, v5.7.3, which provides support for most of the common industry-wide MIBs, including interface counters and TCP/UDP IP stack data.



Cumulus RMP does not prevent customers from extending SNMP features. However, Cumulus Networks encourages the use of higher performance monitoring environments, rather than SNMP.

Contents

(Click to expand)

- [Contents \(see page 217\)](#)
- [Introduction to SNMP \(Simple Network Management Protocol\) \(see page 217\)](#)
- [Configuring Ports for SNMP to Listen for Requests \(see page 218\)](#)
- [Starting the SNMP Daemon \(see page 218\)](#)
- [Configuring SNMP \(see page 219\)](#)
 - [Setting up the Custom Cumulus Networks MIBs \(see page 219\)](#)
 - [Enabling the .1.3.6.1.2.1 Range \(see page 219\)](#)
 - [Enabling Public Community \(see page 220\)](#)
 - [Configuring SNMPv3 \(see page 221\)](#)
- [Configuring Nutanix Prism \(see page 223\)](#)
 - [Cumulus RMP Configuration \(see page 223\)](#)
 - [Nutanix Configuration \(see page 225\)](#)
- [Switch Information Displayed on Nutanix Prism \(see page 228\)](#)
- [Troubleshooting \(see page 228\)](#)
- [Enabling LLDP/CDP on VMware ESXi \(Hypervisor on Nutanix\) \(see page 229\)](#)
- [Enabling LLDP/CDP on Nutanix Acropolis \(Hypervisor on Nutanix Acropolis\) \(see page 231\)](#)
- [snmpwalk the Switch from Another Linux Device \(see page 231\)](#)
- [Troubleshooting Connections without LLDP or CDP \(see page 233\)](#)
- [SNMP Traps \(see page 235\)](#)
 - [snmptrapd.conf \(see page 235\)](#)
 - [Generating Event Notification Traps \(see page 236\)](#)
- [Supported MIBs \(see page 239\)](#)

Introduction to SNMP (Simple Network Management Protocol)

SNMP is an IETF standards-based network management architecture and protocol that traces its roots back to Carnegie-Mellon University in 1982. Since then, it's been modified by programmers at the University of California. In 1995, this code was also made publicly available as the UCD project. After that, `ucd-snmp` was

extended by work done at the University of Liverpool as well as later in Denmark. In late 2000, the project name changed to net-snmp and became a fully-fledged collaborative open source project. The version used by Cumulus Networks is based on the latest `net-snmp` 5.7.3 branch with added custom MIBs and pass through and pass persist scripts.

Configuring Ports for SNMP to Listen for Requests

For security reasons, the default port binding for `snmpd` is the loopback local address; consequently by default, the SNMP service does not listen for SNMP requests from outside the switch. In order to listen to requests from outside the switch, you need to change this binding to a specific IP address (or all interfaces) after configuring security access (community strings, users, and so forth). This is a change from older versions of Cumulus RMP (before version 3.0), which listened to incoming requests on all interfaces by default. The `snmpd` configuration file is `/etc/snmp/snmpd.conf` and should be modified before enabling and starting `snmpd`. The default configuration has no access community strings defined so `snmpd` will not respond to any SNMP requests until this is added.

Starting the SNMP Daemon

The following procedure is the recommended process to start `snmpd` and monitor it using `systemctl`.

To start the SNMP daemon:

1. Start the `snmpd` daemon:

```
cumulus@switch:~$ sudo systemctl start snmpd.service
```

2. Configure the `snmpd` daemon to start automatically after reboot:

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

3. To enable `snmpd` to restart automatically after failure:

- a. Create a file called `/etc/systemd/system/snmpd.service.d/restart.conf`.
- b. Add the following lines:

```
[Service]
Restart=always
RestartSec=60
```

- c. Run `systemctl daemon-reload`.

Once the service is started, SNMP can be used to manage various components on the Cumulus RMP switch.

Configuring SNMP

Cumulus RMP ships with a production usable default `snmpd.conf` file included. This section covers a few basic configuration options in `snmpd.conf`. For more information regarding further configuring this file, refer to the `snmpd.conf` man page.



The default `snmpd.conf` file does not include all supported MIBs or OIDs that can be exposed.



Customers must at least change the default community string for v1 or v2c environments or the `snmpd` daemon will not respond to any requests.

Setting up the Custom Cumulus Networks MIBs



No changes are required in the `/etc/snmp/snmpd.conf` file on the switch, in order to support the custom Cumulus Networks MIBs. The following lines are already included by default:

```
view systemonly included .1.3.6.1.4.1.40310.1
view systemonly included .1.3.6.1.4.1.40310.2
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, several files need to be copied to the server, in order for the custom Cumulus MIB to be recognized on the destination NMS server.

- `/usr/share/snmp/Cumulus-Snmp-MIB.txt`
- `/usr/share/snmp/Cumulus-Counters-MIB.txt`
- `/usr/share/snmp/Cumulus-Resource-Query-MIB.txt`

Enabling the .1.3.6.1.2.1 Range

Some MIBs, including storage information, are not included by default in `snmpd.conf` in Cumulus RMP. This results in some default views on common network tools (like `librenms`) to return less than optimal data. More MIBs can be included, by enabling all the `.1.3.6.1.2.1` range. This simplifies the configuration file, removing concern that any required MIBs will be missed by the monitoring system. Various new MIBs were added for 3.0 and include the following: ENTITY and ENTITY-SENSOR MIB and parts of the BRIDGE-MIB and Q-BRIDGE-MIBs. These are included in the default configuration (Note: the view of the BRIDGE-MIB and Q-BRIDGE-MIB are commented out).



This configuration grants access to a large number of MIBs, including all MIB2 MIBs, which could reveal more data than expected. In addition to being a security vulnerability, it could consume more CPU resources.

To enable the .1.3.6.1.2.1 range:

1. Open `/etc/snmp/snmpd.conf` in a text editor.
2. Make sure the following lines are included in the configuration:

