Centec V580 Hybrid Series Switch

User Guide

Issue	R1.2
Date	2019-06-10

Notice

The purchased products, services and features are stipulated by the contract made between Centec and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute the warranty of any kind, express or implied.

Centec Networks (Suzhou) Co., Ltd.

Address	#Suite 4F - 13/16, Building B, No.5 Xing Han Street, Suzhou Industrial Park, Suzhou, China
Telephone	86-512-62885358
Fax	86-512-62885870
Website	http://www.centecnetworks.com
Email	support@centecnetworks.com

Table of Contents

List of Tables

List of Figures

Revision History

Date	Version	Description
2018-06-20	R1.1	Update document for new product version
2019-06-10	R1.2	Update document for new product version

1 Basic Configuration Guide

1.1 Configuring System Management

1.1.1 Overview

Function Introduction

Banner function is used for configuring messages on the devices. User can specify any messages to notify other users. Improper operations might cause critical situation such as service interrupt, in this case, a notification in advance is necessary. (E.g. to notify users "Don't reboot")

Two types of messages are supported by now:

- login banner. Messages will display on the terminal when user login to the device. "Login mode" is required for displaying this message. Please reference the section of "Configuring User Management".
- exec banner. Messages will display on the terminal when user enter the EXEC mode.

Principle Description

This function displays notification on the terminal to reduce misoperation.

1.1.2 Configuration

Configuring a Login Banner

step 1 Enter the configure mode

```
Switch# configure terminal
```


step 2 Create the notification

User can create a notification (one line or multiple lines) to display on all connected terminals. "Login mode" is required for displaying this message. Please reference the section of "Configuring User Management".

In the following example, the delimiting character is @. All characters between two delimiting characters will display on the terminals when user connect the device.

The message length is at most 99 lines with 1023 character in each line.

```
banner login @admin-login@
```

step 3 Exit the configure mode

```
Switch(config)# exit
```

step 4 Validation

Use the following command to display the configuration

```
switch# show running-config
banner login ^C
 admin-login
^C
```

Configuring an Exec Banner

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create the notification

User can create a notification (one line or multiple lines) to display on all connected terminals. In the following example, the delimiting character is @. All characters between two delimiting characters will display on the terminals when user enter the EXEC mode.

The message length is at most 99 lines with 1023 character in each line.

```
Switch(config)# banner exec @do-not-reboot@
```


step 3 Exit the configure mode

```
Switch(config)# exit
```

step 4 Validation

Use the following command to display the configuration:

```
switch# show running-config
banner exec ^C
 do-not-reboot!
^C
```

1.1.3 Application cases

N/A

1.2 Configuring User Management

1.2.1 Overview

Function Introduction

User management increases the security of the system by keeping the unauthorized users from guessing the password. The user is limited to a specific number of attempts to successfully log in to the switch.

There are three load modes in the switch.

- In “no login” mode, anyone can load the switch without authentication.
- In “login” mode, there is only one default user.
- In “login local” mode, if you want to load the switch you need to have a user account. Local user authentication uses local user accounts and passwords that you create to validate the login attempts of local users. Each switch has a maximum of 32 local user accounts. Before you can enable local user authentication, you must define at least one local user account. You can set up local user accounts by creating a unique username and password combination for each local user. Each username must be fewer than 32 characters.

You can configure each local user account with a privilege level; the valid privilege levels are 1 or 4. Once a local user is logged in, only the commands those are available for that privilege level can be displayed. There is only one user can enter the configure mode at the same time.

The user privilege is defined as following description:

- Privilege1: In this level user only can use basic show command like, “ls”, “dir”, “enable”.
- Privilege2: In this level user can use all show command in Exec mode.
- Privilege3: In this level user can use command including “all PM configuration commands” in CONFIG mode.
- Privilege4: In this level user can use all command including commands that can change one user’s privilege”, “SNMP security commands”, “radius, ssh which related to security commands” and file management command in Exec mode.

If login type is login local, the privilege is form the privilege in user, other the privilege is form line vty.

Principle Description

N/A

1.2.2 Configuration

Configuring the user management in login local mode

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set username and password

```
Switch(config)# username testname privilege 4 password 123abc
```

step 3 Enter the configure mode and set user management mode

```
Switch(config)# line vty 0 7
Switch(config-line)# login local
Switch(config-line)# exit
```

step 4 Exit the configure mode

```
Switch(config)# exit
```


step 5 Validation

After the above setting, login the switch will need a username and password, and user can login with the username and password created before. This is a sample output of the login prompt.

```
Username:
```

After the input the username, a password is required.

```
Username: testname
Password:
```

Authentication succeed:

```
Password:

Switch#
```

Configuring the user service type

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create username and password

```
Switch(config)# username testname privilege 4 password 123abc
```

step 3 Configuring the user service type

```
Switch(config)#username testname service-type telnet
```

step 4 Exit the configure mode

```
Switch(config)# exit
```

step 5 Validation

Telnet is the only method for user which named "testname" to login the device.

Configuring the user management in login mode

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enter the configure mode and set password

```
Switch(config)# line vty 0 7
Switch(config-line)# line-password abc
Switch(config-line)# login
```

step 3 Exit the configure mode

```
Switch(config)# exit
```

step 4 Validation

After the above setting, login the switch will need the line password, and user can login with the password created before. This is a sample output of the login prompt.

```
Password:
```

Configuring Password recovery procedure

If the password is forgotten unfortunately, it can be recovered by following steps.

Step 1 Power on the system. Boot loader will start to run. The follow information will be printed on Console.

```
CPU:   MPC8247 (HiP7 Rev 14, Mask 1.0 1K50M) at 350 MHz
Board: 8247 (PCI Agent Mode)
I2C:   ready
DRAM:  256 MB
In:    serial
Out:   serial
Err:   serial
Net:   FCC1 ETHERNET, FCC2 ETHERNET [PRIME]
Press ctrl+b to stop autoboot: 3
```

Step 2 Press ctrl+b. stop autoboot.

```
Bootrom#
```


Step 3 Under boot loader interface, use the following instructions.

```
Bootrom# boot_flash_nopass
Bootrom# Do you want to  revert  to the default config file ? [Y|N|E]:
```

Please remember your username and password.

Recovering the password may lead configuration lost or service interrupted; we strongly recommend that user should remember the username and password.

1.2.3 Application cases

N/A

1.3 Configuring FTP

1.3.1 Overview

Function Introduction

You can download a switch configuration file from an FTP server or upload the file from the switch to an FTP server. You download a switch configuration file from a server to upgrade the switch configuration. You can overwrite the current startup configuration file with the new one. You upload a switch configuration file to a server for backup purposes. You can use this uploaded configuration for future downloads to the switch or another switch of the same type.

Principle Description

N/A

1.3.2 Configuration

You can copy configurations files to or from an FTP server. The FTP protocol requires a client to send a remote username and password on each FTP request to a server.

Before you begin downloading or uploading a configuration file by using FTP, do these tasks:

- Ensure that the switch has a route to the FTP server. The switch and the FTP server must be in the same network if you do not have a router to route traffic between subnets. Check connectivity to the FTP server by using the ping command.
- If you are accessing the switch through the console or a Telnet session and you do not have a valid username, make sure that the current FTP username is the one that you want to use for the FTP download.
- When you upload a configuration file to the FTP server, it must be properly configured to accept the write request from the user on the switch.

For more information, see the documentation for your FTP server.

FTP connection

Table 1-1 FTP Commands

Command	Description
ftp> ls	List all files in the user directory
ftp> put 1.txt	Upload file 1.txt in current directory to ftp server
ftp> get 1.txt	Download file 1.txt from ftp server to current directory
ftp> delete 1.txt	Delete file 1.txt in ftp server（have read and write server permissions）

Connect to IPv4 FTP server

```
DUT1# ftp mgmt-if 10.10.25.33
```

Downloading a configuration file by using FTP in IPv4 network

step 1 copy the configuration file

```
Switch# copy mgmt-if ftp://test:test@10.10.10.163/ startup-config.conf
flash:/startup-config.conf
```


step 2 Validation

Use the following command to display the configuration

```
Switch# show startup-config
```

Uploading a configuration file by using FTP in IPv4 network

step 1 copy the configuration file

```
Switch# copy flash:/startup-config.conf mgmt-if
ftp://test:test@10.10.10.163/startup-config.conf
```

1.3.3 Application cases

N/A

1.4 Configuring TFTP

1.4.1 Overview

Function Introduction

You can download a switch configuration file from a TFTP server or upload the file from the switch to a TFTP server. You download a switch configuration file from a server to upgrade the switch configuration. You can overwrite the current file with the new one. You upload a switch configuration file to a server for backup purposes; this uploaded file can be used for future downloads to the same or another switch of the same type.

Principle Description

N/A

1.4.2 Configuration

Before you begin downloading or uploading a configuration file by using TFTP, do these tasks:

- Ensure that the workstation acting as the TFTP server is properly configured.
- Ensure that the switch has a route to the TFTP server. The switch and the TFTP server must be in the same network if you do not have a router to route traffic

between subnets. Check connectivity to the TFTP server by using the ping command.

- Ensure that the configuration to be downloaded is in the correct directory on the TFTP server.
- For download operations, ensure that the permissions on the file are set correctly.
- During upload operations, if you are overwriting an existing file (including an empty file, if you had to create one) on the server, ensure that the permissions on the file are set correctly.

Downloading a configuration file by using TFTP in IPv4 network

```
Switch# copy mgmt-if tftp://10.10.10.163/startup-config.conf flash:/startup-config.conf
```

Uploading a configuration file by using TFTP in IPv4 network

```
Switch# copy flash:/startup-config.conf mgmt-if tftp://10.10.10.163/startup-config.conf
```

1.4.3 Application cases

N/A

1.5 Configuring Telnet

1.5.1 Overview

Function Introduction

Telnet is a network protocol used on the Internet or local area networks to provide a bidirectional interactive text-oriented communications facility using a virtual terminal connection. User data is interspersed in-band with Telnet control information in an 8-bit byte oriented data connection over the Transmission Control Protocol (TCP). Telnet was developed in 1969 beginning with RFC 15, extended in RFC 854, and standardized as Internet Engineering Task Force (IETF) Internet Standard STD 8, one of the first Internet standards. Historically, Telnet provided access to a command-line interface (usually, of an operating system) on a remote host. Most network equipment and operating systems with a TCP/IP stack support a Telnet service for remote configuration (including systems based on Windows NT).

Because of security issues with Telnet, its use for this purpose has waned in favor of SSH.

Principle Description

N/A

1.5.2 Configuration

Telnet switch with inner port

Example 1 IPv4 Network

```
Switch# telnet 10.10.29.247
Entering character mode
Escape character is '^]'.
Switch #
```

Telnet switch with management port

Example 1 IPv4 Network

```
Switch# telnet mgmt-if 10.10.29.247
Entering character mode
Escape character is '^]'.
Switch #
```

Configure telnet server

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enable Telnet service

```
Switch(config)# service telnet enable
```

step 3 Exit the configure mode

```
Switch(config)# exit
```

1.5.3 Application cases

N/A

1.6 Configuring SSH

1.6.1 Overview

Function Introduction

The Secure Shell (SSH) is a protocol that provides a secure, remote connection to a device. SSH provides more security for remote connections than Telnet does by providing strong encryption when a device is authenticated. SSH supports the Data Encryption Standard (DES) encryption algorithm, the Triple DES (3DES) encryption algorithm, and password-based user authentication. The SSH feature has an SSH server and an SSH integrated client, which are applications that run on the switch. You can use an SSH client to connect to a switch running the SSH server. The SSH server works with the SSH client supported in this release and with SSH clients. The SSH client also works with the SSH server supported in this release and with SSH servers.

Principle Description

N/A

1.6.2 Topo

Figure 1-1 SSH system application

1.6.3 Username&Password Login Configuration

Create username and password

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create username and password

```
Switch(config)# username testname privilege 4 password aaa
```

Use SSH to connect

```
[root@test1 tftpboot]# ssh testname@10.10.39.101
testname@10.10.39.101's password:
Switch#
```

1.6.4 Secret Key Login Configuration

Create key for SSH

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create a key

```
Switch(config)# rsa key a generate
```

step 3 Create a private key named a.pri with key a and save it to flash

```
Switch(config)# rsa key a export url flash:/a.pri private ssh2
```

step 4 Create a private key named a.pub with key a and save it to flash

```
Switch(config)# rsa key a export url flash:/a.pub public ssh2
```

step 5 Exit the configure mode

```
Switch(config)# exit
```

Import the key

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Import the key a.pub we created as importKey

```
Switch(config)# rsa key importKey import url flash:/a.pub public ssh2
```


step 3 Create username and password

```
Switch(config)# username aaa privilege 4 password abc
```

step 4 Assign the key to user aaa

```
Switch(config)# username aaa assign rsa key importKey
```

step 5 Exit the configure mode

```
Switch(config)# exit
```

Use SSH to connect

step 1 Download the a.pri key on SSH client

step 2 Connect to the client

```
[root@test1 tftpboot]# ssh -i a.pri aaa@10.10.39.101
aaa@10.10.39.101's password:
Switch#
```

1.6.5 Application cases

N/A

1.7 Configuring Time&timezone

1.7.1 Overview

Function Introduction

If no other source of time is available, you can manually configure the time and date after the system is restarted. The time remains accurate until the next system restart. We recommend that you use manual configuration only as a last resort. If you have an outside source to which the switch can synchronize, you do not need to manually set the system clock.

Principle Description

N/A

1.7.2 Configuration

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configuring time and timezone

```
Switch(config)# clock set datetime 11:30:00 10 26 2013
Switch(config)# clock set summer-time dst date 6 1 2013 02:00:00 10 31 2013
02:00:00 120
```

step 3 Exit the configure mode

```
Switch(config)# exit
```

step 4 Validation

Use the following command to display the information of time and date:

```
Switch# show clock
13:31:10 dst Sat Oct 26 2013
```

1.7.3 Application cases

N/A

2 Ethernet Configuration Guide

2.1 Configuring Interface

2.1.1 Overview

Function Introduction

Interface status, speed and duplex are configurable.

When the interface is configured as "no shutdown", it can work normally after cable is connected. When the interface is configured as "shutdown", no matter the cable is connected or not, the interface can not work.

If the device supports combo ports, user can choose to enable copper or fiber mode. The two modes of one port can not work together at same time. The configuration of speed or duplex at combo ports cannot be effective when combo port is working at fiber mode.

The rule of physical port name is as following: interface name format is eth-[slot]-[port]; [slot] is 0 for single pizza-box switch; when stacking is enabled, the [slot] number is according to the configuration. The [port] number is begin with 1, and increase from up to down, from left to right. The following figure shows the interface name of the device:

eth-0-1	eth-0-3	…	eth-0-23
eth-0-2	eth-0-4	…	eth-0-24

Figure 2-1 Interface Name

NOTE To get more information about the interface type and number, please reference to the product spec.

Principle Description

N/A

2.1.2 Configuration

Configuring Interface State

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Turn on an interface

```
Switch#(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
```

step 3 Shut down an interface

```
Switch(config)# interface eth-0-2
Switch(config-if-eth-0-2)# shutdown
```

step 4 Exit the configure mode

```
Switch(config-if-eth-0-2)# end
```

step 5 Validation

Use the following command to display the status of the interfaces:

```
Switch# show interface status
Name       Status      Duplex   Speed    Mode    Type         Description
----------+-----------+--------+--------+-------+------------+----------------
eth-0-1    up          full     10000    access  10GBASE_SR
eth-0-2    admin down  full     10000    access  10GBASE_SR
eth-0-3    up          full     10000    access  10GBASE_SR
eth-0-4    down        full     10000    access  10GBASE_SR
eth-0-5    down        full     10000    access  UNKNOWN
eth-0-6    down        full     10000    access  UNKNOWN
eth-0-7    down        full     10000    access  UNKNOWN
eth-0-8    down        full     10000    access  UNKNOWN
eth-0-9    down        full     10000    access  UNKNOWN
eth-0-10   down        full     10000    access  UNKNOWN
eth-0-11   down        full     10000    access  UNKNOWN
eth-0-12   down        full     10000    access  UNKNOWN
eth-0-13   down        full     10000    access  UNKNOWN
eth-0-14   down        full     10000    access  UNKNOWN
```



```
eth-0-15   down        full     10000    access   UNKNOWN
eth-0-16   down        full     10000    access   UNKNOWN
eth-0-17   down        full     10000    access   UNKNOWN
eth-0-18   down        full     10000    access   UNKNOWN
eth-0-19   down        full     10000    access   UNKNOWN
eth-0-20   down        full     10000    access   UNKNOWN
eth-0-21   down        full     10000    access   UNKNOWN
eth-0-22   down        full     10000    access   UNKNOWN
eth-0-23   down        full     10000    access   UNKNOWN
eth-0-24   down        full     10000    access   UNKNOWN
eth-0-25   down        full     10000    access   UNKNOWN
eth-0-26   down        full     10000    access   UNKNOWN
eth-0-27   down        full     10000    access   UNKNOWN
eth-0-28   down        full     10000    access   UNKNOWN
eth-0-29   down        full     10000    access   UNKNOWN
eth-0-30   down        full     10000    access   UNKNOWN
eth-0-31   down        full     10000    access   UNKNOWN
eth-0-32   down        full     10000    access   UNKNOWN
eth-0-33   down        full     10000    access   UNKNOWN
eth-0-34   down        full     10000    access   UNKNOWN
eth-0-35   down        full     10000    access   UNKNOWN
eth-0-36   down        full     10000    access   UNKNOWN
eth-0-37   down        full     10000    access   UNKNOWN
eth-0-38   down        full     10000    access   UNKNOWN
eth-0-39   down        full     10000    access   UNKNOWN
eth-0-40   down        full     10000    access   UNKNOWN
eth-0-41   down        full     10000    access   UNKNOWN
eth-0-42   down        full     10000    access   UNKNOWN
eth-0-43   down        full     10000    access   UNKNOWN
eth-0-44   down        full     10000    access   UNKNOWN
eth-0-45   down        full     10000    access   UNKNOWN
eth-0-46   down        full     10000    access   UNKNOWN
eth-0-47   down        full     10000    access   UNKNOWN
eth-0-48   down        full     10000    access   UNKNOWN
eth-0-49/1 up          full     10000    access   40GBASE_CR4
eth-0-49/2 up          full     10000    access   40GBASE_CR4
eth-0-49/3 admin down  full     10000    access   UNKNOWN
eth-0-49/4 admin down  full     10000    access   UNKNOWN
eth-0-50   down        full     40000    access   UNKNOWN
eth-0-51   down        full     40000    access   UNKNOWN
eth-0-52   down        full     40000    access   UNKNOWN
eth-0-53   down        full     40000    access   UNKNOWN
eth-0-54   down        full     40000    access   UNKNOWN
```

Configuring Interface Speed

step 1 Enter the configure mode

```
Switch# configure terminal
```


step 2 Enter the interface configure mode and set the speed

Set speed of interface eth-0-1 to 100M

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# speed 100
Switch(config-if-eth-0-1)# no shutdown
```

Set speed of interface eth-0-2 to 1000M

```
Switch(config-if-eth-0-1)# interface eth-0-2
Switch(config-if-eth-0-2)# no shutdown
Switch(config-if-eth-0-2)# speed 1000
```

Set speed of interface eth-0-3 to auto

```
Switch(config-if-eth-0-2)# interface eth-0-3
Switch(config-if-eth-0-3)# no shutdown
Switch(config-if-eth-0-3)# speed auto
```

step 3 Exit the configure mode

```
Switch(config-if-eth-0-3)# end
```

step 4 Validation

Use the following command to display the status of the interfaces:

