Cumulus RMP 3.2.0
User Guide

Table of Contents

Introducing Cumulus RMP

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

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. However, the feature set is customized to the needs of out-of-band management; for a comparison of the features supported in Cumulus RMP, see the Cumulus RMP Features page.

This documentation is current as of December 19, 2016 for version 3.2.0. 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.

What's New in Cumulus Linux 3.2.0

Cumulus Linux 3.2.0 adds a number of new features and platforms to the existing Cumulus Linux 3.x release, including:

- Snapshots
- Full support for previously early access features:
    - Network Command Line Utility (NCLU)
    - Protocol Independent Multicast - Sparse Mode (PIM-SM)
    - TACACS+

Open Source Contributions

Cumulus Networks has forked various software projects, like CFEngine, `Netdev` and some Puppet Labs packages in order to implement various Cumulus Linux 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.

Quick Start Guide

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

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, refer to this reference guide for examples of the Cumulus Linux 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.

Contents

This chapter covers ...

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 are running a Cumulus RMP version earlier than 3.0.0, you must perform a complete install (see page 51). If you already have Cumulus Linux 3.0.0 or later installed on your switch, read Upgrading Cumulus RMP (see page 51) for considerations before start the process.

Getting Started

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 24).

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 with the Network Command Line Utility (NCLU).

Example IP Configuration

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

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

These commands produce the following snippet 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, run `net add hostname`, which modifies both the `/etc/hostname` and `/etc/hosts` files with the desired hostname.

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

⚠ The command prompt in the terminal doesn't reflect the new hostname until you either log out of the switch or start a new shell.

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

Testing Cable Connectivity

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

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

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

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

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

```
cumulus@switch:~$ net show interface all
       Name                       Speed    MTU    Mode
Summary
-----  ------------------------   -------  -----  -------------
--------------------------------------
UP     lo                         N/A      65536  Loopback       IP:
10.0.0.11/32, 127.0.0.1/8, ::1/128
UP     eth0                       1G       1500   Mgmt           IP:
192.168.0.11/24(DHCP)
UP     swp1 (hypervisor_port_1)   1G       1500   Access/L2
Untagged: br0
UP     swp2                       1G       1500   NotConfigured
ADMDN  swp45                      0M       1500   NotConfigured
ADMDN  swp46                      0M       1500   NotConfigured
ADMDN  swp47                      0M       1500   NotConfigured
ADMDN  swp48                      0M       1500   NotConfigured
ADMDN  swp49                      0M       1500   NotConfigured
ADMDN  swp50                      0M       1500   NotConfigured
UP     swp51                      1G       1500   BondMember
Master: bond0(DN)
UP     blue                       N/A      65536  NotConfigured
DN     bond0                      N/A      1500   Bond           Bond
Members: swp51(UN)
UP     br0                        N/A      1500   Bridge/L3      IP:
172.16.1.1/24

Untagged Members: swp1
                                                                 802.1
q Tag: Untagged
                                                                 STP:
RootSwitch(32768)
UP     red                        N/A      65536  NotConfigured
ADMDN  rename13                   0M       1500   NotConfigured
ADMDN  vagrant                    0M       1500   NotConfigured
```

Configuring Switch Ports

Layer 2 Port Configuration

Cumulus RMP does not put all ports into a bridge by default. To configure a front panel port or create a bridge, edit the `/etc/network/interfaces` file. After saving the file, to activate the change, use the `ifup` command.

Examples

Example One

In the following configuration example, the front panel port swp1 is placed into a bridge called `bridge`. The NCLU commands are:

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

The commands above produce the following `/etc/network/interfaces` snippet:

```
auto bridge
iface bridge
    bridge-ports swp1
    bridge-stp on
    bridge-vlan-aware yes
```

Example Two

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

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

This creates the following `/etc/network/interfaces` snippet:

```
auto bridge
iface bridge
      bridge-ports swp1 swp2 swp3 swp4 swp5 swp6 swp7 swp8
swp9 swp10 swp12 swp14 swp15 swp16 swp17 swp18 swp19 swp20
      bridge-stp on
    bridge-vlan-aware yes
```

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:

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

This creates the following `/etc/network/interfaces` snippet:

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

To add an IP address to a bridge interface, run:

```
cumulus@switch:~$ net add bridge address 10.2.2.1/24
cumulus@switch:~$ net add bridge bridge-ports swp1-10,12,14-20
cumulus@switch:~$ net add bridge bridge-stp on
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

This creates the following `/etc/network/interfaces snippet`:

```
auto bridge
iface bridge
      address 10.2.2.1/24
    bridge-ports swp1 swp2 swp3 swp4 swp5 swp6 swp7 swp8 swp9 swp10
swp12 swp14 swp15 swp16 swp17 swp18 swp19 swp20
     bridge-stp on
    bridge-vlan-aware yes
```

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

```
bridge: <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 bridge

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.

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

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

```
cumulus@switch:~$ net show interface lo
    Name    MAC                Speed      MTU   Mode

--  ------  -----------------  -------  -----  --------

UP  lo      00:00:00:00:00:00  N/A      65536  Loopback

IP Details

-------------------------  --------------------

IP:                        127.0.0.1/8, ::1/128

IP Neighbor(ARP) Entries:  0
```

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, configure the `lo` interface with NCLU:

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

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

```
cumulus@switch:~$ net add interface lo ip address 172.16.2.1/24
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

This creates the following snippet in `/etc/network/interfaces`:

```
auto lo
iface lo inet loopback
    address 10.1.1.1/32
    address 172.16.2.1/24
```

System Configuration

Setting Date and Time

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

Contents

This chapter covers ...

Setting the Time Zone

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

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

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

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

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

Alternative: Use the Guided Wizard to Find and Apply a Time Zone

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 RMP 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:

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

# Specify interfaces
interface listen bridge10
```

Related Information

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

Authentication, Authorization, and Accounting

SSH for Remote Access

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

⚠ By default, you cannot use the root account to SSH to a Cumulus Linux switch unless you generate an SSH key (see page 23) or set a password (see page 23) for the account.

Contents

This chapter covers ...

Generate an SSH Key Pair

1. Run the `ssh-keygen` command, and follow the prompts, to generate the key pair:

    Configure a Passwordless System

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

```
cumulus@leaf01:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/cumulus/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/cumulus/.ssh/id_rsa.
Your public key has been saved in /home/cumulus/.ssh/id_rsa.pub.
The key fingerprint is:
5a:b4:16:a0:f9:14:6b:51:f6:f6:c0:76:1a:35:2b:bb cumulus@leaf04
The key's randomart image is:
+---[RSA 2048]----+
|      +.o   o    |
|     o * o . o   |
|    o + o O o    |
|     + . = O     |
|      . S o .    |
|       +   .     |
|      .   E      |
|                 |
|                 |
+-----------------+
```

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

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

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

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

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

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

```
cumulus@leaf04:~$ ssh cumulus@leaf02

Welcome to Cumulus VX (TM)

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

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

Related Information

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

User Accounts

By default, Cumulus 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 19)
- Along with the cumulus group, has both show and edit rights for NCLU

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 24), 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 189).

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:

   ```
   root@host:~# 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:

   ```
   root@host:~# 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 nano /etc/ssh/sshd_config

...

# Authentication:
LoginGraceTime 120
PermitRootLogin yes
StrictModes yes

...
```

3. Restart the `ssh` service:

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

Using sudo to Delegate Privileges

By default, Cumulus 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

This chapter covers ...

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	`ethtool -m swp1`	`%noc ALL=(ALL) NOPASSWD: /sbin/ethtool`

Category	Privilege	Example Command	sudoers Entry
Monitoring	System diagnostics	`cl-support`	`%noc ALL=(ALL) NOPASSWD:/usr/cumulus/bin/cl-support`
Monitoring	Routing diagnostics	`cl-resource-query`	`%noc ALL=(ALL) NOPASSWD:/usr/cumulus/bin/cl-resource-query`
Image management	Install images	`onie-select http://lab/install.bin`	`%noc ALL=(ALL) NOPASSWD:/usr/cumulus/bin/cl-img-install`
Package management	Any apt-get command	`apt-get update or apt-get install`	`%noc ALL=(ALL) NOPASSWD:/usr/bin/apt-get`
Package management	Just apt-get update	`apt-get update`	`%noc ALL=(ALL) NOPASSWD:/usr/bin/apt-get update`
Package management	Install packages	`apt-get install mtr-tiny`	`%noc ALL=(ALL) NOPASSWD:/usr/bin/apt-get install *`

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

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

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

Category	Privilege	Example Command	sudoers Entry
L3	Any iproute command	`ip *`	`%noc ALL=(ALL) NOPASSWD:/bin/ip`

Related Information

- sudo
- Adding Yourself to sudoers

LDAP Authentication and Authorization

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

NSS specifies the order of information sources used to resolve names for each service. Using this with authentication and authorization, it provides the order and location used for user lookup and group mapping on the system. PAM handles the interaction between the user and the system, providing login handling, session setup, authentication of users and authorization of a user actions.

NSS enables PAM to use LDAP for providing user authentication, group mapping and information for other services on the system.

Contents

This chapter covers ...

Configuring LDAP Authentication

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 `libpam-ldapd` package depends on `nslcd`, so to install `libnss-ldapd`, `libpam-ldapd` and `ldap-utils`, you must run:

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

This brings up an interactive prompt asking questions about the LDAP URI, search base distinguished name (DN) and services that should have LDAP lookups enabled. This creates a very basic LDAP configuration, using anonymous bind, and initiating the search for a user under the base DN specified.

Alternatively, these parameters can be pre-seeded using the `debconf-utils`. To use this method, run `apt-get install debconf-utils` and create the pre-seeded parameters using `debconf-set-selections` with the appropriate answers. Run `debconf-show <pkg>` to check the settings. Here is an example of how to preseed answers to the installer questions using `debconf-set-selections`.

Once the install is complete, the *name service LDAP caching daemon* (`nslcd`) will be running. This is the service that handles all of the LDAP protocol interactions, and caches the information returned from the LDAP server. In `/etc/nsswitch.conf`, `ldap` has been appended and is the secondary information source for *passwd*, *group* and *shadow*. The local files (`/etc/passwd`, `/etc/groups` and `/etc/shadow`) are used first, as specified by the `compat` source.

```
passwd: compat ldap
group: compat ldap
shadow: compat ldap
```

You are strongly advised to keep `compat` as the first source in NSS for *passwd*, *group* and *shadow*. This prevents you from getting locked out of the system.

Configuring nslcd.conf

You need to update the main configuration file (`/etc/nslcd.conf`) after installation to accommodate the expected LDAP server settings. The nslcd.conf man page details all the available configuration options. Some of the more important options are related to security and how the queries are handled.

Connection

The LDAP client starts a session by connecting to the LDAP server, by default, on TCP and UDP port 389, or on port 636 for LDAPS. Depending on the configuration, this connection may be unauthenticated (anonymous bind); otherwise, the client must provide a bind user and password. The variables used to define the connection to the LDAP server are the URI and bind credentials.

The URI is mandatory, and specifies the LDAP server location using the FQDN or IP address. It also designates whether to use ldap:// for clear text transport, or ldaps:// for SSL/TLS encrypted transport. Optionally, an alternate port may also be specified in the URI. Typically, in production environments, it is best to utilize the LDAPS protocol. Otherwise all communications are clear text and not secure.

After the connection to the server is complete, the BIND operation authenticates the session. The BIND credentials are optional, and if not specified, an anonymous bind is assumed. This is typically not allowed in most production environments. Configure authenticated (Simple) BIND by specifying the user (*binddn*) and password (*bindpw*) in the configuration. Another option is to use SASL (Simple Authentication and Security Layer) BIND, which provides authentication services using other mechanisms, like Kerberos. Contact your LDAP server administrator for this information since it depends on the configuration of the LDAP server and what credentials are created for the client device.

```
# The location at which the LDAP server(s) should be reachable.
uri ldaps://ldap.example.com
# The DN to bind with for normal lookups.
binddn cn=CLswitch,ou=infra,dc=example,dc=com
bindpw CuMuLuS
```

Search Function

When an LDAP client requests information about a resource, it must connect and bind to the server. Then it performs one or more resource queries depending on what it is looking up. All search queries sent to the LDAP server are created using the configured search *base*, *filter*, and the desired entry (*uid=myuser*) being searched for. If the LDAP directory is large, this search may take a significant amount of time. It is a good idea to define a more specific search base for the common *maps* (*passwd* and *group*).

```
# The search base that will be used for all queries.
base dc=example,dc=com
# Mapped search bases to speed up common queries.
base passwd ou=people,dc=example,dc=com
base group ou=groups,dc=example,dc=com
```

Search Filters

It is also common to use search filters to specify criteria used when searching for objects within the directory. This is used to limit the search scope when authenticating users. The default filters applied are:

```
filter passwd (objectClass=posixAccount)
filter group (objectClass=posixGroup)
```

Attribute Mapping

The *map* configuration allows for overriding the attributes pushed from LDAP. To override an attribute for a given *map**, specify the attribute name and the new value. One example of how this is useful is ensuring the shell is *bash* and the home directory is `/home/cumulus`:

```
map    passwd homeDirectory "/home/cumulus"
map    passwd shell "/bin/bash"
```

*In LDAP, the ***map*** refers to one of the supported maps specified in the manpage for `nslcd.conf` (such as *passwd* or *group*).

Example Configuration

Here is an example configuration using Cumulus RMP.

Troubleshooting

Using nslcd Debug Mode

When setting up LDAP authentication for the first time, Cumulus Networks recommends you turn off this service using `systemctl stop nslcd` 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
```

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

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

If LDAP is configured correctly, the following messages appear after you run the `getent` command:

```
nslcd: DEBUG: accept() failed (ignored): Resource temporarily
unavailable
nslcd: [8e1f29] DEBUG: connection from pid=11766 uid=0 gid=0
nslcd: [8e1f29] <passwd(all)> DEBUG: myldap_search(base="dc=example,
dc=com", filter="(objectClass=posixAccount)")
nslcd: [8e1f29] <passwd(all)> DEBUG: ldap_result(): uid=myuser,
ou=people,dc=example,dc=com
nslcd: [8e1f29] <passwd(all)> DEBUG: ldap_result(): ... 152 more
results
nslcd: [8e1f29] <passwd(all)> DEBUG: ldap_result(): end of results (16
2 total)
```

In the output above, *<passwd(all)>* indicates that the entire directory structure was queried.

A specific user can be queried using the command:

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

You can replace *myuser* with any username on the switch. The following debug output indicates that user *myuser* exists:

```
nslcd: DEBUG: add_uri(ldap://10.50.21.101)
nslcd: version 0.8.10 starting
nslcd: DEBUG: unlink() of /var/run/nslcd/socket failed (ignored): No
such file or directory
nslcd: DEBUG: setgroups(0,NULL) done
nslcd: DEBUG: setgid(110) done
nslcd: DEBUG: setuid(107) done
nslcd: accepting connections
nslcd: DEBUG: accept() failed (ignored): Resource temporarily
unavailable
nslcd: [8b4567] DEBUG: connection from pid=11369 uid=0 gid=0
nslcd: [8b4567] <passwd="cumulus"> DEBUG: myldap_search(base="dc=cumul
usnetworks,dc=com", filter="(&(objectClass=posixAccount)
(uid=cumulus))")
nslcd: [8b4567] <passwd="myuser"> DEBUG: ldap_initialize(ldap://<ip_ad
dress>)
nslcd: [8b4567] <passwd="myuser"> DEBUG: ldap_set_rebind_proc()
nslcd: [8b4567] <passwd="myuser"> DEBUG: ldap_set_option
(LDAP_OPT_PROTOCOL_VERSION,3)
nslcd: [8b4567] <passwd="myuser"> DEBUG: ldap_set_option
(LDAP_OPT_DEREF,0)
nslcd: [8b4567] <passwd="myuser"> DEBUG: ldap_set_option
(LDAP_OPT_TIMELIMIT,0)
nslcd: [8b4567] <passwd="myuser"> DEBUG: ldap_set_option
(LDAP_OPT_TIMEOUT,0)
nslcd: [8b4567] <passwd="myuser"> DEBUG: ldap_set_option
(LDAP_OPT_NETWORK_TIMEOUT,0)
nslcd: [8b4567] <passwd="myuser"> DEBUG: ldap_set_option
(LDAP_OPT_REFERRALS,LDAP_OPT_ON)
nslcd: [8b4567] <passwd="myuser"> DEBUG: ldap_set_option
(LDAP_OPT_RESTART,LDAP_OPT_ON)
nslcd: [8b4567] <passwd="myuser"> DEBUG: ldap_simple_bind_s(NULL,
NULL) (uri="ldap://<ip_address>")
nslcd: [8b4567] <passwd="myuser"> DEBUG: ldap_result(): end of
results (0 total)
```

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

Common Problems

SSL/TLS

- The FQDN of the LDAP server URI does not match the FQDN in the CA-signed server certificate exactly.
- `nslcd` cannot read the SSL certificate, and will report a "Permission denied" error in the debug during server connection negotiation. Check the permission on each directory in the path of the root SSL certificate. Ensure that it is readable by the `nslcd` user.

NSCD

- If the `nscd cache` daemon is also enabled and you make some changes to the user from LDAP, you may want to clear the cache using the commands:

  ```
  nscd --invalidate = passwd
  nscd --invalidate = group
  ```

- The `nscd` package works with `nslcd` to cache name entries returned from the LDAP server. This may cause authentication failures. To work around these issues:
  1. Disable `nscd` by running:

     ```
     cumulus@switch:~$ sudo nscd -K
     ```

  2. Restart the `nslcd` service:

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

  3. Try the authentication again.

LDAP

- The search filter returns wrong results. Check for typos in the search filter. Use `ldapsearch` to test your filter.
- Optionally, configure the basic LDAP connection and search parameters in `/etc/ldap/ldap.conf`
  .

  ```
  # ldapsearch -D 'cn=CLadmin' -w 'CuMuLuS' "(&
  (ObjectClass=inetOrgUser)(uid=myuser))"
  ```

- When a local username also exists in the LDAP database, the order of the information sources in `/etc/nsswitch` can be updated to query LDAP before the local user database. This is generally not recommended. For example, the configuration below ensures that LDAP is queried before the local database.

  ```
  # /etc/nsswitch.conf
  passwd:         ldap compat
  ```

Configuring LDAP Authorization

Linux uses the *sudo* command to allow non-administrator users — like the default *cumulus* user account — to perform privileged operations. To control the users authorized to use sudo, the `/etc/sudoers` file and files located in the `/etc/sudoers.d/` directory have a series of rules defined. Typically, the rules are based on groups, but can also be defined for specific users. Therefore, sudo rules can be added using the group names from LDAP. For example, if a group of users were associated with the group *netadmin*, a rule can be added to give those users sudo privileges. Refer to the sudoers manual (`man sudoers`) for a complete usage description. Here's an illustration of this in `/etc/sudoers`:

```
# The basic structure of a user specification is “who where =
(as_whom) what”.
%sudo ALL=(ALL:ALL) ALL
%netadmin ALL=(ALL:ALL) ALL
```

Active Directory Configuration

Active Directory (AD) is a fully featured LDAP-based NIS server created by Microsoft. It offers unique features that classic OpenLDAP servers lack. Therefore, it can be more complicated to configure on the client and each version of AD is a little different in how it works with Linux-based LDAP clients. Some more advanced configuration examples, from testing LDAP clients on Cumulus RMP with Active Directory (AD /LDAP), are available in our knowledge base.

LDAP Verification Tools

Typically, password and group information is retrieved from LDAP and cached by the LDAP client daemon. To test the LDAP interaction, these command line tools can be used to trigger an LDAP query from the device. This helps to create the best filters and verify the information sent back from the LDAP server.

Identifying a User with the id Command

The `id` command performs a username lookup by following the lookup information sources in NSS for the *passwd* service. This simply returns the user ID, group ID and the group list retrieved from the information source. In the following example, the user *cumulus* is locally defined in `/etc/passwd`, and *myuser* is on LDAP. The NSS configuration has the passwd map configured with the sources `compat ldap`:

```
cumulus@switch:~$ id cumulus
uid=1000(cumulus) gid=1000(cumulus) groups=1000(cumulus),4(adm),27(sud
o)
cumulus@switch:~$ id myuser
uid=1230(myuser) gid=3000(Development) groups=3000(Development),500(Em
ployees),27(sudo)
```

Using getent

The `getent` command retrieves all records found via NSS for a given map. It can also get a specific entry under that map. Tests can be done with the passwd, group, shadow or any other map configured in `/etc /nsswitch.conf`. The output from this command is formatted according to the map requested. Thus, for the passwd service, the structure of the output is the same as the entries in `/etc/passwd`. The same can be said for the group map will output the same as `/etc/group`. In this example, looking up a specific user in the passwd map, the user *cumulus* is locally defined in `/etc/passwd`, and *myuser* is only in LDAP.

```
cumulus@switch:~$ getent passwd cumulus
cumulus:x:1000:1000::/home/cumulus:/bin/bash
cumulus@switch:~$ getent passwd myuser
myuser:x:1230:3000:My Test User:/home/myuser:/bin/bash
```

In the next example, looking up a specific group in the group service, the group *cumulus* is locally defined in `/etc/groups`, and *netadmin* is on LDAP.

```
cumulus@switch:~$ getent group cumulus
cumulus:x:1000:
cumulus@switch:~$ getent group netadmin
netadmin:*:502:larry,curly,moe,shemp
```

Running the command `getent passwd` or `getent group` without a specific request, returns **all** local and LDAP entries for the *passwd* and *group* maps, respectively.

Using LDAP Search

The `ldapsearch` command performs LDAP operations directly on the LDAP server. This does not interact with NSS. This command helps display what the LDAP daemon process is receiving back from the server. The command has many options. The simplest uses anonymous bind to the host and specifies the search DN and what attribute to lookup.

```
cumulus@switch:~$ ldapsearch -H ldap://ldap.example.com -b dc=example,
dc=com -x uid=myuser
```

Click here to expand output of command

```
# extended LDIF
#
# LDAPv3
# base <dc=example,dc=com> with scope subtree
# filter: uid=myuser
# requesting: ALL
#
# myuser, people, example.com
dn: uid=myuser,ou=people,dc=example,dc=com
cn: My User
displayName: My User
gecos: myuser
gidNumber: 3000
givenName: My
homeDirectory: /home/myuser
initials: MU
loginShell: /bin/bash
mail: myuser@example.com
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
objectClass: top
shadowExpire: -1
shadowFlag: 0
shadowMax: 999999
shadowMin: 8
shadowWarning: 7
sn: User
uid: myuser
uidNumber: 1234
# search result
search: 2
result: 0 Success
# numResponses: 2
# numEntries: 1
```

LDAP Browsers

There are some GUI LDAP clients that help to work with LDAP servers. These are free tools to help graphically show the structure of the LDAP database.

- Apache Directory Studio
- LDAPManager

Related Information

- Debian - configuring LDAP authentication
- Debian - configuring PAM to use LDAP

- GitHub - Arthur de Jong nslcd.conf file
- Debian backports

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

Contents

This chapter covers ...

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.

```
cumulus@switch:~$ sudo systemctl start | stop | restart | status |
reload | enable | disable | reenable SERVICENAME.service
```

For example to restart networking, run the command:

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

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 quagga.service` 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.

Keeping systemd Services from Hanging after Starting

If you start, restart or reload any `systemd` service that could be started from another `systemd` service, you must use the `--no-block` option with `systemctl`. Otherwise, that service or even the switch itself may hang after starting or restarting.

Identifying Active Listener Ports for IPv4 and IPv6

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

```
cumulus@switch:~$ sudo netstat -nlp --inet --inet6
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address
State       PID/Program name
tcp        0      0 0.0.0.0:53              0.0.0.0:*
LISTEN      444/dnsmasq
tcp        0      0 0.0.0.0:22              0.0.0.0:*
LISTEN      874/sshd
tcp6       0      0 :::53                   :::*
LISTEN      444/dnsmasq
tcp6       0      0 :::22                   :::*
LISTEN      874/sshd
udp        0      0 0.0.0.0:28450           0.0.0.0:
*                           839/dhclient
udp        0      0 0.0.0.0:53              0.0.0.0:
*                           444/dnsmasq
udp        0      0 0.0.0.0:68              0.0.0.0:
*                           839/dhclient
udp        0      0 192.168.0.42:123        0.0.0.0:
*                           907/ntpd
udp        0      0 127.0.0.1:123           0.0.0.0:
*                           907/ntpd
udp        0      0 0.0.0.0:123             0.0.0.0:
*                           907/ntpd
udp        0      0 0.0.0.0:4784            0.0.0.0:
*                           909/ptmd
udp        0      0 0.0.0.0:3784            0.0.0.0:
*                           909/ptmd
udp        0      0 0.0.0.0:3785            0.0.0.0:
*                           909/ptmd
udp6       0      0 :::58352                :::
*                                839/dhclient
udp6       0      0 :::53                   :::
*                                444/dnsmasq
udp6       0      0 fe80::a200:ff:fe00::123 :::
*                                907/ntpd
udp6       0      0 ::1:123                 :::
*                                907/ntpd
udp6       0      0 :::123                  :::
*                                907/ntpd
udp6       0      0 :::4784                 :::
*                                909/ptmd
udp6       0      0 :::3784                 :::
*                                909/ptmd
```

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                    inactive
Service portwd                  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
```

You can also run `systemctl list-unit-files --type service` to list all services on the switch and see which ones are enabled:

Click here to see output of this command ...

```
cumulus@switch:~$ systemctl list-unit-files --type service
UNIT FILE                                     STATE
aclinit.service                               enabled
acltool.service                               enabled
acpid.service                                 disabled
arp_refresh.service                           enabled
auditd.service                                enabled
autovt@.service                               disabled
bootlog.service                               enabled
bootlogd.service                              masked
bootlogs.service                              masked
bootmisc.service                              masked
checkfs.service                               masked
checkroot-bootclean.service                   masked
checkroot.service                             masked
clagd.service                                 enabled
```

```
clcmd.service                              enabled
console-getty.service                      disabled
console-shell.service                      disabled
container-getty@.service                   static
cron.service                               enabled
cryptdisks-early.service                   masked
cryptdisks.service                         masked
cumulus-aclcheck.service                   static
cumulus-core.service                       static
cumulus-fastfailover.service               enabled
cumulus-firstboot.service                  disabled
cumulus-platform.service                   enabled
cumulus-support.service                    static
dbus-org.freedesktop.hostname1.service     static
dbus-org.freedesktop.locale1.service       static
dbus-org.freedesktop.login1.service        static
dbus-org.freedesktop.machine1.service      static
dbus-org.freedesktop.timedate1.service     static
dbus.service                               static
debian-fixup.service                       static
debug-shell.service                        disabled
decode-syseeprom.service                   static
dhcpd.service                              disabled
dhcpd6.service                             disabled
dhcpd6@.service                            disabled
dhcpd@.service                             disabled
dhcrelay.service                           enabled
dhcrelay6.service                          disabled
dhcrelay6@.service                         disabled
dhcrelay@.service                          disabled
dm-event.service                           disabled
dns-watcher.service                        disabled
dnsmasq.service                            enabled
emergency.service                          static
fuse.service                               masked
getty-static.service                       static
getty@.service                             enabled
halt-local.service                         static
halt.service                               masked
heartbeat-failed@.service                  static
hostname.service                           masked
hsflowd.service                            enabled
hsflowd@.service                           enabled
hwclock-save.service                       enabled
hwclock.service                            masked
hwclockfirst.service                       masked
ifup@.service                              static
initrd-cleanup.service                     static
initrd-parse-etc.service                   static
initrd-switch-root.service                 static
initrd-udevadm-cleanup-db.service          static
killprocs.service                          masked
```

```
kmod-static-nodes.service                  static
kmod.service                               static
ledmgrd.service                            enabled
lldpd.service                              enabled
lm-sensors.service                         enabled
lvm2-activation-early.service              enabled
lvm2-activation.service                    enabled
lvm2-lvmetad.service                       static
lvm2-monitor.service                       enabled
lvm2-pvscan@.service                       static
lvm2.service                               disabled
module-init-tools.service                  static
motd.service                               masked
mountall-bootclean.service                 masked
mountall.service                           masked
mountdevsubfs.service                      masked
mountkernfs.service                        masked
mountnfs-bootclean.service                 masked
mountnfs.service                           masked
mstpd.service                              enabled
netd.service                               enabled
netq-agent.service                         disabled
networking.service                         enabled
ntp.service                                enabled
ntp@.service                               disabled
openvswitch-vtep.service                   disabled
phy-ucode-update.service                   enabled
portwd.service                             enabled
procps.service                             static
ptmd.service                               enabled
pwmd.service                               enabled
quagga.service                             enabled
quotaon.service                            static
rc-local.service                           static
rc.local.service                           static
rdnbrd.service                             disabled
reboot.service                             masked
rescue.service                             static
rmnologin.service                          masked
rsyslog.service                            enabled
screen-cleanup.service                     masked
sendsigs.service                           masked
serial-getty@.service                      disabled
single.service                             masked
smond.service                              enabled
snmpd.service                              disabled
snmpd@.service                             disabled
snmptrapd.service                          disabled
snmptrapd@.service                         disabled
ssh.service                                enabled
ssh@.service                               disabled
sshd.service                               enabled
```

```
stop-bootlogd-single.service               masked
stop-bootlogd.service                      masked
stopssh.service                            enabled
sudo.service                               disabled
switchd-diag.service                       static
switchd.service                            enabled
syslog.service                             enabled
sysmonitor.service                         static
systemd-ask-password-console.service       static
systemd-ask-password-wall.service          static
systemd-backlight@.service                 static
systemd-binfmt.service                     static
systemd-fsck-root.service                  static
systemd-fsck@.service                      static
systemd-halt.service                       static
systemd-hibernate.service                  static
systemd-hostnamed.service                  static
systemd-hybrid-sleep.service               static
systemd-initctl.service                    static
systemd-journal-flush.service              static
systemd-journald.service                   static
systemd-kexec.service                      static
systemd-localed.service                    static
systemd-logind.service                     static
systemd-machined.service                   static
systemd-modules-load.service               static
systemd-networkd-wait-online.service       disabled
systemd-networkd.service                   disabled
systemd-nspawn@.service                    disabled
systemd-poweroff.service                   static
systemd-quotacheck.service                 static
systemd-random-seed.service                static
systemd-readahead-collect.service          disabled
systemd-readahead-done.service             static
systemd-readahead-drop.service             disabled
systemd-readahead-replay.service           disabled
systemd-reboot.service                     static
systemd-remount-fs.service                 static
systemd-resolved.service                   disabled
systemd-rfkill@.service                    static
systemd-setup-dgram-qlen.service           static
systemd-shutdownd.service                  static
systemd-suspend.service                    static
systemd-sysctl.service                     static
systemd-timedated.service                  static
systemd-timesyncd.service                  disabled
systemd-tmpfiles-clean.service             static
systemd-tmpfiles-setup-dev.service         static
systemd-tmpfiles-setup.service             static
systemd-udev-settle.service                static
systemd-udev-trigger.service               static
systemd-udevd.service                      static
```

```
systemd-update-utmp-runlevel.service   static
systemd-update-utmp.service            static
systemd-user-sessions.service          static
udev-finish.service                    static
udev.service                           static
umountfs.service                       masked
umountnfs.service                      masked
umountroot.service                     masked
update-ports.service                   enabled
urandom.service                        static
user@.service                          static
uuidd.service                          static
vboxadd-service.service                enabled
vboxadd-x11.service                    enabled
vboxadd.service                        enabled
vxrd.service                           disabled
vxsnd.service                          disabled
wd_keepalive.service                   enabled
x11-common.service                     masked
ztp-init.service                       enabled
ztp.service                            disabled
191 unit files listed.
lines 147-194/194 (END)
```

Identifying Essential Services

If you need to know which services are required to run when the switch boots, run:

```
cumulus@switch:~$ sudo systemctl list-dependencies --before basic.
target
```

To see which services are needed for networking, run:

```
cumulus@switch:~$ sudo systemctl list-dependencies --after network.
target
network.target
```

```
networking.service
switchd.service
wd_keepalive.service
network-pre.target
```

To identify the services needed for a multi-user environment, run:

```
cumulus@leaf01:~$ sudo systemctl list-dependencies --before multi-user.
target
multi-user.target
```

```
bootlog.service
systemd-readahead-done.service
systemd-readahead-done.timer
systemd-update-utmp-runlevel.service
graphical.target
systemd-update-utmp-runlevel.service
```

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

This chapter covers ...

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

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

```
cumulus@switch:~$ 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.

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

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 52)
  - DHCP/a web server without DHCP options (see page 53)
  - A web server with no DHCP (see page 53)
  - FTP or TFTP without a web server (see page 54)
  - Local file installation (see page 54)
  - USB (see page 54)

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.

Contents

This chapter covers ...

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.

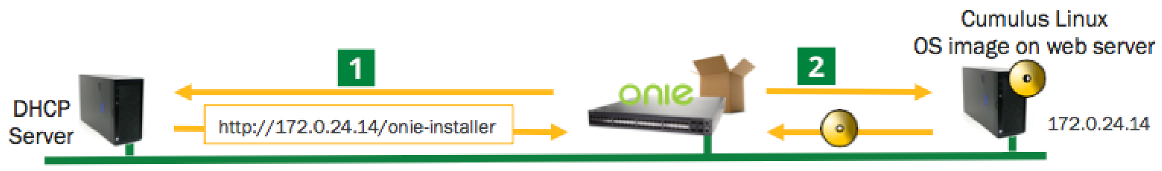
You can name your Cumulus RMP installer binary using any of the ONIE naming schemes mentioned here.

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.


DHCP Server
1
http://172.0.24.14/onie-installer
onie
2
Cumulus Linux OS image on web server
172.0.24.14

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.msdos -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`.

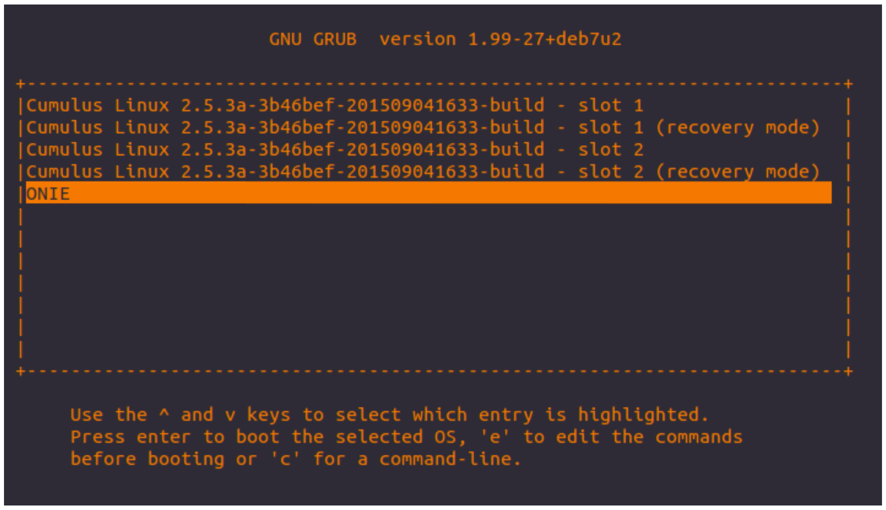
   ⚠ You can also use any of the ONIE naming schemes mentioned here.

   ⓘ 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.

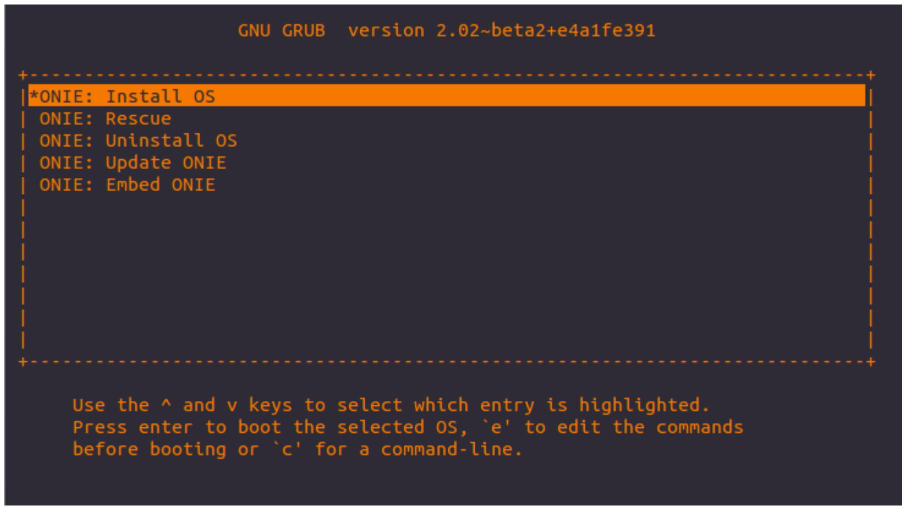


```
                    GNU GRUB  version 1.99-27+deb7u2

+--------------------------------------------------------------------------+
|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.

Related Information

- Open Network Install Environment (ONIE) Home Page

Using Snapshots

Cumulus RMP supports the ability to take snapshots of the complete file system as well as the ability to roll back to a previous snapshot. Snapshots are performed automatically right before and after you upgrade Cumulus RMP and right before and after you commit a switch configuration using NCLU. In addition, you can take a snapshot at any time. You can roll back the entire file system to a specific snapshot or just retrieve specific files.

The primary snapshot components are:

- btrfs — an underlying file system in Cumulus RMP, which supports snapshots.
- snapper — a userspace utility to create and manage snapshots on demand as well as taking snapshots automatically before and after running `apt-get upgrade|install|remove|dist-upgrade`. You can use `snapper` to roll back to earlier snapshots, view existing snapshots, or delete one or more snapshots.
- NCLU — takes snapshots automatically before and after committing network configurations. You can use NCLU to roll back to earlier snapshots, view existing snapshots, or delete one or more snapshots.

Installing the Snapshot Package

If you're upgrading from a version of Cumulus RMP earlier than version 3.2, you need to install the `cumulus-snapshot` package before you can use snapshots.

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

Taking and Managing Snapshots

As described above, snapshots are taken automatically:

- Before and after you update your switch configuration by running `net commit`, via NCLU.
- Before and after you update Cumulus RMP by running `apt-get upgrade|install|remove|dist-upgrade`, via `snapper`.

You can also take snapshots as needed using the `snapper` utility. Run:

```
cumulus@switch:~$ sudo snapper create -d SNAPSHOT_NAME
```

For more information about using `snapper`, run `snapper --help` or `man snapper(8)`.

Viewing Available Snapshots

You can use both NCLU and `snapper` to view available snapshots on the switch.

```
cumulus@switch:~$ net show commit history
  #  Date                             Description
---  -------------------------------
-------------------------------------
 20  Thu 01 Dec 2016 01:43:29 AM UTC  nclu pre  'net commit' (user
cumulus)
 21  Thu 01 Dec 2016 01:43:31 AM UTC  nclu post 'net commit' (user
cumulus)
 22  Thu 01 Dec 2016 01:44:18 AM UTC  nclu pre  '20 rollback' (user
cumulus)
```

```
 23  Thu 01 Dec 2016 01:44:18 AM UTC  nclu post '20 rollback' (user
cumulus)
 24  Thu 01 Dec 2016 01:44:22 AM UTC  nclu pre  '22 rollback' (user
cumulus)
 31  Fri 02 Dec 2016 12:18:08 AM UTC  nclu pre  'ACL' (user cumulus)
 32  Fri 02 Dec 2016 12:18:10 AM UTC  nclu post 'ACL' (user cumulus)
```

However, `net show commit history` only displays snapshots taken when you update your switch configuration. It does not list any snapshots taken directly with `snapper`. To see all the snapshots on the switch, run:

```
cumulus@switch:~$ sudo snapper list
Type   | #  | Pre # | Date                             | User |
Cleanup | Description                               | Userdata
-------+----+-------+----------------------------------+------
+---------+-----------------------------------------+-------------
single | 0  |       |                                  | root
|         | current                                 |
single | 1  |       | Sat 24 Sep 2016 01:45:36 AM UTC | root
|         | first root filesystem                   |
pre    | 20 |       | Thu 01 Dec 2016 01:43:29 AM UTC | root |
number  | nclu pre  'net commit' (user cumulus)  |
post   | 21 | 20    | Thu 01 Dec 2016 01:43:31 AM UTC | root |
number  | nclu post 'net commit' (user cumulus)  |
pre    | 22 |       | Thu 01 Dec 2016 01:44:18 AM UTC | root |
number  | nclu pre  '20 rollback' (user cumulus) |
post   | 23 | 22    | Thu 01 Dec 2016 01:44:18 AM UTC | root |
number  | nclu post '20 rollback' (user cumulus) |
single | 26 |       | Thu 01 Dec 2016 11:23:06 PM UTC | root
|         | test_snapshot                           |
pre    | 29 |       | Thu 01 Dec 2016 11:55:16 PM UTC | root |
number  | pre-apt                                 | important=yes
post   | 30 | 29    | Thu 01 Dec 2016 11:55:21 PM UTC | root |
number  | post-apt                                | important=yes
pre    | 31 |       | Fri 02 Dec 2016 12:18:08 AM UTC | root |
number  | nclu pre  'ACL' (user cumulus)          |
post   | 32 | 31    | Fri 02 Dec 2016 12:18:10 AM UTC | root |
number  | nclu post 'ACL' (user cumulus)          |
```

Viewing Differences between Snapshots

To see a line by line comparison of changes between two snapshots, run:

```
cumulus@switch:~$ sudo snapper diff 20..21
--- /.snapshots/20/snapshot/etc/cumulus/acl/policy.d/50_nclu_acl.
rules    2016-11-30 23:00:42.675092103 +0000
+++ /.snapshots/21/snapshot/etc/cumulus/acl/policy.d/50_nclu_acl.
rules    2016-12-01 01:43:30.029171289 +0000
```

```
@@ -1,7 +0,0 @@
-[iptables]
-# control-plane: acl ipv4 EXAMPLE1 inbound
--A INPUT --in-interface swp+ -j ACCEPT -p tcp -s 10.0.0.11/32 -d
10.0.0.12/32 --dport 110
-
-# swp1: acl ipv4 EXAMPLE1 inbound
--A FORWARD --in-interface swp1 --out-interface swp2 -j ACCEPT -p tcp
-s 10.0.0.11/32 -d 10.0.0.12/32 --dport 110
-
--- /.snapshots/20/snapshot/var/lib/cumulus/nclu/nclu_acl.conf
2016-11-30 23:00:18.030079000 +0000
+++ /.snapshots/21/snapshot/var/lib/cumulus/nclu/nclu_acl.conf
2016-12-01 00:23:10.096136000 +0000
@@ -1,8 +1,3 @@
-acl ipv4 EXAMPLE1 priority 10 accept tcp 10.0.0.11/32 10.0.0.12/32
pop3 outbound-interface swp2

-control-plane
-    acl ipv4 EXAMPLE1 inbound

-iface swp1
-    acl ipv4 EXAMPLE1 inbound
```

You can view the diff for a single file by specifying the name in the command:

```
cumulus@switch:~$ sudo snapper diff 20..21 /var/lib/cumulus/nclu
/nclu_acl.conf
--- /.snapshots/20/snapshot/var/lib/cumulus/nclu/nclu_acl.conf
2016-11-30 23:00:18.030079000 +0000
+++ /.snapshots/21/snapshot/var/lib/cumulus/nclu/nclu_acl.conf
2016-12-01 00:23:10.096136000 +0000
@@ -1,8 +1,3 @@
-acl ipv4 EXAMPLE1 priority 10 accept tcp 10.0.0.11/32 10.0.0.12/32
pop3 outbound-interface swp2

-control-plane
-    acl ipv4 EXAMPLE1 inbound

-iface swp1
-    acl ipv4 EXAMPLE1 inbound
```

For a higher level view, displaying the names of changed/added/deleted files only, run:

```
cumulus@switch:~$ sudo snapper status 20..21
c..... /etc/cumulus/acl/policy.d/50_nclu_acl.rules
c..... /var/lib/cumulus/nclu/nclu_acl.conf
```

Deleting Snapshots

You can remove one or more snapshots using both NCLU and snapper.

Take care when deleting a snapshot, as you cannot restore it once it's been deleted.

To remove a single snapshot or a range of them created with NCLU, run:

```
cumulus@switch:~$ net commit delete SNAPSHOT|SNAPSHOT1-SNAPSHOT2
```

To remove a single snapshot or a range of snapshots using `snapper`, run:

```
cumulus@switch:~$ sudo snapper delete SNAPSHOT|SNAPSHOT1-SNAPSHOT2
```

Snapshot 0 is the running configuration. You can't roll back to it or delete it. However, you can take a snapshot of it.

Snapshot 1 is the root file system.

The `snapper` utility preserves a number of snapshots, and automatically deletes older snapshots once the limit is reached. It does this in two ways.

By default, `snapper` preserves 10 snapshots that are labeled *important*. A snapshot is labeled important if it was created when you run `apt-get`. To change this number, run:

```
cumulus@switch:~$ sudo snapper set-config NUMBER_LIMIT_IMPORTANT=<NUM>
```

You should always make `NUMBER_LIMIT_IMPORTANT` an even number since two snapshots are always taken before and after an upgrade. This does not apply to `NUMBER_LIMIT`, described next.

`snapper` also deletes unlabeled snapshots. The default number of snapshots `snapper` preserves is 5. To change this number, run:

```
cumulus@switch:~$ sudo snapper set-config NUMBER_LIMIT=<NUM>
```

Also, you can prevent snapshots from being taken automatically before and running `apt-get upgrade|install|remove|dist-upgrade`. Edit `/etc/cumulus/apt-snapshot.conf` and set:

```
APT_SNAPSHOT_ENABLE=no
```

Rolling Back to Earlier Snapshots

If you need to restore Cumulus RMP to an earlier state, you can roll back to an older snapshot.

For a snapshot created with NCLU, you can revert to a specific snapshot listed in the output from `net show commit history`, or you can revert to the previous snapshot by specifying *last* when you run:

```
cumulus@switch:~$ net commit rollback SNAPSHOT_NUMBER|last
```

For any snapshot on the switch, you can use `snapper` to roll back to a specific snapshot. When running `snapper rollback`, you must reboot the switch for the rollback to complete:

```
cumulus@switch:~$ sudo snapper rollback SNAPSHOT_NUMBER
cumulus@switch:~$ sudo reboot
```

You can also revert to an earlier version of a specific file instead of rolling back the whole file system:

```
cumulus@switch:~$ sudo snapper undochange 31..32 /etc/cumulus/acl
/policy.d/50_nclu_acl.rules
```

You can also copy the file directly from the snapshot directory:

```
cumulus@switch:~$ cp /.snapshots/32/snapshot/etc/cumulus/acl
/policy.d/50_nclu_acl.rules /etc/cumulus/acl/policy.d/
```

Configuring Automatic Time-based Snapshots

You can configure Cumulus RMP to take hourly snapshots. You need to enable `TIMELINE_CREATE` in the snapper configuration:

```
cumulus@switch:~$ sudo snapper set-config TIMELINE_CREATE=yes
cumulus@switch:~$ sudo snapper get-
config
Key                       | Value
-----------------------+-------
ALLOW_GROUPS              |
ALLOW_USERS               |
BACKGROUND_COMPARISON     | yes
EMPTY_PRE_POST_CLEANUP    | yes
EMPTY_PRE_POST_MIN_AGE    | 1800
FSTYPE                    | btrfs
```

```
NUMBER_CLEANUP         | yes
NUMBER_LIMIT           | 5
NUMBER_LIMIT_IMPORTANT | 10
NUMBER_MIN_AGE         | 1800
QGROUP                 |
SPACE_LIMIT            | 0.5
SUBVOLUME              | /
SYNC_ACL               | no
TIMELINE_CLEANUP       | yes
TIMELINE_CREATE        | yes
TIMELINE_LIMIT_DAILY   | 5
TIMELINE_LIMIT_HOURLY  | 5
TIMELINE_LIMIT_MONTHLY | 5
TIMELINE_LIMIT_YEARLY  | 5
TIMELINE_MIN_AGE       | 1800
```

Caveats and Errata

root Partition Mounted Multiple Times

You may notice that the root partition gets mounted multiple times. This is due to the way the `btrfs` file system handles subvolumes, mounting the root partition once for each subvolume. `btrfs` keeps one subvolume for each snapshot taken, which stores the snapshot data. While all snapshots are subvolumes, not all subvolumes are snapshots.

Cumulus RMP excludes a number of directories when it takes a snapshot of the root file system (and from any rollbacks):

Directory	Reason
/home	Excluded to avoid user data loss on rollbacks.
/var/log, /var/support	Log file and Cumulus support location. Excluded from snapshots to allow post-rollback analysis.
/tmp, /var/tmp	No need to rollback temporary files.
/opt, /var/opt	Third-party software usually are installed in /opt. Exclude /opt to avoid re-installing these applications after rollbacks.
/srv	Contains data for HTTP and FTP servers. Excluded this directory to avoid server data loss on rollbacks.
/usr/local	This directory is used when installing locally built software. Exclude this directory to avoid re-installing these software after rollbacks.
/var/spool	Exclude this directory to avoid loss of mail after a rollback.
/var/lib/libvirt/images	

Directory	Reason
	This is the default directory for libvirt VM images. Exclude from the snapshot. Additionally disable Copy-On-Write (COW) for this subvolume as COW and VM image I/O access patterns do not play nice.
/boot/grub/i386-pc, /boot/grub/x86_64-efi, /boot/grub/arm-uboot	The GRUB kernel modules must stay in sync with the GRUB kernel installed in the master boot record or UEFI system partition.

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.

Before running any `apt-get` commands or after changing the `/etc/apt/sources.list` file, you need to run `apt-get update`.

Contents

This chapter covers ...

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 tcpdump
Package: tcpdump
Status: install ok installed
Priority: optional
Section: net
Installed-Size: 1092
Maintainer: Romain Francoise <rfrancoise@debian.org>
Architecture: amd64
Multi-Arch: foreign
Version: 4.6.2-5+deb8u1
Depends: libc6 (>= 2.14), libpcap0.8 (>= 1.5.1), libssl1.0.0 (>= 1.0.0
)
Description: command-line network traffic analyzer
 This program allows you to dump the traffic on a network. tcpdump
 is able to examine IPv4, ICMPv4, IPv6, ICMPv6, UDP, TCP, SNMP, AFS
 BGP, RIP, PIM, DVMRP, IGMP, SMB, OSPF, NFS and many other packet
 types.
 .
 It can be used to print out the headers of packets on a network
 interface, filter packets that match a certain expression. You can
```

```
  use this tool to track down network problems, to detect attacks
  or to monitor network activities.
Description-md5: f01841bfda357d116d7ff7b7a47e8782
Homepage: http://www.tcpdump.org/
```

⚠ 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
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/ CumulusLinux-1.5/main
tcpreplay amd64 4.6.2-5+deb8u1 [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_4.6.2-5+deb8u1_amd64.deb) ...
Processing triggers for man-db ...
Setting up tcpreplay (4.6.2-5+deb8u1) ...
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
+++-=====================-===================-====================-
=====================================
un  tcpd                      <none>              <none>              (no
description available)
ii  tcpdump                   4.6.2-5+deb8u1      amd64
command-line network traffic analyzer
```

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 jessie main
deb http://security.debian.org/ jessie/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}
```

Related Information

- 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, connect it to the management network, then install Cumulus RMP via ONIE; the initial configuration gets invoked via ZTP. Alternatively, you can insert a USB stick with the configuration so the provisioning process can start automatically.

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

This chapter covers ...

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 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 75)).

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 74) 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 75)).
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 DCHP 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 dc1-tor-sw1 { hardware ethernet 44:38:39:00:1a:6b; fixed-
address 192.168.0.101; option host-name "dc1-tor-sw1"; }
}
```

Detailed Look at HTTP Headers

The following HTTP headers are sent in the request to the web server 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 needed 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" >&
2
  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/FAILU
RE)
 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:
Sate Directory does not exist. Creating it...
May 11 16:37:45 cumulus ztp[400]: ztp [400]: /var/run/ztp.lock: Lock
File does not exist. Creating it...
May 11 16:37:45 cumulus ztp[400]: ztp [400]: /var/lib/cumulus/ztp
/ztp_state.log: State File does not exist. Creating it...
May 11 16:37:45 cumulus ztp[400]: ztp [400]: ZTP LOCAL: Looking for
ZTP local Script
May 11 16:37:45 cumulus ztp[400]: ztp [400]: ZTP LOCAL: Waterfall
search for /var/lib/cumulus/ztp/cumulus-ztp-x86_64-dell_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
/ztp/ztp_state.log: State File does not exist. Creating it...
2016-05-11T16:37:45.185598+00:00 cumulus ztp [400]: ZTP LOCAL:
Looking for ZTP local Script
2016-05-11T16:37:45.485084+00:00 cumulus ztp [400]: ZTP LOCAL:
Waterfall search for /var/lib/cumulus/ztp/cumulus-ztp-x86_64-
dell_s6000_s1220-rUNKNOWN
2016-05-11T16:37:45.486394+00:00 cumulus ztp [400]: ZTP LOCAL:
Waterfall search for /var/lib/cumulus/ztp/cumulus-ztp-x86_64-
dell_s6000_s1220
2016-05-11T16:37:45.488385+00:00 cumulus ztp [400]: ZTP LOCAL:
Waterfall search for /var/lib/cumulus/ztp/cumulus-ztp-x86_64-dell
2016-05-11T16:37:45.489665+00:00 cumulus ztp [400]: ZTP LOCAL:
Waterfall search for /var/lib/cumulus/ztp/cumulus-ztp-x86_64
2016-05-11T16:37:45.490854+00:00 cumulus ztp [400]: ZTP LOCAL:
Waterfall search for /var/lib/cumulus/ztp/cumulus-ztp
2016-05-11T16:37:45.492296+00:00 cumulus ztp [400]: ZTP USB: Looking 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 bootup:

- 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 rebooted.
- 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.

Interface Configuration and Management

`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 97).

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

This chapter covers ...