```
#####
#####
#
#  ACCESS CONTROL
#

# system
view  systemonly  included  .1.3.6.1.2.1
# lldpd (Note: lldpd must be restarted with the -x option
```

```
#    configured in order to send info to snmpd via Agent X
```

```
view  systemonly  included  .1.0.8802.1.1.2
# Cumulus specific
view  systemonly  included  .1.3.6.1.4.1.40310.1
view  systemonly  included  .1.3.6.1.4.1.40310.2
```

3. Restart `snmpd`:

```
# sudo systemctl start snmpd.service
```

Enabling Public Community

The `snmpd` authentication for versions 1 and 2 is disabled by default in Cumulus RMP. This password (called a community string) can be enabled by setting **rocommunity** (for read-only access) or **rwcommunity** (for read-write access). To enable read-only querying by a client:

1. Open `/etc/snmp/snmpd.conf` in a text editor.
2. To allow read-only access using a password *public* from any client IP address (*default*) for the view you defined before with *systemonly*, add the following line to the end of the file, then save it:

```
rocommunity public default -V systemonly
```

3. Restart `snmpd`:

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

Configuring SNMPv3

Since community strings in versions 1 and 2c are sent in the clear, SNMPv3 is often used to enable authentication and encryption. SNMPv3 was first release around 2000. A minimal example is shown here for `/etc/snmp/snmpd.conf` that defines three users, each with a different combination of authentication and encryption. Please change these usernames and passwords before using this in a network:



Make sure you change the usernames and passwords in the sample code below, as the ones used here are for explanatory purposes only.

```
# simple no auth user
createUser user1
# user with MD5 authentication
createUser user2 MD5 user2password
# user with MD5 for auth and DES for encryption
createUser user3 MD5 user3password DES user3encryption
# user999 with MD5 for authentication and DES for encryption

createUser user666 SHA user666password AES user666encryption
createUser user999 MD5 user999password DES user999encryption

# restrict users to certain OIDs
# (Note: creating rouser or rwuser will give
# access regardless of the createUser command above. However,
# createUser without rouser or rwuser will not provide any access).
rouser user1 noauth 1.3.6.1.2.1.1
rouser user2 auth 1.3.6.1.2.1
rwuser user3 priv 1.3.6.1.2.1
rwuser user666
rwuser user999
```

Once you make this configuration and restart the `snmpd` daemon, the user access can be checked with a client — the Debian package called `snmp` contains `snmpget` and `snmpwalk`, as well as other programs that are useful for checking daemon functionality from the switch itself or from another workstation. The following commands check the access for each user defined above from the localhost (the switch itself):

```
# check user1 which has no authentication or encryption (NoauthNoPriv)
snmpget -v 3 -u user1 -l NoauthNoPriv localhost 1.3.6.1.2.1.1.1.0
snmpwalk -v 3 -u user1 -l NoauthNoPriv localhost 1.3.6.1.2.1.1

# check user2 which has authentication but no encryption (authNoPriv)
snmpget -v 3 -u user2 -l authNoPriv -a MD5 -A user2password localhost
1.3.6.1.2.1.1.1.0
snmpget -v 3 -u user2 -l authNoPriv -a MD5 -A user2password localhost
1.3.6.1.2.1.2.1.0
snmpwalk -v 3 -u user2 -l authNoPriv -a MD5 -A user2password localhost
1.3.6.1.2.1

# check user3 which has both authentication and encryption (authPriv)
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
snmpwalk -v 3 -u user3 -l authPriv -a MD5 -A user3password -x DES -X
user3encryption localhost .1.3.6.1.2.1
```

```
snmpwalk -v 3 -u user666 -l authPriv -a SHA -x AES -A user666password -X
user666encryption localhost 1.3.6.1.2.1.1
snmpwalk -v 3 -u user999 -l authPriv -a MD5 -x DES -A user999password -X
user999encryption localhost 1.3.6.1.2.1.1
```

A slightly more secure method of configuring SNMPv3 users without creating cleartext passwords is the following:

1. Install the `net-snmp-config` script that is in `libsnmp-dev` package:

```
cumulus@switch:~$ sudo apt-get update
cumulus@switch:~$ sudo apt-get install libsnmp-dev
```

2. Stop the daemon:

```
cumulus@switch:~$ sudo systemctl stop snmpd
```

3. Use the `net-snmp-config` command to create two users, one with MD5 and DES, and the next with SHA and AES.



The minimum password length is 8 characters and the arguments `-a` and `-x` to `net-snmp-config` have different meanings than they do for `snmpwalk`.