```
Switch# show interface status
Name       Status     Duplex   Speed    Mode    Type         Description
----------+-----------+--------+--------+-------+------------+----------------
eth-0-1    up         a-full   100      access  10GBASE_SR
eth-0-2    up         full     1000     access  10GBASE_SR
eth-0-3    up         a-full   a-1000   access  10GBASE_SR
eth-0-4    up         full     10000    access  10GBASE_SR
eth-0-5    down       full     10000    access  UNKNOWN
eth-0-6    down       full     10000    access  UNKNOWN
eth-0-7    down       full     10000    access  UNKNOWN
eth-0-8    down       full     10000    access  UNKNOWN
eth-0-9    down       full     10000    access  UNKNOWN
eth-0-10   down       full     10000    access  UNKNOWN
eth-0-11   down       full     10000    access  UNKNOWN
eth-0-12   down       full     10000    access  UNKNOWN
eth-0-13   down       full     10000    access  UNKNOWN
eth-0-14   down       full     10000    access  UNKNOWN
eth-0-15   down       full     10000    access  UNKNOWN
eth-0-16   down       full     10000    access  UNKNOWN
eth-0-17   down       full     10000    access  UNKNOWN
eth-0-18   down       full     10000    access  UNKNOWN
eth-0-19   down       full     10000    access  UNKNOWN
eth-0-20   down       full     10000    access  UNKNOWN
eth-0-21   down       full     10000    access  UNKNOWN
eth-0-22   down       full     10000    access  UNKNOWN
```



```
eth-0-23   down        full    10000    access  UNKNOWN
eth-0-24   down        full    10000    access  UNKNOWN
eth-0-25   down        full    10000    access  UNKNOWN
eth-0-26   down        full    10000    access  UNKNOWN
eth-0-27   down        full    10000    access  UNKNOWN
eth-0-28   down        full    10000    access  UNKNOWN
eth-0-29   down        full    10000    access  UNKNOWN
eth-0-30   down        full    10000    access  UNKNOWN
eth-0-31   down        full    10000    access  UNKNOWN
eth-0-32   down        full    10000    access  UNKNOWN
eth-0-33   down        full    10000    access  UNKNOWN
eth-0-34   down        full    10000    access  UNKNOWN
eth-0-35   down        full    10000    access  UNKNOWN
eth-0-36   down        full    10000    access  UNKNOWN
eth-0-37   down        full    10000    access  UNKNOWN
eth-0-38   down        full    10000    access  UNKNOWN
eth-0-39   down        full    10000    access  UNKNOWN
eth-0-40   down        full    10000    access  UNKNOWN
eth-0-41   down        full    10000    access  UNKNOWN
eth-0-42   down        full    10000    access  UNKNOWN
eth-0-43   down        full    10000    access  UNKNOWN
eth-0-44   down        full    10000    access  UNKNOWN
eth-0-45   down        full    10000    access  UNKNOWN
eth-0-46   down        full    10000    access  UNKNOWN
eth-0-47   down        full    10000    access  UNKNOWN
eth-0-48   down        full    10000    access  UNKNOWN
eth-0-49/1 up          full    10000    access  40GBASE_CR4
eth-0-49/2 up          full    10000    access  40GBASE_CR4
eth-0-49/3 admin down  full    10000    access  UNKNOWN
eth-0-49/4 admin down  full    10000    access  UNKNOWN
eth-0-50   down        full    40000    access  UNKNOWN
eth-0-51   down        full    40000    access  UNKNOWN
eth-0-52   down        full    40000    access  UNKNOWN
eth-0-53   down        full    40000    access  UNKNOWN
eth-0-54   down        full    40000    access  UNKNOWN
```

Configuring Interface Duplex

There are 3 duplex mode supported on the device:

- full mode: the interface can transmit and receive packets at same time.
- half mode: the interface can transmit or receive packets at same time.
- auto mode: the interface should negotiate with the other side to decide the duplex mode.

User can choose proper duplex mode according to the network state.

step 1 Enter the configure mode

```
Switch# configure terminal
```


step 2 Enter the interface configure mode and set the duplex

Set duplex of interface eth-0-1 to full

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)# duplex full
```

Set duplex of interface eth-0-1 to half

```
Switch(config-if-eth-0-1)# interface eth-0-2
Switch(config-if-eth-0-2)# no shutdown
Switch(config-if-eth-0-2)# duplex half
```

Set duplex of interface eth-0-1 to auto

```
Switch(config-if-eth-0-2)# interface eth-0-3
Switch(config-if-eth-0-3)# no shutdown
Switch(config-if-eth-0-3)# duplex auto
```

step 3 Exit the configure mode

```
Switch(config-if-eth-0-3)# end
```

step 4 Validation

Use the following command to display the status of the interfaces:

```
Switch# show interface status
Name       Status      Duplex   Speed    Mode    Type         Description
----------+-----------+--------+--------+-------+------------+----------------
eth-0-1    up          full     100      access  10GBASE_SR
eth-0-2    up          half     1000     access  10GBASE_SR
eth-0-3    up          a-full   a-1000   access  10GBASE_SR
eth-0-4    up          full     10000    access  10GBASE_SR
eth-0-5    down        full     10000    access  UNKNOWN
eth-0-6    down        full     10000    access  UNKNOWN
eth-0-7    down        full     10000    access  UNKNOWN
eth-0-8    down        full     10000    access  UNKNOWN
eth-0-9    down        full     10000    access  UNKNOWN
eth-0-10   down        full     10000    access  UNKNOWN
eth-0-11   down        full     10000    access  UNKNOWN
eth-0-12   down        full     10000    access  UNKNOWN
eth-0-13   down        full     10000    access  UNKNOWN
eth-0-14   down        full     10000    access  UNKNOWN
eth-0-15   down        full     10000    access  UNKNOWN
eth-0-16   down        full     10000    access  UNKNOWN
eth-0-17   down        full     10000    access  UNKNOWN
eth-0-18   down        full     10000    access  UNKNOWN
eth-0-19   down        full     10000    access  UNKNOWN
eth-0-20   down        full     10000    access  UNKNOWN
eth-0-21   down        full     10000    access  UNKNOWN
eth-0-22   down        full     10000    access  UNKNOWN
```



```
eth-0-23    down        full     10000    access   UNKNOWN
eth-0-24    down        full     10000    access   UNKNOWN
eth-0-25    down        full     10000    access   UNKNOWN
eth-0-26    down        full     10000    access   UNKNOWN
eth-0-27    down        full     10000    access   UNKNOWN
eth-0-28    down        full     10000    access   UNKNOWN
eth-0-29    down        full     10000    access   UNKNOWN
eth-0-30    down        full     10000    access   UNKNOWN
eth-0-31    down        full     10000    access   UNKNOWN
eth-0-32    down        full     10000    access   UNKNOWN
eth-0-33    down        full     10000    access   UNKNOWN
eth-0-34    down        full     10000    access   UNKNOWN
eth-0-35    down        full     10000    access   UNKNOWN
eth-0-36    down        full     10000    access   UNKNOWN
eth-0-37    down        full     10000    access   UNKNOWN
eth-0-38    down        full     10000    access   UNKNOWN
eth-0-39    down        full     10000    access   UNKNOWN
eth-0-40    down        full     10000    access   UNKNOWN
eth-0-41    down        full     10000    access   UNKNOWN
eth-0-42    down        full     10000    access   UNKNOWN
eth-0-43    down        full     10000    access   UNKNOWN
eth-0-44    down        full     10000    access   UNKNOWN
eth-0-45    down        full     10000    access   UNKNOWN
eth-0-46    down        full     10000    access   UNKNOWN
eth-0-47    down        full     10000    access   UNKNOWN
eth-0-48    down        full     10000    access   UNKNOWN
eth-0-49/1 up           full     10000    access   40GBASE_CR4
eth-0-49/2 up           full     10000    access   40GBASE_CR4
eth-0-49/3 admin down   full     10000    access   UNKNOWN
eth-0-49/4 admin down   full     10000    access   UNKNOWN
eth-0-50    down        full     40000    access   UNKNOWN
eth-0-51    down        full     40000    access   UNKNOWN
eth-0-52    down        full     40000    access   UNKNOWN
eth-0-53    down        full     40000    access   UNKNOWN
eth-0-54    down        full     40000    access   UNKNOWN
```

2.1.3 Application cases

N/A

2.2 Configuring Layer3 Interfaces

2.2.1 Overview

Function Introduction

3 types of Layer3 interface are supported:

- VLAN interfaces: Logical interface with layer3 features. Connect different VLANs via IP address on the VLAN interface. VLAN interfaces can be created and deleted.

- Routed Ports: Ports are physical ports configured to be in Layer 3 mode by using the no switchport in interface configuration command.
- Layer 3 Link Aggregation Ports: Link Aggregation interfaces made up of routed ports.

A Layer 3 switch can have an IP address assigned to each routed port and VLAN interface. All Layer 3 interfaces require an IP address to route traffic. This section shows how to configure an interface as a Layer 3 interface and how to assign an IP address to an interface.

Principle Description

N/A

2.2.2 Configuration

Configuring Routed Port

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enter the interface configure mode and set IP address

```
Switch(config)# interface eth-0-1
Switch(config-if)# no switchport
Switch(config-if)# no shutdown
Switch(config-if)# ip address 1.1.1.1/24
```

step 3 Exit the configure mode

```
Switch(config-if)# end
```

step 4 Validation

Use the following command to display the brief status of the interfaces:

```
Switch# show ip interface brief
Interface              IP-Address      Status                 Protocol
eth-0-1                 1.1.1.1         up                      up
Switch# show ip interface
Interface eth-0-1
  Interface current state: UP
  Internet address(es):
```

```
   1.1.1.1/24 broadcast 1.1.1.255
 The maximum transmit unit is 1500 bytes
 ICMP redirects are always sent
 ARP timeout 01:00:00,  ARP retry interval 1s
 VRRP master of:  VRRP is not configured on this interface
```

Configuring VLAN Interfaces

This chapter describes configuring VLAN interfaces and using them. Several Virtual LAN (VLAN) interfaces can be configured on a single Ethernet interface. Once created, a VLAN interface functions the same as any physical interface, and it can be configured and displayed like any physical interface. Routing protocols, such as, RIP, OSPF and BGP can run across networks using VLAN interfaces.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create a vlan

```
Switch(config)# vlan 10
Switch(config-vlan10)# vlan 10
Switch(config-vlan10)# exit
```

step 3 Enter the interface configure mode and set switch port attributes

```
Switch(config)# interface eth-0-2
Switch(config-if)# switchport mode trunk
Switch(config-if)# switchport trunk allowed vlan all
Switch(config-if)# no shutdown
Switch(config-if)# exit
```

step 4 Enter the vlan interface configure mode and set IP address

```
Switch(config)# interface vlan10
Switch(config-if)# ip address 2.2.2.2/24
```

step 5 Exit the configure mode

```
Switch(config-if)# end
```

step 6 Validation

Use the following command to display the brief status of the interfaces:


```
Switch# show ip interface brief
Interface              IP-Address      Status                  Protocol
vlan10                 2.2.2.2         up                      up

Switch# show ip interface
Interface vlan10
  Interface current state: UP
  Internet address(es):
    2.2.2.2/24 broadcast 2.2.2.255
  The maximum transmit unit is 1500 bytes
  ICMP redirects are always sent
  ARP timeout 01:00:00,  ARP retry interval 1s
  VRRP master of :  VRRP is not configured on this interface
```

2.2.3 Application cases

N/A

2.3 Configuring Interface Errdisable

2.3.1 Overview

Function Introduction

Errdisable is a mechanism to protect the system through shutdown the abnormal interface. If an interface enters errdisable state, there are two ways to recovery it from errdisabled state. The first one is to enable errdisable recovery of this reason before errdisable detection; the interface will be recovered automatically after the configured time. But if errdisable occurred first, then errdisable recovery is enabled, the errdisable will not be recovered automatically. The secondary one is configuring “no shutdown” command on the errdisabled interface.

The flap of interface link state is a potential error caused by hardware or line problem. The administrator can also configure the detection conditions of interface link flap to suppress the flap.

Principle Description

N/A

2.3.2 Configuration

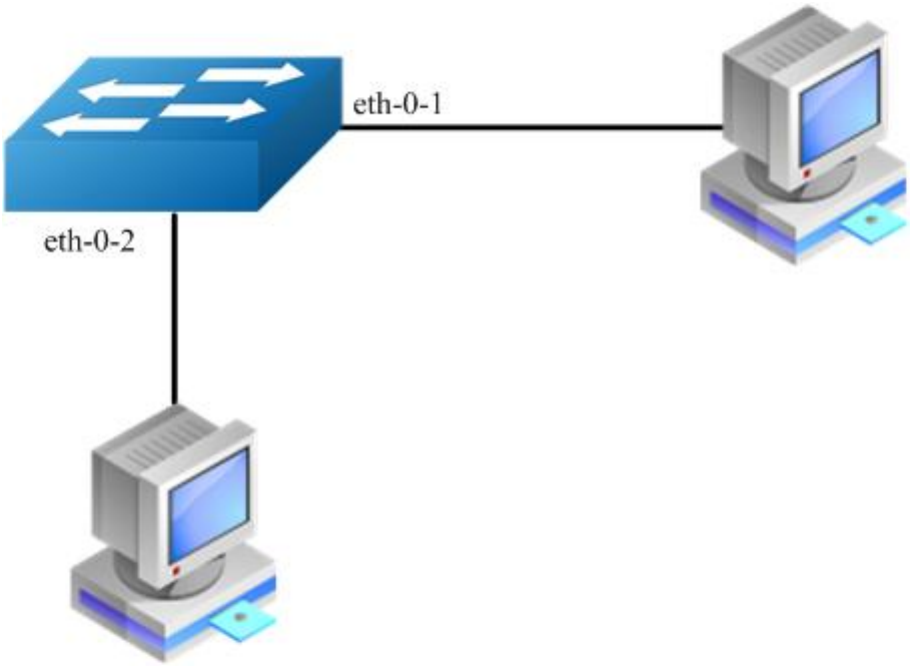

Figure 2-2 Errdisable topology

Configuring Errdisable Detection

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enable detect link flap errdisable

```
Switch(config)# errdisable detect reason link-flap
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

Use the following command to display the configuration of error disable:

```
Switch# show errdisable detect
ErrDisable Reason        Detection status
------------------------+--------------------
bpduguard                Enabled
bpduloop                 Enabled
```



```
port-security            Enabled
link-flap                Enabled
fdb-loop                 Disabled
```

Configuring Errdisable Recovery

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enable errdisable and set recovery interval

```
Switch(config)# errdisable recovery reason link-flap
Switch(config)# errdisable recovery interval 30
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

Use the following command to display the configuration of error disable recovery:

```
Switch# show errdisable recovery
ErrDisable Reason        Timer status
------------------------+--------------------
bpduguard                Disabled
bpduloop                 Disabled
port-security            Disabled
link-flap                Disabled
fdb-loop                 Disabled

Timer interval: 30 seconds
```

Configuring suppress Errdisable link Flap

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set link flap condition

```
 Switch(config)# errdisable flap reason link-flap 20 60
```


step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

Use the following command to display the configuration of error disable flap:

```
Switch# show errdisable flap
ErrDisable Reason        Flaps     Time (sec)
-----------------        ------    ----------
link-flap                20        60
```

Configuring Errdisable fdb-loop

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enable detect fdb-loop

```
Switch(config)# errdisable fdb-loop 40000 300
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

Use the following command to display the configuration of error disable:

```
Switch# show errdisable fdb-loop
Errdisable FDB loop information
Bucket Max Size:      40000
Bucket Token Rate:    300
Current Token Count:  40000
```

Send the packets with same source MAC(0.0.1) to eth-0-1 and eth-0-2 simultaneitily.

```
Switch# show errdisable fdb-loop
Errdisable FDB loop information
Bucket Max Size:      40000
Bucket Token Rate:    300
Current Token Count:  26543

Switch# show errdisable fdb-loop
Errdisable FDB loop information
Bucket Max Size:      40000
```



```
Bucket Token Rate:    300
Current Token Count:  8908

Switch# show errdisable fdb-loop
Errdisable FDB loop information
Bucket Max Size:      40000
Bucket Token Rate:    300
Current Token Count:  40000

Switch# show errdisable recovery
ErrDisable Reason        Timer status
------------------------+--------------------
port-security            Enabled
link-flap                Enabled
fdb-loop                 Enabled

Timer interval: 30 seconds

Interfaces that will be enabled at the next timeout:
Interface        ErrDisable Reason  Time Left(sec)
-----------------+------------------+---------
eth-0-2            Enabled            14
```

The eth-0-2 is errdisabled, but when the time left is 0, the eth-0-2 will be recovered.

Checking Errdisable Status

Administrator can check the interface errdisable status though two commands.

Case 1 Enable errdisable recovery

If link flap errdisable is enabled recovery, the command will display the left time for recovery， Otherwise, will display “unrecovery”.

```
Switch# show errdisable recovery
ErrDisable Reason        Timer Status
-----------------        --------------
bpduguard                Disabled
bpduloop                 Disabled
link-monitor-failure     Disabled
oam-remote-failure       Disabled
port-security            Disabled
link-flap                Enabled
udld                     Disabled
fdb-loop                 Disabled
loopback-detection       Disabled
Timer interval: 300 seconds
Interfaces that will be enabled at the next timeout:
Interface Errdisable Reason Time Left(sec)
```



```
--------- ----------------- --------------
eth-0-3   link-flap         25
```

Case 2 Disalbe errdisable recovery

```
Switch# show errdisable recovery
ErrDisable Reason        Timer Status
-----------------        --------------
bpduguard                Disabled
bpduloop                 Disabled
link-monitor-failure     Disabled
oam-remote-failure       Disabled
port-security            Disabled
link-flap                Disabled
udld                     Disabled
fdb-loop                 Disabled
loopback-detection       Disabled
Timer interval: 300 seconds
```

case 3 Display interface brief information to check errdisable state.