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>
```

`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 97). 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

```
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
```

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

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

```
auto swp1.100
iface swp1.100
auto swp2.100
iface swp2.100
auto br-100
iface br-100
    address 10.0.12.2
/24
    address 2001:dad:
beef::3/64
    bridge-ports
swp1.100 swp2.100
    bridge-stp on
```

Without Child Interfaces Defined

```
auto br-100
iface br-100
    address 10.0.12.2/2
4
    address 2001:dad:
beef::3/64
    bridge-ports swp1.1
00 swp2.100
    bridge-stp on
```

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-depends` option. `--with-depends` 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-depends 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-depends 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
```

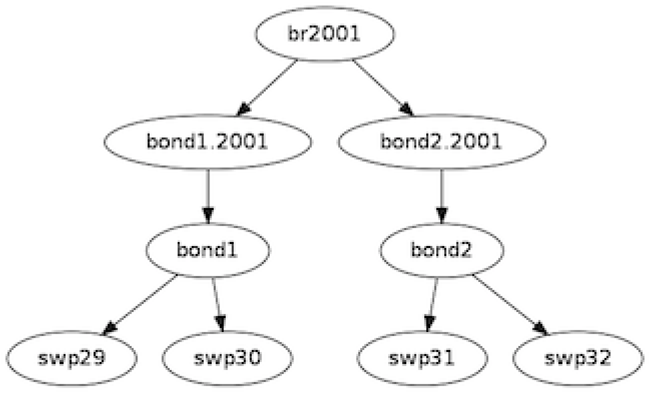
To print the dependency list of a single interface, use:

```
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:

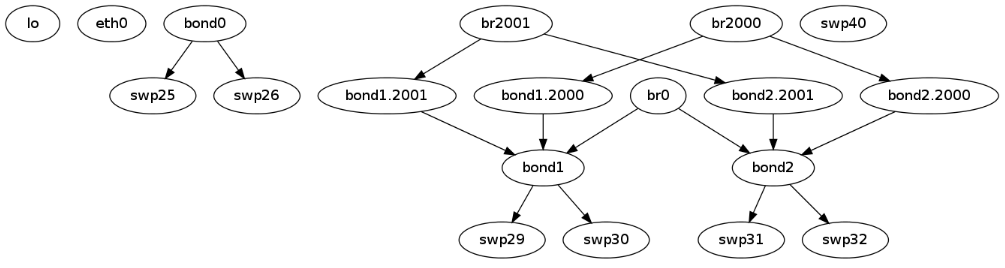
```
cumulus@switch:~$ sudo ifquery --print-dependency=dot br2001
/* Generated by GvGen v.0.9 (http://software.inl.fr/trac/wiki/GvGen)
*/
digraph G {
    compound=true;
    node1 [label="br2001"];
    node2 [label="bond1.2001"];
    node3 [label="bond2.2001"];
    node4 [label="bond1"];
    node5 [label="bond2"];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

IP addresses are configured with the `net add interface` command.

Example IP Address Configuration

The following commands configure three IP addresses for swp1: two IPv4 addresses, and one IPv6 address.

```
cumulus@switch:~$ net add interface swp1 address 12.0.0.1/30
cumulus@switch:~$ net add interface swp1 address 12.0.0.2/30
cumulus@switch:~$ net add interface swp1 address 2001:DB8::1
/126
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

You can specify both IPv4 and IPv6 addresses for the same interface.

These commands create the following code snippet:

```
auto swp1
iface swp1
    address 12.0.0.1/30
    address 12.0.0.2/30
    address 2001:DB8::1/126
```

The address method and address family are added by NCLU when needed, specifically when you are creating DHCP or loopback interfaces.

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

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
```

Specifying IP Address Scope

`ifupdown2` does not honor the configured IP address scope setting in `/etc/network/interfaces`, treating all addresses as global. It does not report an error. Consider this example configuration:

```
auto swp2
iface swp2
    address 35.21.30.5/30
    address 3101:21:20::31/80
    scope link
```

When you run `ifreload -a` on this configuration, `ifupdown2` considers all IP addresses as global.

```
cumulus@switch:~$ ip addr show swp2
5: swp2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
state UP group default qlen 1000
link/ether 74:e6:e2:f5:62:82 brd ff:ff:ff:ff:ff:ff
inet 35.21.30.5/30 scope global swp2
valid_lft forever preferred_lft forever
inet6 3101:21:20::31/80 scope global
valid_lft forever preferred_lft forever
inet6 fe80::76e6:e2ff:fef5:6282/64 scope link
valid_lft forever preferred_lft forever
```

To work around this issue, configure the IP address scope:

Example post-up Configuration

```
cumulus@switch:~$ net add interface swp6 post-up ip address
add 71.21.21.20/32 dev swp6
cumulus@switch:~$ net add interface swp6 scope site
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following code snippet in the `/etc/network/interfaces` file:

```
auto swp6
iface swp6
    post-up ip address add 71.21.21.20/32 dev swp6
    scope site
```

Now it has the correct scope:

```
cumulus@switch:~$ ip addr show swp6
9: swp6: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
state UP group default qlen 1000
link/ether 74:e6:e2:f5:62:86 brd ff:ff:ff:ff:ff:ff
inet 71.21.21.20/32 scope site swp6
valid_lft forever preferred_lft forever
inet6 fe80::76e6:e2ff:fef5:6286/64 scope link
valid_lft forever preferred_lft forever
```

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.

```
cumulus@switch:~$ net add interface swp1 address-purge no
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following configuration snippet in the `/etc/network/interfaces` file:

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

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 96) 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:

```
cumulus@switch:~$ net add interface swp1 post-up /sbin/foo bar
cumulus@switch:~$ net add interface address 12.0.0.1/30
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following configuration in the `/etc/network/interfaces` file:

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

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.

If your `post-up` command also starts, restarts or reloads any `systemd` service, you must use the `--no-block` option with `systemctl`. Otherwise, that service or even the switch itself may hang after starting or restarting.

For example, to restart the `dhcrelay` service after bringing up VLAN 100, first run:

```
cumulus@switch:~$ net add vlan-interface vlan100 post-up
systemctl --no-block restart dhcrelay.service
```

This command creates the following configuration in the `/etc/network/interfaces` file:

```
auto bridge
iface bridge
    bridge-vids 100
    bridge-vlan-aware yes

auto vlan100
iface vlan100
    post-up systemctl --no-block restart dhcrelay.service
    vlan-id 100
    vlan-raw-device bridge
```

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

NCLU supports globs to define port lists (that is, a range of ports). The `glob` keyword is implied when you specify bridge ports and bond slaves:

```
cumulus@switch:~$ net add bridge bridge-ports swp1-4,6,10-12
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

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:

Example Alias Configuration

The following commands create an alias for swp1:

```
cumulus@switch:~$ net add interface swp1 alias
hypervisor_port_1
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following code snippet:

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

You can query the interface description using NCLU. The alias appears in the **Name** column after the actual interface name:

```
cumulus@switch$ net show interface swp1
    Name                          MAC                  Speed      MTU  Mode
--  ------------------------  -----------------  -------  -----
---------
```

```
UP   swp1 (hypervisor_port_1)  44:38:39:00:00:04  1G          1500
Access/L2
```

Interface descriptions also appear in the SNMP OID (see page 194) IF-MIB::ifAlias.

⚠ Aliases are limited to 256 characters.

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.

Related Information

- Debian - Network Configuration

- Linux Foundation - Bonds
- Linux Foundation - Bridges
- Linux Foundation - VLANs
- man ifdown(8)
- man ifquery(8)
- man ifreload
- man ifup(8)
- man ifupdown-addons-interfaces(5)
- man interfaces(5)

Layer 1 and Switch Port Attributes

This chapter discusses the various network interfaces on a switch running Cumulus RMP, how to configure various interface-level settings (if needed) and some troubleshooting commands.

Contents

This chapter covers ...

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)

Interface Settings

Each physical network interface has a number of configurable settings:

- Auto-negotiation

- Duplex
- Link speed
- MTU, or maximum transmission unit

Almost all of these settings are configured automatically for you, depending upon your switch ASIC, although you must always set MTU manually.

⚠ You can only set MTU for logical interfaces. If you try to set auto-negotiation, duplex mode or link speed for a logical interface, an unsupported error gets returned.

Enabling Auto-negotiation

To configure auto-negotiation, set `link-autoneg` to *on* for all the switch ports. For example, to enable auto-negotiation for swp1 through swp52:

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

Any time you enable auto-negotiation, Cumulus RMP restores the default configuration settings specified in the table below (see page).

By default, auto-negotiation is disabled — except on 10G and 1G BASE-T switches, where it's required for links to work at all. And for RJ45-SFP converters, you need to manually configure the settings as described in the default settings table below (see page).

If you disable it later or never enable it, then you have to configure the duplex and link speed settings manually using NCLU — see the relevant sections below. The default speed if you disable auto-negotiation depends on the type of connector used with the port. For example, SFP+ optics default to 10G.

You cannot or should not disable auto-negotiation off for any type of copper cable, including:

- 10G BASE-T
- 10G DAC

However, RJ-45 (10/100/1000 BASE-T) adapters do not work with auto-negotiation enabled. You must manually configure these ports using the settings below (link-autoneg=off, link-speed=1000|100|10, link-duplex=full|half).

Default Interface Configuration Settings

The configuration for each type of interface is described in the following table. Except as noted below, the settings for both sides of the link are expected to be the same.

⚠ If the other side of the link is running a version of Cumulus RMP or Cumulus Linux earlier than 3.2, depending up on the interface type, auto-negotiation may not work on that switch. Cumulus Networks recommends you use the default settings on this switch in this case.

Speed	Auto-negotiation	Manual Configuration Steps	Notes

1000BASE-T (RJ45)	Off	`$ net add interface swp1 link-speed 1000`
		`$ net add interface swp1 link-autoneg off`
		`$ net add interface swp1 link-duplex full`
		
		Configuration in /etc/network/interfaces
		
		`auto swp1`
		`iface swp1`
		`  link-autoneg off`
		`  link-speed 1000`
		`  link-duplex full`
			• The module has two sets of electronics — the port side, which communicates to the switch ASIC, and the RJ45 side.
			• Auto-negotiation is always used on the RJ45 side of the link by the PHY built into the module. This is independent of the switch setting. Set `link-autoneg` to off.
			• Auto-negotiation needs to be enabled on the server side in this scenario.

10G BASE-CR,
10G BASE-LR,
10G BASE-SR,
10G AOC
	Off	`$ net add interface swp1 link-speed 10000`
		`$ net add interface swp1 link-autoneg off`
		`$ net add interface swp1 link-duplex full`	

Speed	Auto-negotiation	Manual Configuration Steps	Notes

		Configuration in /etc/network/interfaces
		
		`auto swp1`
		`iface swp1`
		`  link-autoneg off`
		`  link-speed 10000`
		`  link-duplex full`	

Port Speed and Duplexing

Cumulus RMP supports both half- and full-duplex configurations. Supported port speeds include 100M, 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, you must also specify the duplex mode setting along with it; otherwise, duplex mode defaults to *half-duplex*.

Example Port Speed and Duplexing Configuration

The following NCLU commands configure the port speed and duplex mode for the swp1 interface:

```
cumulus@switch:~$ net add interface swp1 link-speed 10000
cumulus@switch:~$ net add interface link-duplex full
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The above commands create the following `/etc/network/interfaces` code snippet:

```
auto swp1
iface swp1
   link-speed 10000
   link-duplex full
```

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)

MTU

Interface MTU (maximum transmission unit) applies to traffic traversing the management port, front panel /switch ports, bridge, VLAN subinterfaces and bonds — in other words, both physical and logical interfaces.

MTU is the only interface setting that must be set manually.

In Cumulus Linux, `ifupdown2` assigns 1500 as the default MTU setting. You can override this default value by specifying a policy file in `/etc/network/ifupdown2/policy.d/`, like in the following example:

```
cumulus@switch:~$ cat /etc/network/ifupdown2/policy.d/address.json
{
    "address": {
        "defaults": { "mtu": "9000" }
    }
}
```

MTU for a Bridge

The 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), even if another bridge member has a higher MTU value. There is **no** need to specify an MTU on the bridge. Consider this bridge configuration:

```
auto bridge
iface bridge
    bridge-ports bond1 bond2 bond3 bond4 peer5
    bridge-vlan-aware yes
```

```
     bridge-vids 100-110
     bridge-stp on
```

In order for *bridge* to have an MTU of 9000, set the MTU for each of the member interfaces (bond1 to bond 4, and peer5), to 9000 at minimum.

Use MTU 9216 for a bridge

Two common MTUs for jumbo frames are 9216 and 9000 bytes. The corresponding MTUs for the VNIs would be 9166 and 8950.

When configuring MTU for a bond, configure the MTU value directly under the bond interface; the configured value is inherited by member links/slave interfaces. If you need a different MTU on the bond, set it on the bond interface, as this ensures the slave interfaces pick it up. There is no need to specify MTU on the slave interfaces.

VLAN interfaces inherit their MTU settings from their physical devices or their lower interface; for example, swp1.100 inherits its MTU setting from swp1. Hence, specifying an MTU on swp1 ensures that swp1.100 inherits swp1's MTU setting.

Example MTU Configuration

In general, the policy file specified above handles default MTU settings for all interfaces on the switch. If you need to configure a different MTU setting for a subset of interfaces, use NCLU.

The following commands configure an MTU minimum value of 9000 on swp1:

```
cumulus@switch:~$ net add interface swp1 mtu 9000
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following code snippet:

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

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.

To view the MTU setting, use `net show interface <interface>`:

```
cumulus@switch:~$ net show interface swp1
    Name    MAC                Speed     MTU  Mode
```

```
--  ------   -----------------  -------  -----  ---------
UP  swp1     44:38:39:00:00:04  1G        1500  Access/L2
```

Verification and Troubleshooting Commands

Statistics

High-level interface statistics are available with the `net show interface` command:

```
cumulus@switch:~$ net show interface swp1

    Name    MAC                Speed      MTU  Mode
--  ------  -----------------  -------  -----  ---------
UP  swp1    44:38:39:00:00:04  1G        1500  Access/L2


Vlans in disabled State
-------------------------
br0


Counters      TX    RX
----------  ----  ----
errors         0     0
unicast        0     0
broadcast      0     0
multicast      0     0


LLDP
------  ----  ---------------------------
swp1    ====  44:38:39:00:00:03(server01)
```

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|RXPower :|TXPower :
|EthernetComplianceCode)'

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

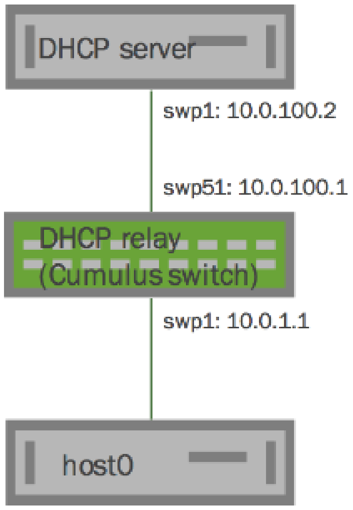
Related Information

- Debian - Network Configuration
- Linux Foundation - VLANs
- Linux Foundation - Bridges
- Linux Foundation - Bonds

Configuring DHCP Relays and Servers

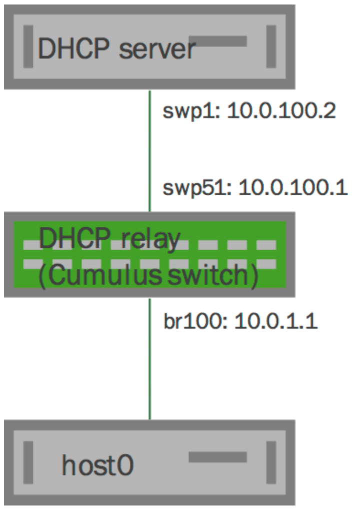
You can configure an interface so it can make DHCP relay requests for IPv4 and IPv6.

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



For the configurations used in this chapter, both the DHCP server and DHCP relay are switches running Cumulus RMP; however, the DHCP server can also be located on a dedicated switch in your environment.

Another way to configure this would be to connect the host to the DHCP relay via a layer 2 bridge instead of a switch port over layer 3:



The `dhcpd` and `dhcrelay` services are disabled by default. After you finish configuring the DHCP relays and servers, you need to start those services.

Contents

This chapter covers ...

Configuring the Host Interfaces

Configure the host interfaces for both IPv4 and IPv6 for DHCP by adding the DHCP port relay to the IPv4 and IPv6 host interfaces.

Example IPv4 Host Interface DHCP Configuration

The example NCLU commands below configure the IPv4 host interface `eth1` for DHCP:

NCLU Commands

```
cumulus@switch:~$ net add interface eth1 ip address dhcp
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU commands above produce the following `/etc/network/interfaces` snippet:

```
auto eth1
iface eth1 inet dhcp
```

Example IPv6 Host Interface DHCP Configuration

The example NCLU commands below configure the IPv6 host interface `eth1` for DHCP:

NCLU commands

```
cumulus@switch:~$ net add interface eth1 ipv6 address dhcp
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU commands above produce the following `/etc/network/interfaces` snippet:

```
auto eth1
iface eth1 inet6 dhcp
```

Configuring the DHCP Relays on Cumulus RMP Switches

Configure the IPv4 and IPv6 DHCP relays on each leaf switch. You need to run two independent instances of `dhcrelay`, one for IPv4 and one for IPv6. The `dhcrelay` feature is part of the `isc-dhcp-relay` services.

Edit the `systemd` launch scripts so that the `dhcrelay` service starts with the switch. The launch scripts for `systemd` are located in `/lib/systemd/system`.

If your launch script also starts, restarts or reloads any `systemd` service, including `dhcrelay.service`, you must use the `--no-block` option with the `systemctl` command. Otherwise, that service or even the switch itself may hang after starting or restarting.

Configuring the DHCP Relay Interfaces

As described above, you can configure the interfaces on the DHCP relay (the Cumulus Linux or Cumulus RMP switch) a number of ways, either as layer 3 or as a layer 2 bridge.

Example Layer 3 Configuration

The following NCLU commands create a layer 3 connected client configuration:

```
cumulus@switch:~$ net add interface swp1 address 10.0.1.1/30
cumulus@switch:~$ net add interface swp51 address 10.0.100.1/30
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following code snippet in `/etc/network/interfaces`:

```
auto swp1
iface swp1
  address 10.0.1.1/30

auto swp51
iface swp51
  address 10.0.100.1/30
```

A layer 2 bridge can be configured in either VLAN-aware (see page 152) or traditional (see page 161) mode.

Example Layer 2 VLAN-aware Bridge

The following commands create a VLAN-aware bridge:

```
cumulus@switch:~$ net add interface swp1 bridge-access 100
cumulus@switch:~$ net add interface swp51 address 10.0.100.1/30
cumulus@switch:~$ net add bridge bridge-ports swp1
cumulus@switch:~$ net add vlan-interface bridge.100 address 10.
0.1.1/24
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following code snippet in `/etc/network/interfaces`:

```
auto swp1
iface swp1
  bridge-access 100

auto swp51
iface swp51
  address 10.0.100.1/30

auto bridge
iface bridge
  bridge-vlan-aware yes
  bridge-ports swp1

auto bridge.100
iface bridge.100
  address 10.0.1.1/24
```

Example Traditional Bridge Configuration

To create a traditional bridge configuration, edit `/etc/network/interfaces` and create a code snippet similar to the one below:

```
auto swp51
iface swp51
  address 10.0.100.1/30

auto br100
iface br100
  address 10.0.1.1/24
  bridge-ports swp1
  bridge-stp yes
```

Configuring IPv4 DHCP Relays

Edit `dhcrelay.service`, as described below. The IPv4 `dhcrelay.service` *Unit* script calls `/etc/default/isc-dhcp-relay` to find launch variables.

```
cumulus@switch:~$ cat /lib/systemd/system/dhcrelay.service
[Unit]
Description=DHCPv4 Relay Agent Daemon
Documentation=man:dhcrelay(8)
After=network-oneline.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
```

The `/etc/default/isc-dhcp-relay` variables file needs to reference both interfaces participating in DHCP relay (facing the server and facing the client) and the IP address of the server. If the client-facing interface is a bridge port, specify the switch virtual interface (SVI) name if using a VLAN-aware bridge (for example, bridge.100), or the bridge name if using traditional bridging (for example, br100).

```
cumulus@switch:~$ net add dhcp relay interface swp1
cumulus@switch:~$ net add dhcp relay interface swp51
cumulus@switch:~$ net add dhcp relay server 10.0.100.2
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following configuration in the `/etc/default/isc-dhcp-relay` file:

```
cumulus@switch:~$ cat /etc/default/isc-dhcp-relay
SERVERS="10.0.100.2"

INTF_CMD="-i swp1 -i swp51"

OPTIONS=""
```

After you've finished configuring the DHCP relay, enable the `dhcrelay` service so the configuration persists between reloads:

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

Configuring IPv6 DHCP Relays

If you're configuring IPv6, you need to create the `dhcrelay6.service` file and populate its content, as described below. The `dhcrelay6.service` *Unit* script calls `/etc/default/isc-dhcp-relay6` to find launch variables.

```
cumulus@switch:~$ cat /lib/systemd/system/dhcrelay6.service
[Unit]
Description=DHCPv6 Relay Agent Daemon
Documentation=man:dhcrelay(8)
After=network-oneline.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
```

The `/etc/default/isc-dhcp-relay6` variables file has a different format than the `/etc/default/isc-dhcp-relay` file used for IPv4 DHCP relays. Make sure to configure the variables appropriately by editing this file:

```
cumulus@switch:$ cat /etc/default/isc-dhcp-relay6
SERVERS=" -u 2001:db8:100::2%swp51"

INTF_CMD="-l swp1"
```

⚠ You cannot use NCLU to configure IPv6 relays.

After you've finished configuring the DHCP relay, enable the `dhcrelay6` service so the configuration persists between reloads:

```
cumulus@switch:~$ sudo systemctl enable dhcrelay6.service
```

Configuring DHCP Server on Cumulus RMP Switches

You can use the following sample configurations for `dhcp.conf` and `dhcpd6.conf` to start both an IPv4 and an IPv6 DHCP server. The configuration files for the two DHCP server instances need to have two pools:

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

Configuring the IPv4 DHCP Server

In a text editor, edit the **dhcpd.conf** file with a configuration similar to the following:

```
cumulus@switch:~$ cat /etc/dhcp/dhcpd.conf
ddns-update-style none;

default-lease-time 600;
max-lease-time 7200;

subnet 10.0.100.0 netmask 255.255.255.0 {
}
subnet 10.0.1.0 netmask 255.255.255.0 {
        range 10.0.1.50 10.0.1.60;
}
```

Just as you did with the DHCP relay scripts, edit the DHCP server configuration file so it can launch the DHCP server when the system boots. Here is a sample configuration:

```
cumulus@switch:~$ cat /etc/default/isc-dhcp-server
DHCPD_CONF="-cf /etc/dhcp/dhcpd.conf"

INTERFACES="swp1"
```

After you've finished configuring the DHCP server, enable the **dhcpd** service immediately:

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

Configuring the IPv6 DHCP Server

In a text editor, edit the **dhcpd6.conf** file with a configuration similar to the following:

```
cumulus@switch:~$ cat /etc/dhcp/dhcpd6.conf
ddns-update-style none;

default-lease-time 600;
max-lease-time 7200;

subnet6 2001:db8:100::/64 {
}
subnet6 2001:db8:1::/64 {
        range6 2001:db8:1::100 2001:db8:1::200;
}
```

Just as you did with the DHCP relay scripts, edit the DHCP server configuration file so it can launch the DHCP server when the system boots. Here is a sample configuration:

```
cumulus@switch:~$ cat /etc/default/isc-dhcp-server6
DHCPD_CONF="-cf /etc/dhcp/dhcpd6.conf"

INTERFACES="swp1"
```

⚠ You cannot use NCLU to configure IPv6 DHCP servers.

After you've finished configuring the DHCP server, enable the `dhcpd6` service immediately:

```
cumulus@switch:~$ sudo systemctl enable dhcpd6.service
```

Troubleshooting the DHCP Relays

If you are experiencing issues with the DHCP relay, you can run the following commands to determine whether or not the issue is with `systemd`. The following commands manually activate the DHCP relay process, and they do not persist when you reboot the switch:

```
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>
```

For example:

```
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
```

See the `man dhcrelay` for more information.

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. STP also provides redundant links for automatic failover when an active link fails. STP is disabled by default on bridges in Cumulus RMP.

Contents

(Click to expand)

Supported Modes

The STP modes Cumulus RMP supports vary depending upon whether the traditional or VLAN-aware bridge driver mode (see page 149) is in use.

Bridges configured in *VLAN-aware (see page 152)* mode operate **only** in RSTP mode. NCLU, the network command line utility for configuring Cumulus RMP, only supports bridges in VLAN-aware mode.

For a bridge configured in *traditional* mode, PVST and PVRST are supported, with the default set to PVRST. Each traditional mode bridge has its own separate STP instance.

Since you cannot use NCLU to configure a traditional mode bridge, you must configure it directly in the `/etc/network/interfaces` file.

Configuring STP for a VLAN-aware Bridge

VLAN-aware (see page 152) bridges only operate in RSTP mode. STP BPDUs are transmitted on the native VLAN.

If a bridge running RSTP (802.1w) receives a common STP (802.1D) BPDU, it will automatically fall back to 802.1D operation. RSTP interoperates with MST seamlessly, creating a single instance of spanning tree, which transmits BPDUs on the native VLAN. RSTP treats the MST domain as if it were one giant switch.

Example STP VLAN-aware Configuration

The following commands configure STP on a VLAN-aware bridge:

```
cumulus@switch:~$ net add bridge bridge-stp yes
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands add `bridge-stp on` to the following code snippet in the `/etc/network/interfaces` file:

```
auto bridge
iface bridge
  bridge-vlan-aware yes
  bridge-vids 100
  bridge-pvid  1
  bridge-ports swp1 swp4 swp5
  bridge-stp on
```

Configuring STP within a Traditional Mode Bridge

Per VLAN Spanning Tree (PVST) creates a spanning tree instance for a bridge. Rapid PVST (PVRST) supports RSTP enhancements for each spanning tree instance. In order to use PVRST with a traditional bridge, a bridge corresponding to the untagged native VLAN must be created, 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 for maximum interoperability.

Manually edit the Linux configuration files ...

To create a traditional mode bridge, configure the bridge stanza under `/etc/network/interfaces`. More information on configuring bridges can be found here. To enable STP on the bridge, include the keyword `bridge-stp on`. swp1 and swp5 are configured for tagging VLAN 100, while swp4 is configured to not tag a VLAN across the link.

```
auto br100
```

```
iface br100
  bridge-ports swp1.100 swp5.100
  bridge-stp on
auto br1
iface br1
  bridge-ports swp1 swp4 swp5
  bridge-stp on
```

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

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

Viewing Bridge and STP Status/Logs

To check STP status for a bridge, run `net show bridge spanning-tree`:

Click to reveal the output ...