```
cumulus@switch:~$ sudo net-snmp-config --create-snmpv3-user -a
md5authpass -x desprivpass -A MD5 -X DES userMD5withDES
cumulus@switch:~$ sudo net-snmp-config --create-snmpv3-user -a
shaauthpass -x aesprivpass -A SHA -X AES userSHAwithAES
cumulus@switch:~$ sudo systemctl start snmpd
```

This adds a `createUser` command in `/var/lib/snmp/snmpd.conf`. Do **not** edit this file by hand, unless you are removing usernames. It also adds the `rwuser` in `/usr/share/snmp/snmpd.conf`. You may want to edit this file and restrict access to certain parts of the MIB by adding `noauth`, `auth` or `priv` to allow unauthenticated access, require authentication or to enforce use of encryption, respectively.

The `snmpd` daemon reads the information from the `/var/lib/snmp/snmpd.conf` file and then the line is removed (eliminating the storage of the master password for that user) and replaced with the key that is derived from it (using the EngineID). This key is a localized key, so that if it is stolen it cannot be used to access other agents. To remove the two users `userMD5withDES` and `userSHAwithAES`, you need simply stop the `snmpd` daemon and edit the files `/var/lib/snmp/snmpd.conf` and `/usr/share/snmp/snmpd.conf`. Simply remove the lines containing the username. Then restart the `snmpd` daemon as in step 3 above.

From a client, you would access the MIB with the correct credentials. (Again, note that the roles of `-x`, `-a` and `-X` and `-A` are reversed on the client side as compared with the `net-snmp-config` command used above.)

```
snmpwalk -v 3 -u userMD5withDES -l authPriv -a MD5 -x DES -A md5authpass -X
desprivpass localhost 1.3.6.1.2.1.1.1
snmpwalk -v 3 -u userSHAwithAES -l authPriv -a SHA -x AES -A shaauthpass -X
aesprivpass localhost 1.3.6.1.2.1.1.1
```

Configuring Nutanix Prism

Nutanix Prism is a graphical user interface (GUI) for managing infrastructures and virtual environments.

Cumulus RMP Configuration

1. SSH to the Cumulus RMP switch that needs to be configured, replacing `[switch]` below as appropriate:

```
cumulus@workbench:~$ ssh cumulus@[switch]
```

2. Confirm the switch is running Cumulus RMP 2.5.5 or newer:

```
cumulus@switch$ cat /etc/lsb-release
DISTRIB_ID="Cumulus RMP"
DISTRIB_RELEASE=2.5.5
DISTRIB_DESCRIPTION=2.5.5-4cd66d9-201512071809-build
```

3. Open the `/etc/snmp/snmpd.conf` file in an editor.
4. Uncomment the following 3 lines in the `/etc/snmp/snmpd.conf` file, and 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
```

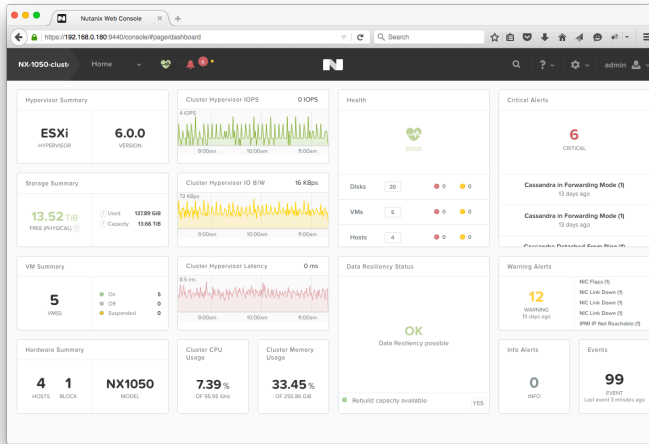
5. Restart `snmpd`:

```
cumulus@switch$ sudo systemctl restart snmpd.service
```

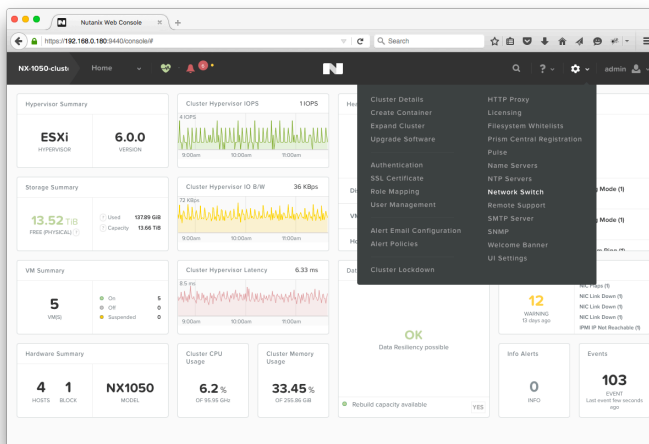
```
Restarting network management services: snmpd.
cumulus@switch$
```


Nutanix Configuration

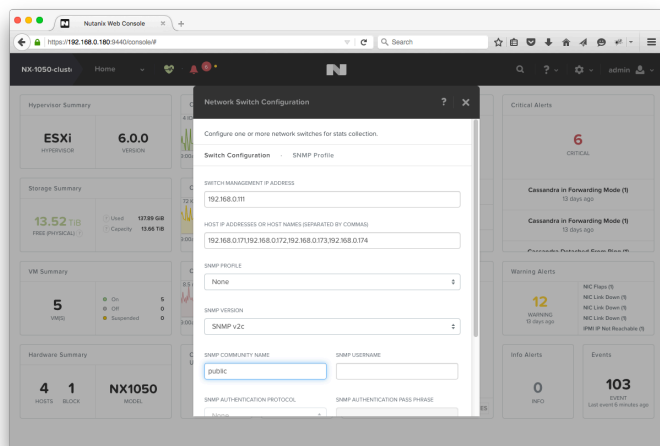
1. Log into the Nutanix Prism. Nutanix defaults to the Home menu, referred to as the Dashboard:



2. Click on the gear icon  in the top right corner of the dashboard, and 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:

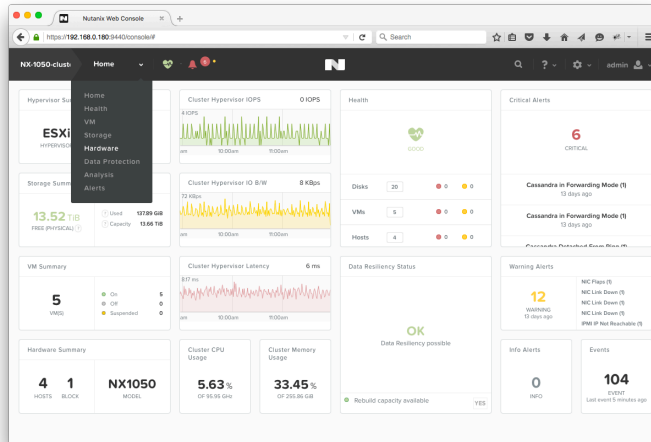


Configuration Parameter	Description	Value Used in Example
Switch Management IP Address	This can be any IP address on the box. In the screenshot above, the eth0 management IP is used.	192.168.0.111
Host IP Addresses or Host Names	IP addresses of Nutanix hosts connected to that particular ToR switch.	192.168.0.171,192.168.0.172,192.168.0.173,192.168.0.174
SNMP Profile	Saved profiles, for easy configuration when hooking up to multiple switches.	None
SNMP Version	SNMP v2c or SNMP v3. Cumulus RMP 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

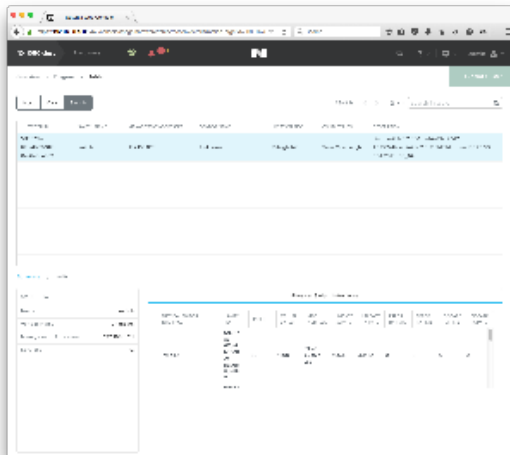


The rest of the values were not touched for this demonstration. They are usually used with SNMP v3.

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:



The switch has been added correctly, when interfaces hooked up to the Nutanix hosts are visible.

Switch Information Displayed on Nutanix Prism

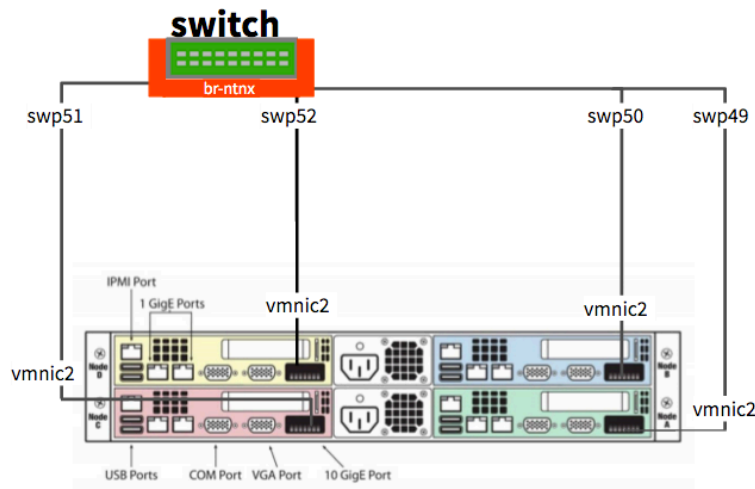
- Physical Interface (e.g. swp1, swp2). This will only display swp interfaces connected to Nutanix hosts by default.
- Switch ID - Unique identifier that Nutanix keeps track of each port ID (see below)
- Index - interface index, in the above demonstration swp49 maps to Index 52 because there is a loopback 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+c12.5+2 #3.
2.65-1+deb7u2+c12.5+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
```

Troubleshooting

To help visualize the following diagram is provided:



Nutanix Node	Physical Port	Cumulus RMP Port
Node A (Green)	vmnic2	swp49
Node B (Blue)	vmnic2	swp50
Node C (Red)	vmnic2	swp51
Node D (Yellow)	vmnic2	swp52

Enabling LLDP/CDP on VMware ESXi (Hypervisor on Nutanix)

- Follow the directions on one of the following websites to enable CDP:
 - http://kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=1003885
 - <http://wahlnetwork.com/2012/07/17/utilizing-cdp-and-ldp-with-vsphere-networking/>
- e.g. 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

```

The **both** means CDP is now running, and the `lldpd` daemon on Cumulus RMP is capable of "seeing" CDP devices.