```
Switch# show interface status
Port       Status     Duplex   Speed    Mode    Type         Description
-----------------------------------------------------------------------------
eth-0-1    up         a-full   a-1000   TRUNK   1000BASE_SX
eth-0-2    down       auto     auto     TRUNK   Unknown
eth-0-3    errdisable a-full   a-1000   TRUNK   1000BASE_SX
eth-0-4    down       auto     auto     ACCESS  Unknown
```

2.3.3 Application cases

N/A

2.4 Configuring MAC Address Table

2.4.1 Overview

Function Introduction

MAC address table contains address information for the switch to forward traffic between ports. The address table includes these types of address:

- Dynamic address: the source address learnt by the switch and will be aged after aging time if this address is not hit. We only support IVL learning mode.
- Static address: the source address manually added by administrators.

Principle Description

Following is a brief description of terms and concepts used to describe the MAC address table:

- IVL: Independent VLAN Learning: for a given set of VLANs, if a given individual MAC Address is learned in one VLAN, it can't be used in forwarding decisions taken for that address relative to any other VLAN in the given set.
- SVL: Shared VLAN Learning: for a given set of VLANs, if an individual MAC Address is learned in one VLAN, it can be used in forwarding decisions taken for that address relative to all other VLANs in the given set.

Reference to standard: IEEE 802.1D，IEEE 802.1Q

2.4.2 Configuration

Configuring Address Aging Time

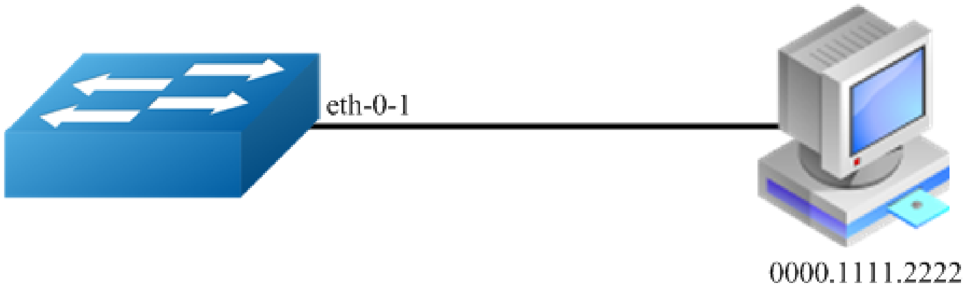

Figure 2-3 Mac address aging

The aging time is not exact time. If aging time set to N, then the dynamic address will be aged after N~2N interval. The default aging time is 300 seconds.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set dynamic address aging time

```
Switch(config)# mac-address-table ageing-time 10
```

step 3 Exit the configure mode

```
Switch(config)# end
```


step 4 Validation

Use the following command to display the aging time:

```
Switch# show mac address-table ageing-time
MAC address table ageing time is 10 seconds
```

Configuring Static Unicast Address

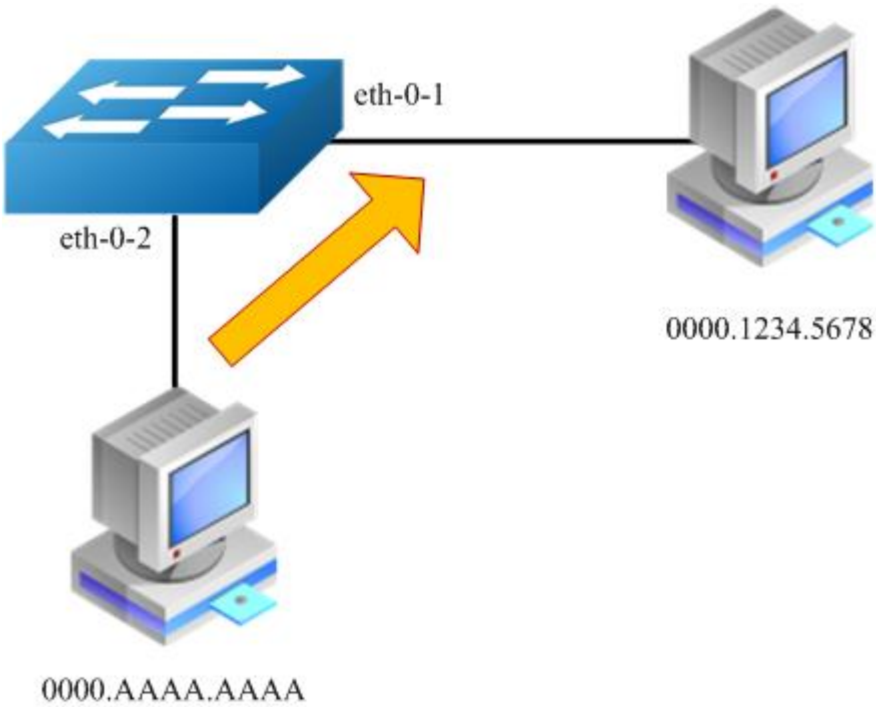

Figure 2-4 Static mac address table

Unicast address can be only bound to one port. According to the picture, Mac-Da 0000.1234.5678 should forward via eth-0-1.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set static mac address table

```
Switch(config)# mac-address-table 0000.1234.5678 forward eth-0-1 vlan 1
```

step 3 Exit the configure mode

```
Switch(config)# end
```


step 4 Validation

Use the following command to display the mac address table:

```
Switch# show mac-address-table
            Mac Address Table
----------------------------------------------
VLAN ID MAC Address      Type         Port
-------+----------------+------------+--------
1       0000.1234.5678   static       eth-0-1
```

Configuring MAC Filter Address

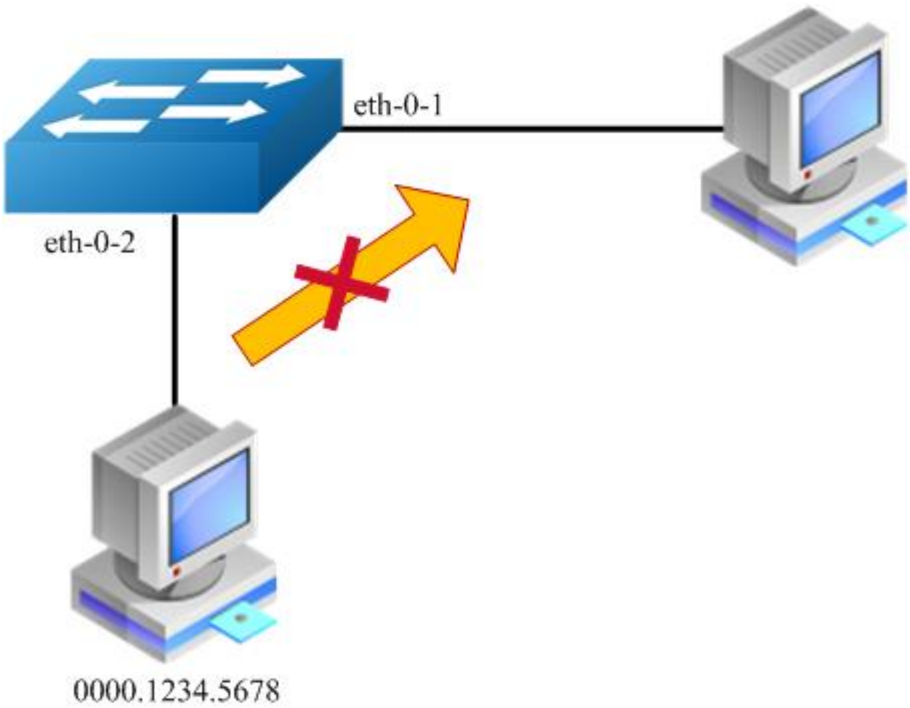

Figure 2-5 mac address filter

MAC filter will discard these frames whose source or destination address is set to discard. The MAC filter has higher priority than MAC address.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Add unicast address to be discarded

```
Switch(config)# mac-address-table 0000.1234.5678 discard
```


step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

Use the following command to display the mac address filter:

```
Switch# show macfilter address-table
MAC Filter Address Table
---------------------------------
Current count       : 0
Max count           : 128
Left count          : 128
---------------------------------
Mac Address:
0000.1234.5678
```

2.4.3 Application cases

N/A

2.5 Configuring VLAN

2.5.1 Overview

Function Introduction

VLAN (Virtual Local Area Network) is a switched network that is logically segmented the network into different broadcast domain so that packets are only switched between ports that are designated for the same VLAN. Each VLAN is considered as a logical network, and packets send to stations that do not belong to the same VLAN must be forwarded through a router.

Reference to standard: IEEE 802.1Q

Principle Description

Following is a brief description of terms and concepts used to describe the VLAN:

- VID: VLAN identifier
- LAN: Local Area Network
- VLAN: Virtual LAN

➢ PVID: Port VID, the untagged or priority-tagged frames will be assigned with this VID

Tagged Frame: Tagged Frame is inserted with 4 Bytes VLAN Tag, show in the picture below:

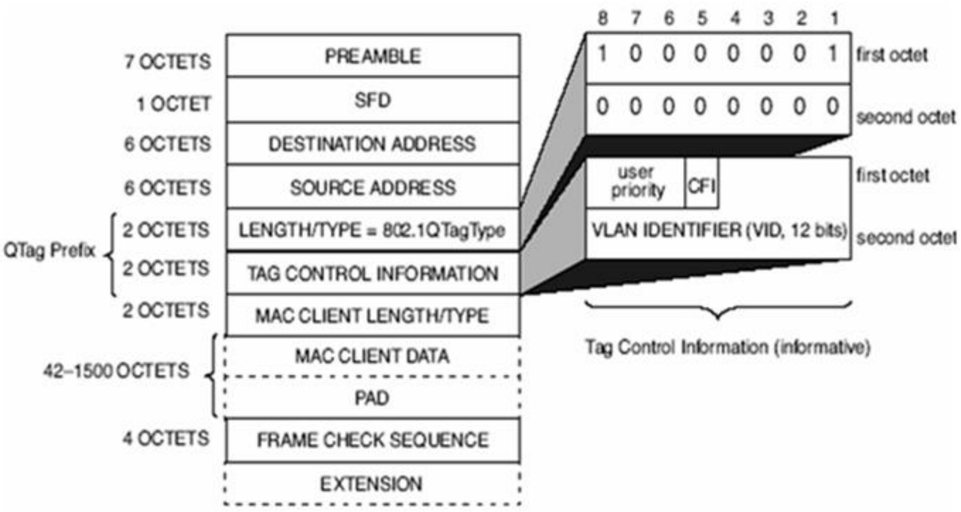

Figure 2-6 Tagged Frame

Trunk Link: Both tagged and untagged frames can be transmitted on this link. Trunk link allow for multiple VLANs to cross this link, show in the picture below:

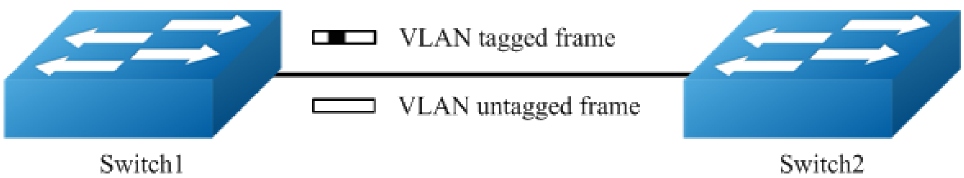

Figure 2-7 Trunk link

Access Link: Only untagged frames can be transmitted on this link. Access link is at the edge of the network, where end stations attach, show in the picture below:

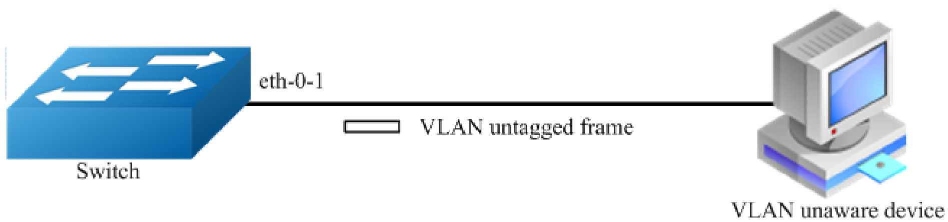

Figure 2-8 Access link

2.5.2 Configuration

Configuring Access Port

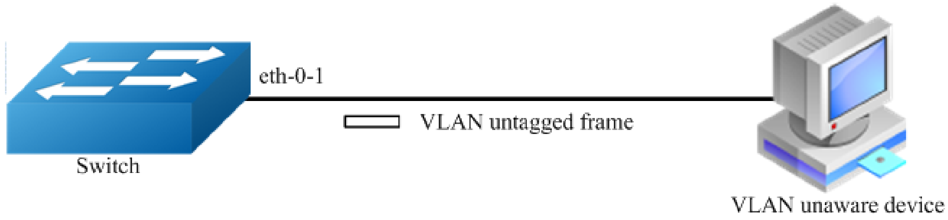

Figure 2-9 Access link

Access port only receives untagged or priority-tagged frames, and transmits untagged frames.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create vlan

```
Switch(config)# vlan 2
Switch(config-vlan2)# exit
```

step 3 Enter the interface configure mode, set the switch port mode and bind to the vlan

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# switchport mode access
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)# switchport access vlan 2
```

step 4 Exit the configure mode

```
Switch(config-if-eth-0-1)# end
```

step 5 Validation

Use the following command to display the information of the switch port interface:

```
Switch# show interface switchport interface eth-0-1
 Interface name          : eth-0-1
 Switchport mode         : access
 Ingress filter          : Enable
 Acceptable frame types  : vlan-untagged only
```



```
Default Vlan              : 2
Configured Vlans          : 2
```

Use the following command to display the vlan brief information:

```
Switch# show vlan 2
(u)-Untagged, (t)-Tagged
VLAN ID  Name             State    Instance Member ports
--------+----------------+--------+--------+----------------------------------
2        VLAN0002         Active   0        eth-0-1(u)
```

Configuring Trunk Port

Trunk port receives tagged, untagged, and priority-tagged frames, and transmits both untagged and tagged frames. If trunk port receives an untagged frame, this frame will be assigned to the VLAN of the trunk port's PVID; if a frame send out from the trunk port and the frame's VID is equal to the trunk port's PVID, this frame will be send out without VLAN tag.

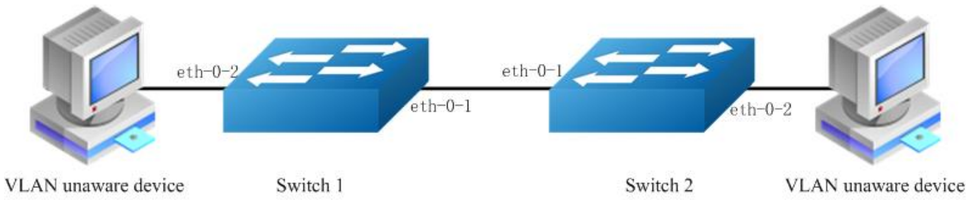

Figure 2-10 Trunk link

Network topology is shown in the picture above. The following configuration steps are same for Switch1 and Switch2.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create vlan

```
Switch(config)# vlan 10
Switch(config-vlan10)# exit
```

step 3 Enter the interface configure mode, set the switch port mode and bind to the vlan

Set eth-0-1's switch port mode as trunk, set native vlan as 10, and allow all VLANs on this interface:


```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)# switchport mode trunk
Switch(config-if-eth-0-1)# switchport trunk allowed vlan all
Switch(config-if-eth-0-1)# switchport trunk native vlan 10
Switch(config-if-eth-0-1)# exit
```

Set eth-0-2's switch port mode as access, and bind to vlan 10:

```
Switch(config)# interface eth-0-2
Switch(config-if-eth-0-2)# no shutdown
Switch(config-if-eth-0-2)# switchport access vlan 10
Switch(config-if-eth-0-2)# end
```

step 4 Exit the configure mode

```
Switch(config-if)# end
```

step 5 Validation

Use the following command to display the information of the switch port interface:

```
Switch# show interface switchport interface eth-0-1
 Interface name           : eth-0-1
 Switchport mode          : trunk
 Ingress filter           : Enable
 Acceptable frame types   : all
 Default Vlan             : 10
 Configured Vlans         : 1,10
```

Use the following command to display the vlan brief information:

```
Switch# show vlan 10
VLAN ID  Name             State    Instance Member ports
--------+----------------+--------+--------+-----------------------------------
10       VLAN0010         Active   0        eth-0-1(u)       eth-0-2(u)
```

2.5.3 Application cases

N/A

2.6 Configuring Link Aggregation

2.6.1 Overview

Function Introduction

This chapter contains a sample configuration of Link Aggregation Control Protocol (LACP) . LACP is based on the 802.3ad IEEE specification. It allows bundling of several physical interfaces to form a single logical channel providing enhanced performance and redundancy. The aggregated interface is viewed as a single link to each switch. The spanning tree views it as one interface. When there is a failure in one physical interface, the other interfaces stay up and there is no disruption. This implementation supports the aggregation of maximum 16 physical Ethernet links into a single logical channel. LACP enables our device to manage link aggregation group between other devices that conform to the 802.3ad protocol. By using the LACP, the switch learns the identity of partners supporting LACP and the capabilities of each port. It then dynamically groups ports with same properties into a single logical bundle link.

Reference to standard IEEE 802.3ad.

Principle Description

N/A

2.6.2 Configuration

Configure channel-group

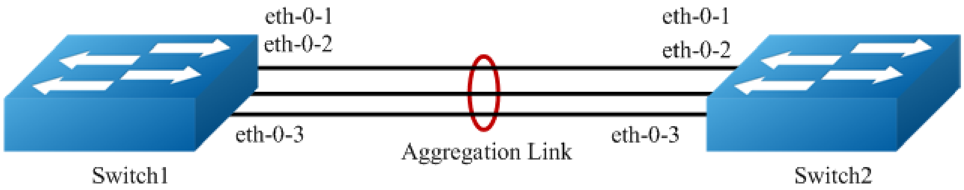

Figure 2-11 LACP

The configurations of Switch1 and Switch2 are as below:

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set the global attributes of LACP

Set the system priority of this switch. This priority is used for determining the system that is responsible for resolving conflicts in the choice of aggregation groups. A lower numerical value has a higher priority. Set the load balance mode. In this case we choose source MAC address for load balance.

Switch1 configuration:

```
Switch(config)# lacp system-priority 2000
Switch(config)# port-channel load-balance set src-mac
```

Switch2 configuration:

```
Switch(config)# lacp system-priority 2000
Switch(config)# port-channel load-balance set src-mac
```

step 3 Enter the interface configure mode and add the interface to the channel group

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-9)# no shutdown
Switch(config-if-eth-0-9)# channel-group 1 mode active
Switch(config-if-eth-0-9)# exit
Switch(config)# interface eth-0-2
Switch(config-if-eth-0-10)# channel-group 1 mode active
Switch(config-if-eth-0-10)# no shutdown
Switch(config-if-eth-0-10)# exit
Switch(config)# interface eth-0-3
Switch(config-if-eth-0-11)# channel-group 1 mode active
Switch(config-if-eth-0-11)# no shutdown
Switch(config-if-eth-0-11)# exit
```

step 4 Exit the configure mode

```
Switch(config)# end
```

step 5 Validation

Use the following command to display the information of the channel-group:

```
Switch# show channel-group summary
Port-channel load-balance hash-field-select:
    src-mac dst-mac src-ip dst-ip
Flags:  s - suspend           T - standby
        w - wait              B - in Bundle
        R - Layer3            S - Layer2
        D - down/admin down   U - in use

Aggregator  Protocol  Ports
-----------+---------+-------------------------------------------------------
agg1(SU)    LACP      eth-0-1(B)    eth-0-2(B)   eth-0-3(B)
DUT1# show interface agg1
Interface agg1
  Interface current state: UP
  Hardware is LAG, address is 4020.577c.4909
  Bandwidth 120000000 kbits
  Index 2049 , Metric 1
  Speed - 40Gb/s ,  Duplex - full ,  Media type is Aggregation
```



```
Link speed type is force link,   Link duplex type is force link
Admin input flow-control is off, output flow-control is off
Oper input flow-control is off, output flow-control is off
The Maximum Frame Size is 1632 bytes
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
  0 packets input, 0 bytes
  Received 0 unicast, 0 broadcast, 0 multicast
  0 runts, 0 giants, 0 input errors, 0 CRC
  0 frame, 0 overrun, 0 pause input
  0 packets output, 0 bytes
  Transmitted 0 unicast, 0 broadcast, 0 multicast
  0 underruns, 0 output errors, 0 pause output
```

Configuring Static-channel-group

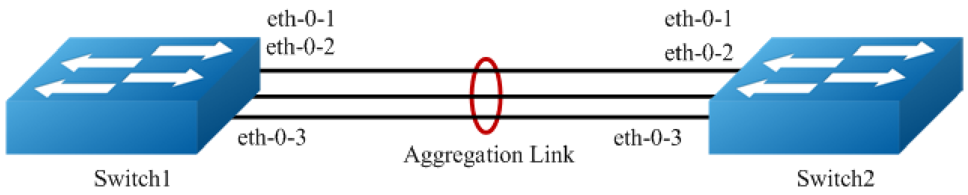

Figure 2-12 Static Agg

The configurations of Switch1 and Switch2 are as below:

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enter the interface configure mode and add the interface to the channel group