```
cumulus@switch:~$ net show bridge spanning-tree
bridge CIST info
  enabled         yes
  bridge id       1.000.44:39:39:FF:40:90
  designated root 1.000.44:39:39:FF:40:90
  regional root   1.000.44:39:39:FF:40:90
  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 253343s
  topology change count      4
  topology change            no
  topology change port       peerlink
  last topology change port  leaf03-04
bridge:exit01-02 CIST info
  enabled            no                       role
Disabled
  port id            8.004                    state
discarding
  external port cost 305                      admin external cost  0
  internal port cost 305                      admin internal cost  0
  designated root    1.000.44:38:39:00:00:27 dsgn external cost   0
  dsgn regional root 1.000.44:38:39:00:00:27 dsgn internal cost   0
  designated bridge  1.000.44:38:39:00:00:27 designated port      8.00
4
```

```
   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         2                        Num TX TCN           0
   Num RX BPDU         0                        Num RX TCN           0
   Num Transition FWD  0                        Num Transition BLK   2
   bpdufilter port     no
   clag ISL            no                       clag ISL Oper UP     no
   clag role           primary                  clag dual conn mac   00:0
0:00:00:00:00
   clag remote portID F.FFF                     clag system mac      44:3
9:39:FF:40:90
bridge:leaf01-02 CIST info
   enabled             yes                      role
Designated
   port id             8.003                    state
forwarding
   external port cost 10000                     admin external cost  0
   internal port cost 10000                     admin internal cost  0
   designated root     1.000.44:39:39:FF:40:90 dsgn external cost    0
   dsgn regional root 1.000.44:39:39:FF:40:90 dsgn internal cost     0
   designated bridge   1.000.44:39:39:FF:40:90 designated port       8.00
3
   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         253558                   Num TX TCN           2
   Num RX BPDU         253373                   Num RX TCN           4
   Num Transition FWD 126675                    Num Transition BLK   1266
94
   bpdufilter port     no
   clag ISL            no                       clag ISL Oper UP     no
   clag role           primary                  clag dual conn mac   44:3
9:39:FF:40:94
   clag remote portID F.FFF                     clag system mac      44:3
9:39:FF:40:90
bridge:leaf03-04 CIST info
   enabled             yes                      role
Designated
   port id             8.001                    state
forwarding
   external port cost 10000                     admin external cost  0
   internal port cost 10000                     admin internal cost  0
   designated root     1.000.44:39:39:FF:40:90 dsgn external cost    0
```

```
  dsgn regional root 1.000.44:39:39:FF:40:90 dsgn internal cost    0
  designated bridge  1.000.44:39:39:FF:40:90 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    2                       disputed              no
  bpdu guard port    no                      bpdu guard error      no
  network port       no                      BA inconsistent       no
  Num TX BPDU        130960                  Num TX TCN            6
  Num RX BPDU        4                       Num RX TCN            1
  Num Transition FWD 2                       Num Transition BLK    1
  bpdufilter port    no
  clag ISL           no                      clag ISL Oper UP      no
  clag role          primary                 clag dual conn mac    44:3
9:39:FF:40:93
  clag remote portID F.FFF                   clag system mac       44:3
9:39:FF:40:90
bridge:peerlink CIST info
  enabled            yes                     role
Designated
  port id            F.002                   state
forwarding
  external port cost 10000                   admin external cost   0
  internal port cost 10000                   admin internal cost   0
  designated root    1.000.44:39:39:FF:40:90 dsgn external cost    0
  dsgn regional root 1.000.44:39:39:FF:40:90 dsgn internal cost    0
  designated bridge  1.000.44:39:39:FF:40:90 designated port       F.00
2
  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        126700                  Num TX TCN            2
  Num RX BPDU        6                       Num RX TCN            3
  Num Transition FWD 2                       Num Transition BLK    1
  bpdufilter port    no
  clag ISL           yes                     clag ISL Oper UP      yes
  clag role          primary                 clag dual conn mac    00:0
0:00:00:00:00
  clag remote portID F.FFF                   clag system mac       44:3
9:39:FF:40:90
```

Using Linux to Check Spanning Tree Status (Advanced)

Using Linux to check STP status ...

`mstpctl` is the utility provided by the `mstpd` service to configure STP. The `mstpd` daemon is an open source project used by Cumulus RMP to implement IEEE802.1D 2004 and IEEE802.1Q 2011.

`mstpd` is started by default when the switch boots. `mstpd` logs and errors are located in `/var/log/syslog`.

`mstpd` is the preferred utility for interacting with STP on Cumulus RMP. `brctl` also provides certain methods for configuring STP; however, they are not as complete as the tools offered in `mstpd` and output from brctl can be misleading in some cases.

To get the bridge state, use:

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

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
  last topology change port  swp5
```

To get the `mstpd` bridge port state, use:

```
cumulus@switch:~$ sudo mstpctl showport br2
 E swp1 8.001 forw F.000.00:14:01:01:01:00 F.000.00:14:01:01:01:00 8.0
01 Desg
   swp4 8.002 forw F.000.00:14:01:01:01:00 F.000.00:14:01:01:01:00 8.0
02 Desg
```

```
 E swp5 8.003 forw F.000.00:14:01:01:01:00 F.000.00:14:01:01:01:00 8.0
03 Desg
cumulus@switch:~$ sudo mstpctl showportdetail br2 swp1
 br2:swp1 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      8.00
1
  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
```

Customizing Spanning Tree Protocol

There are a number of ways you can customize STP in Cumulus RMP. You should exercise extreme caution with many of the settings below to prevent malfunctions in STP's loop avoidance.

Spanning Tree Priority

If you have an MSTI (multiple spanning tree instance), you can set the *tree priority* for a bridge. The bridge with the lowest priority is elected the *root bridge*. The priority must be a number between *0* and *65535* and must be a multiple of 4096; the default is *32768*.

For `msti`, only 0 is supported currently.

To set the tree priority, run:

```
cumulus@switch:~$ net add bridge mstpctl-treeprio 8192
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

PortAdminEdge/PortFast Mode

`PortAdminEdge` is equivalent to the PortFast feature offered by other vendors. It enables or disables the *initial edge state* of a port in a bridge.

All ports configured with PortAdminEdge bypass the listening and learning states to move immediately to forwarding.

Using PortAdminEdge mode has the potential to cause loops if it is not accompanied by the BPDU guard (see page 121) feature.

While it is common for edge ports to be configured as access ports for a simple end host, this is not mandatory. In the data center, edge ports typically connect to servers, which may pass both tagged and untagged traffic.

Example VLAN-aware Bridge Configuration

To configure PortAdminEdge mode, use the `mstpctl-bpduguard` and `mstpctl-portadminedge` NCLU configuration commands:

```
cumulus@switch:~$ net add interface swp5 mstpctl-bpduguard yes
cumulus@switch:~$ net add interface swp5 mstpctl-portadminedge
yes
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU commands above create the following code snippet:

```
auto swp5
iface swp5
    mstpctl-bpduguard yes
    mstpctl-portadminedge yes
```

PortAutoEdge

PortAutoEdge is an enhancement to the standard PortAdminEdge (PortFast) mode, which allows for the automatic detection of edge ports. PortAutoEdge enables and disables the *auto transition* to/from the edge state of a port in a bridge.

Edge ports and access ports are not the same thing. Edge ports transition directly to the forwarding state and skip the listening and learning stages. Upstream topology change notifications are not generated when an edge port's link changes state. Access ports only forward untagged traffic; however, there is no such restriction on edge ports, which can forward both tagged and untagged traffic.

When a BPDU is received on a port configured with portautoedge, the port ceases to be in the edge port state and transitions into a normal STP port.

When BPDUs are no longer received on the interface, the port becomes an edge port, and transitions through the discarding and learning states before resuming forwarding.

To configure PortAutoEdge for an interface:

```
cumulus@switch:~$ net add interface swp1 mstpctl-portautoedge yes
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

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.

Example BPDU Guard Configuration

To configure BPDU guard, set the `mstpctl-bpduguard` value for the interface:

```
cumulus@switch:~$ net add interface swp5 mstpctl-bpduguard yes
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

This creates the following stanza in the `/etc/network/interfaces` file:

```
auto swp5
iface swp5
    mstpctl-bpduguard 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:

```
cumulus@switch:~$ net show bridge spanning-tree | grep bpdu
 bpdu guard port     yes                        bpdu guard error     yes
```

The only way to recover a port that has been placed in the disabled state is to manually 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 dsgn external cost   0
  dsgn regional root 8.000.6C:64:1A:00:4F:9C dsgn 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 dsgn external cost   0
  dsgn regional root 8.000.6C:64:1A:00:4F:9C dsgn 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    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
```

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 a 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 in order for bridge assurance to operate properly.

The default setting for bridge assurance is off. This means that there is no difference between disabling bridge assurance on an interface and not configuring bridge assurance on an interface.

Example Bridge Assurance Configuration

To enable bridge assurance on an interface, add the `mstpctl-portnetwork` option to the interface:

```
cumulus@switch:~$ net add interface swp1 mstpctl-portnetwork
yes
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

This creates the following interface stanza:

```
auto swp1
iface swp1
    mstpctl-portnetwork yes
```

You can monitor logs for bridge assurance messages by doing the following:

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

BPDU Filter

You can enable `bpdufilter` on a switch port, which filters BPDUs in both directions. This effectively disables STP on the port as no BPDUs are transiting.

Using BDPU filter inappropriately can cause layer 2 loops. Use this feature deliberately and with extreme caution.

Example BPDU Filter Configuration

To configure the BPDU filter, add the `mstpctl-portbpdufilter` option to the interface:

```
cumulus@switch:~$ net add interface swp6 mstpctl-
portbpdufilter yes
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following stanza in the `/etc/network/interfaces` file:

```
auto swp6
iface swp6
    mstpctl-portbpdufilter yes
```

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

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 for each physical port in one of three ways:

- By configuring the settings with NCLU.
- By editing `/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.

Example NCLU Configuration

The commands below create an example storm control configuration, with broadcast and multicast storm control enabled, and set to 400 packets per second (pps) and 3000 pps, respectively:

```
cumulus@switch:~$ net add interface swp1 post-up echo 400 >
/cumulus/switchd/config/interface/$IFACE/storm_control
/broadcast
cumulus@switch:~$ net add interface swp1 post-up echo 3000 >
/cumulus/switchd/config/interface/$IFACE/storm_control
/multicast
cumulus@switch:~$ net add interface swp1 post-down echo 0 >
/cumulus/switchd/config/interface/$IFACE/storm_control
/broadcast
cumulus@switch:~$ net add interface swp1 post-down echo 0 >
/cumulus/switchd/config/interface/$IFACE/storm_control
/multicast
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following configuration in the `/etc/network/interfaces` file:

```
auto swp1
iface swp1
  post-up echo 400 > /cumulus/switchd/config/interface/$IFACE
/storm_control/broadcast
  post-up echo 3000 > /cumulus/switchd/config/interface/$IFACE
/storm_control/multicast
  post-down echo 0 > /cumulus/switchd/config/interface/$IFACE
/storm_control/broadcast
  post-down echo 0 > /cumulus/switchd/config/interface/$IFACE
/storm_control/multicast
```

Example /etc/cumulus/switchd.conf Configuration

To enable broadcast and multicast storm control at 400 packets per second (pps) and 3000 pps, respectively, for swp1 with `/etc/cumulus/switchd.conf`:

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

Example Configuration with All Possible 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 swp4
  bridge-stp on
  mstpctl-maxage 20
  mstpctl-ageing 300
  mstpctl-fdelay 15
  mstpctl-maxhops 20
  mstpctl-txholdcount 6
  mstpctl-forcevers rstp
  mstpctl-treeprio 32768
  mstpctl-treeportprio swp3=128
  mstpctl-hello 2
  mstpctl-portpathcost swp1=0 swp2=0
  mstpctl-portadminedge swp1=no swp2=no
  mstpctl-portautoedge swp1=yes swp2=yes
  mstpctl-portp2p swp1=no swp2=no
  mstpctl-portrestrrole swp1=no swp2=no
  mstpctl-portrestrtcn swp1=no swp2=no
  mstpctl-portnetwork swp1=no
  mstpctl-bpduguard swp1=no swp2=no
  mstpctl-portbpdufilter swp4=yes
```

Configuring Other Spanning Tree Parameters

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.

For a comparison of STP parameter configuration between `mstpctl` and other vendors, please read this knowledge base article.

The table below describes the configuration parameters available.

NCLU configuration for these values is done on the interfaces, not the bridge itself.

Parameter	Description
`mstpctl-maxage`	Sets the bridge's *maximum age* to `<max_age>` seconds. The default is *20*. The maximum age must meet the condition 2 * (Bridge Forward Delay - 1 second) >= Bridge Max Age.

Parameter	Description
`mstpctl-ageing`	Sets the Ethernet (MAC) address *ageing time* in `<time>` seconds for the bridge when the running version is STP, but not RSTP/MSTP. The default is *300*.
`mstpctl-fdelay`	Sets the bridge's *bridge forward delay* to `<time>` seconds. The default is *15*. The bridge forward delay must meet the condition 2 * (Bridge Forward Delay - 1 second) >= Bridge Max Age.
`mstpctl-maxhops`	Sets the bridge's *maximum hops* to `<max_hops>`. The default value is *20*.
`mstpctl-txholdcount`	Sets the bridge's *bridge transmit hold count* to `<tx_hold_count>`. The default is *6*.
`mstpctl-forcevers`	Sets the bridge's *force STP version* to either RSTP/STP. MSTP is not supported currently. The default is *RSTP*.
	Sets the bridge's *tree priority* 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 *root bridge*. The default is *32768*. For `msti`, only 0 is supported currently.
`mstpctl-treeportprio`	Sets the *priority* of port `<port>` to `<priority>` 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*. For `msti`, only *0* is supported currently.
`mstpctl-hello`	Sets the bridge's *bridge hello time* to `<time>` seconds. The default is *2*.
`mstpctl-portpathcost`	Sets the *port cost* of the port `<port>` in bridge `<bridge>` to `<cost>`. The default is *0*. `mstpd` supports only long mode; that is, 32 bits for the path cost.
`mstpctl-portadminedge`	Enables/disables the *initial edge state* of the port `<port>` in bridge `<bridge>`. The default is *no*.
`mstpctl-portautoedge`	Enables/disables the *auto transition* to/from the edge state of the port `<port>` in bridge `<bridge>`. The default is *yes*.

Parameter	Description
	portautoedge is an enhancement to the standard PortAdminEdge (PortFast) mode, which allows for the automatic detection of edge ports. ⚠ Edge ports and access ports are not the same thing. Edge ports transition directly to the forwarding state and skip the listening and learning stages. Upstream topology change notifications are not generated when an edge port's link changes state. Access ports only forward untagged traffic; however, there is no such restriction on edge ports, which can forward both tagged and untagged traffic. When a BPDU is received on a port configured with portautoedge, the port ceases to be in the edge port state and transitions into a normal STP port. When BPDUs are no longer received on the interface, the port becomes an edge port, and transitions through the discarding and learning states before resuming forwarding.
`mstpctl-portp2p`	Enables/disables the *point-to-point detection mode* of the port `<port>` in bridge `<bridge>`. The default is *auto*.
`mstpctl-portrestrrole`	Enables/disables the ability of the port `<port>` in bridge `<bridge>` to take the *root role*. The default is *no*.
`mstpctl-portrestrtcn`	Enables/disables the ability of the port `<port>` in bridge `<bridge>` to propagate *received topology change notifications*. The default is *no*.
`mstpctl-portnetwork`	Enables/disables the *bridge assurance capability* for a network port `<port>` in bridge `<bridge>`. The default is *no*.
`mstpctl-bpduguard`	Enables/disables the *BPDU guard configuration* of the port `<port>` in bridge `<bridge>`. The default is *no*.
`mstpctl-portbpdufilter`	Enables/disables the *BPDU filter* functionality for a port `<port>` in bridge `<bridge>`. The default is *no*.

Caveats and Errata

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

Related Information

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

- Sourceforge - mstpd project

- Wikipedia - Spanning Tree Protocol
- brctl(8)
- bridge-utils-interfaces(5)
- ifupdown-addons-interfaces(5)
- mstpctl(8)
- mstpctl-utils-interfaces(5)

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/syslog` 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

This chapter covers ...

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/syslog  | grep lldp
```

```
2016-11-23T14:28:53.937458-05:00 switch lldpd[20333]: protocol LLDP
enabled
2016-11-23T14:28:53.938119-05:00 switch lldpd[20333]: protocol CDPv1
enabled
2016-11-23T14:28:53.938796-05:00 switch lldpd[20333]: protocol CDPv2
enabled
2016-11-23T14:28:53.939480-05:00 switch lldpd[20333]: libevent 2.0.21-
stable initialized with epoll method
2016-11-23T14:28:53.940140-05:00 switch lldpd[20333]: enable SNMP
subagent
2016-11-23T14:28:53.941247-05:00 switch lldpd[20333]: NET-SNMP
version 5.7.3 AgentX subagent connected
2016-11-23T14:28:54.049638-05:00 switch lldpcli[20330]: LLDP PortID
TLV type set to new value : ifname
2016-11-23T14:28:54.050519-05:00 switch lldpcli[20330]: lldpd should
resume operations
```

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:

```
cumulus@switch:~$ sudo lldpcli configure system interface pattern "swp
*"
```

To configure inactive interfaces:

```
cumulus@switch:~$ sudo lldpcli configure system interface pattern-
blacklist "eth0"
```

The active interface list always overrides the inactive interface list.

To reset any interface list to none:

```
cumulus@switch:~$ sudo 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

# Add "-x" to DAEMON_ARGS to start SNMP subagent

# Enable CDP by default
```

```
DAEMON_ARGS="-x -c"
```

Caveats and Errata

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

Related Information

- GitHub - lldpd project
- Wikipedia - Link Layer Discovery Protocol

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

This chapter covers ...

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 140) 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 135).

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.

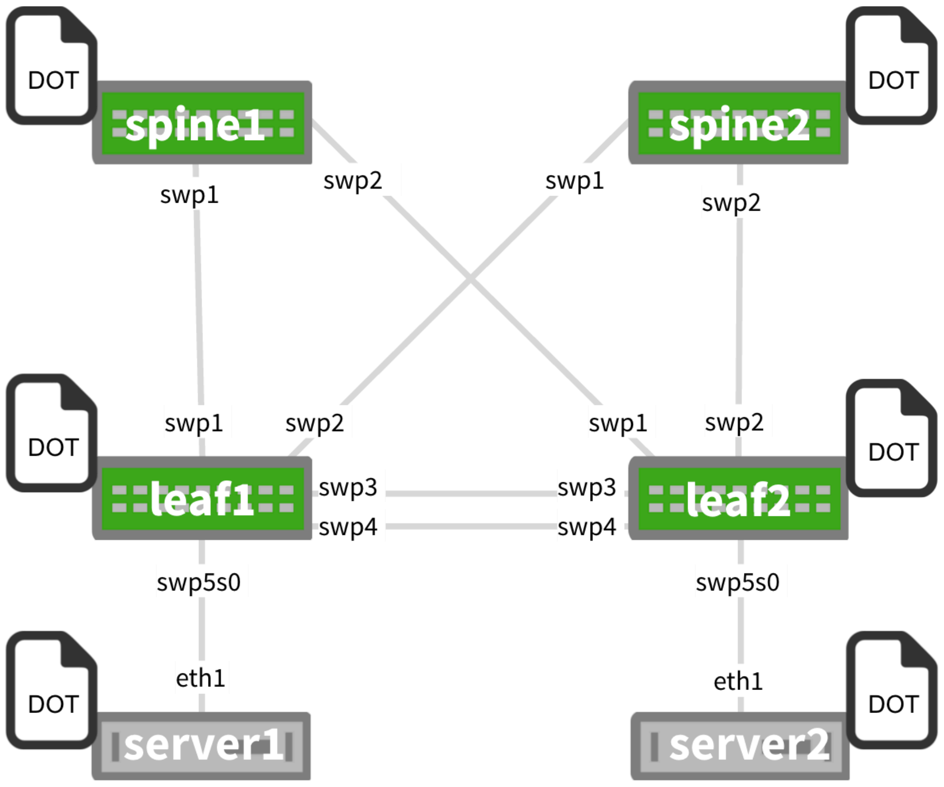
At startup, `ptmd` connects to `lldpd`, the LLDP daemon, over a Unix socket and retrieves the neighbor name and port information. It then compares the retrieved port information with the configuration information that it read from the topology file. If there is a match, then it is a PASS, else it is 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";
}
```



ptmd Scripts

`ptmd` executes scripts at `/etc/ptm.d/if-topo-pass` and `/etc/ptm.d/if-topo-fail` for each interface that goes through a change, 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 141).

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.

Further, to avoid errors when starting the `ptmd` process, make sure that `/etc/hosts` and `/etc/hostname` both reflect the hostname you are using in the `topology.dot` file.

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

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=port
descr"]
}
```

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_ty
pe=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.

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 operational state of the ports configured on the switch and information about BFD sessions from `ptmd`. `ptmctl` parses the CSV notifications sent by `ptmd`.

See `man ptmctl` for more information.

ptmctl Examples

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         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         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.

By simply opening the topology file with Graphviz, you can ensure that it is readable and that the file format is correct.

If you edit `topology.dot` file from a Windows system, be sure to double check the file formatting; there may be extra characters that keep the graph from working correctly.

Related Information

- 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, which 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 include:

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

The Cumulus RMP uses version 1 of the LAG protocol.

Contents

This chapter covers ...

Hash Distribution

Egress traffic through a bond is distributed to a slave based on a packet hash calculation, providing load balancing over the slaves; many conversation flows are distributed over all available slaves to load balance the total traffic. Traffic for a single conversation flow always hashes to the same slave.

The hash calculation uses packet header data to pick which slave to transmit the packet to:

- 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.

In a failover event, the hash calculation is adjusted to steer traffic over available slaves.

Creating a Bond

Bonds can be created and configured using the Network Command Line Utility (NCLU). Follow the steps below to create a new bond:

1. SSH into the switch.
2. Add a bond using the `net add bond` command, replacing `[bond-name]` with the name of the bond, and `[slaves]` with the list of slaves:

```
cumulus@switch:~$ net add bond [bond-name] bond-slaves [slaves]
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The name of the bond must be:

- Compliant with Linux interface naming conventions.
- Unique within the switch.

Configuration Options

The configuration options, and their default values, are listed in the table below.

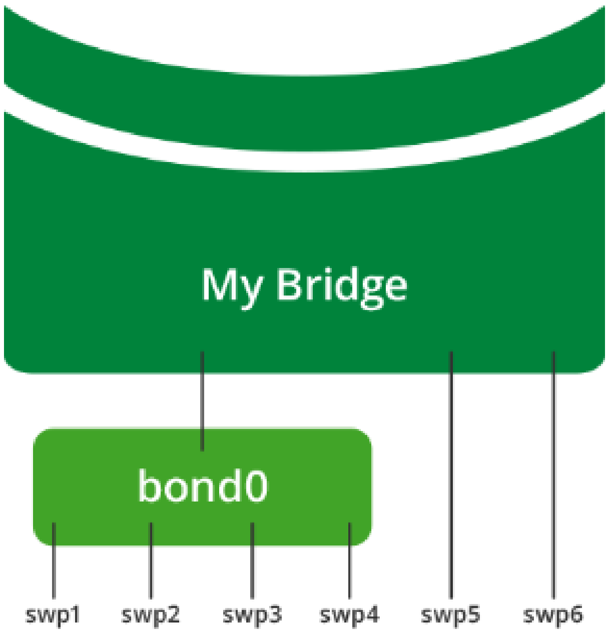
Each bond configuration option, except for `bond-slaves,` is set to the recommended value by default in Cumulus RMP. They should only be configured if a different setting is needed. For more information on configuration values, refer to the Related Information section below.

Configuration Option	Description	Default Value
`bond-mode`	The defined bonding mode. Cumulus RMP *only* supports IEEE 802.3ad link aggregation mode. This setting **must not** be changed.	`802.3ad`
`bond-slaves`	The list of slaves in the bond.	N/A
`bond-miimon`	Defines how often the link state of each slave is inspected for failures.	`100`
`bond-use-carrier`	Determines the link state.	`1`
`bond-xmit-hash-policy`	Hash method used to select the slave for a given packet. This setting **must not** be changed.	`layer3+4`
`bond-lacp-rate`	Sets the rate to ask the link partner to transmit LACP control packets.	`1`
		`1`

Configuration Option	Description	Default Value
`bond-min-links`	Defines the minimum number of links that must be active before the bond is put into service. A value greater than `1` is useful if higher level services need to ensure a minimum aggregate bandwidth level before activating a bond. Keeping `bond-min-links` set to `1` indicates the bond must have at least one active member. If the number of active members drops below the `bond-min-links` setting, the bond 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`.	

Example Configuration: Bonding 4 Slaves

In the following example, the front panel port interfaces swp1-swp4 are slaves in bond0, while swp5 and swp6 are not part of bond0.



Example Bond Configuration

The following commands create a bond with four slaves:

```
cumulus@switch:~$ net add bond bond0 address 10.0.0.1/30
cumulus@switch:~$ net add bond bond0 bond-slaves swp1-4
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create this code snippet in the `/etc/network/interfaces` file:

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

⚠ If you are intending that the bond become part of a bridge, you don't need to specify an IP address.

When networking is started on switch, bond0 is created as MASTER and interfaces swp1-swp4 come up in SLAVE mode, as seen in the `ip link show` command:

```
cumulus@switch:~$ ip link show
...

3: swp1: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc
pfifo_fast master bond0 state UP mode DEFAULT qlen 500
    link/ether 44:38:39:00:03:c1 brd ff:ff:ff:ff:ff:ff
4: swp2: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc
pfifo_fast master bond0 state UP mode DEFAULT qlen 500
    link/ether 44:38:39:00:03:c1 brd ff:ff:ff:ff:ff:ff
5: swp3: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc
pfifo_fast master bond0 state UP mode DEFAULT qlen 500
    link/ether 44:38:39:00:03:c1 brd ff:ff:ff:ff:ff:ff
6: swp4: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc
pfifo_fast master bond0 state UP mode DEFAULT qlen 500
    link/ether 44:38:39:00:03:c1 brd ff:ff:ff:ff:ff:ff

...