2. After the next CDP interval, the Cumulus RMP box will pick 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 `netshow` to look at `lldp` information:

```

cumulus@switch$ netshow lldp
-----
To view the legend, rerun "netshow" cmd with the "--legend" option
-----
Local Port      Speed      Mode      Remote Port  Remote
Host            Summary

```

```

-----
eth0          1G          Mgmt          ==== swp32          swoob.vsockt.
local IP: 192.168.0.111/24(DHCP)
swp49         10G(SFP+)    Access/L2    ==== vmnic2         NX-1050-
A             Untagged: br-ntnx
swp50         10G(SFP+)    Access/L2    ==== vmnic2         NX-1050-
B             Untagged: br-ntnx
swp51         10G(SFP+)    Access/L2    ==== vmnic2         NX-1050-
C             Untagged: br-ntnx
swp52         10G(SFP+)    Access/L2    ==== vmnic2         NX-1050-
D             Untagged: br-ntnx

```

Enabling LLDP/CDP on Nutanix Acropolis (Hypervisor on Nutanix Acropolis)

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 RMP. LLDP should be enabled for each interface on the host. Refer to <https://community.mellanox.com/docs/DOC-1522> for setup instructions.

snmpwalk the Switch from Another Linux Device

One of the most important ways to troubleshoot is to snmpwalk the switch from another Linux device that can reach the switch running Cumulus RMP. For this demonstration, another switch running Cumulus RMP within the network is used.

1. Open `/etc/apt/sources.list` in an editor.
2. Add the following line, and save the file:

```
deb http://ftp.us.debian.org/debian/ jessie main non-free
```

3. Update the switch:

```
cumulus@switch2$ sudo apt-get update
```

4. Install the snmp and snmp-mibs-downloader packages:

```
cumulus@switch2$ sudo apt-get install snmp snmp-mibs-downloader
```

5. Verify that the "mibs :" line is commented out in `/etc/snmp/snmp.conf`:

```
#
# 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. Perform an snmpwalk on the switch. The switch running snmpd in the demonstration is using IP address 192.168.0.111. It is possible to snmpwalk the switch from itself, following these instructions, ruling out an snmp problem vs networking problem.

```
cumulus@switch2$ snmpwalk -c public -v2c 192.168.0.111
```

Output Examples

```
IF-MIB::ifPhysAddress.2 = STRING: 74:e6:e2:f5:a2:80
IF-MIB::ifPhysAddress.3 = STRING: 0:e0:ec:25:b8:54
IF-MIB::ifPhysAddress.4 = STRING: 74:e6:e2:f5:a2:81
IF-MIB::ifPhysAddress.5 = STRING: 74:e6:e2:f5:a2:82
IF-MIB::ifPhysAddress.6 = STRING: 74:e6:e2:f5:a2:83
IF-MIB::ifPhysAddress.7 = STRING: 74:e6:e2:f5:a2:84
IF-MIB::ifPhysAddress.8 = STRING: 74:e6:e2:f5:a2:85
IF-MIB::ifPhysAddress.9 = STRING: 74:e6:e2:f5:a2:86
IF-MIB::ifPhysAddress.10 = STRING: 74:e6:e2:f5:a2:87
IF-MIB::ifPhysAddress.11 = STRING: 74:e6:e2:f5:a2:88
IF-MIB::ifPhysAddress.12 = STRING: 74:e6:e2:f5:a2:89
IF-MIB::ifPhysAddress.13 = STRING: 74:e6:e2:f5:a2:8a
IF-MIB::ifPhysAddress.14 = STRING: 74:e6:e2:f5:a2:8b
IF-MIB::ifPhysAddress.15 = STRING: 74:e6:e2:f5:a2:8c
IF-MIB::ifPhysAddress.16 = STRING: 74:e6:e2:f5:a2:8d
IF-MIB::ifPhysAddress.17 = STRING: 74:e6:e2:f5:a2:8e
IF-MIB::ifPhysAddress.18 = STRING: 74:e6:e2:f5:a2:8f
IF-MIB::ifPhysAddress.19 = STRING: 74:e6:e2:f5:a2:90
```

Any information gathered here should verify that snmpd is running correctly on the Cumulus RMP side, reducing locations where a problem may reside.

Troubleshooting Tips Table for snmp walks

Run snmpwalk from	If it works	If it does not work
switch (switch to be monitored)	snmpd is serving information correctly Problem resides somewhere else (e.g. network connectivity, Prism misconfiguration)	Is snmpd misconfigured or installed incorrectly?
switch2 (another Cumulus RMP switch in the network)	snmpd is serving information correctly and network reachability works between switch and switch2 Problems resides somewhere else (e.g. can Prism reach switch , Prism misconfiguration)	Network connectivity is not able to grab information? Is there an iptables rule blocking? Is the snmp walk being run correctly?
Nutanix Prism CLI (ssh to the cluster IP address)	snmpd is serving information correctly and network reachability works between switch and the Nutanix Appliance Problems resides somewhere else (e.g. The GUI might be misconfigured)	Is the right community name being used in the GUI? Is snmp v2c being used?

Troubleshooting 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 (e.g. 0c:c4:7a:09:a2:43 represents vmnic0 which is tied to NX-1050-A).
3. List out all the MAC addresses associated to 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;p`

```
cumulus@switch$ brctl showmacs br-ntnx | grep 0c:c4:7a:09:a2:43
swp49      0c:c4:7a:09:a2:43      0      no      4.58
cumulus@switch$
```

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
-----
-----
cumulus@switch$

```

SNMP Traps

snmptrapd.conf

The Net-SNMP trap daemon configuration file, `/etc/snmptrapd.conf`, is used to configure how incoming traps should be processed. For more information about specific configuration options within the file, run the following command in a terminal:

```
cumulus@switch:~$ man 5 snmptrapd.conf
```

```

#####
###

```

```

#
# EXAMPLE-trap.conf:
#   An example configuration file for configuring the Net-SNMP snmptrapd
#   agent.
#
#####
###
#
# This file is intended to only be an example.  If, however, you want
# to use it, it should be placed in /etc/snmp/snmptrapd.conf.
# When the snmptrapd agent starts up, this is where it will look for it.
#
# All lines beginning with a '#' are comments and are intended for you
# to read.  All other lines are configuration commands for the agent.
#
# PLEASE: read the snmptrapd.conf(5) manual page as well!