```
Switch# configure terminal
Switch(config)# interface eth-0-9
Switch(config-if-eth-0-9)# no shutdown
Switch(config-if-eth-0-9)# static-channel-group 1
Switch(config-if-eth-0-9)# exit
Switch(config)# interface eth-0-10
Switch(config-if-eth-0-10)# static-channel-group 1
Switch(config-if-eth-0-10)# no shutdown
Switch(config-if-eth-0-10)# exit
Switch(config)# interface eth-0-11
Switch(config-if-eth-0-11)# static-channel-group 1
Switch(config-if-eth-0-11)# no shutdown
Switch(config-if-eth-0-11)# exit
```


step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

Use the following command to display the information of the channel-group:

```
Switch# show channel-group summary
Port-channel load-balance hash-field-select:
    dst-mac src-ip dst-ip
Flags:  s - suspend           T - standby
        w - wait              B - in Bundle
        R - Layer3            S - Layer2
        D - down/admin down   U - in use

Aggregator  Protocol  Ports
-----------+---------+----------------------------------------------------
agg1(SU)    Static    eth-0-1(B)    eth-0-2(B)   eth-0-3(B)
```

2.6.3 Application cases

N/A

2.7 Configuring MSTP

2.7.1 Overview

Function Introduction

The MSTP (Multiple Spanning Tree Algorithm and Protocol (IEEE 802.1Q-2005)) enables multiple VLANs to be mapped to the same spanning-tree instance, thereby reducing the number of spanning-tree instances needed to support a large number of VLANs. The MSTP provides for multiple forwarding paths for data traffic and enables load balancing. It improves the fault tolerance of the network because a failure in one instance (forwarding path) does not affect other instances (forwarding paths). The most common initial deployment of MSTP is in the backbone and distribution layers of a Layer 2 switched network; this deployment provides the highly-available network required in a service-provider environment.

When the switch is in the multiple spanning-tree (MST) modes, the Rapid Spanning Tree Protocol (RSTP), which is based on IEEE 802.1w, is automatically enabled. The RSTP provides rapid convergence of the spanning tree through explicit handshaking that eliminates the IEEE 802.1D forwarding delay and quickly transitions root ports and designated ports to the forwarding state.

MSTP bridge port has 5 roles:

- Root - A forwarding port that is the best port from non-root bridge to root bridge
- Master - A port provides connectivity from the Region to a CIST Root of outside the Region
- Designated - A forwarding port for every LAN segment
- Alternate - An alternate path to the root bridge
- Backup - A backup/redundant path to a segment where another bridge port already connects
- Disabled - Not strictly part of STP, a network administrator can manually disable a port

MSTP switch port has 3 states:

- Discarding - No user data is sent over the port
- Learning - The port is not forwarding frames, but is populating its MAC-address-table
- Forwarding - The port is fully operational

Principle Description

Reference to IEEE 802.1Q-2005

2.7.2 Configuration

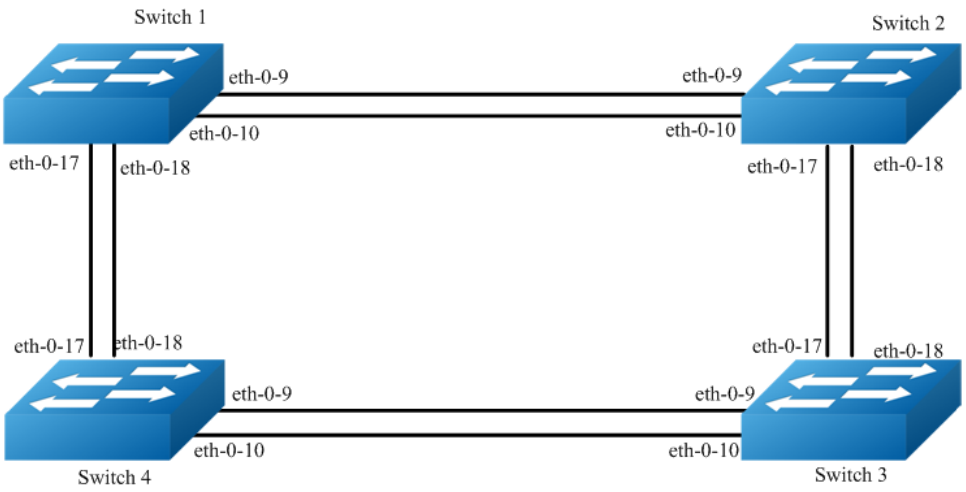

Figure 2-13 MSTP

The configurations of Switch1-Switch4 are as blow. The configurations of these 4 Switches are same if there is no special description.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set the mode of STP

```
Switch(config)# spanning-tree mode mstp
```

step 3 Create vlan

```
Switch(config)# vlan range 10,20
```

step 4 Enter the MSTP configure mode， create region and instance. Bind the vlan to the instance.

```
Switch(config)# spanning-tree mode mstp
Switch(config)# spanning-tree enable
Switch(config)# spanning-tree mst configuration
Switch(config-mst)# region RegionName
Switch(config-mst)# instance 1 vlan 10
Switch(config-mst)# instance 2 vlan 20
Switch(config-mst)# exit
```


step 5 Enter the interface configure mode, set the attributes of the interfaces

```
Switch(config)# interface range eth-0-9,eth-0-10,eth-0-17,eth-0-18
Switch(config-if-range)# switchport mode trunk
Switch(config-if-range)# switchport trunk allowed vlan all
Switch(config-if-range)# no shutdown
Switch(config-if-range)# exit
```

step 6 Enable STP and set priority for each swicth

Switch1:

```
Switch# configure terminal
Switch(config)# spanning-tree priority 0
```

Switch2:

```
Switch# configure terminal
Switch(config)# spanning-tree instance 1 priority 0
```

Switch3:

```
Switch# configure terminal
Switch(config)# spanning-tree instance 2 priority 0
```

step 7 Exit the configure mode

```
Switch(config)# end
```

step 8 Validation

Use the following command to display the information of MSTP on Switch1:

```
Switch# show spanning-tree mst brief
-----[Spanning-tree Enabled][Mode MSTP]-----
##### MST0   : 1
Root ID     Priority    0 (0x0000)
            Address     6682.2594.1000
            Hello Time 2  sec  Max Age 20 sec  Forward Delay 15 sec

Reg Root ID Priority    0 (0x0000)
            Address     6682.2594.1000

Bridge ID   Priority    0 (0x0000)
            Address     6682.2594.1000
            Hello Time 2  sec  Max Age 20 sec  Forward Delay 15 sec
            Aging Time 300 sec

Interface    Role         State             Cost        Priority.Number   Type
------------+------------+-----------------+--------------------+--------+------
eth-0-9      Designated   Forwarding        500              128.9        P2p
eth-0-10     Designated   Forwarding        500              128.10       P2p
```



```
eth-0-17     Designated   Forwarding        500             128.17       P2p
eth-0-18     Designated   Forwarding        500             128.18       P2p

##### MST1   : 10
Root ID      Priority     1 (0x0001)
             Address      4ebd.d541.b200
Bridge ID    Priority     32769 (0x8001)
             Address      6682.2594.1000
Interface    Role         State             Cost         Priority.Number   Type
------------+------------+-----------------+--------------------+--------+------
eth-0-9      Rootport     Forwarding        500             128.9        P2p
eth-0-10     Alternate    Discarding        500             128.10       P2p
eth-0-17     Designated   Forwarding        500             128.17       P2p
eth-0-18     Designated   Forwarding        500             128.18       P2p

##### MST2   : 20
Root ID      Priority     2 (0x0002)
             Address      d86b.60dc.6400
Bridge ID    Priority     32770 (0x8002)
             Address      6682.2594.1000
Interface    Role         State             Cost         Priority.Number   Type
------------+------------+-----------------+--------------------+--------+------
eth-0-9      Rootport     Forwarding        500             128.9        P2p
eth-0-10     Alternate    Discarding        500             128.10       P2p
eth-0-17     Alternate    Discarding        500             128.17       P2p
eth-0-18     Alternate    Discarding        500             128.18       P2p
```

Use the following command to display the information of MSTP on Switch2:

```
Switch# show spanning-tree mst brief
-----[Spanning-tree Enabled][Mode MSTP]-----
##### MST0   : 1
Root ID      Priority     0 (0x0000)
             Address      6682.2594.1000
             Hello Time 2  sec  Max Age 20 sec  Forward Delay 15 sec

Reg Root ID Priority      0 (0x0000)
             Address      6682.2594.1000

Bridge ID    Priority     32768 (0x8000)
             Address      4ebd.d541.b200
             Hello Time 2  sec  Max Age 20 sec  Forward Delay 15 sec
             Aging Time 300 sec

Interface    Role         State             Cost         Priority.Number   Type
------------+------------+-----------------+--------------------+--------+------
eth-0-9      Rootport     Forwarding        500             128.9        P2p
eth-0-10     Alternate    Discarding        500             128.10       P2p
eth-0-17     Designated   Forwarding        500             128.17       P2p
eth-0-18     Designated   Forwarding        500             128.18       P2p

##### MST1   : 10
Root ID      Priority     1 (0x0001)
             Address      4ebd.d541.b200
Bridge ID    Priority     1 (0x0001)
             Address      4ebd.d541.b200
```



```
Interface    Role        State           Cost        Priority.Number   Type
------------+-----------+----------------+-------------------+--------+------
eth-0-9      Designated  Forwarding      500              128.9        P2p
eth-0-10     Designated  Forwarding      500              128.10       P2p
eth-0-17     Designated  Forwarding      500              128.17       P2p
eth-0-18     Designated  Forwarding      500              128.18       P2p

##### MST2   : 20
Root ID      Priority    2 (0x0002)
             Address     d86b.60dc.6400
Bridge ID    Priority    32770 (0x8002)
             Address     4ebd.d541.b200
Interface    Role        State           Cost        Priority.Number   Type
------------+-----------+----------------+-------------------+--------+------
eth-0-9      Designated  Forwarding      500              128.9        P2p
eth-0-10     Designated  Forwarding      500              128.10       P2p
eth-0-17     Rootport    Forwarding      500              128.17       P2p
eth-0-18     Alternate   Discarding      500              128.18       P2p
```

Use the following command to display the information of MSTP on Switch3:

```
Switch# show spanning-tree mst brief
-----[Spanning-tree Enabled][Mode MSTP]-----
##### MST0   : 1
Root ID      Priority    0 (0x0000)
             Address     6682.2594.1000
             Hello Time 2  sec  Max Age 20 sec  Forward Delay 15 sec

Reg Root ID Priority    0 (0x0000)
             Address     6682.2594.1000

Bridge ID    Priority    32768 (0x8000)
             Address     d86b.60dc.6400
             Hello Time 2  sec  Max Age 20 sec  Forward Delay 15 sec
             Aging Time 300 sec

Interface    Role        State           Cost        Priority.Number   Type
------------+-----------+----------------+-------------------+--------+------
eth-0-9      Alternate   Discarding      500              128.9        P2p
eth-0-10     Alternate   Discarding      500              128.10       P2p
eth-0-17     Rootport    Forwarding      500              128.17       P2p
eth-0-18     Alternate   Discarding      500              128.18       P2p

##### MST1   : 10
Root ID      Priority    1 (0x0001)
             Address     4ebd.d541.b200
Bridge ID    Priority    32769 (0x8001)
             Address     d86b.60dc.6400
Interface    Role        State           Cost        Priority.Number   Type
------------+-----------+----------------+-------------------+--------+------
eth-0-9      Designated  Forwarding      500              128.9        P2p
eth-0-10     Designated  Forwarding      500              128.10       P2p
eth-0-17     Rootport    Forwarding      500              128.17       P2p
eth-0-18     Alternate   Discarding      500              128.18       P2p

##### MST2   : 20
```



```
Root ID     Priority    2 (0x0002)
            Address     d86b.60dc.6400
Bridge ID   Priority    2 (0x0002)
            Address     d86b.60dc.6400
Interface    Role         State             Cost        Priority.Number   Type
------------+------------+-----------------+--------------------+--------+------
eth-0-9      Designated   Forwarding        500              128.9         P2p
eth-0-10     Designated   Forwarding        500              128.10        P2p
eth-0-17     Designated   Forwarding        500              128.17        P2p
eth-0-18     Designated   Forwarding        500              128.18        P2p
```

Use the following command to display the information of MSTP on Switch4:

```
Switch# show spanning-tree mst brief
-----[Spanning-tree Enabled][Mode MSTP]-----
##### MST0   : 1
Root ID     Priority    0 (0x0000)
            Address     6682.2594.1000
            Hello Time 2  sec  Max Age 20 sec  Forward Delay 15 sec

Reg Root ID Priority    0 (0x0000)
            Address     6682.2594.1000

Bridge ID   Priority    32768 (0x8000)
            Address     945e.43e8.b100
            Hello Time 2  sec  Max Age 20 sec  Forward Delay 15 sec
            Aging Time 300 sec

Interface    Role         State             Cost        Priority.Number   Type
------------+------------+-----------------+--------------------+--------+------
eth-0-9      Designated   Forwarding        500              128.9         P2p
eth-0-10     Designated   Forwarding        500              128.10        P2p
eth-0-17     Rootport     Forwarding        500              128.17        P2p
eth-0-18     Alternate    Discarding        500              128.18        P2p

##### MST1   : 10
Root ID     Priority    1 (0x0001)
            Address     4ebd.d541.b200
Bridge ID   Priority    32769 (0x8001)
            Address     945e.43e8.b100
Interface    Role         State             Cost        Priority.Number   Type
------------+------------+-----------------+--------------------+--------+------
eth-0-9      Alternate    Discarding        500              128.9         P2p
eth-0-10     Alternate    Discarding        500              128.10        P2p
eth-0-17     Rootport     Forwarding        500              128.17        P2p
eth-0-18     Alternate    Discarding        500              128.18        P2p

##### MST2   : 20
Root ID     Priority    2 (0x0002)
            Address     d86b.60dc.6400
Bridge ID   Priority    32770 (0x8002)
            Address     945e.43e8.b100
Interface    Role         State             Cost        Priority.Number   Type
------------+------------+-----------------+--------------------+--------+------
eth-0-9      Rootport     Forwarding        500              128.9         P2p
eth-0-10     Alternate    Discarding        500              128.10        P2p
```



```
eth-0-17      Designated    Forwarding          500             128.17        P2p
eth-0-18      Designated    Forwarding          500             128.18        P2p
```

2.7.3 Application cases

N/A

3 IP Service Configuration Guide

3.1 Configuring Arp

3.1.1 Overview

Function Introduction

The Address Resolution Protocol (ARP) is a protocol used to dynamically map between Internet host addresses and Ethernet addresses. ARP caches Internet-Ethernet address mappings. When an interface requests a mapping for an address not in the cache, ARP queues the message, which requires the mapping, and broadcasts a message on the associated network requesting the address mapping. If a response is provided, the new mapping is cached and any pending message is transmitted. ARP will queue at most one packet while waiting for a response to a mapping request; only the most recently transmitted packet is kept. If the target host does not respond after 3 requests, the host is considered to be down, allowing an error to be returned to transmission attempts during this interval. If a target host does not send message for a period (normally one hour), the host is considered to be uncertainty, and several requests (normally 6, 3 unicast and 3 broadcast) will send to the host before delete the ARP entry. ARP entries may be added, deleted or changed manually. Manually added entries may be temporary or permanent.

Principle Description

N/A

3.1.2 Configuration

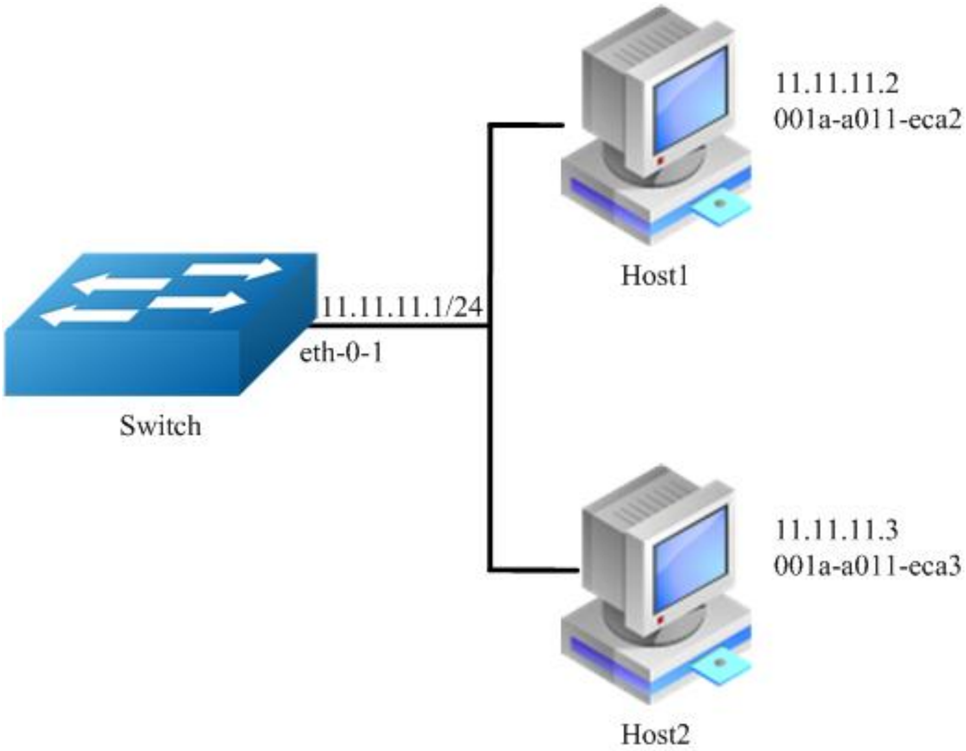

Figure 3-1 arp

In this configuration example, interface eth-0-1 assigned with address 11.11.11.1/24, on subnet 11.11.11.0/24, there are two hosts, and their IP addresses are 11.11.11.2, 11.11.11.3, MAC address are 001a-a011-eca2, 001a-a011-eca3. ARP entry of host 11.11.11.2 is added manually, the entry of host 11.11.11.3 is added dynamically. Time-out period of ARP entries for interface eth-0-1 configure to 40 minutes, ARP request retry delay on interface eth-0-1 configure to 2 seconds.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configure the layer 3 interface and set the ip address

```
Switch (config)# interface eth-0-1
Switch (config-if-eth-0-1)# no switchport
Switch (config-if-eth-0-1)# no shutdown
Switch (config-if-eth-0-1)# ip address 11.11.11.1/24
```


step 3 Configure ARP aging timeout value and the ARP retry interval value

```
Switch (config-if-eth-0-1)# arp retry-interval 2
Switch (config-if-eth-0-1)# arp timeout 2400
Switch (config-if-eth-0-1)# exit
```

step 4 Add a static ARP entry

```
Switch (config)# arp 11.11.11.2 001a.a011.eca2
```

step 5 Exit the configure mode

```
Switch(config)# end
```

step 6 Validation

Use the following command to display the information of the ARP entry:

```
Switch# show ip arp
Protocol   Address          Age (min)  Hardware Addr    Interface
 ----------+----------------+----------+----------------+----------
Internet   11.11.11.1       -          001e.080a.544b   eth-0-1
Internet   11.11.11.2       -          001a.a011.eca2   eth-0-1
Internet   11.11.11.3       1          001a.a011.eca3   eth-0-1
```

Use the following command to display the information of the ARP configurations on the interface:

```
Switch# show interface eth-0-1
Interface eth-0-1
  Interface current state: UP
  Hardware is Port, address is 001e.080a.544b
  Bandwidth 10000000 kbits
  Index 13 , Metric 1
  Speed - auto ,  Duplex - auto ,  Media type is 10GBASE_SR
  Link speed type is autonegotiation,   Link duplex type is autonegotiation
  Admin input flow-control is off, output flow-control is off
  Oper input flow-control is off, output flow-control is off
  The Maximum Frame Size is 1632 bytes
  VRF binding: not bound
  No Virtual private Wire service configured
  ARP timeout is 2400s ,  ARP retry interval 2s
  ARP Proxy is disabled, Local ARP Proxy is disabled
  The maximum transmit unit (MTU) is 1514 bytes
  Internet primary address:
    11.11.11.1/24 broadcast 11.11.11.255
    5 minute input rate 0 bits/sec, 0 packets/sec
    5 minute output rate 0 bits/sec, 0 packets/sec
    0 packets input, 0 bytes
    Received 0 unicast, 0 broadcast, 0 multicast
    0 runts, 0 giants, 0 input errors, 0 CRC
```



```
    0 frame, 0 overrun, 0 pause input
    0 packets output, 0 bytes
    Transmitted 0 unicast, 0 broadcast, 0 multicast
    0 underruns, 0 output errors, 0 pause output
```

3.1.3 Application cases

N/A

3.2 Configuring Arp limit

3.2.1 Overview

Function Introduction

ARP number limit feature is used to limit the number of dynamic ARP entry on L3 port. After the number of ARP entry reaches the limit for the number, the new ARP address can't be learnt on the interface. At this moment, it is considered as ARP number-limit violation.

If a number-limit violation occurs, the packets to be forwarded will be dropped.

There are three number-limit violation mode: Protect/Restrict/Errdisable

- If the violation is protect, the interface will drop packets directly if violation happened.
- If the violation is restrict, the interface will drop packets and print log if violation happened.
- If the violation is errdisable, the interface will not only drop packets and print log, the interface also errdisabled if violation happened.

Note: If the number-limit is enabled on the L3 port, when modify the number-limit maximum of the L3 port, the learnt dynamic ARP will be cleared first and learn the dynamic ARP again.

If change the number-limit enable/disable status of L3 port, the learnt dynamic ARP also will be cleared first and learn the dynamic ARP again.

The default value of number-limit maximum of every L3 port is 512.

ARP rate limit feature is used to limit the number of ARP packet receives from L3 port in 30s. If the number of ARP packet receives from one L3 port reaches the configured value, it is considered as ARP attacks occurs.

If an ARP attack occurs, the packets to be forwarded will be dropped.

There are two mode processes ARP attacks: Restrict, Errdisable

- If the action is restrict, the interface will drop packets and print log if receive packets number in 30s is more than configured value.
- If the action is errdisable, the interface will not only drop packets and print log, the interface also errdisabled if receive packets number in 30s is more than configured value.

Note: If the rate-limit enable/disable status change, modify the rate-limit maximum or modify violation mode of the L3 port, the packet statistics and abnormal flag will be cleared.

The default value of rate-limit maximum of every L3 port is 150.

Principle Description

N/A

3.2.2 Configuration

Configuration ARP Number limit

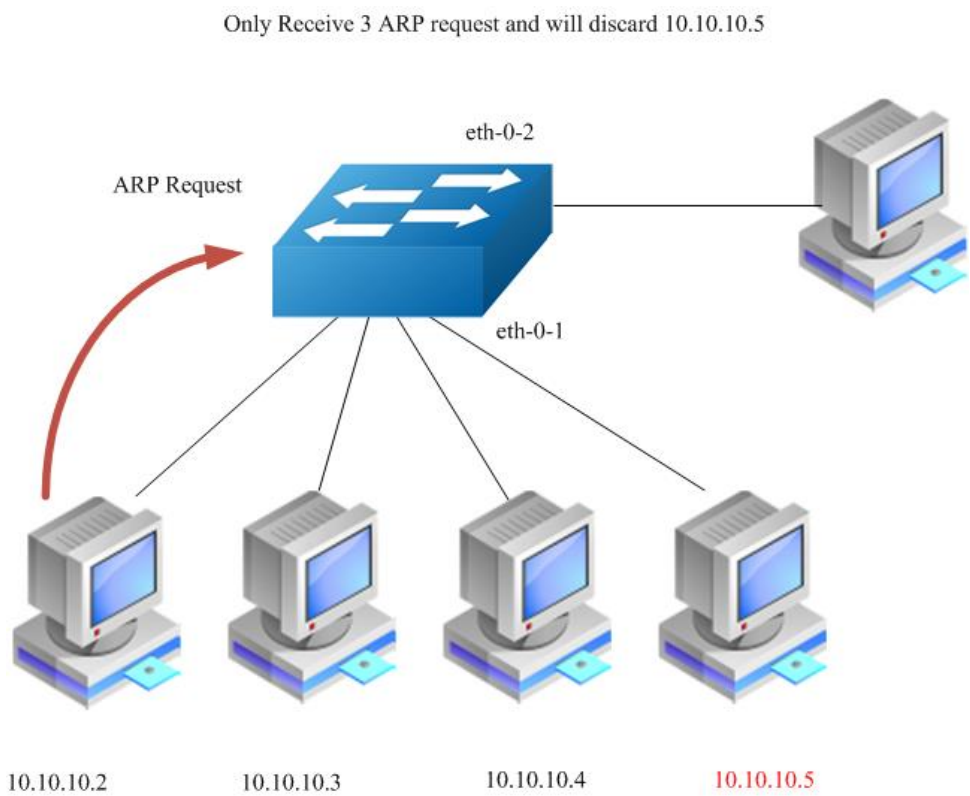

Figure 3-2 ARP Number limit

In this configuration example, interface eth-0-1 configured as L3 port, enable number-limit, set maximum to 3, set violation mode to restrict. System learn 10.10.10.2, 10.10.10.3, 10.10.10.4 first, upto the maximum of number-limit, so ARP packet of 10.10.10.5 will be discarded.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configure the layer 3 interface and set the ip address

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no switchport
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)# ip address 10.10.10.1/24
```

step 3 Configure ARP number limit

```
Switch(config-if-eth-0-1)# ip arp number-limit enable
Switch(config-if-eth-0-1)# ip arp number-limit maximum 3
Switch(config-if-eth-0-1)# ip arp number-limit violation restrict
```

step 4 Exit the configure mode

```
Switch(config-if-eth-0-1)# end
```

step 5 Validation

Use the following command to display the information of the ARP number limit:

```
Switch# show ip arp number-limit
ArpNumLimit Port  MaxLimitNum  CurrentNum  ViolationMode
-----------------+------------+-----------+--------------
eth-0-1           3            3           restrict
DUT1# show ip arp number-limit interface eth-0-1
Arp number limit                    : enabled
Arp number limit violate mode       : discard packet and log
Arp number limit maximum            : 3
ARP number limit total number       : 3
```


Configuration ARP Rate limit

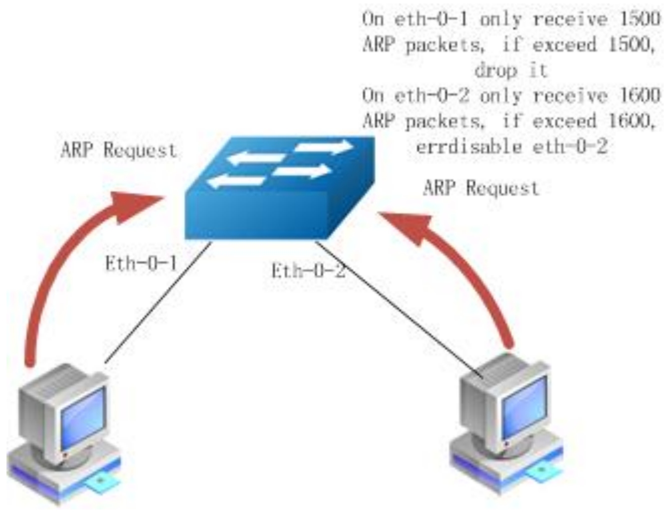

Figure 3-3 ARP Rate limit

In this configuration example, interface eth-0-1 configured as L3 port, enable rate-limit, set rate-limit maximum to 1500; interface eth-0-2 configured as L3 port, enable rate-limit, set rate-limit maximum to 1600, set violation mode to errdisable. If receive ARP packets number in 30s on interface eth-0-1 upto 1500, then the ARP packet receives from interface eth-0-1 will be droped. If receive ARP packets number in 30s on interface eth-0-2 upto 1600, then the ARP packet receives from interface eth-0-1 will be droped and set eth-0-2 errdisabled.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configure the layer 3 interface and set the IP address

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no switchport
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)# ip address 10.10.10.1/24
Switch(config-if-eth-0-1)# exit
Switch(config)# interface eth-0-2
Switch(config-if-eth-0-2)# no switchport
Switch(config-if-eth-0-2)# no shutdown
Switch(config-if-eth-0-2)# ip address 20.20.20.1/24
Switch(config-if-eth-0-2)# exit
```


step 3 Configure ARP rate limit

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# ip arp rate-limit enable
Switch(config-if-eth-0-1)# ip arp rate-limit maximum 1500
Switch(config-if-eth-0-1)# exit
Switch(config)# interface eth-0-2
Switch(config-if-eth-0-2)# ip arp rate-limit enable
Switch(config-if-eth-0-2)# ip arp rate-limit maximum 1600
Switch(config-if-eth-0-2)# ip arp rate-limit violation errdisable
Switch(config-if-eth-0-2)# exit
```

step 4 Exit the configure mode

```
Switch(config-if-eth-0-2)# end
```

step 5 Validation

Use the following command to display the information of the ARP rate limit:

```
Switch# show ip arp rate-limit
The statistics is counting at 14.12s in 30.00s's time period
Port        States   MaxLimitNum  CurrentNum  Violation     Abnormal
-----------+--------+------------+-----------+-------------+-------
eth-0-1     enable   1500         0           restrict      N
eth-0-2     enable   1600         0           errdisable    N
```

3.2.3 Application cases

N/A

3.3 Configuring ARP proxy

3.3.1 Overview

Function Introduction

Proxy ARP, the most common method for learning about other routes, enables an Ethernet host with no routing information to communicate with hosts on other networks or subnets. The host assumes that all hosts are on the same local Ethernet and that they can use ARP to determine their MAC addresses. If a switch receives an ARP request for a host that is not on the same network as the sender, the switch evaluates whether it has the best route to that host. If it does, it sends an ARP reply packet with its own Ethernet MAC address, and the host that sent the request sends the packet to the switch, which forwards it to the intended host.

Proxy ARP treats all networks as if they are local and performs ARP requests for every IP address. Proxy ARP can be separated to 2 parts: Proxy ARP and local Proxy ARP. Local Proxy ARP is always used in the topology where the Device is enabled port isolate but still need to do communicating via routing. Internet Control Message Protocol (ICMP) redirects are disabled on interfaces where the local proxy ARP feature is enabled.

Principle Description

N/A

3.3.2 Configuration

Configuring ARP Proxy

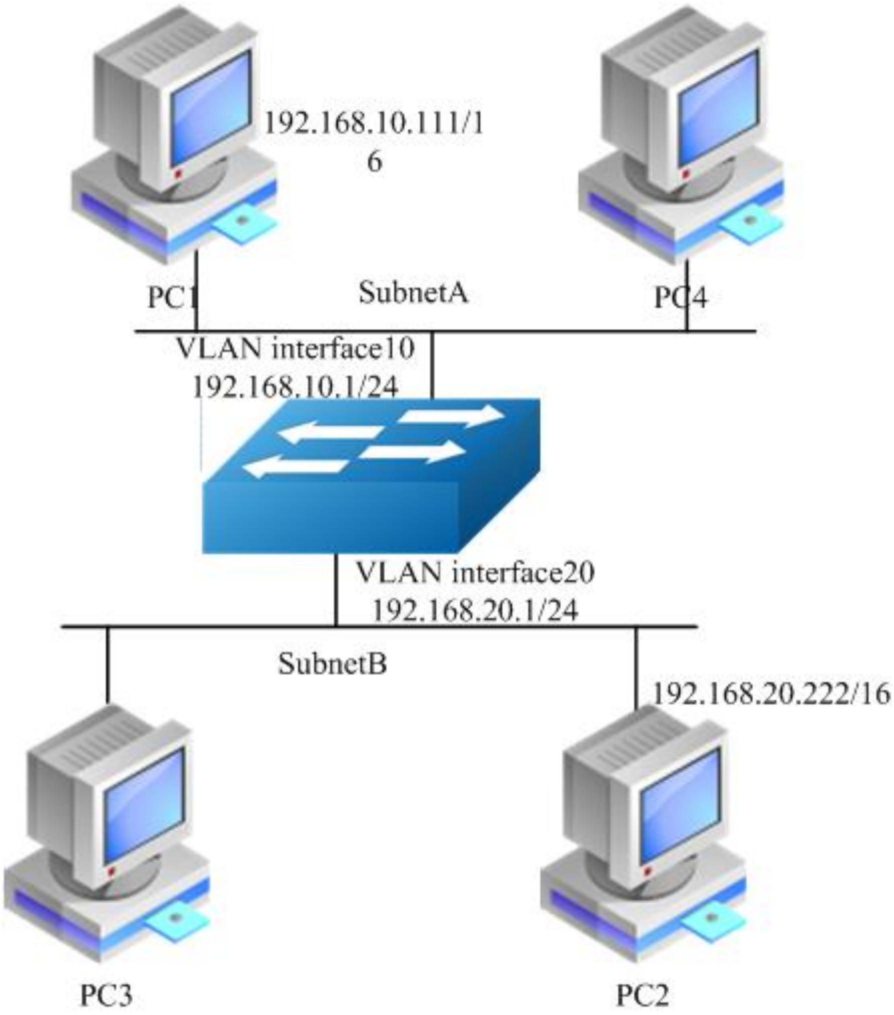

Figure 3-4 arp proxy

As seen in the above topology, PC1 is belonged to VLAN10 and PC2 is belonged to VLAN20. If ARP proxy feature is not enabled, then PC1 and PC2 cannot communicate with each other. As following, these steps are shown to enable ARP proxy feature for both VLAN interface 10 and VLAN interface 20.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create vlan

```
Switch(config)# vlan 10
Switch(config-vlan10)# exit
Switch(config)# vlan 20
Switch(config-vlan20)# exit
```

step 3 Enter the interface configure mode, set the switch port mode and bind to the vlan

```
Switch(config)# interface eth-0-22
Switch(config-if-eth-0-22)# switchport access vlan 10
Switch(config-if-eth-0-22)# no shutdown
Switch(config-if-eth-0-22)# exit

Switch(config)# interface eth-0-23
Switch(config-if-eth-0-23)# switchport access vlan 20
Switch(config-if-eth-0-23)# no shutdown
Switch(config-if-eth-0-23)# exit
```

step 4 Create the vlan interface, configure the IP address, and enable ARP proxy

```
Switch(config)# interface vlan 10
Switch(config-if-vlan10)# ip address 192.168.10.1/24
Switch(config-if-vlan10)# proxy-arp enable
Switch(config-if-vlan10)# exit

Switch(config)# interface vlan 20
Switch(config-if-vlan20)# ip address 192.168.20.1/24
Switch(config-if-vlan20)# proxy-arp enable
Switch(config-if-vlan20)# exit
```

step 5 Exit the configure mode

```
Switch(config)# end
```


step 6 Validation

Use the following command to display the information of the ARP proxy configuration on the switch:

```
Switch# show ip interface vlan 10
Interface vlan10
  Interface current state: UP
  Internet address(es):
    192.168.10.1/24  broadcast 192.168.10.255/24
  The maximum transmit unit is 1500 bytes
  ICMP redirects are always sent
  ARP timeout 01:00:00,  ARP retry interval 1s
  ARP Proxy is enabled, Local ARP Proxy is disabled

Switch# show ip interface vlan 20
Interface vlan20
  Interface current state: UP
  Internet address(es):
    192.168.20.1/24  broadcast 192.168.20.255/24
  The maximum transmit unit is 1500 bytes
  ICMP redirects are always sent
  ARP timeout 01:00:00,  ARP retry interval 1s
  ARP Proxy is enabled, Local ARP Proxy is disabled
```

Use the following command to display the information of the ARP entry on the switch:

```
Switch# show ip arp
 Protocol    Address          Age (min)  Hardware Addr   Interface
 Internet    192.168.10.1            -   7cc3.11f1.aa00  vlan10
 Internet    192.168.10.111          5   0cf9.11b6.6e2e  vlan10
 Internet    192.168.20.1            -   7cc3.11f1.aa00  vlan20
 Internet    192.168.20.222          6   5a94.031f.2357  vlan20
```

Use the following command to display the information on PC1:

```
[Host:~]$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 0C:F9:11:B6:6E:2E
          inet addr:192.168.10.111  Bcast:192.168.255.255  Mask:255.255.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1600  Metric:1
          RX packets:11 errors:0 dropped:0 overruns:0 frame:0
          TX packets:10 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:588 (588.0 b)  TX bytes:700 (700.0 b)
          Interrupt:5

[Host:~]$ arp -a
? (192.168.20.222) at 7c:c3:11:f1:aa:00 [ether] on eth0
[Host: ~]$ route -v
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.0     *               255.255.0.0     U     0      0        0 eth0
```



```
[Host:~]$ ping 192.168.20.222
PING 192.168.20.222 (192.168.20.222) 56(84) bytes of data.
64 bytes from 192.168.20.222: icmp_seq=0 ttl=63 time=189 ms
64 bytes from 192.168.20.222: icmp_seq=1 ttl=63 time=65.2 ms
--- 192.168.20.222 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 65.209/127.226/189.244/62.018 ms, pipe 2
```

Use the following command to display the information on PC2:

```
[Host:~]$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 5A:94:03:1F:23:57
          inet addr:192.168.20.222  Bcast:192.168.255.255  Mask:255.255.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1600  Metric:1
          RX packets:14 errors:0 dropped:0 overruns:0 frame:0
          TX packets:17 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:784 (784.0 b)  TX bytes:1174 (1.1 KiB)
          Interrupt:5

[Host:~]$ arp -a
? (192.168.10.111) at 7c:c3:11:f1:aa:00 [ether] on eth0

[Host: ~]$ route -v
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.0     *               255.255.0.0     U     0      0        0 eth0

[Host: ~]$ ping 192.168.10.111
PING 192.168.10.111 (192.168.10.111) 56(84) bytes of data.
64 bytes from 192.168.10.111: icmp_seq=0 ttl=63 time=53.8 ms
64 bytes from 192.168.10.111: icmp_seq=1 ttl=63 time=65.8 ms
--- 192.168.10.111 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1007ms
rtt min/avg/max/mdev = 53.832/59.842/65.852/6.010 ms, pipe 2
```

Configuring Local ARP Proxy

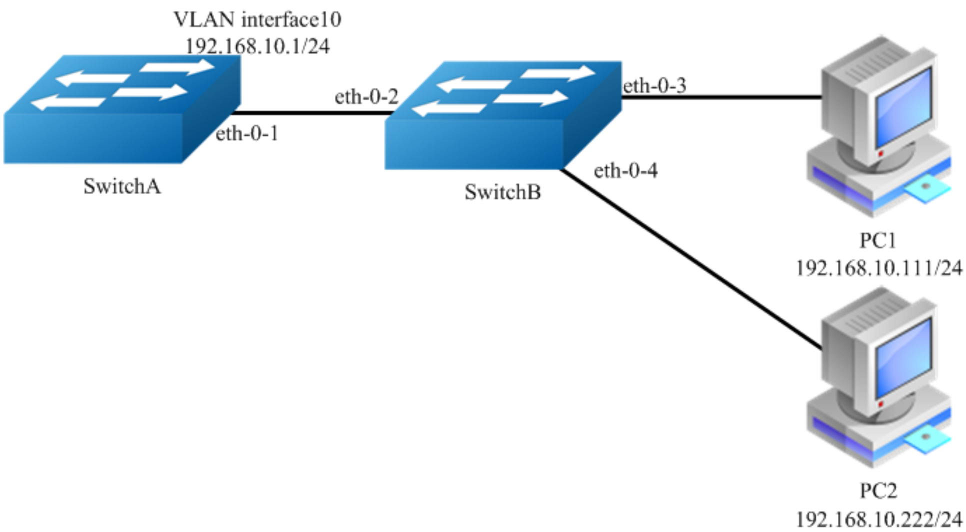

Figure 3-5 local arp proxy

As the above topology, eth-0-2, eth-0-3 and eth-0-4 are belonging to VLAN 10. eth-0-3 and eth-0-4 are both in port isolate group 1, and eth-0-2 is in port isolate group 3, so packets received in eth-0-3 cannot flood to eth-0-4, but packets received in eth-0-2 can flood to both eth-0-3 and eth-0-4. PC1 is connecting with port eth-0-3 and PC2 is connecting with port eth-0-4.Configure as the following step for communicating with PC1 and PC2.