55: bond0: <BROADCAST,MULTICAST,MASTER,UP,LOWER_UP> mtu 1500 qdisc
noqueue state UP mode DEFAULT
    link/ether 44:38:39:00:03:c1 brd ff:ff:ff:ff:ff:ff
```

⚠ All slave interfaces within a bond have the same MAC address as the bond. Typically, the first slave added to the bond donates its MAC address as the bond MAC address, while 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.

Caveats and Errata

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

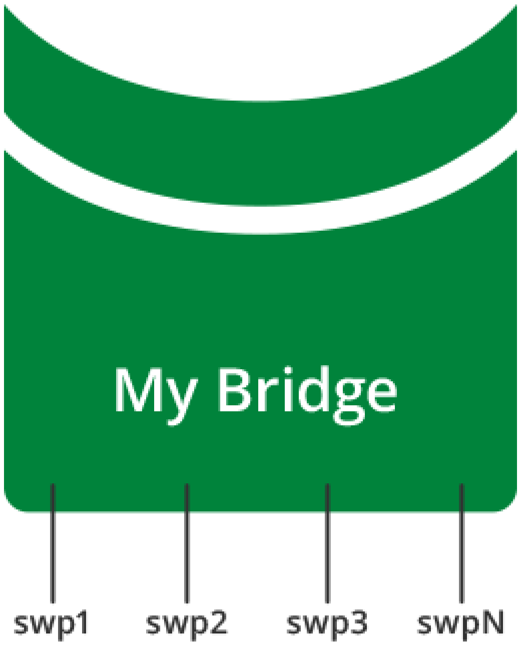
Related Information

- Linux Foundation - Bonding
- 802.3ad (Accessible writeup)
- Wikipedia - Link aggregation

Ethernet Bridging - VLANs

Ethernet bridges provide a means for hosts to communicate through layer 2, by connecting all of the physical and logical interfaces in the system into a single layer 2 domain. The bridge is a logical interface with a MAC address and an MTU (maximum transmission unit). The bridge MTU is the minimum MTU among all its members. 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 (see page 151).

Bridge members can be individual physical interfaces, bonds or logical interfaces that traverse an 802.1Q VLAN trunk.



Cumulus Networks recommends using *VLAN-aware mode* bridges, rather than *traditional mode* bridges. The bridge driver in Cumulus Linux is capable of VLAN filtering, which allows for configurations that are similar to incumbent network devices. While Cumulus Linux 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.

Cumulus Linux does not put all ports into a bridge by default.

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

Contents

This chapter covers ...

Creating a VLAN-aware Bridge

To learn about VLAN-aware bridges and how to configure them, read VLAN-aware Bridge Mode for Large-scale Layer 2 Environments (see page 152).

Creating a Traditional Mode Bridge

To create a traditional mode bridge, see Traditional Mode Bridges (see page 161).

Configuring Bridge MAC Addresses

The MAC address for a frame is learned when the frame enters the bridge via an interface. The MAC address is recorded in the bridge table, and the bridge forwards the frame to its intended destination by looking up the destination MAC address. The MAC entry is then maintained for a period of time defined by the `bridge-ageing` configuration option. If the frame is seen with the same source MAC address before the MAC entry age is exceeded, the MAC entry age is refreshed; if the MAC entry age is exceeded, the MAC address is deleted from the bridge table.

The following example output shows a MAC address table for the bridge:

```
cumulus@switch:~$ net show bridge macs
VLAN      Master    Interface    MAC                  TunnelDest
State      Flags    LastSeen
--------  --------  -----------  -----------------  ------------
---------  -------  -----------------
untagged  bridge    swp1         44:38:39:00:00:03
00:00:15
untagged  bridge    swp1         44:38:39:00:00:04
permanent           20 days, 01:14:03
```

Configuring an SVI (Switch VLAN Interface)

Bridges can be included as part of a routing topology after being assigned an IP address. This enables hosts within the bridge to communicate with other hosts outside of the bridge, via a *switch VLAN interface* (SVI), which provides layer 3 routing. The IP address of the bridge is typically from the same subnet as the bridge's member hosts.

⚠ 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 unreachable.

To configure the SVI, use NCLU:

```
cumulus@switch:~$ net add bridge bridge-ports swp1-2
cumulus@switch:~$ net add bridge bridge-stp on
cumulus@switch:~$ net add vlan-interface vlan10 address 10.100.100.1
/24
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following SVI configuration in the `/etc/network/interfaces` file:

```
auto bridge
iface bridge
    bridge-ports swp1 swp2
    bridge-vids 10
    bridge-vlan-aware yes
    bridge-stp on

auto vlan10
iface vlan10
    address 10.100.100.1/24
    vlan-id 10
    vlan-raw-device bridge
```

Alternately, you can use the *bridge.VLAN-ID* naming convention for the SVI. The following example configuration can be manually created in the `/etc/network/interfaces` file, which functions identically to the above configuration:

```
auto bridge
iface bridge
    bridge-ports swp1 swp2
    bridge-vids 10
    bridge-vlan-aware yes
    bridge-stp on

bridge.10
bridge.10
    address 10.100.100.1/24
```

Caveats and Errata

- A bridge cannot contain multiple subinterfaces of the **same** port. Attempting this configuration results in an error.
- In environments where both VLAN-aware and traditional bridges are in use, if a traditional bridge has a subinterface of a bond that is a normal interface in a VLAN-aware bridge, the bridge will be flapped when the traditional bridge's bond subinterface is brought down.

Related Information

- Linux Foundation - Bridges
- Linux Foundation - VLANs
- Linux Journal - Linux as an Ethernet Bridge
- Comparing Traditional Bridge Mode to VLAN-aware Bridge Mode

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 (see page 161), 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 113). 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.

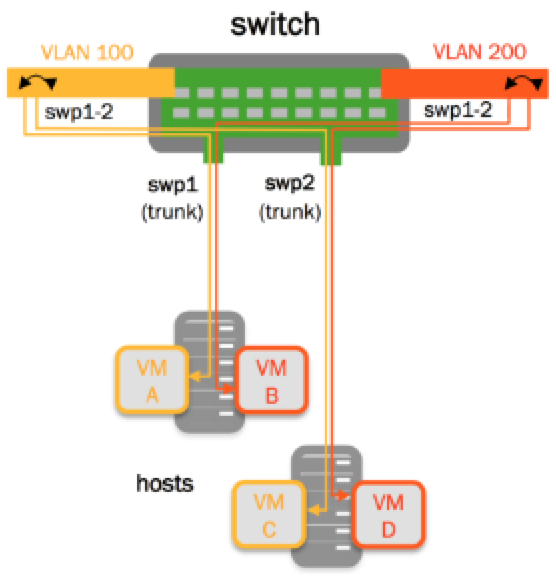
You can configure both VLAN-aware and traditional mode bridges on the same network in Cumulus RMP; however you should not have more than one VLAN-aware bridge on a given switch.

Contents

This chapter covers ...

Configuring a VLAN-aware Bridge

VLAN-aware bridges can be configured with the Network Command Line Utility (NCLU). The example below shows the NCLU commands required to create a VLAN-aware bridge configured for STP, that contains two switch ports, and includes 3 VLANs — the tagged VLANs 100 and 200 and the untagged (native) VLAN of 1:



```
cumulus@switch:~$ net add bridge
bridge-ports swp1-2 bridge-vids
100,200 bridge-pvid 1 bridge-stp
on
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
cumulus@switch:~$ net show
configuration files

...

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

...
```

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 Identifer*. 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**.

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
```

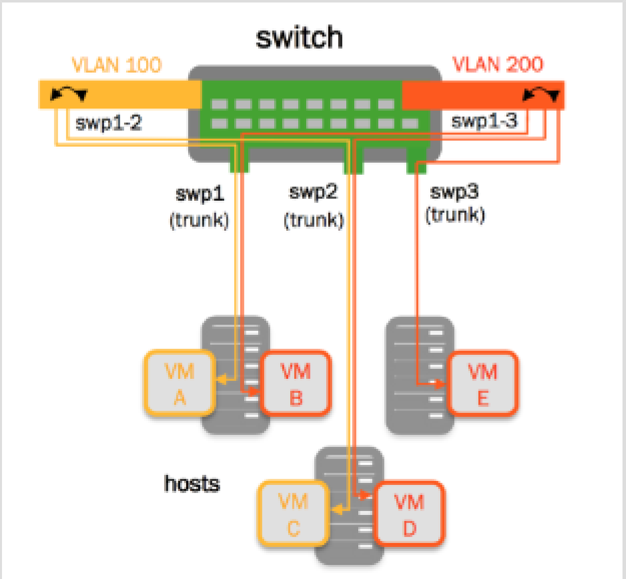
```
auto bridge
iface bridge
    bridge-vlan-
aware yes
    bridge-ports
swp1 swp2
    bridge-vids
100 200
    bridge-stp on
```

Example Configurations

VLAN Filtering/VLAN Pruning

By default, the bridge port inherits the bridge VIDs. A port's configuration can override the bridge VIDs, by using the `bridge-vids` attribute:

```
cumulus@switch:~$ net add bridge
bridge-ports swp1-3 bridge-vids
100,200 bridge-pvid 1 bridge-stp
on
cumulus@switch:~$ net add
interface swp3 bridge-vids 200
```



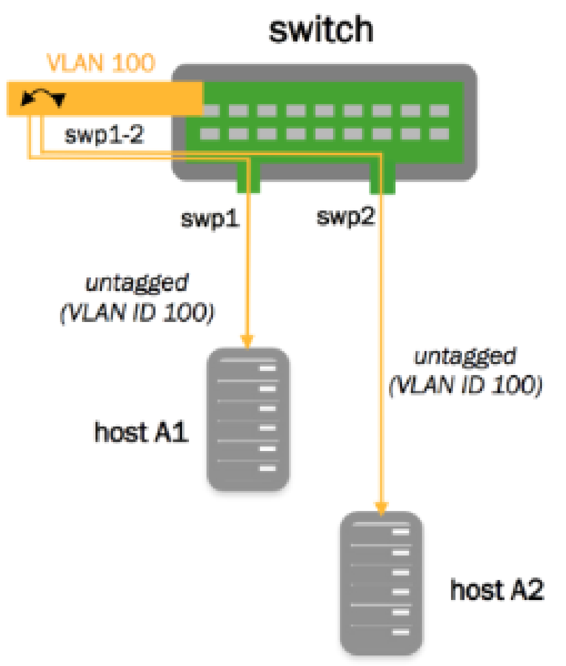
```
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
cumulus@switch:~$ net show
configuration files

...
auto bridge
iface bridge
    bridge-vlan-aware yes
    bridge-ports swp1 swp2 swp3
    bridge-vids 100 200
    bridge-pvid 1
    bridge-stp on

auto swp3
iface swp3
  bridge-vids 200
```

Untagged/Access Ports

Access ports ignore all tagged packets. In the configuration below, swp1 and swp2 are configured as access ports, while all untagged traffic goes to VLAN 100 as specified in the example below:



```
cumulus@switch:~$ net add bridge
bridge-ports swp1-2 bridge-vids
100,200 bridge-pvid 1 bridge-stp
on
cumulus@switch:~$ net add
interface swp1 bridge-access 100
cumulus@switch:~$ net add
interface swp2 bridge-access 100
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
cumulus@switch:~$ net show
configuration files

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

auto swp1
iface swp1
```

	`bridge-access 100`
	
	`auto swp2`
	`iface swp2`
	`bridge-access 100`

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` to the switch port. 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`:

```
cumulus@switch:~$ net add interface swp9 bridge-allow-untagged no
```

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

```
cumulus@switch$ net show bridge vlan
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.

```
cumulus@switch:~$ net add vlan-interface vlan100 address 192.168.10.1/
24
cumulus@switch:~$ net add vlan-interface vlan100 address 2001:db8::1/3
2
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following configuration in the `/etc/network/interfaces` file:

```
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
```

⚠ In the above configuration, if your switch is configured for multicast routing, you do not need to specify **`bridge-igmp-querier-src`**, as there is no need for a static IGMP querier configuration on the switch. Otherwise, the static IGMP querier configuration helps to probe the hosts to refresh their IGMP reports.

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

```
cumulus@switch:~$ net add vlan 1-2000
```

Configuring Multiple Ports in a Range

The `bridge-ports` attribute takes a range of numbers. The "swp1-52" in the example below indicates that swp1 through swp52 are part of the bridge, which is a shortcut that saves you from enumerating each port individually:

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

These commands create the following configuration in the `/etc/network/interfaces` file:

```
auto bridge
iface bridge
      bridge-vlan-aware yes
      bridge-ports swp1 swp2 swp3 ... swp51 swp52
      bridge-stp on
      bridge-vids 310 700 707 712 850 910
```

Access Ports and Pruned VLANs

The following example configuration contains an access port and switch port that are *pruned*; that is, they only send and receive traffic tagged to/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.

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

...

# 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
```

Large Bond Set Configuration

The configuration below demonstrates a VLAN-aware bridge with a large set of bonds. The bond configurations are generated from a Mako template.

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

...
#
# 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
```

Caveats and Errata

- **Spanning Tree Protocol (STP):** VLAN-aware mode supports a single instance of STP across all VLANs, as STP is enabled on a per-bridge basis. 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` remains the user space protocol daemon.
  Cumulus RMP supports Rapid Spanning Tree Protocol (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 1000 VLANs by default; this range is 3000-3999. The reserved range can be modified if it conflicts with any user-defined VLANs, as long the new range is a contiguous set of VLANs with IDs anywhere between 2 and 4094, and the minimum size of the range is 300 VLANs.
  To configure the reserved range:
  1. Open `/etc/cumulus/switchd.conf` in a text editor.
  2. Uncomment the following line, specify the new range, and save the file.

     ```
     resv_vlan_range
     ```

  3. Restart `switchd` so the new range takes effect.

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

     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 149) can utilize VLAN translation.
- **Converting bridges between supported modes:** Traditional mode bridges cannot be automatically converted to/from a VLAN-aware bridge. The original configuration must be deleted, and all member switch ports must be brought down, then a new bridge can be created.

Traditional Mode Bridges

Cumulus Networks recommends you use a VLAN-aware bridge (see page 152) on your switch. You use traditional mode bridges only if you need to run more than one bridge on the switch or if you need to use PVSTP+.

Contents

This chapter covers ...

Creating a Traditional Mode Bridge

You configure traditional mode bridges in `/etc/network/interfaces` file. To create a traditional mode bridge:

1. Open the `/etc/network/interfaces` file in a text editor.
2. Add a new stanza to create the bridge, and save the file. The example below creates a bridge with STP enabled and the MAC address ageing timer configured to a lower value than the default:

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

Configuration Option	Description	Default Value
bridge-ports	List of logical and physical ports belonging to the logical bridge.	N/A
bridge-ageing	Maximum amount of time before a MAC addresses learned on the bridge expires from the bridge MAC cache.	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 113).	off

The name of the bridge must be:

- Compliant with Linux interface naming conventions.
- Unique within the switch.

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.

3. Reload the network configuration using the `ifreload` command:

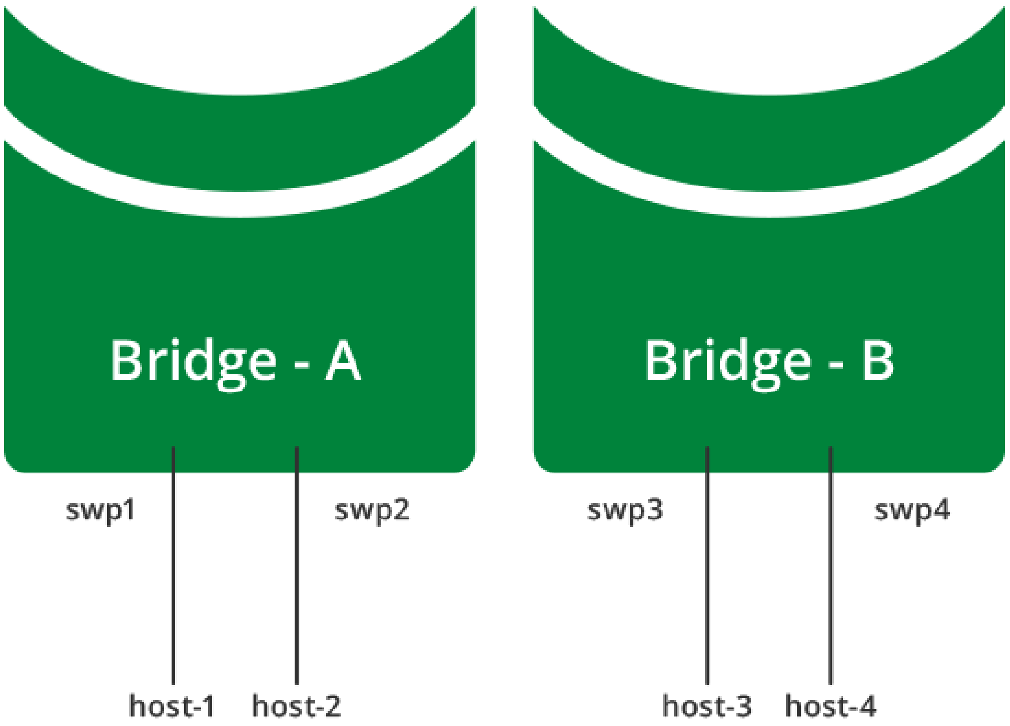
```
cumulus@switch:~$ sudo ifreload -a
```

You can configure multiple bridges, in order to logically divide a switch into multiple layer 2 domains. This allows for hosts to communicate with other hosts in the same domain, while separating them fro hosts in other domains.

You can create only one VLAN-aware bridge on a switch.

The diagram below shows a multiple bridge configuration, where host-1 and host-2 are connected to bridge-A, while host-3 and host-4 are connected to bridge-B. This means that:

- host-1 and host-2 can communicate with each other.
- host-3 and host-4 can communicate with each other.
- host-1 and host-2 cannot communicate with host-3 and host-4.



This example configuration looks like this in the `/etc/network/interfaces` file:

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

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

Using Trunks in Traditional Bridge 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.

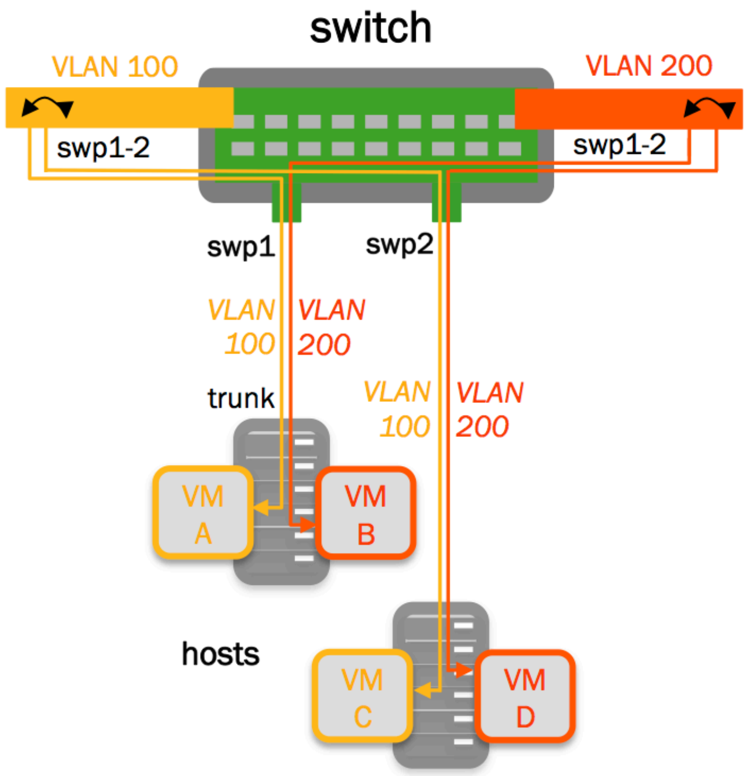
802.1Q also identifies an *untagged* frame as belonging to the *native* VLAN (most network devices default their native VLAN to 1). The concept of native, non-native, tagged or untagged has generated confusion due to mixed terminology and vendor-specific implementations. Some clarification is in order:

- 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



To create the above example, add the following configuration to the `/etc/network/interfaces` file:

```
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
```

VLAN Tagging Examples

You can find more examples of VLAN tagging in this chapter (see page 165).

VLAN Tagging

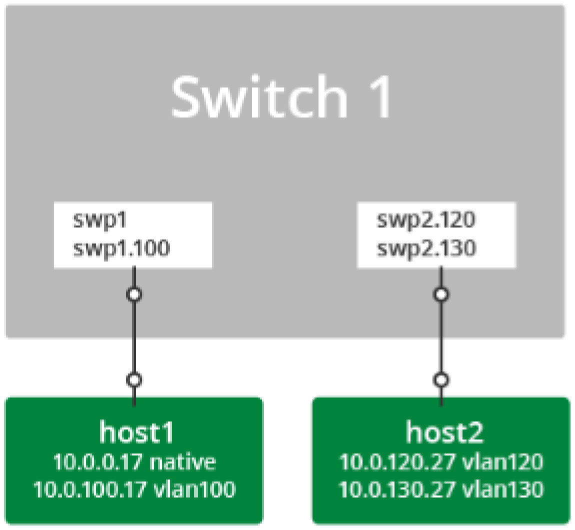
This article shows two examples of VLAN tagging (see page), one basic and one more advanced. They both demonstrate the streamlined interface configuration from `ifupdown2`.

Contents

This chapter covers ...

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, edit `/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
```

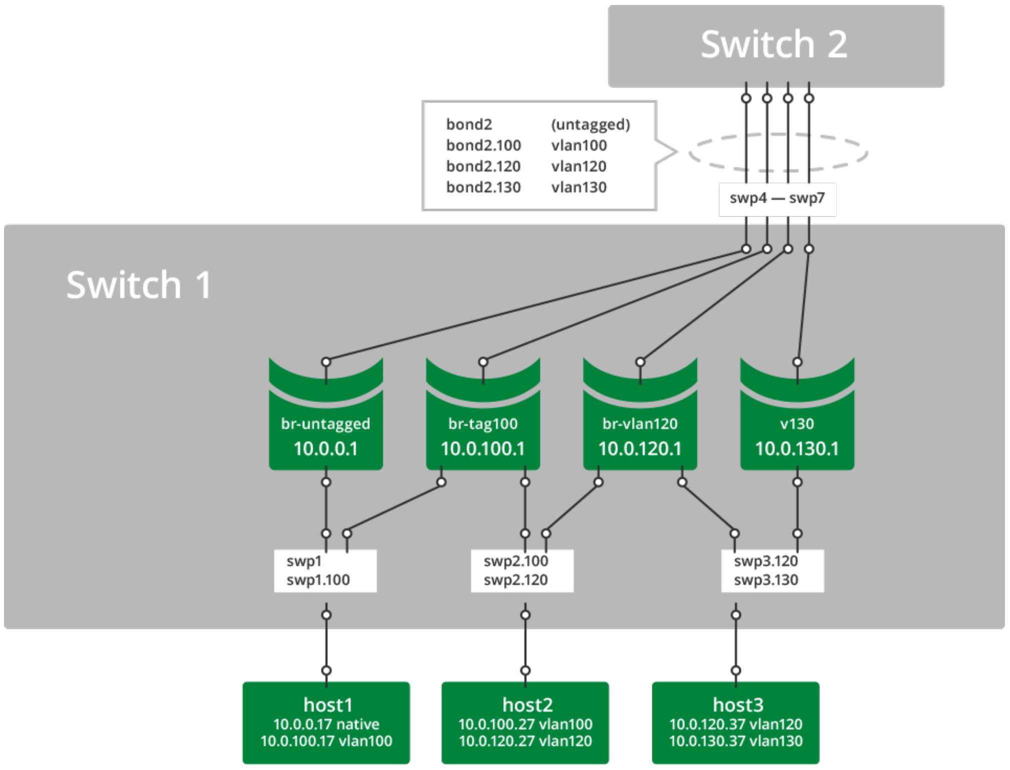
```
auto swp2
iface swp2

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

# configure the bridges  - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - -

auto br-untagged
iface br-untagged
   address 10.0.0.1/24
   bridge-ports swp1 bond2
   bridge-stp on

auto br-tag100
iface br-tag100
   address 10.0.100.1/24
   bridge-ports swp1.100 swp2.100 bond2.100
   bridge-stp on

auto br-vlan120
iface br-vlan120
   address 10.0.120.1/24
   bridge-ports swp2.120 swp3.120 bond2.120
   bridge-stp on

auto v130
iface v130
    address 10.0.130.1/24
    bridge-ports swp3.130 bond2.130
    bridge-stp on

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

To verify:

```
cumulus@switch:~$ sudo mstpctl showbridge br-tag100
br-tag100 CIST info
  enabled         yes
  bridge id       8.000.44:38:39:00:32:8B
  designated root 8.000.44:38:39:00:32:8B
  regional root   8.000.44:38:39:00:32:8B
  root port       none
  path cost     0          internal path cost   0
  max age       20         bridge max age       20
  forward delay 15         bridge forward delay 15
  tx hold count 6          max hops             20
  hello time    2          ageing time          300
  force protocol version     rstp
  time since topology change 333040s
  topology change count      1
  topology change            no
  topology change port        swp2.100
  last topology change port   None
```

```
cumulus@switch:~$ sudo mstpctl showportdetail br-tag100  | grep -B 2
state
br-tag100:bond2.100 CIST info
  enabled            yes                      role
Designated
  port id            8.003                    state
forwarding
--
br-tag100:swp1.100 CIST info
  enabled            yes                      role
Designated
  port id            8.001                    state
forwarding
--
br-tag100:swp2.100 CIST info
  enabled            yes                      role
Designated
  port id            8.002                    state
forwarding
```

```
cumulus@switch:~$ cat /proc/net/vlan/config
VLAN Dev name     | VLAN ID
Name-Type: VLAN_NAME_TYPE_RAW_PLUS_VID_NO_PAD
bond2.100       | 100  | bond2
bond2.120       | 120  | bond2
bond2.130       | 130  | bond2
swp1.100        | 100  | swp1
swp2.100        | 100  | swp2
swp2.120        | 120  | swp2
swp3.120        | 120  | swp3
swp3.130        | 130  | swp3
```

```
cumulus@switch:~$ cat /proc/net/bonding/bond2
Ethernet Channel Bonding Driver: v3.7.1 (April 27, 2011)

Bonding Mode: IEEE 802.3ad Dynamic link aggregation
Transmit Hash Policy: layer3+4 (1)
MII Status: up
MII Polling Interval (ms): 100
Up Delay (ms): 0
Down Delay (ms): 0

802.3ad info
```

```
LACP rate: fast
Min links: 0
Aggregator selection policy (ad_select): stable
Active Aggregator Info:
    Aggregator ID: 3
    Number of ports: 4
    Actor Key: 33
    Partner Key: 33
    Partner Mac Address: 44:38:39:00:32:cf

Slave Interface: swp4
MII Status: up
Speed: 10000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 44:38:39:00:32:8e
Aggregator ID: 3
Slave queue ID: 0

Slave Interface: swp5
MII Status: up
Speed: 10000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 44:38:39:00:32:8f
Aggregator ID: 3
Slave queue ID: 0

Slave Interface: swp6
MII Status: up
Speed: 10000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 44:38:39:00:32:90
Aggregator ID: 3
Slave queue ID: 0

Slave Interface: swp7
MII Status: up
Speed: 10000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 44:38:39:00:32:91
Aggregator ID: 3
Slave queue ID: 0
```

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 swp9.10
0
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 152) 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`:

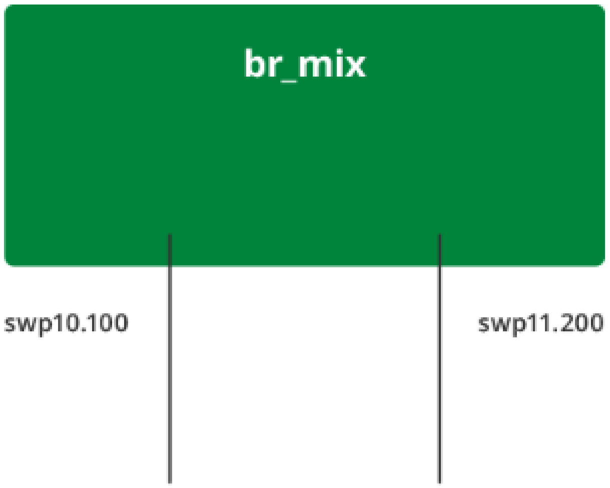
```
cumulus@switch:~$ echo net.bridge.bridge-allow-multiple-vlans = 1 |
sudo tee /etc/sysctl.d/multiple_vlans.conf
net.bridge.bridge-allow-multiple-vlans = 1
cumulus@switch:~$ sudo sysctl -p /etc/sysctl.d/multiple_vlans.conf
net.bridge.bridge-allow-multiple-vlans = 1
```

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
```



Routing

This chapter discusses routing on switches running Cumulus RMP.

Contents

This chapter covers ...

Managing Static Routes

You manage static routes using NCLU or the Cumulus Linux `ip route` command. The routes are added to the Quagga routing table, and are then updated into the kernel routing table as well.