```

```
#
snmpTrapdAddr localhost
forward default {{global['snmp_server']}}
```

Generating Event Notification Traps

The Net-SNMP agent provides a method to generate SNMP trap events, via the Distributed Management (DisMan) Event MIB, for various system events, including linkup/down, exceeding the temperature sensor threshold, CPU load, or memory threshold, or other SNMP MIBs.

Enabling MIB to OID Translation

MIB names can be used instead of OIDs, by installing the `snmp-mibs-downloader`, to download SNMP MIBs to the switch prior to enabling traps. This greatly improves the readability of the `snmpd.conf` file.

1. Open `/etc/apt/sources.list` in a text editor.
2. Add the non-free repository, and save the file:

```
cumulus@switch:~$ deb http://ftp.us.debian.org/debian/ jessie main non-free
```

3. Update the switch:

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

4. Install the `snmp-mibs-downloader`:

```
apt-get snmp-mibs-downloader
```

5. Open the `/etc/snmp/snmpd.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. Once the configuration has been confirmed, remove or comment out the `non-free` repository in `/etc/apt/sources.list`.

```
#deb http://ftp.us.debian.org/debian/ jessie main non-free
```

Configuring Trap Events

The following configurations should be made in `/etc/snmp/snmp.conf`, in order to enable specific types of traps. Once configured, restart the `snmpd` service to apply the changes.

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

Defining Access Credentials

An SNMPv3 username is required to authorize the DisMan service. The example code below uses `cumulusUser` as the username.

```
createUser cumulusUser
iquerySecName cumulusUser
rouser cumulusUser
```

Defining Trap Receivers

The example code below creates a trap receiver that is capable of receiving SNMPv2 traps.

```
trap2sink 192.168.1.1 public
```



Although the traps are sent to an SNMPv2 receiver, the SNMPv3 user is still required.



It is possible to define multiple trap receivers, and to use the domain name instead of IP address in the `trap2sink` directive.

Configuring LinkUp/Down Notifications

The `linkUpDownNotifications` directive is used to configure linkup/down notifications when the operational status of the link changes.

```
linkUpDownNotifications yes
```



The default frequency for checking link up/down is 60 seconds. The default frequency can be changed using the `monitor` directive directly instead of the `linkUpDownNotifications` directive. See `man snmpd.conf` for details.

Configuring Temperature Notifications

Temperature sensor information for each available sensor is maintained in the the `lmSensors` MIB. Each platform may contain a different number of temperature sensors. The example below generates a trap event when any temperature sensors exceeds a threshold of 68 degrees (centigrade). It monitors each `lmTempSensorsValue`. When the threshold value is checked and exceeds the `lmTempSensorsValue`, a trap is generated. The `-o lmTempSensorsDevice` option is used to instruct SNMP to also include the `lmTempSensorsDevice` MIB in the generated trap. The default frequency for the `monitor` directive is 600 seconds. The default frequency may be changed using the `-r` option.:

```
monitor lmTempSensor -o lmTempSensorsDevice lmTempSensorsValue > 68000
```

Alternatively, temperature sensors may be monitored individually. To monitor the sensors individually, first use the `sensors` command to determine which sensors are available to be monitored on the platform.

```
#sensors

CY8C3245-i2c-4-2e
Adapter: i2c-0-mux (chan_id 2)
fan5: 7006 RPM (min = 2500 RPM, max = 23000 RPM)
fan6: 6955 RPM (min = 2500 RPM, max = 23000 RPM)
fan7: 6799 RPM (min = 2500 RPM, max = 23000 RPM)
fan8: 6750 RPM (min = 2500 RPM, max = 23000 RPM)
temp1: +34.0 C (high = +68.0 C)
temp2: +28.0 C (high = +68.0 C)
temp3: +33.0 C (high = +68.0 C)
temp4: +31.0 C (high = +68.0 C)
temp5: +23.0 C (high = +68.0 C)
```

Configure a `monitor` command for the specific sensor using the `-I` option. The `-I` option indicates that the monitored expression is applied to a single instance. In this example, there are five temperature sensors available. The following monitor directive can be used to monitor only temperature sensor three at five minute intervals.

```
monitor -I -r 300 lmTemSensor3 -o lmTempSensorsDevice.3 lmTempSensorsValue.  
3 > 68000
```

Configuring 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:

```
monitor MemFreeTotal -o memTotalReal memTotalFree < 1000000
```

Configuring Processor Load Notifications

To monitor CPU load for 1, 5 or 15 minute intervals, use the `load` directive in conjunction with the `monitor` directive. The following example will generate a trap when the 1 minute interval reaches 12%, the 5 minute interval reaches 10% or the 15 minute interval reaches 5%.

```
load 12 10 5  
monitor -r 60 -o laNames -o laErrMsg "laTable" laErrorFlag !=0
```

Configuring Disk Utilization Notifications

To monitor disk utilization for all disks, use the `includeAllDisks` directive in conjunction with the `monitor` directive. The example code below generates a trap when a disk is 99% full:

```
includeAllDisks 1%  
monitor -r 60 -o dskPath -o DiskErrMsg "dskTable" diskErrorFlag !=0
```

Configuring Authentication Notifications

To generate authentication failure traps, use the `authtrappable` directive:

```
authtrappable 1
```

Supported MIBs

Below are the MIBs supported by Cumulus RMP, as well as suggested uses for them. The overall Cumulus RMP MIB is defined in `/usr/share/snmp/Cumulus-Snmp-MIB.txt`.