The configurations of switch A and switch B are same if there is no special description.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create vlan

```
Switch(config)# vlan 10
Switch(config-vlan10)# exit
```

step 3 Enter the interface configure mode, set the switch port mode and bind to the vlan

Switch A configuration:

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# switchport access vlan 10
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)# exit
```

Switch B configuration:

```
Switch(config)# interface range eth-0-2 - 4
Switch(config-if-range)# switchport access vlan 10
Switch(config-if-range)# no shutdown
Switch(config-if-range)# exit
```

step 4 Create the vlan interface, configure the IP address, and enable local ARP proxy

Switch A configuration:

```
Switch(config)# interface vlan 10
Switch(config-if-vlan10)# ip address 192.168.10.1/24
Switch(config-if-vlan10)# local-proxy-arp enable
Switch(config-if-vlan10)# exit
```


step 5 Configuring port isolation(optional)

Switch B configuration:

After configuring port isolation as blow, eth-0-3 and eth-0-4 on swichB are isolated in layer 2 network.

```
Switch(config)# port-isolate mode l2
Switch(config)# interface eth-0-3 - 4
Switch(config-if-range)# port-isolate group 1
Switch(config-if-range)# exit

Switch(config)# interface eth-0-2
Switch(config-if-eth-0-2)# port-isolate group 3
Switch(config-if-eth-0-2)# exit
```

step 6 Validation

Use the following command to display the information of the ARP entry on switchA:

```
Switch# show ip arp
 Protocol    Address          Age (min)  Hardware Addr   Interface
 Internet    192.168.10.1             -  eeb4.2a8d.6c00  vlan10
 Internet    192.168.10.111           0  34b0.b279.5f67  vlan10
 Internet    192.168.10.222           0  2a65.9618.57fa  vlan10
```

Use the following command to display the information of the local ARP proxy configurations on the interface of switchA:

```
Switch# show ip interface vlan 10
Interface vlan10
  Interface current state: UP
  Internet address(es):
    192.168.10.1/24  broadcast 192.168.10.255/24
  The maximum transmit unit is 1500 bytes
  ICMP redirects are never sent
  ARP timeout 01:00:00,  ARP retry interval 1s
  ARP Proxy is disabled, Local ARP Proxy is enabled
```

Use the following command to display the information on PC1:

```
[Host: ~]$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 34:B0:B2:79:5F:67
          inet addr:192.168.10.111  Bcast:192.168.10.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1600  Metric:1
          RX packets:22 errors:0 dropped:0 overruns:0 frame:0
          TX packets:28 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1344 (1.3 KiB)  TX bytes:2240 (2.1 KiB)
          Interrupt:5

[Host: ~]$ arp -a
```



```
? (192.168.10.222) at ee:b4:2a:8d:6c:00 [ether] on eth0

[Host: ~]$ ping 192.168.10.222
PING 192.168.10.222 (192.168.10.222) 56(84) bytes of data.
64 bytes from 192.168.10.222: icmp_seq=0 ttl=63 time=131 ms
64 bytes from 192.168.10.222: icmp_seq=1 ttl=63 time=159 ms
--- 192.168.10.222 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1003ms
rtt min/avg/max/mdev = 131.078/145.266/159.454/14.188 ms, pipe 2
```

Use the following command to display the information on PC2:

```
[Host:~]$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 2A:65:96:18:57:FA
          inet addr:192.168.10.222  Bcast:192.168.10.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1600  Metric:1
          RX packets:19 errors:0 dropped:0 overruns:0 frame:0
          TX packets:20 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1148 (1.1 KiB)  TX bytes:1524 (1.4 KiB)
          Interrupt:5

[Host:~]$ arp -a
? (192.168.10.111) at ee:b4:2a:8d:6c:00 [ether] on eth0

[Host: ~]$ ping 192.168.10.111
PING 192.168.10.111 (192.168.10.111) 56(84) bytes of data.
64 bytes from 192.168.10.111: icmp_seq=0 ttl=63 time=198 ms
64 bytes from 192.168.10.111: icmp_seq=1 ttl=63 time=140 ms
64 bytes from 192.168.10.111: icmp_seq=2 ttl=63 time=146 ms
--- 192.168.10.111 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2008ms
rtt min/avg/max/mdev = 140.196/161.959/198.912/26.267 ms, pipe 2
```

3.3.3 Application cases

N/A

3.4 Configuring DHCP Client

3.4.1 Overview

Function Introduction

Dynamic Host Configuration Protocol(DHCP) client can acquire IP address and configuration dynamically from DHCP server by DHCP. If client and server is on the same physical subnet, client can communicate with server directly, otherwise they need DHCP relay agent which is used to forward DHCP messages. DHCP client can request IP address from DHCP server by broadcasting DHCP messages. After received IP address and lease correspond to it, client will configure itself and set the expired time. When half past the lease, client will sent DHCP messages for a new lease to use the IP address continually. If it success, DHCP client will renew the lease. DHCP client can send option request to server, which may be one or several of router, static-route, classless-static-route, classless-static-route-ms, tftp-server-address, dns-nameserver , domain-name, netbios-nameserver and vendor-specific. By default, options include router, static-route, classless-static-route, classless-static-route-ms, tftp-server-address will be requested from server. We can cancel one or several of these option requests by command.

Principle Description

N/A

3.4.2 Configuration

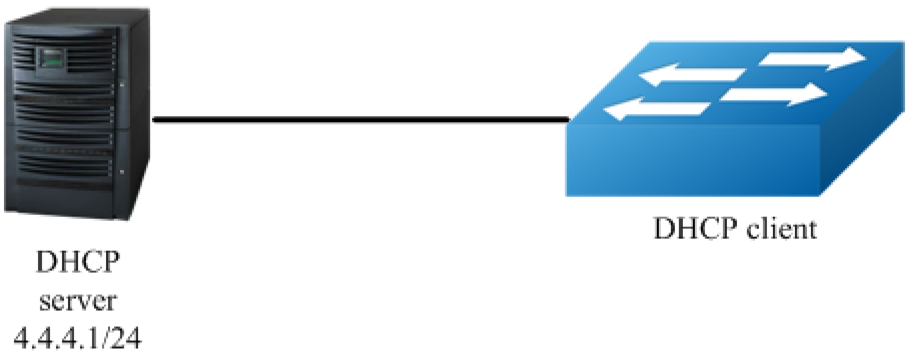

Figure 3-6 dhcp client

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enter the interface configure mode

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no switchport
Switch(config-if-eth-0-1)# no shutdown
```

step 3 disable static-route and enable DHCP client

```
Switch(config-if-eth-0-1)# no dhcp client request static-route
Switch(config-if-eth-0-1)# ip address dhcp
```


step 4 Exit the configure mode

```
Switch(config-if-eth-0-1)# end
```

step 5 Validation

Check interface configuration:

```
Switch# show running-config interface eth-0-1
Building configuration...
!
interface eth-0-1
 no switchport
 ip address dhcp
 no dhcp client request static-route
!
```

Check all DHCP client status:

```
Switch# show dhcp client verbose
DHCP client informations:
============================================================
eth-0-1 DHCP client information:
  Current state: BOUND
  Allocated IP: 4.4.4.199 255.255.255.0
  Lease/renewal/rebinding: 1187/517/1037 seconds
  Lease from 2011-11-18 05:59:59 to 2011-11-18 06:19:59
  Will Renewal in 0 days 0 hours 8 minutes 37 seconds
  DHCP server: 4.4.4.1
  Transaction ID: 0x68857f54
  Client ID: switch-7e39.3457.b700-eth-0-1
```

Show DHCP client statistics:

```
Switch# show dhcp client statistics
DHCP client packet statistics:
============================================================
DHCP OFFERS    received: 1
DHCP ACKs      received: 2
DHCP NAKs      received: 0
DHCP Others    received: 0
DHCP DISCOVER      sent: 1
DHCP DECLINE       sent: 0
DHCP RELEASE       sent: 0
DHCP REQUEST       sent: 2
DHCP packet send failed: 0
```

3.4.3 Application cases

N/A

3.5 Configuring DHCP Relay

3.5.1 Overview

Function Introduction

DHCP relay agent is any host that forwards DHCP packets between clients and servers. Relay agents are used to forward requests and replies between clients and servers when they are not on the same physical subnet. Relay agent forwarding is distinct from the normal forwarding of an IP router, where IP datagram are switched between networks somewhat transparently. By contrast, relay agents receive DHCP messages and then generate a new DHCP message to send out on another interface. The relay agent sets the gateway address (girder field of the DHCP packet) and, if configured, adds the relay agent information option (option82) in the packet and forwards it to the DHCP server. The reply from the server is forwarded back to the client after removing option 82.

Principle Description

N/A

3.5.2 Configuration

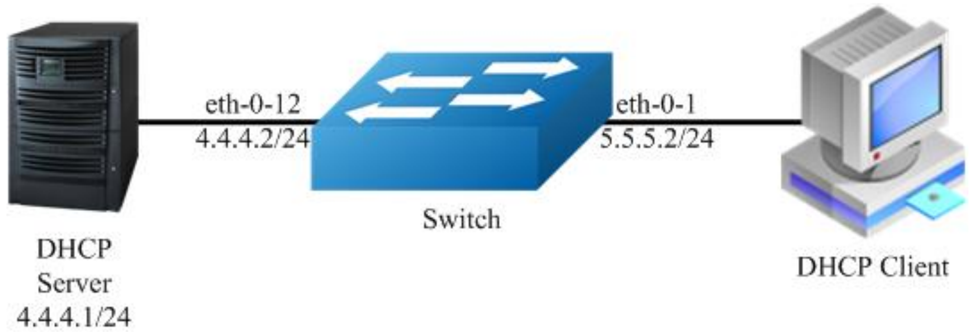

Figure 3-7 DHCP relay

This figure is the networking topology for testing DHCP relay functions. We need two Linux boxes and one Switch to construct the test bed.

- Computer A is used as DHCP server.
- Computer B is used as DHCP client.
- Switch is used as DHCP relay agent.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enter the interface configure mode，set the attributes and IP address

```
Switch(config)# interface eth-0-12
Switch(config-if-eth-0-12)# no switchport
Switch(config-if-eth-0-12)# ip address 4.4.4.2/24
Switch(config-if-eth-0-12)# no shutdown
Switch(config-if-eth-0-12)# exit

Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no switchport
Switch(config-if-eth-0-1)# ip address 5.5.5.2/24
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)# exit
```

step 3 Create a dhcp server

```
Switch(config)# dhcp-server 1 4.4.4.1
```

step 4 Enable DHCP server and option82 for the interface

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# dhcp relay information trusted
Switch(config-if-eth-0-1)# dhcp-server 1
Switch(config-if-eth-0-1)# exit
```

step 5 Enable DHCP server and DHCP relay globally

```
Switch(config)# service dhcp enable
Switch(config)# dhcp relay
```

step 6 Validation

Check the interface configuration

```
Switch# show running-config interface eth-0-12
Building configuration...
interface eth-0-12
 no switchport
 ip address 4.4.4.2/24
!
Switch # show running-config interface eth-0-1
Building configuration...
interface eth-0-1
 no switchport
 dhcp relay information trusted
```



```
dhcp-server 1
ip address 5.5.5.2/24
```

Check the dhcp service status

```
Switch# show services
Networking services configuration:
Service Name     Status         Port       Protocol
----------------+-------------+----------+----------
dhcp             enable         67/68      UDP
http             enable         80         TCP
telnet           enable         23         TCP
ssh              enable         22         TCP
snmp             disable        161        UDP
```

Check the dhcp server group configuration

```
Switch# show dhcp-server
DHCP server group information:
============================================================
group 1 ip address list:
[1] 4.4.4.1
```

Check the dhcp relay statistics

```
Switch# show dhcp relay statistics
DHCP relay packet statistics:
=============================================================
Client relayed packets:  20
Server relayed packets:  20

Client error packets:     20
Server error packets:     0
Bogus GIADDR drops:       0
Bad circuit ID packets:   0
Corrupted agent options:  0
Missing agent options:    0
Missing circuit IDs:      0
```

Check your computer IP address from DHCP server

```
Ipconfig/all
Dhcp Enabled. . . . . . . . . . . : Yes
Autoconfiguration Enabled . . . . : Yes
IP Address. . . . . . . . . . . . : 5.5.5.1
Subnet Mask . . . . . . . . . . . : 255.255.255.0
Default Gateway . . . . . . . . . : 5.5.5.2
DHCP Server . . . . . . . . . . . : 4.4.4.1
DNS Servers . . . . . . . . . . . : 4.4.4.1
```

3.5.3 Application cases

N/A

4 IP Routing Configuration Guide

4.1 Configuring IP Unicast-Routing

4.1.1 Overview

Function Introduction

Static routing is a concept describing one way of configuring path selection of routers in computer networks. It is the type of routing characterized by the absence of communication between routers regarding the current topology of the network. This is achieved by manually adding routes to the routing table. The opposite of static routing is dynamic routing, sometimes also referred to as adaptive routing.

In these systems, routes through a data network are described by fixed paths (statically). These routes are usually entered into the router by the system administrator. An entire network can be configured using static routes, but this type of configuration is not fault tolerant. When there is a change in the network or a failure occurs between two statically defined nodes, traffic will not be rerouted. This means that anything that wishes to take an affected path will either have to wait for the failure to be repaired or the static route to be updated by the administrator before restarting its journey. Most requests will time out (ultimately failing) before these repairs can be made. There are, however, times when static routes can improve the performance of a network. Some of these include stub networks and default routes.

Principle Description

N/A

4.1.2 Configuration

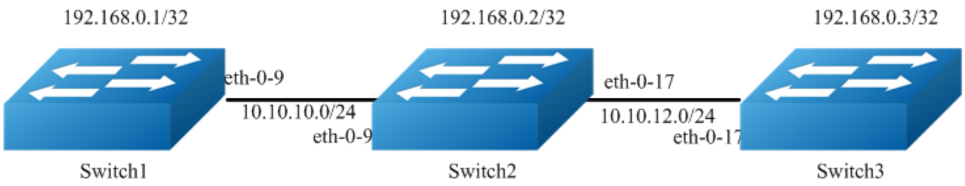

Figure 4-1 ip unicast routing

This example shows how to enable static route in a simple network topology.

There are 3 static routes on Switch1, one is to achieve remote network 10.10.12.0/24, the other two are to achieve the loopback addresses on Switch2 and Switch3. There is a default static route on Switch3, that is, static routes use same gateway or nexthop address. There are 2 static routes on Switch2, both of them are to achieve the remote switch's loopback address.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enter the interface configure mode, set the attributes and IP address

Configure on Switch1:

```
Switch(config)# interface eth-0-9
Switch(config-if-eth-0-9)# no shutdown
Switch(config-if-eth-0-9)# no switchport
Switch(config-if-eth-0-9)# ip address 10.10.10.1/24
Switch(config-if-eth-0-9)# exit

Switch(config)# interface loopback0
Switch(config-if-loopback0)# no shutdown
Switch(config-if-loopback0)# no switchport
Switch(config-if-loopback0)# ip address 192.168.0.1/32
Switch(config-if-loopback0)# exit
```

Configure on Switch2:

```
Switch(config)# interface eth-0-9
Switch(config-if-eth-0-9)# no shutdown
Switch(config-if-eth-0-9)# no switchport
Switch(config-if-eth-0-9)# ip address 10.10.10.2/24
Switch(config-if-eth-0-9)# exit

Switch(config)# interface eth-0-17
Switch(config-if-eth-0-17)# no shutdown
Switch(config-if-eth-0-17)# no switchport
```



```
Switch(config-if-eth-0-17)# ip address 10.10.12.2/24
Switch(config-if-eth-0-17)# exit

Switch(config)# interface loopback0
Switch(config-if-loopback0)# no shutdown
Switch(config-if-loopback0)# no switchport
Switch(config-if-loopback0)# ip address 192.168.0.2/32
Switch(config-if-loopback0)# exit
```

Configure on Switch3:

```
Switch(config)# interface eth-0-17
Switch(config-if-eth-0-17)# no shutdown
Switch(config-if-eth-0-17)# no switchport
Switch(config-if-eth-0-17)# ip address 10.10.12.3/24
Switch(config-if-eth-0-17)# exit

Switch(config)# interface loopback0
Switch(config-if-loopback0)# no switchport
Switch(config-if-loopback0)# no shutdown
Switch(config-if-loopback0)# ip address 192.168.0.3/32
Switch(config-if-loopback0)# exit
```

step 3 Configuring static route

Configure on Switch1:

Note: Specify the destination prefix and mask for the network for which a gateway is required, for example, 10.10.12.0/24. Add a gateway for each of them (in this case 10.10.10.2 for all). Since R2 is the only next hop available, you can configure a default route instead of configuring the same static route for individual addresses.

```
Switch(config)# ip route 10.10.12.0/24 10.10.10.2
Switch(config)# ip route 192.168.0.2/32 10.10.10.2
Switch(config)# ip route 192.168.0.3/32 10.10.10.2
```

Configure on Switch2:

```
Switch(config)# ip route 192.168.0.1/32 10.10.10.1
Switch(config)# ip route 192.168.0.3/32 10.10.12.3
```

Configure on Switch3:

Note:Specify 10.10.12.2 as a default gateway to reach any network. Since 10.10.12.2 is the only route available you can specify it as the default gateway instead of specifying it as the gateway for individual network or host addresses.