To add a static route, run:

```
cumulus@switch:~$ net add ip route 203.0.113.0/24 198.51.100.2
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands create the following configuration in the `/etc/quagga/Quagga.conf` file:

```
!
ip route 203.0.113.0/24 198.51.100.2
!
```

To delete a static route, run:

```
cumulus@switch:~$ net del ip route 203.0.113.0/24 198.51.100.2
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

To view static routes, run:

```
cumulus@switch:~$ net show route static
RIB entry for static
====================
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, P - PIM, T - Table,
       > - selected route, * - FIB route
S>* 203.0.113.0/24 [1/0] via 198.51.100.2, swp3
```

Static Routing via ip route

A static route can also be created by adding `post-up ip route add` command to a switch port configuration. For example:

```
cumulus@switch:~$ net add interface swp3 address 198.51.100.1/24 post-
up ip route add 203.0.113.0/24 via 198.51.100.2
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

These commands produce the following configuration in the `/etc/network/interfaces` file:

```
auto swp3
iface swp3
    address 198.51.100.1/24
    post-up ip route add 203.0.113.0/24 via 198.51.100.2
```

If an IPv6 address is assigned to a DOWN interface, the associated route is still installed into the routing table. The type of IPv6 address doesn't matter: link local, site local and global all exhibit the same problem.

If the interface is bounced up and down, then the routes are no longer in the route table.

The `ip route` command allows manipulating the kernel routing table directly from the Linux shell. See `man ip(8)` for details. `quagga` monitors the kernel routing table changes and updates its own routing table accordingly.

To display the routing table:

```
cumulus@switch:~$ ip route show
default via 10.0.1.2 dev eth0
10.0.1.0/24 dev eth0  proto kernel  scope link  src 10.0.1.52
192.0.2.0/24 dev swp1  proto kernel  scope link  src 192.0.2.12
192.0.2.10/24 via 192.0.2.1 dev swp1  proto zebra  metric 20
192.0.2.20/24  proto zebra  metric 20
     nexthop via 192.0.2.1  dev swp1 weight 1
     nexthop via 192.0.2.2  dev swp2 weight 1
192.0.2.30/24 via 192.0.2.1 dev swp1  proto zebra  metric 20
192.0.2.40/24 dev swp2  proto kernel  scope link  src 192.0.2.42
192.0.2.50/24 via 192.0.2.2 dev swp2  proto zebra  metric 20
192.0.2.60/24 via 192.0.2.2 dev swp2  proto zebra  metric 20
```

```
192.0.2.70/24  proto zebra  metric 30
     nexthop via 192.0.2.1  dev swp1 weight 1
     nexthop via 192.0.2.2  dev swp2 weight 1
198.51.100.0/24 dev swp3  proto kernel  scope link  src 198.51.100.1
198.51.100.10/24 dev swp4  proto kernel  scope link  src 198.51.100.11
198.51.100.20/24 dev br0  proto kernel  scope link  src 198.51.100.21
```

Caveats and Errata

Adding IPv6 Default Route with src Address on eth0 Fails without Adding Delay

Attempting to install an IPv6 default route on eth0 with a source address fails at reboot or when running `ifup` on eth0.

The first execution of `ifup -dv` returns this warning and does not install the route:

```
cumulus@switch:~$ sudo ifup -dv eth0
warning: eth0: post-up cmd '/sbin/ip route add default via 2001:620:
5ca1:160::1 /
src 2001:620:5ca1:160::45 dev eth0' failed (RTNETLINK answers:
Invalid argument)<<<<<<<<<<
```

Running `ifup` a second time on eth0 successfully installs the route.

There are two ways you can work around this issue.

- Add a sleep 2 to the eth0 interface in `/etc/network/interfaces`:

```
cumulus@switch:~$ net add interface eth0 address 2001:620:5ca1:16
0::45/64 post-up /bin/sleep 2s
cumulus@switch:~$ net add interface eth0 post-up /sbin/ip route
add default via 2001:620:5ca1:160::1 src 2001:620:5ca11:160::45
dev eth0
```

- Exclude the `src` parameter to the `ip route add` that causes the need for the delay. If the `src` parameter is removed, the route is added correctly.

```
cumulus@switch:~$ net add interface eth0 post-up /sbin/ip route
add default via 2001:620:5ca1:160::1 dev eth0
```

```
cumulus@switch:~$ ifdown eth0
Stopping NTP server: ntpd.
Starting NTP server: ntpd.
cumulus@switch:~$ ip -6 r s
cumulus@switch:~$ ifup eth0
```

```
Stopping NTP server: ntpd.
Starting NTP server: ntpd.
cumulus@switch:~$ ip -6 r s
2001:620:5ca1:160::/64 dev eth0  proto kernel  metric 256
fe80::/64 dev eth0  proto kernel  metric 256
default via 2001:620:5ca1:160::1 dev eth0  metric 1024
```

Related Information

- Linux IP - ip route command
- Quagga docs - static route commands

Management VRF

Management VRF provides a separation between the out-of-band management network and the in-band data plane network. For all VRFs, the *main* routing table is the default table for all of the data plane switch ports. With management VRF, a second table, *mgmt*, is used for routing through eth0.

Cumulus RMP only supports eth0 as the management interface. VLAN subinterfaces, bonds, bridges and the front panel switch ports are not supported as management interfaces.

When management VRF is enabled, logins to the switch are set into the management VRF context. IPv4 and IPv6 networking applications run by an administrator communicate out the management network by default. This default context does not impact services run through `systemd` and the `systemctl` command, and does not impact commands examining the state of the switch; for example, using the `ip` command to list links, neighbors or routes.

The Hurricane2 ASIC used by the Penguin Arctica 4804IP-RMP switch that runs Cumulus RMP does not support *VRF* (virtual routing tables and forwarding).

Contents

This chapter covers ...

Enabling Management VRF

To enable management VRF on eth0, complete the following steps:

1. Configure management VRF on the switch.

Example Management VRF Configuration

The example NCLU commands below create a VRF called *mgmt*:

> The management VRF must be named `mgmt` to differentiate from a data plane VRF.

```
cumulus@switch:~$ net add vrf mgmt address 127.0.0.1/8
cumulus@switch:~$ net add vrf mgmt vrf-table auto
cumulus@switch:~$ net add interface eth0 vrf mgmt
cumulus@switch:~$ net pending
cumulus@switch:~$ net commit
```

The NCLU commands above create the following snippets in `/etc/networking/interfaces`:

```
auto mgmt
iface mgmt
    address 127.0.0.1/8
    vrf-table auto

auto eth0
iface eth0 inet dhcp
    vrf mgmt
```

2. Reboot the switch to activate the mgmt VRF:

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

Bringing the Management VRF Up after Downing It with ifdown

If you take down the management VRF using `ifdown`, to bring it back up you need to do one of two things:

- Use `ifup --with-depends <vrf>`
- Use `ifreload -a`

For example:

```
cumulus@switch:~$ sudo ifdown mgmt
cumulus@switch:~$ sudo ifup --with-depends mgmt
```

Enabling NTP

To enable NTP to run in the mgmt VRF:

1. Configure the *mgmt* VRF as described in the Enabling Management VRF section above (see page 178).
2. If NTP is running, stop the service.

   ```
   cumulus@switch:~$ sudo systemctl stop ntp.service
   ```

   By default, NTP is running in the default VRF, and to automatically start when the system boots; the NTP service needs to be stopped, disabled, and then restarted once the **mgmt** VRF is configured.

3. Disable NTP from automatically starting in the default VRF:

   ```
   cumulus@switch:~$ sudo systemctl disable ntp.service
   ```

4. Start NTP in the mgmt VRF.

   ```
   cumulus@switch:~$ sudo systemctl start ntp@mgmt
   ```

5. Verify that NTP peers are active.

   ```
   cumulus@switch:~$ ntpq -pn
        remote           refid      st t when poll reach   delay
   offset  jitter
   =================================================================
   =============
   *38.229.71.1     204.9.54.119     2 u   42   64  377   31.275   -
   0.625   3.105
   -104.131.53.252  209.51.161.238   2 u   47   64  377   16.381   -
   5.251   0.681
   +45.79.10.228    200.98.196.212   2 u   44   64  377   42.998
   0.115   0.585
   +74.207.240.206  127.67.113.92    2 u   43   64  377   73.240   -
   1.623   0.320
   ```

6. Enable ntp@mgmt so it starts when the switch boots:

```
cumulus@switch:~$ sudo systemctl enable ntp@mgmt
```

Enabling snmpd

To enable `snmpd` to run in the mgmt VRF:

1. Configure the mgmt VRF as described in the Enabling Management VRF section above (see page 178)
.
2. Stop the `snmpd` daemon if it is running.

   ```
   cumulus@switch:~$ sudo systemctl stop snmpd.service
   ```
3. Disable `snmpd` to ensure it does not try to start in the default VRF if the system is rebooted.

   ```
   cumulus@switch:~$ sudo systemctl disable snmpd.service
   ```
4. Start `snmpd` in the the mgmt VRF:

   ```
   cumulus@switch:~$ sudo systemctl start snmpd@mgmt
   ```
5. Enable snmpd@mgmt so it starts when the switch boots:

   ```
   cumulus@switch:~$ sudo systemctl enable snmpd@mgmt
   ```

Using ping or traceroute

By default, issuing a `ping` or `traceroute` assumes the packet should be sent to the dataplane network (the main routing table). If you wish to use `ping` or `traceroute` on the management network, use the `-I` flag for ping and `-i` for `traceroute`.

```
cumulus@switch:~$ ping -I mgmt
```

Or:

```
cumulus@switch:~$ sudo traceroute -i mgmt
```

SNMP Traps Use eth0 Only

SNMP cannot use a switch port to send data. For any SNMP traps, this traffic gets sent out to eth0. Cumulus Networks plans to support switch ports in the future.

For SNMP, this restriction only applies to traps. SNMP polling is not affected.

Using SSH within a Management VRF Context

If you SSH to the switch through a switch port, it works as expected. If you need to SSH from the device out a switch port, use `vrf exec default ssh <ip_address_of_swp_port>`. For example:

```
cumulus@switch:~$ sudo vrf exec default ssh 10.23.23.2 10.3.3.3
```

Viewing the Routing Tables

When you look at the routing table with `ip route show`, you are looking at the switch port (*main*) table. You can also see the data plane routing table with `ip route show table main`.

To look at information about eth0 (the management routing table), use `ip route show table mgmt`.

```
cumulus@switch:~$ net show route table mgmt
default via 192.168.0.1 dev eth0


cumulus@switch:~$ net show route
default via 10.23.23.3 dev swp17  proto zebra  metric 20
10.3.3.3 via 10.23.23.3 dev swp17
10.23.23.0/24 dev swp17  proto kernel  scope link  src 10.23.23.2
192.168.0.0/24 dev eth0  proto kernel  scope link  src 192.168.0.11
```

Viewing a Single Route

Note that if you use `ip route get` to return information about a single route, the command resolves over the *mgmt* table by default. To get information about the route in the switching silicon, use:

```
cumulus@switch:~$ net show route <addr>
```

To get the route for any VRF, run:

```
cumulus@switch:~$ net show route vrf mgmt <addr>
```

Using the mgmt Interface Class

In `ifupdown2` interface classes (see page 83) are used to create a user-defined grouping for interfaces. The special class *mgmt* is available to separate the switch's management interfaces from the data interfaces. This allows you to manage the data interfaces by default using `ifupdown2` commands. Performing operations on the *mgmt* interfaces requires specifying the `--allow-mgmt` option, which prevents inadvertent outages on the management interfaces. Cumulus RMP by default brings up all interfaces in both the *auto* (default) class and the *mgmt* interface class when the switch boots.

You configure the management interface in `/etc/network/interfaces`. In the example below, the management interface, eth0, and the mgmt VRF stanzas are added to the *mgmt* interface class:

```
auto lo
iface lo inet loopback

allow-mgmt eth0
iface eth0 inet dhcp
    vrf mgmt

allow-mgmt mgmt
iface mgmt
    address 127.0.0.1/8
    vrf-table auto
```

When you run `ifupdown2` commands against the interfaces in the mgmt class, include --allow=mgmt with the commands. For example, to see which interfaces are in the mgmt interface class, run:

```
cumulus@switch:~$ ifquery l --allow=mgmt
eth0
mgmt
```

To reload the configurations for interfaces in the mgmt class, run:

```
cumulus@switch:~$ sudo ifreload --allow=mgmt
```

However, you can still bring the management interface up and down using `ifup eth0` and `ifdown eth0`.

Management VRF and DNS

Cumulus RMP supports both DHCP and static DNS entries over management VRF through IP FIB rules. These rules are added to direct lookups to the DNS addresses out of the management VRF. However, nameservers configured through DHCP are automatically updated, while statically configured nameservers (configured in `/etc/resolv.conf`) only get updated when you run `ifreload -a`.

Because DNS lookups are forced out of the management interface using FIB rules, this could affect data plane ports if there are overlapping addresses.

Monitoring and Troubleshooting

This chapter introduces monitoring and troubleshooting Cumulus RMP.

Contents

This chapter covers ...

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.

Getting General System Information

Two commands are helpful for getting general information about the switch and the version of Cumulus Linux you are running. These are helpful with system diagnostics and if you need to submit a support request to Cumulus Networks.

For information about the version of Cumulus Linux running on the switch, run `net show version`, which displays the contents of `/etc/lsb-release`:

```
cumulus@switch:~$ net show version
DISTRIB_ID="Cumulus Linux"
DISTRIB_RELEASE=3.2.0
DISTRIB_DESCRIPTION="Cumulus Linux 3.2.0"
```

For general information about the switch, run `net show system`:

```
cumulus@switch:~$ net show system
Cumulus Version 3.2.0
Build: Cumulus Linux 3.2.0
Uptime: 6 days, 1:00:52
```

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 `switchd`.

Directory	Description
etc	Is a replica of the switch's `/etc` directory. `/etc` 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 `/var/log` directory. Most Cumulus RMP log files are located in this directory. Notable log files include `switchd.log` and `daemon.log` log files, and `syslog`. For more information, read this knowledge base article.
proc	Is a replica of the switch's `/proc` directory. In Linux, `/proc` 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.
support	Is a set of files containing further system information, which is obtained by `cl-support` running commands such as `ps -aux`, `netstat -i`, 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 196).

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 *imfile*	Enables the `rsyslog` module to watch file contents.
$InputFileName	The file to be sent to the `syslog` server. In this example, you are going to send changes made to `/var/log/switchd.log` to the `syslog` server.
$InputFileStateFile	This is used by `rsyslog` to track state of the file being monitored. This must be unique for each file being monitored.
$InputFileTag	Defines the `syslog` tag that will precede the `syslog` messages. In this example, all logs are prefaced with *switchd*.
$InputFileSeverity	Defines the logging severity level sent to the `syslog` server.
$InputFileFacility	Defines the logging format. *local7* is common.
$InputFilePollInterval	Defines how frequently in seconds `rsyslog` 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 `syslog` 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.

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 228)
- watchdog

Contents

This chapter covers ...

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: 114
TLV Name             Code Len Value
-------------------- ---- --- -----
Product Name         0x21   4 4804
Part Number          0x22  14 R0596-F0009-00
Device Version       0x26   1 2
Serial Number        0x23  19 D1012023918PE000012
Manufacture Date     0x25  19 10/09/2013 20:39:02
Base MAC Address     0x24   6 00:E0:EC:25:7B:D0
MAC Addresses        0x2A   2 53
Vendor Name          0x2D  17 Penguin Computing
Label Revision       0x27   4 4804
Manufacture Country  0x2C   2 CN
CRC-32               0xFE   4 0x96543BC5
(checksum valid)
```

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
tmp75-i2c-6-48
Adapter: i2c-1-mux (chan_id 0)
temp1:         +39.0 C  (high = +75.0 C, hyst = +25.0 C)

tmp75-i2c-6-49
Adapter: i2c-1-mux (chan_id 0)
temp1:         +35.5 C  (high = +75.0 C, hyst = +25.0 C)

ltc4215-i2c-7-40
Adapter: i2c-1-mux (chan_id 1)
in1:          +11.87 V
in2:          +11.98 V
power1:        12.98 W
curr1:         +1.09 A

max6651-i2c-8-48
Adapter: i2c-1-mux (chan_id 2)
fan1:         13320 RPM  (div = 1)
fan2:         13560 RPM
```

Command Options

Usage: `sensors [OPTION]... [CHIP]...`

Option	Description
-c, –config-file	Specify a config file; use - after `-c` to read the config file from `stdin`; by default, `sensors` references the configuration file in `/etc/sensors.d/`.
-s, –set	Executes set statements in the config file (root only); `sensors -s` is run once at boot time and applies all the settings to the boot drivers.

Option	Description
-f, –fahrenheit	Show temperatures in degrees Fahrenheit.
-A, –no-adapter	Do not show the adapter for each chip.
–bus-list	Generate bus statements for `sensors.conf`.

If `[CHIP]` is not specified in the command, all chip 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 228).

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 `cpld` registers. `smond` utilizes these registers to read all sources, which impacts the health of the system unit, determines the unit's health, and updates the system LEDs.

Use `smonctl` to display sensor information for the various system units:

```
cumulus@switch:~$ 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 `watchdog` Debian package. `wd_keepalive` writes to a file called `/dev/watchdog` periodically to keep the switch from resetting, at least once per minute. Each write delays the reboot time by another minute. After one minute of inactivity where `wd_keepalive` doesn't write to `/dev/watchdog`, the switch resets itself.

The watchdog is enabled by default on 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`.

Related Information

- packages.debian.org/search?keywords=lshw
- 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
```

This chapter covers ...

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 `tar.gz` output.	This is the hostname of the switch where `cl-support` 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.
-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 `cl-support` script that generated the `cl_support` file. It is copied so Cumulus Networks knows exactly which files were included and which weren't. This helps to fix future `cl-support` requests in the future.
core	Contains the core files generated from the Cumulus RMP HAL (hardware abstraction layer) process, `switchd`.
etc	`etc` is the core system configuration directory. `cl-support` replicates the switch's `/etc` directory. `/etc` contains all the general Linux configuration files, as well as configurations for the system's network interfaces, `quagga`, `monit`, and other packages.
var/log	`/var` 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 `/var`. `cl-support` includes only the `log` subdirectory of the `var` system-level directory and replicates the switch's `/var/log` directory. Most Cumulus RMP log files are located in this directory. Notable log files include `switchd.log` and `daemon.log` log files, and `syslog`. For more information, read this knowledge base article.

Directory	Description
proc	`proc` (short for processes) provides system statistics through a directory-and-file interface. In Linux, `/proc` contains runtime system information (like system memory, devices mounted, and hardware configuration). `cl-support` simply replicates the switch's `/proc` directory to determine the current state of the system.
support	`support` 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 `ps -aux`, `netstat -i`, and so forth — even the routing tables are included.

Here is more information on the file structure:

- Troubleshooting the etc Directory (see page 201) — 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 198) — This guide highlights the most important log files to look at. Keep in mind, `cl-support` includes all of the log files.
- Troubleshooting the support Directory (see page 212) — This is an explanation of the `support` directory included in the `cl-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?
/var/log/alternatives.log	Information from the update-alternatives are logged into this log file.	
/var/log/apt	Information the `apt` utility can send logs here; for example, from `apt-get install` and `apt-get remove`.	
/var/log/audit/	Contains log information stored by the Linux audit daemon, `auditd`.	
/var/log/auth.log	Authentication logs. Note that Cumulus RMP does not write to this log file; but because it's a standard file, Cumulus RMP creates it as a zero length file.	
/var/log/autoprovision	Logs output generated by running the zero touch provisioning (see page 71) script.	

Log	Description	Why is this important?
/var/log/boot.log	Contains information that is logged when the system boots.	
/var/log/btmp	This file contains information about failed login attempts. Use the `last` command to view the `btmp` file. For example: `cumulus@switch:~$ last -f /var/log/btmp | more`	
/var/log/cron.log	Log file for cron jobs. Note that Cumulus RMP does not write to this log file; but because it's a standard file, Cumulus RMP creates it as a zero length file.	
/var/log/daemon.log	Contains information logged by the various background daemons that run on the system. Note that Cumulus RMP does not write to this log file; but because it's a standard file, Cumulus RMP creates it as a zero length file.	
/var/log/debug	Debugging information. Note that Cumulus RMP does not write to this log file; but because it's a standard file, Cumulus RMP creates it as a zero length file.	
/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 `dmesg` command.	`dmesg` 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 `dpkg` command.	
/var/log/faillog	Contains failed user login attempts. Use the `faillog` command to display the contents of this file.	
/var/log/fsck/*	The `fsck` utility is used to check and optionally repair one or more Linux filesystems.	
/var/log/installer/*	Directory containing files related to the installation of Cumulus RMP.	
/var/log/kern.log	Logs produced by the kernel and handled by `syslog`.	

Log	Description	Why is this important?
	Note that Cumulus RMP does not write to this log file; but because it's a standard file, Cumulus RMP creates it as a zero length file.	
/var/log /lastlog	Formats and prints the contents of the last login log file.	
/var/log/lpr. log	Printer logs. Note that Cumulus RMP does not write to this log file; but because it's a standard file, Cumulus RMP creates it as a zero length file.	
/var/log/mail. log	Mail server logs. Also includes `mail.err`, `mail.info` and `mail.warn`. Note that Cumulus RMP does not write to this log file; but because it's a standard file, Cumulus RMP creates it as a zero length file.	
/var/log /messages	General messages and system-related information. Note that Cumulus RMP does not write to this log file; but because it's a standard file, Cumulus RMP creates it as a zero length file.	
/var/log/monit. log	`monit` is a utility for managing and monitoring processes, files, directories and filesystems on a Unix system.	
/var/log/netd. log	Log file for NCLU.	
/var/log/news /*	The **news** command keeps you informed of news concerning the system. Note that Cumulus RMP does not write to this log file; but because it's a standard file, Cumulus RMP creates it as a zero length file.	
/var/log /ntpstats	Logs for network configuration protocol.	
/var/log /snapper.log	Log file for snapshots (see page 58).	These logs are valuable for the snapshots you take on your switch.
/var/log /switchd.log/	The HAL log for Cumulus RMP.	This is specific to Cumulus

Log	Description	Why is this important?
		RMP. Any `switchd` 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 `grep` this file to see what occurred during a problem.
/var/log /watchdog	Hardware watchdog (see page 195) log files.	
/var/log/wtmp	Login records file.	

Troubleshooting the etc Directory

The `cl-support` (see page 196) script replicates the `/etc` directory.

Files that `cl-support` deliberately excludes are:

File	Description
/etc/nologin	`nologin` prevents unprivileged users from logging into the system.
/etc /alternatives	`update-alternatives` 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 `cl-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 `/etc/adduser.conf` contains defaults for the programs `adduser`, `addgroup`, `deluser`, and `delgroup`.	
adjtime	Corrects the time to synchronize the system clock.	
apt		

File	Description	Why is this important?
	`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.
audisp	The directory that contains `audisp-remote.conf`, which is the file that controls the configuration of the audit remote logging subsystem.	
audit	The directory that contains the `/etc/audit/auditd.conf`, 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 `/usr/share/bash-completion/bash_completion`.	
bash_completion.d	This folder contains app-specific code for Bash completion on Cumulus RMP, such as `mstpctl`.	
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 `bindresvport`.	
ca-certificates	The folder for `ca-certificates`. 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 `/usr/share/ca-certificates`.	
calendar	The system-wide default calendar file.	
chef	This is an example of something that is not included by default. In this instance, `cl-support` included the `chef` folder for some reason.	

File	Description	Why is this important?
		This is not installed by default, but this tool could have been installed or configured incorrectly, which is why it's included in the `cl-support` output.
cron.d	`cron` is a daemon that executes scheduled commands .	
cron.daily	See above.	
cron.hourly	See above.	
cron.monthly	See above.	
cron.weekly	See above.	
crontab	See above.	
cumulus	This directory contains the following: • ACL information, stored in the `acl` directory. • `switchd` configuration file, `switchd.conf`. • `qos`, which is under the `datapath` directory. • The routing protocol process priority, `nice.conf`. • The breakout cable configuration, under `ports.conf`.	This folder is specific to Cumulus RMP and does not exist on other Linux platforms. For example, while you can configure `iptables`, to hardware accelerate rules into the hardware you need to use `cl-acltool` and have the rules under the /etc/cumulus /acl/policy.d/<filename.rules)
debconf.conf	Debconf is a configuration system for Debian packages .	
debian_version	The complete Debian version string.	
debsums-ignore	`debsums` verifies installed package files against their MD5 checksums. This file identifies the packages to ignore.	
default	This folder contains files with configurable flags for many different applications (most installed by default or added manually). For example, `/etc/default /networking` has a flag for `EXCLUDE_INTERFACES=` , which is set to nothing by default, but a user could change it to something like swp3.	

deluser.conf	The file `/etc/deluser.conf` contains defaults for the programs `deluser` and `delgroup`.	
dhcp	This directory contains DHCP-specific information.	
dpkg	The package manager for Debian.	
e2fsck.conf	The configuration file for `e2fsck` . It controls the default behavior of `e2fsck` while it checks ext2, ext3 or ext4 filesystems.	
environment	Utilized by pam_env for setting and unsetting environment variables.	
ethertypes	This file can be used to show readable characters instead of hexadecimal numbers for the protocols. For example, 0x0800 will be represented by IPv4.	
fstab	Static information about the filesystems.	
fstab.d	The directory that can contain additional `fstab` information; it is empty by default.	
fw_env.config	Configuration file utilized by U-Boot.	
gai.conf	Configuration file for sorting the return information from getaddrinfo.	
groff	The directory containing information for `groffer`, an application used for displaying Unix man pages.	
group	The `/etc/group` file is a text file that defines the groups on the system.	
group-	Backup for the `/etc/group` file.	
gshadow	`/etc/gshadow` contains the shadowed information for group accounts.	
gshadow-	Backup for the `/etc/gshadow` file.	
host.conf	Resolver configuration file, which contains options like `multi` that determines whether `/etc/hosts` 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. `hosts.allow=Access` 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 `init` and `init.d` are explained well here.	
inittab	The format of the `inittab` file used by the `sysv`-compatible `init` process.	
inputrc	The initialization file utilized by `readline`.	
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 `ip`.	
issue	`/etc/issue` 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 `ldconfig` tool, which configures dynamic linker run-time bindings.	

ld.so.conf.d	The directory that contains additional `ld.so.conf` configuration (see above).	
ldap	The directory containing the `ldap.conf` configuration file used to set the system-wide default to be applied when running LDAP clients.	
libaudit.conf	Configuration file utilized by get_auditfail_action.	
libnl-3	Directory for the configuration relating to the `libnl` 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 `lldpcli` at startup.	
localtime	Copy of the original data file for `/etc/timezone`.	
logcheck	Directory containing `logcheck.conf` and logfiles utilized by the `log check` 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 `cat /etc/lsb-release` for output.	This shows you the version of the operating system you are running; also compare this to the output of `cl-img-select`.
magic	Used by the `file` command to determine file type. `magic` tests check for files with data in particular fixed formats.	
magic.mime	The `magic` MIME type causes the `file` command to output MIME type strings rather than the more traditional human readable ones.	

File	Description	Why is this important?
mailcap	The `mailcap` file is read by the metamail program to determine how to display non-text at the local site.	
mailcap.order	The order of entries in the `/etc/mailcap` file can be altered by editing the `/etc/mailcap.order` 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 `mke2fs`, which is a program that creates an ext, ext3 or ext4 filesystem.	
modprobe.d	Configuration directory for `modprobe`, which is a utility that can add and remove modules from the Linux kernel.	
modules	The kernel modules to load at boot time.	
monit	`monit` is a utility for monitoring services on a Unix system; this directory has configuration files beneath it.	
motd	The contents of `/etc/motd` ("message of the day") are displayed by `pam_motd` after a successful login but just before it executes the login shell.	
mtab	The programs `mount` and `umount` maintain a list of currently mounted filesystems in the `/etc/mtab` file. If no arguments are given to `mount`, this list is printed.	
nanorc	The GNU `nano rcfile`.	
network	Contains the network interface configuration for `ifup` and `ifdown`.	The main configuration file is under `/etc/network/interfaces`. 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 `conf.db` file, which is used by `ovsdb-server`.	
openvswitch-vtep	Configuration files used for the VTEP daemon and `ovsdb-server`.	
opt	Host-specific configuration files for add-on applications installed in `/opt`.	
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 `/etc/passwd`.	
perl	Perl is an available scripting language. `/etc/perl` contains configuration files specific to Perl.	
profile	`/etc/profile` is utilized by `sysprofile`, 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 134) passes or fails.	Cumulus RMP-specific folder for PTM (prescriptive topology manager).
python	`python` is an available scripting language.	
python2.6	The 2.6 version of `python`.	
python2.7	The 2.7 version of `python`.	
rc.local	The `/etc/rc.local` 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 `/usr/local`. Most installations don't need `/etc/rc.local`; it's provided for the minority of cases where it's needed.	
rc0.d	Like `rc.local`, 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 `id=2`, which indicates that the default runlevel will be 2 when the multi-user state is entered, and the scripts in /etc/rc2.d/ 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 *single* 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 `/etc/rmt` is provided for compatibility with other Unix-like systems, some of which have utilities that expect to find (and execute) `rmt` in the `/etc` directory on remote systems.	
rpc	The `rpc` file contains human-readable names that can be used in place of RPC program numbers.	
rsyslog.conf	The `rsyslog.conf` file is the main configuration file for `rsyslogd`, which logs system messages on *nix systems.	
rsyslog.d	The directory containing additional configuration for `rsyslog.conf` (see above).	
securetty	This file lists terminals into which the root user can log in.	
security	The `/etc/security` directory contains security-related configurations files. Whereas PAM concerns itself with the methods used to authenticate any given user, the files under `/etc/security` are concerned with just what a user can or cannot do. For example, the `/etc/security/access.conf` file contains a list of which users are allowed to log in and from what host (for example, using telnet). The `/etc/security/limits.conf` file contains various system limits, such as maximum number of processes.	
selinux	NSA Security-Enhanced Linux.	
sensors.d	The directory from which the `sensors` program loads its configuration; this is unique for each hardware platform. See also Monitoring System Hardware (see page 191).	
sensors3.conf	The `sensors.conf` file describes how `libsensors`, 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?
	`services` 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.	
shadow	shadow is a file that contains the password information for the system's accounts and optional aging information.	
shadow-	The backup for the `/etc/shadow` file.	
shells	The pathnames of valid login shells.	
skel	The skeleton directory (usually `/etc/skel`) is used to copy default files and also sets a umask for the creation used by `pam_mkhomedir`.	
snmp	Interface functions to the SNMP (simple network management protocol) toolkit.	
ssh	The `ssh` configuration.	
ssl	The OpenSSL `ssl` library implements the Secure Sockets Layer (SSL v2/v3) and Transport Layer Security (TLS v1) protocols. This directory holds certificates and configuration.	
staff-group-for-usr-local	Use `cat` or `more` on this file to learn more information, see bugs.debian.org/299007.	
sudoers	The `sudoers` policy plugin determines a user's `sudo` privileges.	
sudoers.d	The directory file containing additional `sudoers` configuration (see above).	
sysctl.conf	Configures kernel parameters at boot.	
sysctl.d	The directory file containing additional configuration (see above).	
systemd	`systemd` system and service manager.	
terminfo	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 presents the output from several commands, rather than being a copy of the switch's filesystem. For example:

File	Equivalent Command	Description
support /ip. addr	`cumulus@switch:~$ ip addr show`	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

This chapter covers ...

- Understanding the "RTNETLINK answers: Invalid argument" Error when Adding a Port to a Bridge (see page 217)

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.