MIB Name	Suggested Uses
BRIDGE and Q-BRIDGE	The dot1dBasePortEntry and dot1dBasePortIfIndex tables in the BRIDGE-MIB and dot1qBase, dot1qFdbEntry, dot1qTpFdbEntry, dot1qTpFdbStatus, and the dot1qVlanStaticName tables in the Q-BRIDGE-MIB tables. You must uncomment the <code>bridge_pp.py pass_persist</code> script in <code>/etc/snmp/snmpd.conf</code> .
CUMULUS-COUNTERS-MIB	Discard counters: Cumulus RMP also includes its own counters MIB, defined in <code>/usr/share/snmp/Cumulus-Counters-MIB.txt</code> . It has the OID <code>.1.3.6.1.4.1.40310.2</code> .
CUMULUS-RESOURCE-QUERY-MIB	Cumulus RMP includes its own resource utilization MIB, which is similar to using <code>cl-resource-query</code> (see page 166). It monitors L3 entries by host, route, nexthops, ECMP groups and L2 MAC/BDPU entries. The MIB is defined in <code>/usr/share/snmp/Cumulus-Resource-Query-MIB.txt</code> , and has the OID <code>.1.3.6.1.4.1.40310.1</code> .
DISMAN-EVENT	Trap monitoring
ENTITY	From RFC 4133, the temperature sensors, fan sensors, power sensors, and ports are covered.
ENTITY-SENSOR	Physical sensor information (temperature, fan, and power supply) from RFC 3433.
HOST-RESOURCES	Users, storage, interfaces, process info, run parameters
IF-MIB	Interface description, type, MTU, speed, MAC, admin, operation status, counters
IP (includes ICMP)	IPv4, IPv4 addresses, counters, netmasks
IPv6	IPv6 counters
IP-FORWARD	IP routing table
LLDP	L2 neighbor info from <code>lldpd</code> (note, you need to enable the SNMP subagent (see page) in LLDP). <code>lldpd</code> needs to be started with the <code>-x</code> option to enable connectivity to <code>snmpd</code> (AgentX).
LM-SENSORS MIB	Fan speed, temperature sensor values, voltages. This is deprecated since the ENTITY-SENSOR MIB has been added.
NET-SNMP-AGENT	Agent timers, user, group config

MIB Name	Suggested Uses
NET-SNMP-EXTEND	Agent timers, user, group config
NET-SNMP-EXTEND-MIB	(See also this knowledge base article on extending NET-SNMP in Cumulus RMP to include data from power supplies, fans and temperature sensors.)
NET-SNMP-VACM	Agent timers, user, group config
NOTIFICATION-LOG	Local logging
SNMP-FRAMEWORK	Users, access
SNMP-MPD	Users, access
SNMP-TARGET	
SNMP-USER-BASED-SM	Users, access
SNMP-VIEW-BASED-ACM	Users, access
SNMPv2	SNMP counters (For information on exposing CPU and memory information via SNMP, see this knowledge base article .)
TCP	TCP related information
UCD-SNMP	System memory, load, CPU, disk IO
UDP	UDP related information



The ENTITY MIB does not currently show the chassis information in Cumulus RMP 3.0.

Index

A

- access ports [141](#)
- active listener ports [33](#)
- apt-get [46](#)
- arp cache [200](#)
- auto-negotiation [81](#)
- autoprovisioning [52](#)

B

- BFD [116](#), [118](#)
 - Bidirectional Forwarding Detection [116](#)
 - echo function [118](#)
- bonds [123](#)
- boot recovery [165](#)
- bpdufilter [104](#)
 - and STP [104](#)
- BPDU guard [101](#)
 - and STP [101](#)
- brctl [10](#), [91](#), [128](#), [129](#)
 - and STP [91](#)
- bridge assurance [101](#)
 - and STP [101](#)
- bridges [126](#), [127](#), [127](#), [128](#), [129](#), [130](#), [130](#), [133](#), [136](#), [141](#), [141](#), [147](#)
 - access ports [141](#)
 - adding interfaces [128](#), [129](#)
 - adding IP addresses [133](#)
 - MAC addresses [130](#)
 - MTU [130](#)
 - physical interfaces [127](#)
 - trunk ports [141](#)
 - untagged frames [136](#)
 - VLAN-aware [127](#), [147](#)

C

- cable connectivity [8](#)
- cabling [112](#)
 - Prescriptive Topology Manager [112](#)

- cl-acltool 201
- cl-cfg 37
- cl-netstat 195
- cl-resource-query 37, 166
- cl-support 160
- Cumulus Linux 38, 156
 - installing 38
 - reserved VLAN ranges 156
- Cumulus RMP 45
 - upgrading 45
- cumulus user 18

D

- daemons 31
- date 14
 - setting 14
- deb 51
- debugging 159
- decode-syseeprom 168
- dmidecode 169
- dpkg 49
- dpkg-reconfigure 13
- duplex interfaces 79

E

- echo function 118, 118
 - BFD 118
 - PTM 118
- ERSPAN 202
 - network troubleshooting 202
- Ethernet management port 7
- ethtool 84, 194
 - switch ports 84

G

- globs 75
- Graphviz 112

H

- hardware [167](#)
 - monitoring [167](#)
- hash distribution [126](#)
- host entries [166](#)
 - monitoring [166](#)
- hostname [8](#)
- hsflowd [215](#)
- hwclock [15](#)