```
Switch(config)# ip route 0.0.0.0/0 10.10.12.2
```


step 4 Exit the configure mode

```
Switch(config)# end
```

step 5 Validation

Use the following command to display the route information on Switch1:

```
Switch# show ip route
Codes: C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, P - PIM,
       > - selected route, * - FIB route,
       [*] - [AD/Metric]

C>* 10.10.10.0/24 is directly connected, eth-0-9
S>* 10.10.12.0/24 [1/0] via 10.10.10.2, eth-0-9
C>* 192.168.0.1/32 is directly connected, loopback0
S>* 192.168.0.2/32 [1/0] via 10.10.10.2, eth-0-9
S>* 192.168.0.3/32 [1/0] via 10.10.10.2, eth-0-9
```

Use the following command to display the route information on Switch2:

```
Switch# show ip route
Codes: C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, P - PIM,
       > - selected route, * - FIB route,
       [*] - [AD/Metric]

C>* 10.10.10.0/24 is directly connected, eth-0-9
C>* 10.10.12.0/24 is directly connected, eth-0-17
S>* 192.168.0.1/32 [1/0] via 10.10.10.1, eth-0-9
C>* 192.168.0.2/32 is directly connected, loopback0
S>* 192.168.0.3/32 [1/0] via 10.10.12.3, eth-0-17
```

Use the following command to display the route information on Switch3:

```
Switch# show ip route
Codes: C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, P - PIM,
       > - selected route, * - FIB route,
       [*] - [AD/Metric]

S>* 0.0.0.0/0 [1/0] via 10.10.12.2, eth-0-17
C>* 10.10.12.0/24 is directly connected, eth-0-17
C>* 192.168.0.3/32 is directly connected, loopback0
```

4.1.3 Application cases

N/A

4.2 Configuring OSPF

4.2.1 Overview

Function Introduction

OSPF is an Interior Gateway Protocol (IGP) designed expressly for IP networks, supporting IP subnet ting and tagging of externally derived routing information. OSPF also allows packet authentication and uses IP multicast when sending and receiving packets.

The implementation conforms to the OSPF Version 2 specifications with these key features:

- Definition of stub areas is supported: Routes learned through any IP routing protocol can be redistributed into another IP routing protocol. At the intradomain level, this means that OSPF can import routes learned through RIP. OSPF routes can also be exported into RIP.
- Plain text and MD5 authentication among neighboring routers within an area is supported: Configurable routing interface parameters include interface output cost, retransmission interval, interface transmit delay, router priority, router dead and hello intervals, and authentication key.
- Virtual links are not supported: Not-so-stubby-areas (NSSAs) per RFC 1587 are not supported now. OSPF typically requires coordination among many internal routers, area border routers (ABRs) connected to multiple areas, and autonomous system boundary routers (ASBRs). The minimum configuration would use all default parameter values, no authentication, and interfaces assigned to areas. If you customize your environment, you must ensure coordinated configuration of all routers.

Principle Description

Reference to RFC 2328

4.2.2 Configuration

Basic OSPF Parameters Configuration

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configure the Routing process and associate the network with a specified OSPF area

```
Switch(config)# router ospf
Switch(config-router)# network 10.10.10.0/24 area 0
Switch(config-router)# quit
```

Note:use the following command to delete the routing process

```
Switch(config)# no router ospf
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

```
Switch# show running-config
router ospf
 network 10.10.10.0/24 area 0
```

Enabling OSPF on an Interface

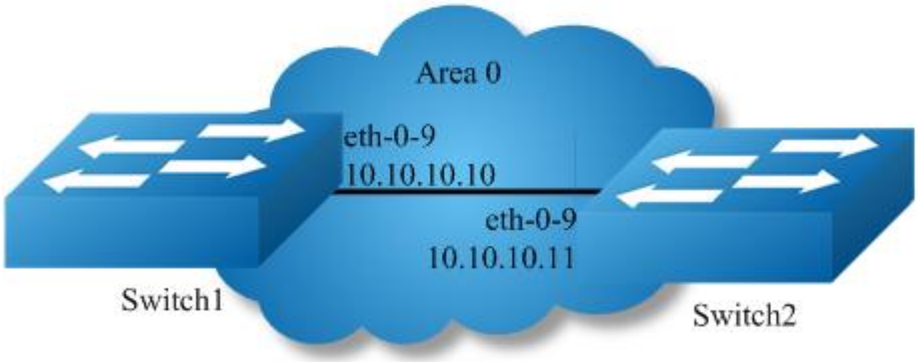

Figure 4-2 ospf

This example shows the minimum configuration required for enabling OSPF on an interface Switch1 and 2 are two routers in Area 0 connecting to network 10.10.10.0/24

NOTE Configure one interface so that it belongs to only one area. However, you can configure different interfaces on a router to belong to different areas.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enter the interface configure mode, set the attributes and IP address

Configure on Switch1:

```
Switch(config)# interface eth-0-9
Switch(config-if-eth-0-9)# no switchport
Switch(config-if-eth-0-9)# no shutdown
Switch(config-if-eth-0-9)# ip address 10.10.10.10/24
Switch(config-if-eth-0-9)# exit
```

Configure on Switch2:

```
Switch(config)# interface eth-0-9
Switch(config-if-eth-0-9)# no switchport
Switch(config-if-eth-0-9)# no shutdown
Switch(config-if-eth-0-9)# ip address 10.10.10.11/24
Switch(config-if-eth-0-9)# exit
```

step 3 Configure the Routing process and associate the network with a specified OSPF area

Configure on Switch1:

```
Switch(config)# router ospf
Switch(config-router)# network 10.10.10.0/24 area 0
```

Configure on Switch2:

```
Switch(config)# router ospf
Switch(config-router)# network 10.10.10.0/24 area 0
```

Note: To using OSPF among two devices which are directly connected, the area IDs must be same. The ospf process IDs can be same or different.

step 4 Exit the configure mode

```
Switch(config-router)# end
```

step 5 Validation

Use the following command to display the database of ospf:

```
Switch# show ip ospf database

            OSPF Router with ID (10.10.10.10)

                Router Link States (Area 0)

Link ID         ADV Router      Age  Seq#       CkSum  Link count
10.10.10.10     10.10.10.10       26 0x80000006 0x1499 1
10.10.10.11     10.10.10.11       27 0x80000003 0x1895 1

                Net Link States (Area 0)

Link ID         ADV Router      Age  Seq#       CkSum
10.10.10.10     10.10.10.10       26 0x80000001 0xdfd8
```

Use the following command to display the interface of ospf:

```
Switch# show ip ospf interface
eth-0-9 is up
  ifindex 119, MTU 1500 bytes, BW 40000 Mbit <UP,BROADCAST,RUNNING,MULTICAST>
  Internet Address 10.10.10.10/24, Broadcast 10.10.10.255, Area 0.0.0.0
  MTU mismatch detection:enabled
  Router ID 10.10.10.10, Network Type BROADCAST, Cost: 3
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 10.10.10.10, Interface Address 10.10.10.10
  Backup Designated Router (ID) 10.10.10.11, Interface Address 10.10.10.11
  Multicast group memberships: OSPFAllRouters OSPFDesignatedRouters
  Timer intervals configured, Hello 10s, Dead 40s, Wait 40s, Retransmit 5s
    Hello due in 3.465s
  Neighbor Count is 1, Adjacent neighbor count is 1
```

Use the following command to display the neighbor of ospf:

Switch1：

```
Switch# show ip ospf neighbor

Neighbor ID     Pri State           Dead Time Address         Interface
RXmtL RqstL DBsmL
10.10.10.11      1   Full/Backup     34.107s   10.10.10.11     eth-0-9:10.10.10.10
0     0     0
```

Switch2：


```
Switch# show ip ospf neighbor

Neighbor ID     Pri State           Dead Time Address         Interface
RXmtL RqstL DBsmL
10.10.10.10      1   Full/DR         38.682s   10.10.10.10     eth-0-9:10.10.10.11
1     0     0
```

Use the following command to display the ospf routes:

```
Switch# show ip route
Codes: C - connected, S - static, R - RIP,
   O - OSPF, I - IS-IS, B - BGP, P - PIM,
   > - selected route, * - FIB route,
   [*] - [AD/Metric]

O   10.10.10.0/24 [110/3] is directly connected, eth-0-9, 03:40:16
C>* 10.10.10.0/24 is directly connected, eth-0-9
```

Configuring OSPF Cost

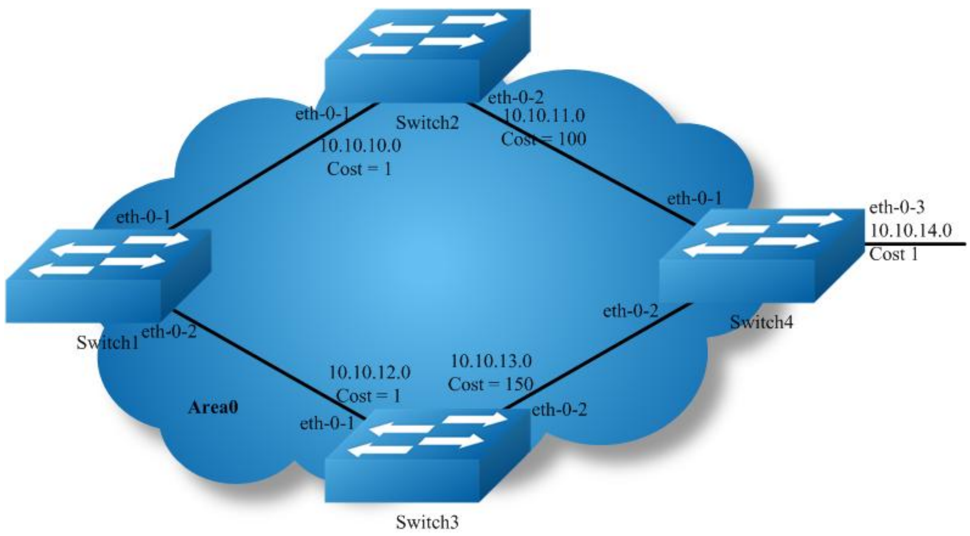

Figure 4-3 ospf cost

You can make a route the preferred route by changing its cost. In this example, cost has been configured to make Switch2 the next hop for Switch1.

The default cost on each interface is 3(40G speed). Interface eth2 on Switch2 has a cost of 100 and interface eth2 on Switch3 has a cost of 150. The total cost to reach(Switch4 network 10.10.14.0) through Switch2 and Switch3:

Switch2: 3+3+100 = 106

Switch3: 3+3+150 = 156

Therefore, Switch1 chooses Switch2 as its next hop for destination Switch4

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enter the interface configure mode, set the attributes and IP address. Set the ospf cost under the interface configure mode

Configure on Switch1:

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no switchport
Switch(config-if-eth-0-1)# ip address 10.10.10.1/24
Switch(config-if-eth-0-1)# exit
Switch(config)# interface eth-0-2
Switch(config-if-eth-0-2)# no switchport
Switch(config-if-eth-0-2)# ip address 10.10.12.1/24
Switch(config-if-eth-0-2)# exit
```

Configure on Switch2:

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no switchport
Switch(config-if-eth-0-1)# ip address 10.10.10.2/24
Switch(config-if-eth-0-1)# exit
Switch(config)# interface eth-0-2
Switch(config-if-eth-0-2)# no switchport
Switch(config-if-eth-0-2)# ip address 10.10.11.2/24
Switch(config-if-eth-0-2)# ip ospf cost 100
Switch(config-if-eth-0-2)# exit
```

Configure on Switch3:

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no switchport
Switch(config-if-eth-0-1)# ip address 10.10.12.2/24
Switch(config-if-eth-0-1)# exit
Switch(config)# interface eth-0-2
Switch(config-if-eth-0-2)# no switchport
Switch(config-if-eth-0-2)# ip address 10.10.13.2/24
Switch(config-if-eth-0-2)# ip ospf cost 150
Switch(config-if-eth-0-2)# exit
```

Configure on Switch4:

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no switchport
Switch(config-if-eth-0-1)# ip address 10.10.11.1/24
Switch(config-if-eth-0-1)# exit
Switch(config)# interface eth-0-2
Switch(config-if-eth-0-2)# no switchport
Switch(config-if-eth-0-2)# ip address 10.10.13.1/24
Switch(config-if-eth-0-2)# exit
Switch(config)# interface eth-0-3
```



```
Switch(config-if-eth-0-3)# no switchport
Switch(config-if-eth-0-3)# ip address 10.10.14.1/24
Switch(config-if-eth-0-3)# exit
```

step 3 Configure the Routing process and associate the network with a specified OSPF area

Configure on Switch1:

```
Switch(config)# router ospf 100
Switch(config-router)# network 10.10.10.0/24 area 0
Switch(config-router)# network 10.10.12.0/24 area 0
Switch(config-router)# exit
```

Configure on Switch2:

```
Switch(config)# router ospf 100
Switch(config-router)# network 10.10.10.0/24 area 0
Switch(config-router)# network 10.10.11.0/24 area 0
Switch(config-router)# exit
```

Configure on Switch3:

```
Switch(config)# router ospf 100
Switch(config-router)# network 10.10.12.0/24 area 0
Switch(config-router)# network 10.10.13.0/24 area 0
Switch(config-router)# exit
```

Configure on Switch4:

```
Switch(config)# router ospf 100
Switch(config-router)# network 10.10.11.0/24 area 0
Switch(config-router)# network 10.10.13.0/24 area 0
Switch(config-router)# network 10.10.14.0/24 area 0
Switch(config-router)# exit
```

step 4 Exit the configure mode

```
Switch(config)# end
```

step 5 Validation

Use the following command to display the ospf routes:

Switch1：

```
Switch# show ip route
Codes: C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, P - PIM,
       > - selected route, * - FIB route,
       [*] - [AD/Metric]

```



```
O   10.10.10.0/24 [110/3] is directly connected, eth-0-1, 00:04:03
C>* 10.10.10.0/24 is directly connected, eth-0-1
O>* 10.10.11.0/24 [110/103] via 10.10.10.2, eth-0-1, 00:02:41
O   10.10.12.0/24 [110/3] is directly connected, eth-0-2, 00:04:14
C>* 10.10.12.0/24 is directly connected, eth-0-2
O>* 10.10.13.0/24 [110/106] via 10.10.10.2, eth-0-1, 00:02:25
```

Switch2:

```
Switch# show ip route
Codes: C - connected, S - static, R - RIP,
   O - OSPF, I - IS-IS, B - BGP, P - PIM,
   > - selected route, * - FIB route,
   [*] - [AD/Metric]

O   10.10.10.0/24 [110/3] is directly connected, eth-0-1, 00:04:38
C>* 10.10.10.0/24 is directly connected, eth-0-1
O   10.10.11.0/24 [110/100] is directly connected, eth-0-2, 00:05:19
C>* 10.10.11.0/24 is directly connected, eth-0-2
O>* 10.10.12.0/24 [110/6] via 10.10.10.1, eth-0-1, 00:04:30
O>* 10.10.13.0/24 [110/103] via 10.10.11.1, eth-0-2, 00:04:15
```

Switch3:

```
Switch# show ip route
Codes: C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, P - PIM,
       > - selected route, * - FIB route,
       [*] - [AD/Metric]

O>* 10.10.10.0/24 [110/6] via 10.10.12.1, eth-0-1, 00:13:14
O>* 10.10.11.0/24 [110/106] via 10.10.12.1, eth-0-1, 00:13:14
O   10.10.12.0/24 [110/3] is directly connected, eth-0-1, 00:13:14
C>* 10.10.12.0/24 is directly connected, eth-0-1
O   10.10.13.0/24 [110/109] via 10.10.12.1, eth-0-1, 00:13:14
C>* 10.10.13.0/24 is directly connected, eth-0-2
```

Switch4:

```
Switch# show ip route
Codes: C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, P - PIM,
       > - selected route, * - FIB route,
       [*] - [AD/Metric]

O>* 10.10.10.0/24 [110/6] via 10.10.11.2, eth-0-1, 00:16:06
O   10.10.11.0/24 [110/3] is directly connected, eth-0-1, 00:16:11
C>* 10.10.11.0/24 is directly connected, eth-0-1
O>* 10.10.12.0/24 [110/6] via 10.10.13.2, eth-0-2, 00:14:37
O   10.10.13.0/24 [110/3] is directly connected, eth-0-2, 00:16:24
C>* 10.10.13.0/24 is directly connected, eth-0-2
```


Configuring OSPF Authentication

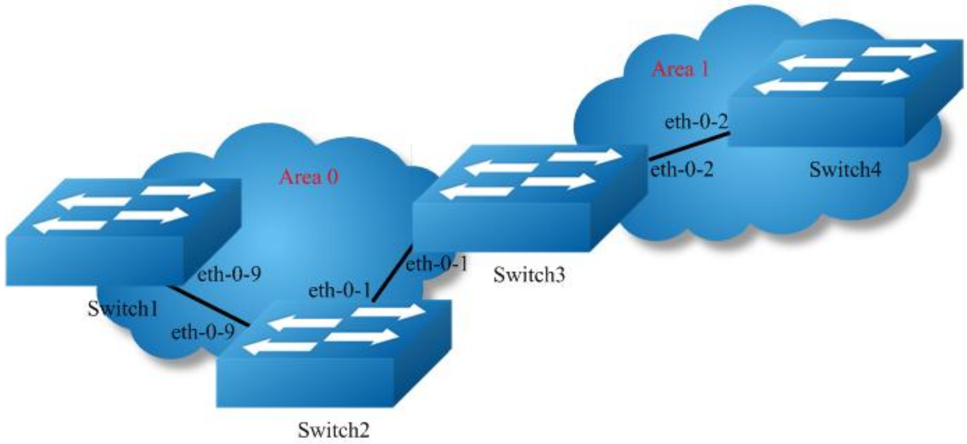

Figure 4-4 ospf authentication

In our implementation there are two types of OSPF authentications--Null authentication (Type 0) and MD5 (Type 2) authentication. With null authentication, routing exchanges over the network are not authenticated. In Simple Text authentication, the authentication type is the same for all routers that communicate using OSPF in a network. For MD5 authentication, you configure a key and a key-id on each router. The router generates a message digest on the basis of the key, key ID and the OSPF packet and adds it to the OSPF packet.

The Authentication type can be configured on a per-interface basis or a per-area basis. Additionally, Interface and Area authentication can be used together. Area authentication is used for an area and interface authentication is used for a specific interface in the area. If the Interface authentication type is different from Area authentication type, Interface authentication type overrides the Area authentication type. If the Authentication type is not specified for an interface, the Authentication type for the area is used. The authentication command descriptions contain details of each type of authentication. Refer to the OSPF Command Reference for OSPF authentication commands.