```
cumulus@switch:~$ 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. The line `bond-xmit-hash-policy layer3+7` below fails because it should read `bond-xmit-hash-policy layer3+4`.

```
cumulus@switch:~$ sudo ifquery --check bond0
iface bond0
    bond-xmit-hash-policy layer3+7  [fail]
    bond-slaves swp25 swp26         [pass]
    address 14.0.0.9/30             [pass]
    address 2001:ded:beef:2::1/64   [pass]
```

⚠ `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 bond1 is removed, you must remove the reference to it from the br0 configuration. Otherwise, if you reload the configuration with `ifreload -a`, bond1 is still part of br0.

MTU Set on a Logical Interface Fails with Error: "Numerical result out of range"

This error occurs when the MTU (see page) 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

This chapter covers ...

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 `uid` 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 `cl-netstat`).
-t <tag name>	Saves statistics with `<tag name>`.
-v	Prints `cl-netstat` 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 / 30.2
0 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

This chapter covers ...

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

`mz` 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=
```

Related Information

- www.perihel.at/sec/mz/mzguide.html
- en.wikipedia.org/wiki/Ping
- en.wikipedia.org/wiki/Traceroute

Using NCLU to Troubleshoot Your Network Configuration

The Network Command Line Utility (NCLU) can quickly return a lot of information about your network configuration.

Contents

This chapter covers ...

Using net show Commands

Running `net show` and pressing TAB displays all available command line arguments usable by `net`. The output looks like this:

```
cumulus@switch$ net show <TAB>
    bgp              :  Border Gateway Protocol
    bridge           :  A layer2 bridge
    clag             :  Multi-Chassis Link Aggregation
    commit           :  apply the commit buffer to the system
    configuration    :  Settings, configuration state, etc
    counters         :  show netstat counters
    hostname         :  System hostname
    igmp             :  Internet Group Management Protocol
    interface        :  An interface such as swp1, swp2, etc
    ip               :  Internet Protocol version 4
    ipv6             :  Internet Protocol version 6
    lldp             :  Link Layer Discovery Protocol
    lnv              :  Lightweight Network Virtualization
    mroute           :  Configure static unicast route into MRIB for
multicast RPF lookup
    msdp             :  Multicast Source Discovery Protocol
    ospf             :  Open Shortest Path First (OSPFv2)
    ospf6            :  Open Shortest Path First (OSPFv3)
    pim              :  Protocol Independent Multicast
    route            :  Static routes
    route-map        :  Route-map
    system           :  System information
    version          :  Version number
```

Showing Interfaces

To show all available interfaces that are physically UP, run `netshow interface`:

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

    Name    Speed    MTU    Mode           Summary
--  ------  -------  -----  -------------
--------------------------------------
UP  lo      N/A      65536  Loopback       IP: 10.0.0.11/32, 127.0.0.1
/8, ::1/128
UP  eth0    1G       1500   Mgmt           IP: 192.168.0.11/24(DHCP)
UP  swp1    1G       1500   Access/L2      Untagged: br0
UP  swp2    1G       1500   NotConfigured
UP   swp51   1G        1500   NotConfigured
UP   swp52   1G        1500   NotConfigured
UP   blue    N/A       65536  NotConfigured
UP   br0     N/A       1500   Bridge/L3      IP: 172.16.1.1/24
                                             Untagged Members: swp1
                                             802.1q Tag: Untagged
                                             STP: RootSwitch(32768)
UP   red     N/A       65536  NotConfigured
```

Whereas `net show interface all` displays every interface regardless of state:

```
cumulus@switch:~$ net show interface all
      Name     Speed    MTU    Mode           Summary
----- -------  -------  -----  -------------
--------------------------------------
UP    lo       N/A      65536  Loopback       IP: 10.0.0.11/32, 127.0
.0.1/8, ::1/128
UP    eth0     1G       1500   Mgmt           IP: 192.168.0.11/24(DHC
P)
UP    swp1     1G       1500   Access/L2      Untagged: br0
UP    swp2     1G       1500   NotConfigured
ADMDN swp45    0M       1500   NotConfigured
ADMDN swp46    0M       1500   NotConfigured
ADMDN swp47    0M       1500   NotConfigured
ADMDN swp48    0M       1500   NotConfigured
ADMDN swp49    0M       1500   NotConfigured
ADMDN swp50    0M       1500   NotConfigured
UP    swp51    1G       1500   NotConfigured
UP    swp52    1G       1500   NotConfigured
UP    blue     N/A      65536  NotConfigured
UP    br0      N/A      1500   Bridge/L3      IP: 172.16.1.1/24
                                              Untagged Members: swp1
                                              802.1q Tag: Untagged
                                              STP: RootSwitch(32768)
UP    red      N/A      65536  NotConfigured
ADMDN vagrant  0M       1500   NotConfigured
```

You can get information about the switch itself by running `net show system`:

```
cumulus@switch:~$ netshow system
Arctica 4804IP
Cumulus Version 3.2.0
Build: Cumulus RMP 3.2.0
Uptime: 2 days, 21:31:00
```

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:

`net show int` runs `net show interface`
`net show sys` runs `net show system`

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

This chapter covers ...

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.

Related Information

- 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

This chapter covers ...

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 base 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 `sudo 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`:

```
cumulus@switch:~$ 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 acces). 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
```

Syntax	Meaning
rocommunity	Read-only community; (*rwcommunity* is for read-write access).

Syntax	Meaning
public	Plain text password.
default	"default" allows connections from any system. "localhost" allows requests only from the local host. A restricted source can either be a specific hostname (or address), or a subnet, represented as IP/MASK (like 10.10.10.0/255.255.255.0), or IP/BITS (like 10.10.10.0/24), or the IPv6 equivalents.
systemonly	The name of this particular SNMP view. This is a user-defined value.

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.service
```

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.service
```

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/llib/snmp/snpmd.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
```

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

4. Install the `snmp` and `snmp-mibs-downloader` packages:

```
cumulus@switch:~$ 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. Run the following command, which rules out an SNMP problem against a networking problem.

```
cumulus@switch:~$ snmpwalk -c public -v2c 192.168.0.111
```

Here is some sample output:

```
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 snmpwalks

Run snmpwalk from	If it works	If it does not work
switch (switch to monitor)	`snmpd` is serving information correctly. The problem resides somewhere else. For example, network connectivity, or Prism misconfiguration.	Is snmpd misconfigured or installed incorrectly?
switch2 (another Cumulus Linux or Cumulus RMP switch in the network)	`snmpd` is serving information correctly and network reachability works between **switch** and **switch2**. The problem resides somewhere else. For example, Prism cannot reach **switch**, or there is a Prism misconfiguration.	Network connectivity is not able to grab information? Is there an `iptables` rule blocking? Is the `snmpwalk` being run correctly?
Nutanix Prism (see page 244) **CLI** (SSH to the cluster IP address)	`snmpd` is serving information correctly and network reachability works between **switch** and the **Nutanix Appliance**. The problem resides somewhere else. For example, the GUI might be misconfigured.	Is the right community name being used in the GUI? Is `snmp v2c` being used?

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:

```
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.

Monitoring Fans, Power Supplies, or Transformers

SNMP can be configured to monitor the operational status of an Entity MIB or Entity-Sensor MIB. The operational status, given as a value of ok(1), unavailable(2), or nonoperational(3), can be determined by adding the following example configuration to `/etc/snmp/snmpd.conf`, and adjusting the values:

- Using the `entPhySensorOperStatus` integer:

```
# without installing extra MIBS we can check the check Fan1
status
# if the Fan1 index is 100011001
monitor -I -r 10  -o 1.3.6.1.2.1.47.1.1.1.1.7.100011001 "Fan1
Not OK"  1.3.6.1.2.1.99.1.1.1.5.100011001 > 1
# Any Entity Status non OK (greater than 1)
monitor  -r 10  -o 1.3.6.1.2.1.47.1.1.1.1.7  "Sensor Status
Failure"  1.3.6.1.2.1.99.1.1.1.5 > 1
```

The `entPhySensorOperStatus` integer can be found by walking the `entPhysicalName` table.

To get all sensor information, run `snmpwalk` on the entPhysicalName table. For example:

```
cumulus@leaf01:~$ snmpwalk -v 2c -cpublic localhost .1.3.6.1.2.1.
47.1.1.1.1.7
iso.3.6.1.2.1.47.1.1.1.1.7.100000001 = STRING: "PSU1Temp1"
iso.3.6.1.2.1.47.1.1.1.1.7.100000002 = STRING: "PSU2Temp1"
iso.3.6.1.2.1.47.1.1.1.1.7.100000003 = STRING: "Temp1"
iso.3.6.1.2.1.47.1.1.1.1.7.100000004 = STRING: "Temp2"
iso.3.6.1.2.1.47.1.1.1.1.7.100000005 = STRING: "Temp3"
iso.3.6.1.2.1.47.1.1.1.1.7.100000006 = STRING: "Temp4"
iso.3.6.1.2.1.47.1.1.1.1.7.100000007 = STRING: "Temp5"
iso.3.6.1.2.1.47.1.1.1.1.7.100011001 = STRING: "Fan1"
iso.3.6.1.2.1.47.1.1.1.1.7.100011002 = STRING: "Fan2"
iso.3.6.1.2.1.47.1.1.1.1.7.100011003 = STRING: "Fan3"
iso.3.6.1.2.1.47.1.1.1.1.7.100011004 = STRING: "Fan4"
iso.3.6.1.2.1.47.1.1.1.1.7.100011005 = STRING: "Fan5"
iso.3.6.1.2.1.47.1.1.1.1.7.100011006 = STRING: "Fan6"
iso.3.6.1.2.1.47.1.1.1.1.7.100011007 = STRING: "PSU1Fan1"
iso.3.6.1.2.1.47.1.1.1.1.7.100011008 = STRING: "PSU2Fan1"
iso.3.6.1.2.1.47.1.1.1.1.7.110000001 = STRING: "PSU1"
iso.3.6.1.2.1.47.1.1.1.1.7.110000002 = STRING: "PSU2"
```

- Using the OID name:

```
# for a specific fan called Fan1 with an index 100011001
monitor -I -r 10  -o entPhysicalName.100011001 "Fan1 Not OK"
entPhySensorOperStatus.100011001 > 1
# for any Entity Status not OK ( greater than 1)
monitor  -r 10  -o entPhysicalName  "Sensor Status Failure"
entPhySensorOperStatus > 1
```

⚠ The OID name can be used if the `snmp-mibs-download` package is installed.

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:~$ sudo deb http://ftp.us.debian.org/debian/
jessie main non-free
```

3. Update the switch:

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

4. Install the `snmp-mibs-downloader`:

```
cumulus@switch:~$ sudo apt-get snmp-mibs-downloader
```

5. Open the `/etc/snmp/snmp.conf` file to verify that the `mibs :` line is commented out:

```
#
# As the snmp packages come without MIB files due to license
reasons, loading
# of MIBs is disabled by default. If you added the MIBs you can
reenable
# loading them by commenting out the following line.
#mibs :
```

6. Open the `/etc/default/snmpd` file to verify that the `export MIBS=` line is commented out:

```
# This file controls the activity of snmpd and snmptrapd

# Don't load any MIBs by default.
# You might comment this lines once you have the MIBs Downloaded.
#export MIBS=
```

7. 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 lmTempSenesorsDevice` 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 lmTemSensor -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.

```
cumulus@switch:~$ sudo 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 laErrMessage "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 `authtrapenable` directive:

```
authtrapenable 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 `bridge_pp.py pass_persist` script in `/etc/snmp/snmpd.conf`.
CUMULUS-COUNTERS-MIB	Discard counters: Cumulus RMP also includes its own counters MIB, defined in `/usr/share/snmp/Cumulus-Counters-MIB.txt`. It has the OID `.1.3.6.1.4.1.40310.2`
CUMULUS-RESOURCE-QUERY-MIB	Cumulus RMP includes its own resource utilization MIB, which is similar to using cl-resource-query (see page 190). It monitors L3 entries by host, route, nexthops, ECMP groups and L2 MAC/BDPU entries. The MIB is defined in `/usr/share/snmp/Cumulus-Resource-Query-MIB.txt`, and has the OID `.1.3.6.1.4.1.40310.1.`
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 `lldpd` (note, you need to enable the SNMP subagent (see page) in LLDP). `lldpd` needs to be started with the `-x` option to enable connectivity to `snmpd` (AgentX).
LM-SENSORS MIB	Fan speed, temperature sensor values, voltages. This is deprecated since the ENTITY-SENSOR MIB has been added.
NET-SNMP-AGENT	Agent timers, user, group config
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.

Using Nutanix Prism as a Monitoring Tool

Nutanix Prism is a graphical user interface (GUI) for managing infrastructure and virtual environments. In order to use it, you need to take special steps within Cumulus RMP before you can configure Prism.

Contents

This chapter covers ...

Configuring Cumulus RMP

1. SSH to the Cumulus RMP switch that needs to be configured, replacing `[switch]` below as appropriate:

```
cumulus@switch:~$ ssh cumulus@[switch]
```

2. Confirm the switch is running Cumulus RMP 2.5.5 or newer:

```
cumulus@switch:~$ net show system
Arctica 4804IP
Cumulus RMP 3.2.0
Build: Cumulus RMP 3.2.0
Uptime: 8 days, 1:07:17
```

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)

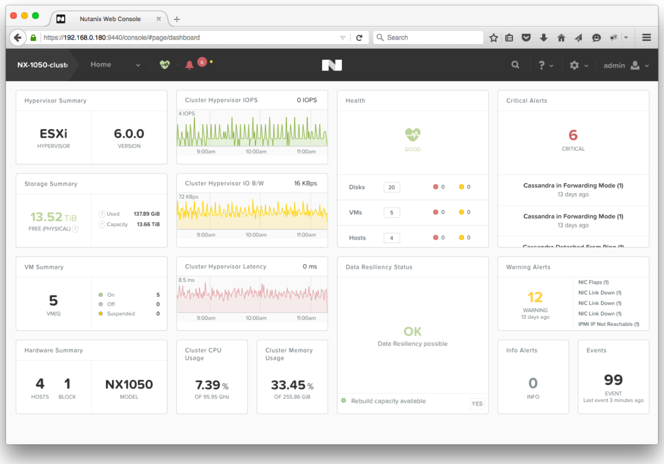
```
# BRIDGE-MIB and Q-BRIDGE-MIB tables
view   systemonly  included   .1.3.6.1.2.1.17
```

5. Restart `snmpd`:

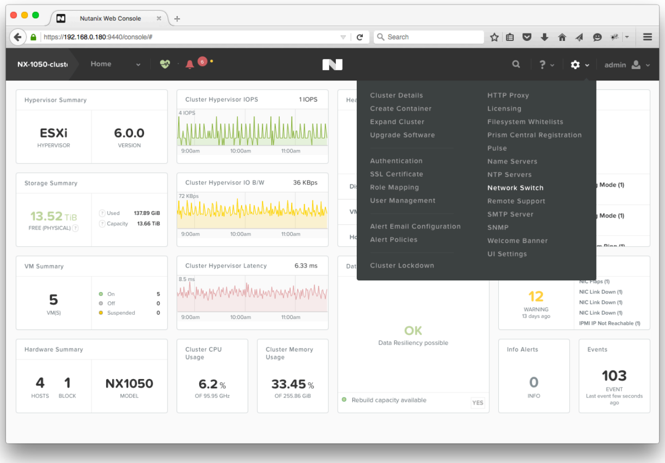
```
cumulus@switch:~$ sudo systemctl restart snmpd.service
Restarting network management services: snmpd.
```

Configuring Nutanix

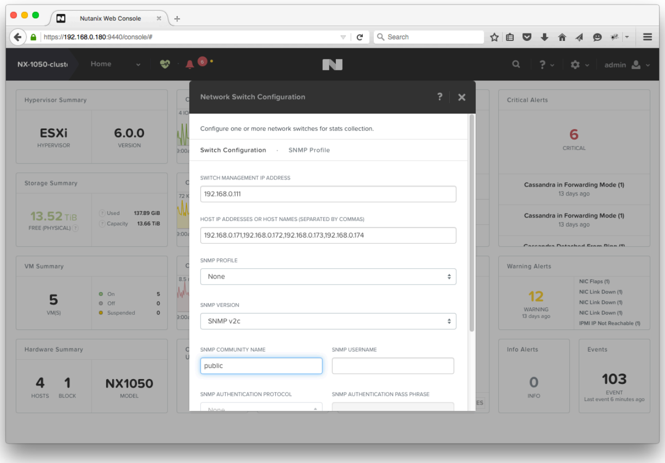
1. Log into the Nutanix Prism. Nutanix defaults to the Home menu, referred to as the Dashboard:



2. Click on the gear icon in the top right corner of the dashboard, 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:


Network Switch Configuration

Configure one or more network switches for stats collection.

Switch Configuration · SNMP Profile

SWITCH MANAGEMENT IP ADDRESS: 192.168.0.111

HOST IP ADDRESSES OR HOST NAMES (SEPARATED BY COMMAS): 192.168.0.171,192.168.0.172,192.168.0.173,192.168.0.174

SNMP PROFILE: None

SNMP VERSION: SNMP v2c

SNMP COMMUNITY NAME: public

SNMP USERNAME:

SNMP AUTHENTICATION PROTOCOL: None

SNMP AUTHENTICATION PASS PHRASE:

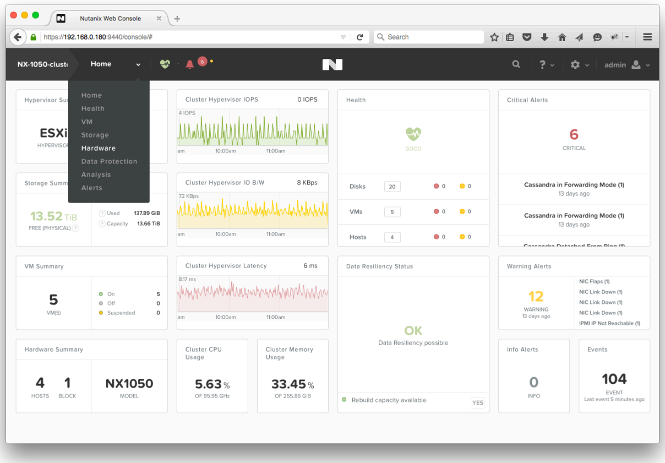
Configuration Parameter	Description	Value Used in Example
Switch Management IP Address	This can be any IP address on the box. In the screenshot above, the eth0 management IP is used.	192.168.0.111
Host IP Addresses or Host Names		192.168.0.171,192.168.0.172,192.168.0.173,192.168.0.174

Configuration Parameter	Description	Value Used in Example
	IP addresses of Nutanix hosts connected to that particular ToR switch.	
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

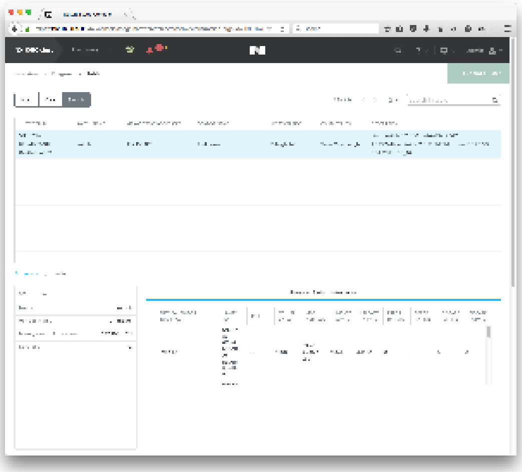
The rest of the values were not touched for this demonstration. They are usually used with SNMP v3.

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)

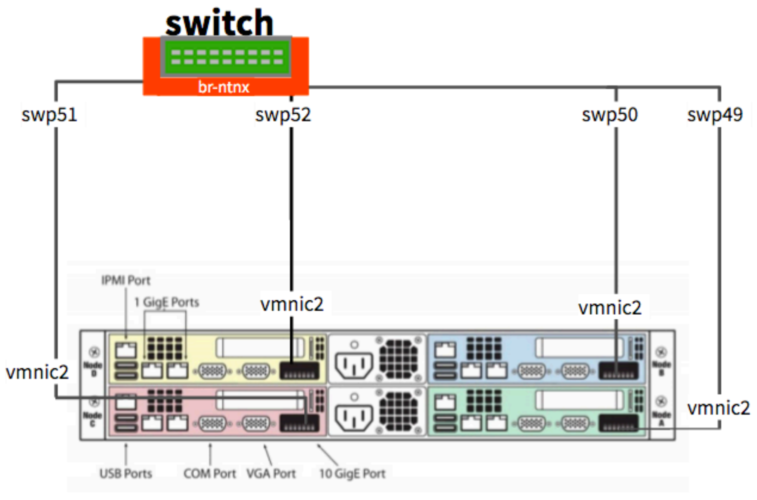
- Unicast TX Packets (Transmitted)
- Error RX Packets (Received)
- Error TX Packets (Transmitted)
- Discard RX Packets (Received)
- Discard TX Packets (Transmitted)

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-swi
tch
    Switch ID                 : 00051a76-f711-89b6-0000-000000003bac::
5f13678e-6ffd-4b33-912f-f1aa6e8da982
    Name                      : switch
    Switch Management Address : 192.168.0.111
    Description               : Linux switch 3.2.65-1+deb7u2+cl2.5+2 #
3.2.65-1+deb7u2+cl2.5+2 SMP Mon Jun 1 18:26:59 PDT 2015 x86_64
    Object ID                 : enterprises.40310
    Contact Information       : Admin <admin@company.com>
    Location Information      : Raleigh, NC
    Services                  : 72
    Switch Vendor Name        : Unknown
    Port Ids                  : 00051a76-f711-89b6-0000-000000003bac::
5f13678e-6ffd-4b33-912f-f1aa6e8da982:52, 00051a76-f711-89b6-0000-00000
0003bac::5f13678e-6ffd-4b33-912f-f1aa6e8da982:53, 00051a76-f711-89b6-0
000-000000003bac::5f13678e-6ffd-4b33-912f-f1aa6e8da982:54, 00051a76-
f711-89b6-0000-000000003bac::5f13678e-6ffd-4b33-912f-f1aa6e8da982:55
```

Troubleshooting a Nutanix Node

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)

1. Follow the directions on one of the following websites to enable CDP:
   - kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=1003885
   - wahlnetwork.com/2012/07/17/utilizing-cdp-and-lldp-with-vsphere-networking/

   For example, switch CDP on:

   ```
   root@NX-1050-A:~] esxcli network vswitch standard set -c
   both -v vSwitch0
   ```

   Then confirm it is running:

   ```
   root@NX-1050-A:~] esxcli network vswitch standard list -v
   vSwitch0
   vSwitch0
      Name: vSwitch0
      Class: etherswitch
      Num Ports: 4082
      Used Ports: 12
      Configured Ports: 128
      MTU: 1500
      CDP Status: both
      Beacon Enabled: false
      Beacon Interval: 1
      Beacon Threshold: 3
      Beacon Required By:
      Uplinks: vmnic3, vmnic2, vmnic1, vmnic0
      Portgroups: VM Network, Management Network
   ```