I

- ifdown [65](#)
- ifquery [70](#), [189](#)
- ifup [65](#)
- ifupdown [64](#)
- ifupdown2 [64](#), [74](#), [139](#), [188](#), [189](#), [189](#)
 - excluding interfaces [189](#)
 - logging [189](#)
 - purging IP addresses [74](#)
 - troubleshooting [188](#)
 - VLAN tagging [139](#)
- interface counters [195](#)
- interface dependencies [69](#)
- interfaces [78](#), [83](#)
 - statistics [83](#)
- IP addresses [74](#)
 - purging [74](#)
- iproute2 [193](#)
 - failures [193](#)

L

- LACP [123](#)
- layer 3 access ports [10](#)
 - configuring [10](#)
- LDAP [27](#)
- link aggregation [123](#)
- Link Layer Discovery Protocol [106](#)
- LLDP [106](#)
- lldpcli [107](#)
- lldpd [106](#), [113](#)

logging [162](#), [189](#), [189](#)
 ifupdown2 [189](#)
 networking service [189](#)
loopback interface [11](#)
 configuring [11](#)
lshw [169](#)

M

MAC entries [166](#)
 monitoring [166](#)
Mako templates [76](#), [191](#)
 debugging [191](#)
monitoring [13](#), [159](#), [166](#), [171](#), [194](#), [214](#), [217](#)
 hardware watchdog [171](#)
 Net-SNMP [217](#)
 network traffic [214](#)
mstpctl [91](#), [143](#)
MTU [81](#), [130](#), [193](#)
 bridges [130](#)
 failures [193](#)
multiple bridges [131](#)
mz [200](#)
 traffic generator [200](#)

N

name switch service [26](#)
Net-SNMP [217](#)
networking service [189](#)
 logging [189](#)
network interfaces [78](#)
network traffic [214](#)
 monitoring [214](#)
NSS [26](#)
 name switch service [26](#)
NTP [15](#)
 time [15](#)
ntpd [15](#)

O

open source contributions [5](#)

P

- packages [46](#)
 - managing [46](#)
- PAM [26](#)
 - pluggable authentication modules [26](#)
- parent interfaces [72](#)
- password [18](#)
 - default [18](#)
- passwordless access [18](#)
- passwords [7](#)
- Per VLAN Spanning Tree [91](#)
 - PVST [91](#)
- ping [199](#)
- pluggable authentication modules [26](#)
- port lists [75](#)
- port speeds [79](#)
- Prescriptive Topology Manager [112](#)
- privileged commands [21](#)
- PTM [112](#), [118](#)
 - echo function [118](#)
 - Prescriptive Topology Manager [112](#)
- ptmctl [120](#)
- ptmd [112](#)
- PTM scripts [114](#), [119](#)
- PVRST [91](#)
 - Rapid PVST [91](#)
- PVST [91](#)
 - Per VLAN Spanning Tree [91](#)

Q

- QSFP [196](#)

R

- Rapid PVST [91](#)
 - PVRST [91](#)
- remote access [17](#)
- repositories [51](#)
 - other packages [51](#)
- reserved VLAN ranges [156](#)

- restart [37](#)
 - switchd [37](#)
- root user [7, 18](#)
- routes [166](#)
 - monitoring [166](#)
- RSTP [91](#)

S

- sensors command [169](#)
- serial console management [7](#)
- services [31](#)
- sFlow [214](#)
- sFlow visualization tools [216](#)
- SFP [84, 196](#)
 - switch ports [84](#)
- single user mode [165](#)
- smonctl [170](#)
- smond [170](#)
- snmpd [217](#)
- sources.list [51](#)
- SPAN [202](#)
 - network troubleshooting [202](#)
- spanning tree parameters [94](#)
- Spanning Tree Protocol [90, 148](#)
 - STP [90](#)
 - VLAN-aware bridges [148](#)
- SSH [17](#)
- SSH keys [17](#)
- static routing [157](#)
 - with ip route [157](#)
- storm control [105](#)
 - STP [105](#)
- STP [90, 101, 105](#)
 - and bridge assurance [101](#)
 - Spanning Tree Protocol [90](#)
 - storm control [105](#)
- sudo [18, 21](#)
- sudoers [21, 21](#)
 - examples [21](#)
- switchd [35, 35, 37](#)
 - configuring [35](#)
 - file system [35](#)
 - restarting [37](#)

- switch ports [9](#)
 - configuring [9](#)
- syslog [162](#)
- system management [159](#)

T

- templates [76](#)
- time [14](#)
 - setting [14](#)
- time zone [8](#), [13](#)
- topology [112](#)
 - data center [112](#)
- traceroute [199](#)
- traffic distribution [126](#)
- traffic generator [200](#)
 - mz [200](#)
- troubleshooting [159](#), [165](#)
 - single user mode [165](#)
- trunk ports [136](#), [141](#)
- tzdata [13](#)

U

- untagged frames [136](#)
 - bridges [136](#)
- user accounts [18](#), [18](#)
 - cumulus [18](#)
 - root [18](#)
- user authentication [26](#)
- user commands [74](#)
 - interfaces [74](#)

V

- visudo [21](#)
- VLAN-aware bridges [127](#), [147](#), [148](#), [148](#)
 - configuring [148](#)
 - Spanning Tree Protocol [148](#)
- VLAN tagging [139](#), [139](#), [141](#)
 - advanced example [141](#)
 - basic example [139](#)
- VLAN translation [146](#)

W

watchdog [171](#)
 monitoring [171](#)

Z

zero touch provisioning [52](#), [54](#)
 USB [54](#)
ZTP [52](#)