   The **both** means CDP is now running, and the `lldp` dameon 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 `net show` to look at `lldp` information:

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

Local Port    Speed    Mode                 Remote Port
Remote  Host     Summary
------------  -------  -------------  ----  -----------------
---------------  -------------------------
eth0          1G       Mgmt           ====  swp6
oob-mgmt-switch  IP: 192.168.0.11/24(DHCP)
swp1          1G       Access/L2      ====  44:38:39:00:00:03
server01         Untagged: br0
swp51         1G       NotConfigured  ====  swp1
spine01
swp52         1G       NotConfigured  ====  swp1
spine02
```

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.

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`:

```
cumulus@switch:~$ brctl showmacs br-ntnx | grep 0c:c4:7a:09:a2:43
swp49     0c:c4:7a:09:a2:43    0    no             4.58
```

vmnic1 is now hooked up to swp49. This matches what is seen in `lldp`:

```
cumulus@switch:~$ lldpctl show neighbor swp49
-----------------------------------------------------------------
--------------
LLDP neighbors:
-----------------------------------------------------------------
--------------
Interface:    swp49, via: CDPv2, RID: 6, Time: 0 day, 01:11:12
  Chassis:
    ChassisID:    local NX-1050-A
    SysName:      NX-1050-A
    SysDescr:     Releasebuild-2494585 running on VMware ESX
    MgmtIP:       0.0.0.0
    Capability:   Bridge, on
  Port:
    PortID:       ifname vmnic2
    PortDescr:    vmnic2
-----------------------------------------------------------------
--------------
```

Monitoring Best Practices

The following monitoring processes are considered best practices for reviewing and troubleshooting potential issues with Cumulus Linux environments. In addition, several of the more common issues have been listed, with potential solutions included.

Contents

This chapter covers ...

Overview

This document aims to provide two sets of outputs:

1. Metrics that can be polled from Cumulus Linux and used in trend analysis
2. Critical log messages that can be monitored for triggered alerts

Trend Analysis via Metrics

A metric is a quantifiable measure that is used to track and assess the status of a specific infrastructure component. It is a check collected over time. Examples of metrics include bytes on an interface, CPU utilization and total number of routes.

Metrics are more valuable when used for trend analysis.

Alerting via Triggered Logging

Triggered issues are normally sent to syslog, but could go to another log file depending on the feature. On Cumulus Linux rsyslog handles all logging including local and remote logging. Logs are the best method to use for generating alerts when the system transitions from a stable steady state.

Sending logs to a centralized collector, then creating an alerts based on critical logs is optimal solution for alerting.

Hardware

The `smond` process provides monitoring functionality for various switch hardware elements. Mininmum /maximum values are output, depending on the flags applied to the basic command. The hardware elements and applicable commands/flags are listed in the table below:

Hardware Element	Monitoring Command/s	Interval Poll

Temperature	`cumulus@switch:~$ smonctl -j`
	`cumulus@switch:~$ smonctl -j -s TEMP[X]`
		600 seconds

Fan	`cumulus@switch:~$ smonctl -j`
	`cumulus@switch:~$ smonctl -j -s FAN[X]`
		600 seconds

Hardware Element	Monitoring Commands/s	Interval Poll

PSU	`cumulus@switch:~$ smonctl -j`
	`cumulus@switch:~$ smonctl -j -s PSU[X]`
		600 seconds

PSU Fan	`cumulus@switch:~$ smonctl -j`
	`cumulus@switch:~$ smonctl -j -s PSU[X]Fan[X]`
		600 seconds

PSU Temperature	`cumulus@switch:~$ smonctl -j`
	`cumulus@switch:~$ smonctl -j -s PSU[X]Temp`
	`[X]`
		600 seconds

Voltage	`cumulus@switch:~$ smonctl -j`
	`cumulus@switch:~$ smonctl -j -s Volt[X]`
		600 seconds

Front Panel LED	`cumulus@switch:~$ ledmgrd -d`
	`cumulus@switch:~$ ledmgrd -j`
		600 seconds

Hardware Logs	Log Location	Log Entries

High temperature	`/va`
	`r`
	`/lo`
	`g`
	`/sy`
		`/usr/sbin/smond : : Temp2(Near the CPU (Right)):`
		`Temperature(51 C) is HIGH for last 71240 secs`
		`/usr/sbin/smond : : Temp3(Top right corner):`
		`Temperature(43 C) is HIGH for last 71240 secs`
		`/usr/sbin/smond : : Temp4:  Avg state is CRITICAL for`

Hardware Logs	Log Location	Log Entries
	`slog`	`last 60 secs. Last 10 values: [85.0, 85.5, 86.125, 86.625, 87.0, 87.625, 88.0, 88.625, 88.875, 89.25]. Generating cl-support and shutting down system...`

Fan speed issues	`/var/log/syslog`	`/usr/sbin/smond : : Fan5(Fan Tray 3): state changed from OK to HIGH`
		`/usr/sbin/smond : : Fan5(Fan Tray 3): Fan speed 28912 RPM greater than 19000 RPM`
		`/usr/sbin/smond : : Fan5(Fan Tray 3): state changed from HIGH to OK`
		`...`
		`/usr/sbin/smond : : Fan1: state changed from OK to ABSENT`
		`/usr/sbin/smond : : Fan1:  Fan speed is at 0 RPM (not working or absent)`

PSU failure	`/var/log/syslog`	`/usr/sbin/smond : : PSU1: state changed from UNKNOWN to BAD`

System Data

Cumulus Linux includes a number of ways to monitor various aspects of system data. In addition, alerts are issued in high risk situations.

CPU Idle Time

When a CPU reports five high CPU alerts within a span of 5 minutes, an alert is logged.

Short High CPU Bursts

Short bursts of high CPU can occur during switchd churn or routing protocol startup. Do not set alerts for these short bursts.

System Element	Monitoring Command/s	Interval Poll

CPU utilization	`cumulus@switch:~$ cat /proc/stat`
	`cumulus@switch:~$ top -b -n 1`
		30 seconds

CPU Logs	Log Location	Log Entries

High CPU	`/var/log /syslog`	`sysmonitor: High CPU use: 91%`
		`sysmonitor: CPU use no longer high: 58%`
		
		`sysmonitor: Critically high load average: 1.31 (1.31)`
		`sysmonitor: High load average: 1.24 (1.24)`
		`sysmonitor: Load Average no longer high: 0.88 (0.88)`

Cumulus Linux 3.0 and later monitors CPU, memory and disk space via sysmonitor. The configurations for the thresholds are stored in **/etc/cumulus/sysmonitor.conf**. More information is available via **man sysmonitor**.

CPU measure	Thresholds
Use	Alert: 90% Crit: 95%
Process Load	Alarm: 95% Crit: 125%

Click here to see differences between Cumulus Linux 2.5 ESR and 3.0 and later...

CPU Logs	Log Location	Log Entries
High CPU		

CPU Logs	Log Location	Log Entries

	`/var/log/syslog`	`jdoo[2803]: 'localhost' cpu system usage of 41.1% matches resource limit [cpu system usage>30.0%]`
		`jdoo[4727]: 'localhost' sysloadavg(15min) of 111.0 matches resource limit [sysloadavg(15min)>110.0]`

In Cumulus Linux 2.5, CPU logs are created with each unique threshold:

CPU measure	< 2.5 Threshold
User	70%
System	30%
Wait	20%

Cumulus Linux 2.5, CPU and Memory warnings are generated via jdoo. The configuration for the thresholds are stored in **/etc/jdoo/jdoorc.d/cl-utilities.rc**.

Memory Usage

When the memory utilization exceeds 90% a warning is logged and a cl-support is generated.

System Element	Monitoring Command/s	Interval Poll

Memory utilization	`cumulus@switch:~$ cat /proc/meminfo`
	`cumulus@switch:~$ cat /usr/bin/free`
		30 seconds

Disk Usage

When monitoring disk utilization **tmpfs** can be excluded from monitoring.

System Element	Monitoring Command/s	Interval Poll
Disk utilization	`cumulus@switch:~$ /bin/df -x tmpfs`	300 seconds

Process Restart

In Cumulus Linux 3.0 and later, systemd is responsible for monitoring and restarting processes.

Process Element	Monitoring Command/s
View processes monitored by systemd	`cumulus@switch:~$ systemctl status`

Click here to changes from Cumulus Linux 2.5 ESR to 3.0 and later...

Cumulus Linux 2.5.2 through 2.5 ESR uses a forked version of monit called jdoo to monitor processes. If the process ever fails, jdoo then invokes init.d to restart the process.

Process Element	Monitoring Command/s
View processes monitored by jdoo	`cumulus@switch:~$ jdoo summary`
View process restarts	`cumulus@switch:~$ sudo cat /var/log/syslog`
View current process state	`cumulus@switch:~$ ps -aux`

Layer 1 Protocols and Interfaces

Link and port state interface transitions are logged to **/var/log/syslog** and **/var/log/switchd.log**.

Interface Element	Monitoring Command/s

Link state	`cumulus@switch:~$ cat /sys/class/net/[iface]/operstate`
	
	`cumulus@switch:~$ net show interface all json`

Link speed	`cumulus@switch:~$ cat /sys/class/net/[iface]/speed`
	
	`cumulus@switch:~$ net show interface all json`

Port state	`cumulus@switch:~$ ip link show`
	
	`cumulus@switch:~$ net show interface all json`

Bond state	`cumulus@switch:~$ cat /proc/net/bonding/[bond]`
	
	`cumulus@switch:~$ net show interface all json`

Interface counters are obtained from either querying the hardware or the Linux kernel. The two outputs should align, but the Linux kernel aggregates the output from the hardware.

Interface Counter Element	Monitoring Command/s	Interval Poll

Interface counters	`cumulus@switch:~$ cat /sys/class/net/[iface]/statistics/[stat_name]`
	
	`cumulus@switch:~$ net show counters json`
	`cumulus@switch:~$ cl-netstat -j`
	`cumulus@switch:~$ ethtool -S [iface]`
		10 seconds

Layer 1 Logs	Log Location	Log Entries

Link failure/Link flap	`/var/log/switchd.log`	`switchd[5692]: nic.c:213 nic_set_carrier: swp17: setting kernel carrier: down`
		`switchd[5692]: netlink.c:291 libnl: swp1, family 0, ifi 20, oper down`
		`...`
		`switchd[5692]: nic.c:213 nic_set_carrier: swp1: setting kernel carrier: up`
		`switchd[5692]: netlink.c:291 libnl: swp17, family 0, ifi 20, oper up`

Unidirectional link	`/var/log/switchd.log`
	`/var/log/ptm.log`
		`ptmd[7146]: ptm_bfd.c:2471 Created new session 0x1 with peer 10.255.255.11 port swp1`
		`ptmd[7146]: ptm_bfd.c:2471 Created new session 0x2 with peer fe80::4638:39ff:fe00:5b port swp1`
		`...`
		`ptmd[7146]: ptm_bfd.c:2471 Session 0x1 down to peer 10.255.255.11, Reason 8`
		`ptmd[7146]: ptm_bfd.c:2471 Detect timeout on session 0x1 with peer 10.255.255.11, in state 1`

Layer 1 Logs	Log Location	Log Entries
Bond Negotiation • Working	`/var/log/syslog`	`kernel: [73946.052292] bonding: Bond1 is being created...` `kernel: [73946.062957] Bond1: Enslaving swp49 as a backup interface with an up link` `kernel: [73946.081609] Bond1: Enslaving swp50 as a backup interface with an up link` `kernel: [73946.090636] IPv6: ADDRCONF(NETDEV_UP): Bond1: link is not ready` `kernel: [74590.925353] IPv6: ADDRCONF (NETDEV_CHANGE): Bond1: link becomes ready`
Bond Negotiation • Failing	`/var/log/syslog`	`kernel: [73946.052292] bonding: Bond1 is being created...` `kernel: [73946.062957] Bond1: Enslaving swp49 as a backup interface with an up link` `kernel: [73946.081609] Bond1: Enslaving swp50 as a backup interface with an up link` `kernel: [73946.090636] IPv6: ADDRCONF(NETDEV_UP): Bond1: link is not ready`
MLAG peerlink negotiation • Working	`/var/log/syslog`	`lldpd[998]: error while receiving frame on swp50: Network is down` `lldpd[998]: error while receiving frame on swp49: Network is down` `kernel: [76174.262893] peerlink: Setting ad_actor_system to 44:38:39:00:00:11` `kernel: [76174.264205] 8021q: adding VLAN 0 to HW filter on device peerlink` `mstpd: one_clag_cmd: setting (1) peer link: peerlink` `mstpd: one_clag_cmd: setting (1) clag state: up`

Layer 1 Logs	Log Location	Log Entries

		`mstpd: one_clag_cmd: setting system-mac 44:39:39:ff:40:94`
		`mstpd: one_clag_cmd: setting clag-role secondary`

	`/var/log/clagd.log`	`clagd[14003]: Cleanup is executing.`
		`clagd[14003]: Cannot open file "/tmp/pre-clagd.q7XiO`
		`clagd[14003]: Cleanup is finished`
		`clagd[14003]: Beginning execution of clagd version 1`
		`clagd[14003]: Invoked with: /usr/sbin/clagd --daemon`
		`clagd[14003]: Role is now secondary`
		`clagd[14003]: HealthCheck: role via backup is second`
		`clagd[14003]: HealthCheck: backup active`
		`clagd[14003]: Initial config loaded`
		`clagd[14003]: The peer switch is active.`
		`clagd[14003]: Initial data sync from peer done.`
		`clagd[14003]: Initial handshake done.`
		`clagd[14003]: Initial data sync to peer done.`

MLAG peerlink negotiation
• Failing
	`/var/log/syslog`	`lldpd[998]: error while receiving frame on swp50: Network is down`
		`lldpd[998]: error while receiving frame on swp49: Network is down`
		`kernel: [76174.262893] peerlink: Setting ad_actor_system to 44:38:39:00:00:11`
		`kernel: [76174.264205] 8021q: adding VLAN 0 to HW filter on device peerlink`
		`mstpd: one_clag_cmd: setting (1) peer link: peerlink`
		`mstpd: one_clag_cmd: setting (1) clag state: down`

Layer 1 Logs	Log Location	Log Entries

		`mstpd: one_clag_cmd: setting system-mac 44:39:39:ff:40:94`
		`mstpd: one_clag_cmd: setting clag-role secondary`

	`/var/log/clagd.log`	`clagd[14291]: Cleanup is executing.`
		`clagd[14291]: Cannot open file "/tmp/pre-clagd.vXpXOcM8h9/brbatch" for reading: No such file or directory`
		`clagd[14291]: Cleanup is finished`
		`clagd[14291]: Beginning execution of clagd version 1.3.0`
		`clagd[14291]: Invoked with: /usr/sbin/clagd --daemon 169.254.255.2 peerlink.4094 44:39:39:FF:40:94 --priority 32768 --backupIp 192.168.0.1`
		`clagd[14291]: Role is now secondary`
		`clagd[14291]: Initial config loaded`

MLAG port negotiation
• Working
	`/var/log/syslog`	`kernel: [77419.112195] bonding: server01 is being created...`
		`lldpd[998]: error while receiving frame on swp1: Network is down`
		`kernel: [77419.122707] 8021q: adding VLAN 0 to HW filter on device swp1`
		`kernel: [77419.126408] server01: Enslaving swp1 as a backup interface with a down link`
		`kernel: [77419.177175] server01: Setting ad_actor_system to 44:39:39:ff:40:94`
		`kernel: [77419.190874] server01: Warning: No 802.3 ad response from the link partner for any adapters in the bond`
		`kernel: [77419.191448] IPv6: ADDRCONF(NETDEV_UP): server01: link is not ready`
		`kernel: [77419.191452] 8021q: adding VLAN 0 to HW filter on device server01`
		`kernel: [77419.192060] server01: link status`

Layer 1 Logs	Log Location	Log Entries

		`definitely up for interface swp1, 1000 Mbps full duplex`
		`kernel: [77419.192065] server01: now running without any active interface!`
		`kernel: [77421.491811] IPv6: ADDRCONF (NETDEV_CHANGE): server01: link becomes ready`
		`mstpd: one_clag_cmd: setting (1) mac 44:38:39:00:00:17 <server01, None>`

	`/var/log/clagd.log`	`clagd[14003]: server01 is now dual connected.`

MLAG port negotiation
• Failing
	`/var/log/syslog`	`kernel: [79290.290999] bonding: server01 is being created...`
		`kernel: [79290.299645] 8021q: adding VLAN 0 to HW filter on device swp1`
		`kernel: [79290.301790] server01: Enslaving swp1 as a backup interface with a down link`
		`kernel: [79290.358294] server01: Setting ad_actor_system to 44:39:39:ff:40:94`
		`kernel: [79290.373590] server01: Warning: No 802.3ad response from the link partner for any adapters in the bond`
		`kernel: [79290.374024] IPv6: ADDRCONF(NETDEV_UP): server01: link is not ready`
		`kernel: [79290.374028] 8021q: adding VLAN 0 to HW filter on device server01`
		`kernel: [79290.375033] server01: link status definitely up for interface swp1, 1000 Mbps full duplex`
		`kernel: [79290.375037] server01: now running without any active interface!`

	Location	

	`/var/log/clagd.log`	`clagd[14291]: Conflict (server01): matching clag-id (1) not configured on peer`
		`...`
		`clagd[14291]: Conflict cleared (server01): matching clag-id (1) detected on peer`

MLAG port negotiation
• Flapping
	`/var/log/syslog`	`mstpd: one_clag_cmd: setting (0) mac 00:00:00:00:00:00 <server01, None>`
		`mstpd: one_clag_cmd: setting (1) mac 44:38:39:00:00:03 <server01, None>`

	`/var/log/clagd.log`	`clagd[14291]: server01 is no longer dual connected`
		`clagd[14291]: server01 is now dual connected.`

Prescriptive Topology Manager (PTM) uses LLDP information to compare against a topology.dot file that describes the network. It has built in alerting capabilities, so it is preferable to use PTM on box rather than polling LLDP information regularly. The PTM code is available on the Cumulus Networks github repository. Additional PTM, BFD and associated logs are documented in the code.

⚠ Peering information should be tracked through PTM. For more information, refer to the Prescriptive Topology Manager documentation.

Neighbor Element	Monitoring Command/s	Interval Poll
LLDP Neighbor	`cumulus@switch:~$ lldpctl -f json`	300 seconds

Prescriptive Topology Manager	`cumulus@switch:~$ ptmctl -j [-d]`	Triggered

Layer 2 Protocols

Spanning tree is a protocol that prevents loops in a layer 2 infrastructure. In a stable state, the spanning tree protocol should stably converge. Monitoring the Topology Change Notifications (TCN) in STP helps identify when new BPDUs were received.

Interface Counter Element	Monitoring Command/s	Interval Poll

STP TCN Transitions	`cumulus@switch:~$ mstpctl showbridge json`
	`cumulus@switch:~$ mstpctl showport json`
		60 seconds

MLAG/CLAG peer state	`cumulus@switch:~$ clagctl status`
	`cumulus@switch:~$ clagd -j`
	`cumulus@switch:~$ cat /var/log/clagd.log`
		60 seconds

MLAG/CLAG peer MACs	`cumulus@switch:~$ clagctl dumppeermacs`
	`cumulus@switch:~$ clagctl dumpourmacs`
		300 seconds

Layer 2 Logs	Log Location	Log Entries

Spanning Tree Working	`/var/log/syslog`	`kernel: [1653877.190724] device swp1 entered promiscuous mode`
		`kernel: [1653877.190796] device swp2 entered`

Layer 2 Logs	Log Location	Log Entries

		`promiscuous mode`
		`mstpd: create_br: Add bridge bridge`
		`mstpd: clag_set_sys_mac_br: set bridge mac 00:00:00:00:00:00`
		`mstpd: create_if: Add iface swp1 as port#2 to bridge bridge`
		`mstpd: set_if_up: Port swp1 : up`
		`mstpd: create_if: Add iface swp2 as port#1 to bridge bridge`
		`mstpd: set_if_up: Port swp2 : up`
		`mstpd: set_br_up: Set bridge bridge up`
		`mstpd: MSTP_OUT_set_state: bridge:swp1:0 entering blocking state(Disabled)`
		`mstpd: MSTP_OUT_set_state: bridge:swp2:0 entering blocking state(Disabled)`
		`mstpd: MSTP_OUT_flush_all_fids: bridge:swp1:0 Flushing forwarding database`
		`mstpd: MSTP_OUT_flush_all_fids: bridge:swp2:0 Flushing forwarding database`
		`mstpd: MSTP_OUT_set_state: bridge:swp1:0 entering learning state(Designated)`
		`mstpd: MSTP_OUT_set_state: bridge:swp2:0 entering learning state(Designated)`
		`sudo: pam_unix(sudo:session): session closed for user root`
		`mstpd: MSTP_OUT_set_state: bridge:swp1:0 entering forwarding state(Designated)`
		`mstpd: MSTP_OUT_set_state: bridge:swp2:0 entering forwarding state(Designated)`
		`mstpd: MSTP_OUT_flush_all_fids: bridge:swp2:0 Flushing forwarding database`
		`mstpd: MSTP_OUT_flush_all_fids: bridge:swp1:0 Flushing forwarding database`

Spanning Tree Blocking	`/var/log/syslog`	`mstpd: MSTP_OUT_set_state: bridge:swp2:0 entering blocking state(Designated)`
		`mstpd: MSTP_OUT_set_state: bridge:swp2:0 entering learning state(Designated)`
		`mstpd: MSTP_OUT_set_state: bridge:swp2:0`

Layer 2 Logs	Log Location	Log Entries

		`entering forwarding state(Designated)`
		`mstpd: MSTP_OUT_flush_all_fids: bridge:swp2:0`
		`Flushing forwarding database`
		`mstpd: MSTP_OUT_flush_all_fids: bridge:swp2:0`
		`Flushing forwarding database`
		`mstpd: MSTP_OUT_set_state: bridge:swp2:0`
		`entering blocking state(Alternate)`
		`mstpd: MSTP_OUT_flush_all_fids: bridge:swp2:0`
		`Flushing forwarding database`

Layer 3 Protocols

When Quagga boots up for the first time, there will be a different log file for each daemon that has been activated. If the log file is ever edited (ie. through vtysh or Quagga.conf), the integrated configuration sends all logs to the same file.

In order to send Quagga logs to syslog, apply the configuration **log syslog** in **vtysh**.

BGP

When monitoring BGP, c heck if BGP peers are operational. There is not much value in alerting on the current operational state of the peer as monitoring the transition is more valuable, and this is done by monitoring syslog.

Monitoring the routing table provides trending on the size of the infrastructure. This is especially useful when integrated with host based solutions (ie. RoH) when the routes track with the number of applications available.

BGP Element	Monitoring Command/s	Interval Poll

BGP peer failure	`cumulus@switch:~$ vtysh -c "show ip bgp summary json"`
	`cumulus@switch:~$ cl-bgp summary show json`
		60 seconds

BGP route table	`cumulus@switch:~$ vtysh -c "show ip bgp json"`
	`cumulus@switch:~$ cl-bgp route show`
		600 seconds

BGP Logs	Log Location	Log Entries
BGP peer down	`/var/log/syslog /var/log/quagga/*.log`	`bgpd[3000]: %NOTIFICATION: sent to neighbor swp1 4/0 (Hold Timer Expired) 0 bytes bgpd[3000]: %ADJCHANGE: neighbor swp1 Down BGP Notification send`

OSPF

When monitoring OSPF, check if OSPF peers are operational. There is not much value in alerting on the current operational state of the peer as monitoring the transition is more valuable, and this is done by monitoring syslog.

Monitoring the routing table provides trending on the size of the infrastructure. This is especially useful when integrated with host based solutions (ie. RoH) when the routes track with the number of applications available.

OSPF Element	Monitoring Command/s	Interval Poll

OSPF protocol peer failure	`cumulus@switch:~$ vtysh -c "show ip ospf neighbor all json"`
	`cumulus@switch:~$ cl-ospf summary show json`
		60 seconds

OSPF link state database	`cumulus@switch:~$ vtysh - c "show ip ospf database"`	600 seconds

Route and Host Entries

OSPF Element	Monitoring Command/s	Interval Poll
Host Entries		600 seconds

OSPF Element	Monitoring Command/s	Interval Poll

	`cumulus@switch:~$ cl-resource-query`
	`cumulus@switch:~$ cl-resource-query -k`	

Route Entries	`cumulus@switch:~$ cl-resource-query`
	`cumulus@switch:~$ cl-resource-query -k`
		600 seconds

Routing Logs

Layer 3 Logs	Log Location	Log Entries

Routing protocol process crash	`/var/log/syslog`	`watchquagga[3044]: bgpd state -> down : read returned EOF`
		`cumulus-core: Running cl-support for core files bgpd.3030.1470341944.core.core_helper`
		`core_check.sh[4992]: Please send /var/support/cl_support__spine01_20160804_201905.tar.xz to Cumulus support.`
		`watchquagga[5241]: Forked background command [pid 6665]: /usr/sbin/service quagga restart bgpd`
		`watchquagga[7719]: watchquagga 0.99.24+cl3u2 watching [zebra bgpd ospfd], mode [phased zebra restart]`
		`watchquagga[7719]: zebra state -> up : connect succeeded`
		`watchquagga[7719]: bgpd state -> up : connect succeeded`
		`watchquagga[7719]: ospfd state -> up : connect succeeded`

Logging

The table below covers the various log files, and what they should be used for:

OSPF Element	Monitoring Command/s	Log Location
Syslog	Catch all log file. Identifies memory leaks and CPU spikes.	`/var/log/syslog`
Switchd functionality	Hardware Abstraction Layer (HAL).	`/var/log/switchd.log`
Routing daemons	Quagga zebra daemon details	`/var/log/daemon.log`
Routing protocol	The log file is configurable in Quagga. When quagga first boots, it boots using the non-integrated config so each routing protocol has its own log file. After booting up, quagga switches over to using the integrated configuration which means that all logs go to a single place. To edit where log files go use the command **log file <location>**. By default, Quagga logs are not sent to syslog. This can be enabled using the command **log syslog <level>**. After this, logs go through rsyslog and into **/var/log/syslog**.	`/var/log/quagga/zebra.log` `/var/log/quag`

OSPF Element	Monitoring Command/s	Log Location
		`ga/{protocol}.log /var/log/quagga/Quagga.log`

Protocols and Services

NTP

Run the following command to confirm the NTP process is working correctly, and that the switch clock is synced with NTP:

```
cumulus@switch:~$ /usr/bin/ntpq -p
```

Device Management

Device Access Logs

Access Logs	Log Location	Log Entries
User Authentication and Remote Login	`/var/log/syslog`	`sshd[31830]: Accepted publickey for cumulus from 192.168.0.254 port 45582 ssh2: RSA 38:e6:3b:cc:04:ac:41:5e:c9:e3:93:9d:cc:9e:48:25 sshd[31830]: pam_unix(sshd:session): session opened for user cumulus by (uid=0)`

	Location	

Device Super User Command Logs

Super User Command Logs	Log Location	Log Entries

Executing commands using sudo	`/var/log/syslog`	`sudo:  cumulus : TTY=pts/2 ; PWD=/home/cumulus ; USER=root ; COMMAND=/bin/uname -a`
		`sudo: pam_unix(sudo:session): session opened for user root by cumulus(uid=0)`
		`sudo: pam_unix(sudo:session): session closed for user root`

Index