In the example below, Switch1 and 2 are configured for both the interface and area authentications. The authentication type of interface eth-0-9 on Switch1 and interface eth-0-9 on Switch2 is null authentication mode The authentication type of interface eth-0-1 on Switch2 and interface eth-0-1 on Switch3 is md5 authentication mode The authentication type of interface eth-0-2 on Switch3 and interface eth-0-2 on Switch4 is MD5 authentication mode in area1,if you define area 1 authentication type first, you needn’t define interface authentication type, only define authentication key value.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enter the interface configure mode, set the attributes and IP address. Set the ospf authentication under the interface configure mode

Configure on Switch1:

```
Switch(config)# interface eth-0-9
Switch(config-if-eth-0-9)# no switchport
Switch(config-if-eth-0-9)# ip address 9.9.9.1/24
Switch(config-if-eth-0-9)# exit
```

Configure on Switch2:

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no switchport
Switch(config-if-eth-0-1)# ip address 1.1.1.1/24
Switch(config-if-eth-0-1)# ip ospf authentication message-digest
Switch(config-if-eth-0-1)# exit

Switch(config)# interface eth-0-9
Switch(config-if-eth-0-9)# no switchport
Switch(config-if-eth-0-9)# ip address 9.9.9.2/24
Switch(config-if-eth-0-9)# exit
```

Configure on Switch3:

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no switchport
Switch(config-if-eth-0-1)# ip address 1.1.1.2/24
Switch(config-if-eth-0-1)# ip ospf authentication message-digest
Switch(config-if-eth-0-1)# exit
Switch(config)# interface eth-0-2
Switch(config-if-eth-0-2)# no switchport
Switch(config-if-eth-0-2)# ip address 2.2.2.1/24
Switch(config-if-eth-0-2)# ip ospf authentication message-digest
Switch(config-if-eth-0-2)# ip ospf message-digest-key 1 md5 key1
Switch(config-if-eth-0-2)# exit
```

Configure on Switch4:

```
Switch(config)# interface eth-0-2
Switch(config-if-eth-0-2)# no switchport
Switch(config-if-eth-0-2)# ip address 2.2.2.2/24
Switch(config-if-eth-0-2)# ip ospf authentication message-digest
Switch(config-if-eth-0-2)# ip ospf message-digest-key 1 md5 key1
Switch(config-if-eth-0-2)# exit
```


step 3 Configure the Routing process and associate the network with a specified OSPF area

Configure on Switch1:

```
Switch(config)# router ospf
Switch(config-router)# network 9.9.9.0/24 area 0
Switch(config-router)# exit
```

Configure on Switch2:

```
Switch(config)# router ospf
Switch(config-router)# network 9.9.9.0/24 area 0
Switch(config-router)# network 1.1.1.0/24 area 0
Switch(config-router)# exit
```

Configure on Switch3:

```
Switch(config)# router ospf
Switch(config-router)# area 1 authentication message-digest
Switch(config-router)# network 2.2.2.0/24 area 1
Switch(config-router)# network 1.1.1.0/24 area 0
Switch(config-router)# exit
```

Configure on Switch4:

```
Switch(config)# router ospf
Switch(config-router)# area 1 authentication message-digest
Switch(config-router)# network 2.2.2.0/24 area 1
Switch(config-router)# exit
```

step 4 Exit the configure mode

```
Switch(config)# end
```

step 5 Validation

Use the following command to display the neighbor of ospf:

Switch1：

```
Switch# show ip ospf neighbor

Neighbor ID     Pri State           Dead Time Address         Interface
RXmtL RqstL DBsmL
9.9.9.2           1   Full/DR         37.347s   9.9.9.2         eth-0-9:9.9.9.1
0     0     0
```

Switch2：

```
Switch# show ip ospf neighbor

```



```
Neighbor ID      Pri State            Dead Time Address          Interface
RXmtL RqstL DBsmL
9.9.9.1          1    Full/Backup      32.633s   9.9.9.1          eth-0-9:9.9.9.2
0      0      0
10.10.10.13      1    Full/DR          33.811s   1.1.1.2          eth-0-1:1.1.1.1
0      0      0
```

Switch3：

```
Switch# show ip ospf neighbor

Neighbor ID      Pri State            Dead Time Address          Interface
RXmtL RqstL DBsmL
2.2.2.2          1    Full/Backup      35.738s   2.2.2.2          eth-0-2:2.2.2.1
0      0      0
9.9.9.2          1    Full/Backup      30.681s   1.1.1.1          eth-0-1:1.1.1.2
0      0      0
```

Switch4：

```
Switch# show ip ospf neighbor

Neighbor ID      Pri State            Dead Time Address          Interface
RXmtL RqstL DBsmL
10.10.10.13      1    Full/DR          34.501s   2.2.2.1          eth-0-2:2.2.2.2
0      0      0
```

Use the following command to display the interface of ospf:

Switch3：

```
Switch# show ip ospf interface
eth-0-2 is up
  ifindex 113, MTU 1500 bytes, BW 40000 Mbit <UP,BROADCAST,RUNNING,MULTICAST>
  Internet Address 2.2.2.1/24, Broadcast 2.2.2.255, Area 0.0.0.1
  MTU mismatch detection:enabled
  Router ID 10.10.10.13, Network Type BROADCAST, Cost: 3
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 10.10.10.13, Interface Address 2.2.2.1
  Backup Designated Router (ID) 2.2.2.2, Interface Address 2.2.2.2
  Saved Network-LSA sequence number 0x80000002
  Multicast group memberships: OSPFAllRouters OSPFDesignatedRouters
  Timer intervals configured, Hello 10s, Dead 40s, Wait 40s, Retransmit 5s
    Hello due in 5.078s
  Neighbor Count is 1, Adjacent neighbor count is 1
  Message digest authentication enabled
        Youngest key id is 1
```

Use the following command to display the protocol state of ospf process:

Switch3：

```
Switch# show ip ospf
 OSPF Routing Process, Router ID: 2.2.2.1
 Supports only single TOS (TOS0) routes
```



```
This implementation conforms to RFC2328
RFC1583Compatibility flag is disabled
OpaqueCapability flag is disabled
Initial SPF scheduling delay 0 millisec(s)
Minimum hold time between consecutive SPFs 50 millisec(s)
Maximum hold time between consecutive SPFs 5000 millisec(s)
Hold time multiplier is currently 1
SPF algorithm last executed 9.158s ago
Last SPF duration 645 usecs
SPF timer is inactive
LSA minimum interval 5000 msecs
LSA minimum arrival 1000 msecs
Write Multiplier set to 20
Refresh timer 10 secs
This router is an ABR, ABR type is: Alternative Cisco
Number of external LSA 0. Checksum Sum 0x00000000
Number of opaque AS LSA 0. Checksum Sum 0x00000000
Number of areas attached to this router: 2
Area ID: 0.0.0.0 (Backbone)
  Number of interfaces in this area: Total: 1, Active: 1
  Number of fully adjacent neighbors in this area: 1
  Area has no authentication
  SPF algorithm executed 6 times
  Number of LSA 9
  Number of router LSA 4. Checksum Sum 0x00024415
  Number of network LSA 3. Checksum Sum 0x000120a4
  Number of summary LSA 2. Checksum Sum 0x0001066c
  Number of ASBR summary LSA 0. Checksum Sum 0x00000000
  Number of NSSA LSA 0. Checksum Sum 0x00000000
  Number of opaque link LSA 0. Checksum Sum 0x00000000
  Number of opaque area LSA 0. Checksum Sum 0x00000000

Area ID: 0.0.0.1
  Shortcutting mode: Default, S-bit consensus: ok
  Number of interfaces in this area: Total: 1, Active: 1
  Number of fully adjacent neighbors in this area: 1
  Area has message digest authentication
  Number of full virtual adjacencies going through this area: 0
  SPF algorithm executed 5 times
  Number of LSA 8
  Number of router LSA 3. Checksum Sum 0x0000c07b
  Number of network LSA 2. Checksum Sum 0x00014ae2
  Number of summary LSA 3. Checksum Sum 0x00006b1b
  Number of ASBR summary LSA 0. Checksum Sum 0x00000000
  Number of NSSA LSA 0. Checksum Sum 0x00000000
  Number of opaque link LSA 0. Checksum Sum 0x00000000
  Number of opaque area LSA 0. Checksum Sum 0x00000000
```


5 Security Configuration Guide

5.1 Configuring Time-Range

5.1.1 Overview

Function Introduction

A time range is created that defines specific absolute times or periodic times of the day and week in order to implement time-based function, such as ACLs. The time range is identified by a name and then referenced by a function, which by itself has no relevance. Therefore, the time restriction is imposed on the function itself. The time range relies on the system clock.

Principle Description

N/A

5.1.2 Configuration

Create an absolute time range

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create a time-range and set absolute time

```
Switch(config)# time-range test-absolute
Switch(config-time-range-test-absolute)# absolute start 1:1:2 1 1 2012 end 1:1:3 1
7 2012
Switch(config-time-range-test-absolute)# exit
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

```
DUT1# show time-range
time-range test-absolute
   absolute start 01:01:02 01 01 2012 end 01:01:03 01 07 2012
```

Create a periodic time range

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create a time-range and set periodic time

```
Switch(config)# time-range test-periodic
Switch(config-time-range-test-periodic)# periodic 1:1 mon to 1:1 wed
Switch(config-time-range-test-periodic)# exit
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

```
DUT1# show time-range
time-range test-periodic
   periodic 01:01 Mon to 01:01 Wed
```

5.1.3 Application cases

N/A

5.2 Configuring ACL

5.2.1 Overview

Function Introduction

Access control lists (ACLs) classify traffic with the same characteristics. The ACL can have multiple access control entries (ACEs), which are commands that match fields against the contents of the packet. ACLs can filter packets received on interface by many fields such as ip address, mac address and deny or permit the packets.

In this document, it will provide two kinds of ACL: IP ACL, MAC ACL. In IP ACL, it will filter the ipv4 packets and ARP packets; In MAC ACL, it will filter all the packets with L2 header.

Principle Description

The following terms and concepts are used to describe ACL：

- **Access control entry (ACE)**: Each ACE includes an action element (permit or deny) and a series of filter element based on criteria such as source address, destination address, protocol, and protocol-specific parameters.
- **MAC ACL**: MAC ACL can filter packet by mac-sa and mac-da, and the mac-address can be masked, or configured as host id, or configured as any to filter all MAC addresses. MAC ACL can also filter other L2 fields such as COS, VLAN-ID, INNER-COS, INNER-VLAN-ID, L2 type.
- **IP ACL**: IP ACL can filter packet by ip-sa and ip-da, and ip-address can be masked, or configured as host id, or configured as any to filter all IPv4 address. IP ACL can also filter other L3 fields such as DSCP, L4 protocol and L4 fields such as TCP port, UDP port, and so on.
- **Time Range**: Time range can define a period of time only between which the ACE can be valid if the ACE is associated to the time range.

5.2.2 Configuration

ACL Configuration

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create access list

mac access list:

```
Switch(config)# mac access-list aaa
Switch(config-mac-acl-aaa)# 10 permit src-mac host 0000.0000.1111 dest-mac any
Switch(config-mac-acl-aaa)# 20 deny src-mac any
Switch(config-mac-acl-aaa)# exit
```

ip access list:

```
Switch(config)# ip access-list bbb
Switch(config-ip-acl-bbb)# 10 permit src-ip host 1.1.1.1 dest-ip any
```



```
Switch(config-ip-acl-bbb)# 20 deny src-ip any dest-ip any
Switch(config-ip-acl-bbb)# exit
```

step 3 Exit the configure mode

```
Switch(config)# end
```

ACL Configuration with time-range

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create time-range

```
Switch(config)# time-range tr1
Switch(config-time-range-tr1)# periodic 9:10 daily to 20:00
Switch(config-time-range-tr1)# exit
Switch(config)# time-range tr2
Switch(config-time-range-tr2)# absolute start 12:00:00 10 10 2018 end 0:0:0 10 25
2018
Switch(config-time-range-tr2)# exit
```

step 3 Create rules with time-range

```
Switch(config)# mac access-list tracl1
Switch(config-mac-acl-tracl1)# permit src-mac host 0000.0000.2222 time-range tr1
Switch(config-mac-acl-tracl1)# deny src-mac host 0000.0000.3333 time-range tr2
Switch(config-mac-acl-tracl1)# exit
Switch(config)# ip access-list tracl2
Switch(config-ip-acl-tracl2)# permit src-ip host 2.2.2.2 time-range tr1
Switch(config-ip-acl-tracl2)# deny src-ip host 3.3.3.3 time-range tr2
Switch(config-ip-acl-tracl2)# exit
```

step 4 Exit the configure mode

```
Switch(config)# end
```

Apply the ACL to the CoPP

step 1 Enter the configure mode

```
Switch# configure terminal
```


step 2 Create class-map, and bind the access list

```
Switch(config)# class-map cmap1
Switch(config-class-map-cmap1)# match access-list aaa
Switch(config-class-map-cmap1)# match access-list bbb
Switch(config-class-map-cmap1)# match access-list tracl1
Switch(config-class-map-cmap1)# match access-list tracl2
Switch(config-class-map-cmap1)# exit
```

step 3 Create policy-map and bind the class map

```
Switch(config)# policy-map pmap1
Switch(config-policy-map-pmap1)# class cmap1
Switch(config-policy-map-pmap1-cmap-cmap1)# exit
Switch(config-policy-map-pmap1)# exit
```

step 4 Apply the policy to CoPP

```
Switch(config)# control-plane
Switch(config-control-plane)# policy input pmap1
```

步骤 5 退出配置模式

step 5 Exit the configure mode

```
Switch(config)# end
```

step 6 Validation

The result of show mac access-list:

```
Switch# show mac access-list
mac access-list aaa
10 permit src-mac host 0000.0000.1111 dest-mac any
20 deny src-mac any
mac access-list tracl1
10 permit src-mac host 0000.0000.2222 time-range tr1 (invalid)
20 deny src-mac host 0000.0000.3333 time-range tr2 (valid)
```

The result of show ip access-list:

```
Switch# show ip access-list
ip access-list bbb
10 permit src-ip host 1.1.1.1 dest-ip any
20 deny src-ip any dest-ip any
ip access-list tracl2
10 permit src-ip host 2.2.2.2 time-range tr1 (invalid)
20 deny src-ip host 3.3.3.3 time-range tr2 (valid)
```

The result of show clock and show time-range info:


```
Switch# show clock
05:46:36 UTC Fri Oct 21 2018

Switch# show time-range
time-range tr1
periodic 09:10 daily to 20:00
time-range tr2
absolute start 12:00:00 10 10 2018 end 00:00:00 10 25 2018

Switch# show time-range info
time-range tr1
mac access-list tracl1 sequence-num 10
ip access-list tracl2 sequence-num 10
time-range tr2
mac access-list tracl1 sequence-num 20
ip access-list tracl2 sequence-num 20
```

The result of show class-map and show policy-map:

```
Switch# show class-map
class-map cmap1
 match access-list aaa
 match access-list bbb
 match access-list tracl1
 match access-list tracl2

Switch# show policy-map
policy-map pmap1
 class cmap1
```

5.2.3 Application cases

N/A

5.3 Configuring AAA and Radius

5.3.1 Overview

Function Introduction

Authentication verifies users before they are allowed access to the network and network services. System can use AAA authentication methods and Non-AAA authentication methods. RADIUS Authentication is one of AAA authentication methods. RADIUS is a distributed client/server system that secures networks against unauthorized access. RADIUS is widely used protocol in network environments. It is commonly used for embedded network devices such as routers, modem servers, switches, etc. RADIUS clients run on support routers and switches.

Clients send authentication requests to a central RADIUS server, which contains all user authentication and network service access information.

Principle Description

Terminology:

- AAA: Authentication Authorization and Accounting
- RADIUS: Remote Authentication Dial-In User Service
- TACACS+: Terminal Access Controller Access Control System Plus

Reference to IETF RFC2865, RFC2866 for RADIUS; Reference to IETF RFC1321 for MD5.

5.3.2 Configuration

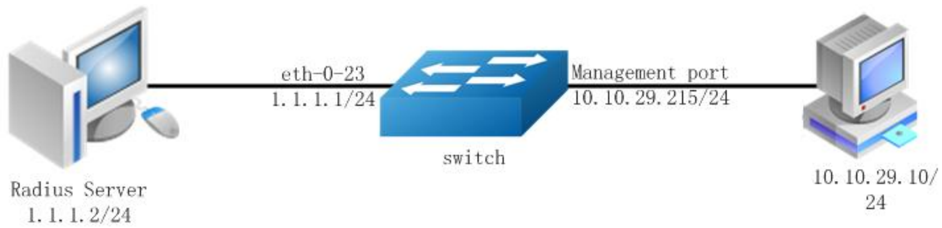

Figure 5-1 RADIUS authentication application

5.3.3 AAA Configuration

The figure above is the networking topology for RADIUS authentication functions. We need one Switch and two computers for this test.

One computer as RADIUS server, its ip address of the eth0 interface is 1.1.1.2/24.

Switch has RADIUS authentication function. The ip address of interface eth-0-23 is 1.1.1.1/24. The management ip address of switch is 10.10.29.215, management port is connected the PC for test login, PC’s ip address is 10.10.29.10.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enable AAA

```
Switch(config)# aaa new-model
Switch(config)# aaa authentication login radius-login radius local
```



```
Switch(config)# aaa authorization exec radius-author radius
Switch(config)# aaa accounting exec radius-acct start-stop radius
```

step 3 Configure Radius server

```
Switch(config)# radius-server host 1.1.1.2 auth-port 1819 key keyname
```

step 4 Configure a layer 3 interface and set ip address

```
Switch(config)# interface eth-0-23
Switch(config-if-eth-0-23)# no shutdown
Switch(config-if-eth-0-23)# no switchport
Switch(config-if-eth-0-23)# ip address 1.1.1.1/24
Switch(config-if-eth-0-23)# quit
```

step 5 set authentication mode

```
Switch(config)# line vty 0 7
Switch(config-line)# login authentication radius-login
Switch(config-line)# authorization exec radius-author
Switch(config-line)# accounting exec radius-acct
Switch(config-line)# privilege level 4
Switch(config-line)# no line-password
```

step 6 Exit the configure mode

```
Switch(config-line)# end
```

step 7 Validation

You can use command show authentication status in switch:

```
Switch# show aaa status
AAA status:
     Authentication enable
     Authorization  enable
     Accounting     enable
```

You can use command show keys in switch:

```
Switch# show aaa method-lists all
Authen queue = AAA_ML_AUTHEN_LOGIN
   Name = default   state = ALIVE: local
   Name = radius-login   state = ALIVE: radius local
Author queue = AAA_ML_AUTHOR_SHELL
   Name = default   state = ALIVE: local
   Name = radius-author   state = ALIVE: radius
Account queue = AAA_ML_ACCT_SHELL
   Name = default   state = ALIVE: none
   Name = radius-acct   state = ALIVE: radius
```



```
Account queue = AAA_ML_ACCT_COMMAND
   Name = default    state = ALIVE: none
```

Telnet output:

Figure 5-2 Telnet connecting test

NOTE Don’t forget to turn RADIUS authentication feature on.

Make sure the cables is linked correctly You can use command to check log messages if Switch can’t do RADIUS authentication:

```
Switch# show logging buffer
```

Radius server configuration (Using WinRadius for example)

Set ip address for PC:

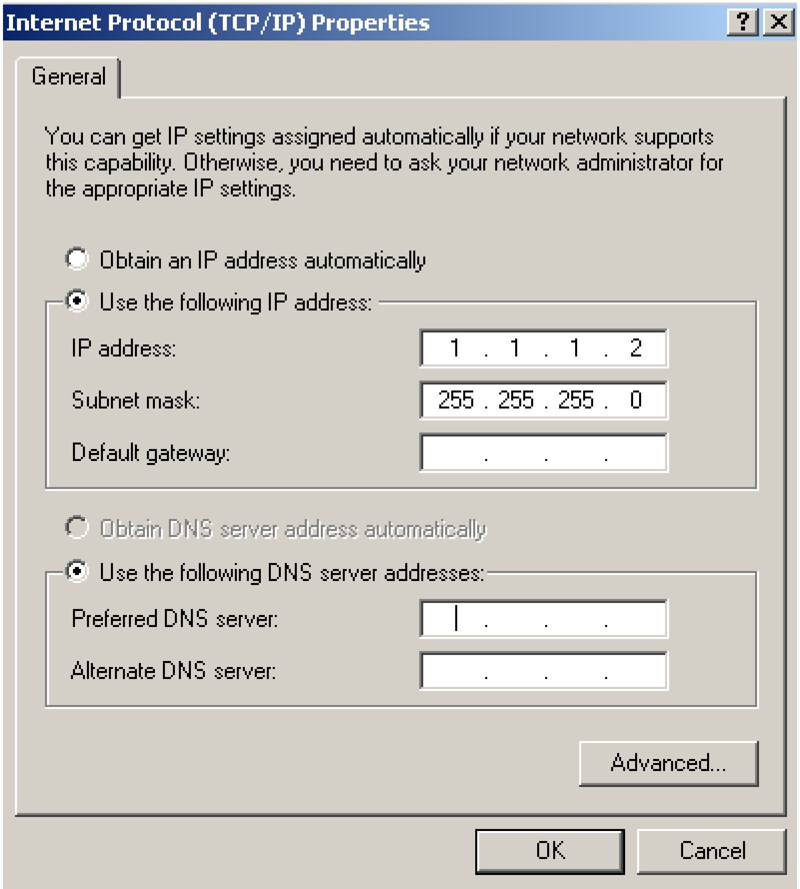

Figure 5-3 Set IP address for PC

Connectivity test between server and switch:

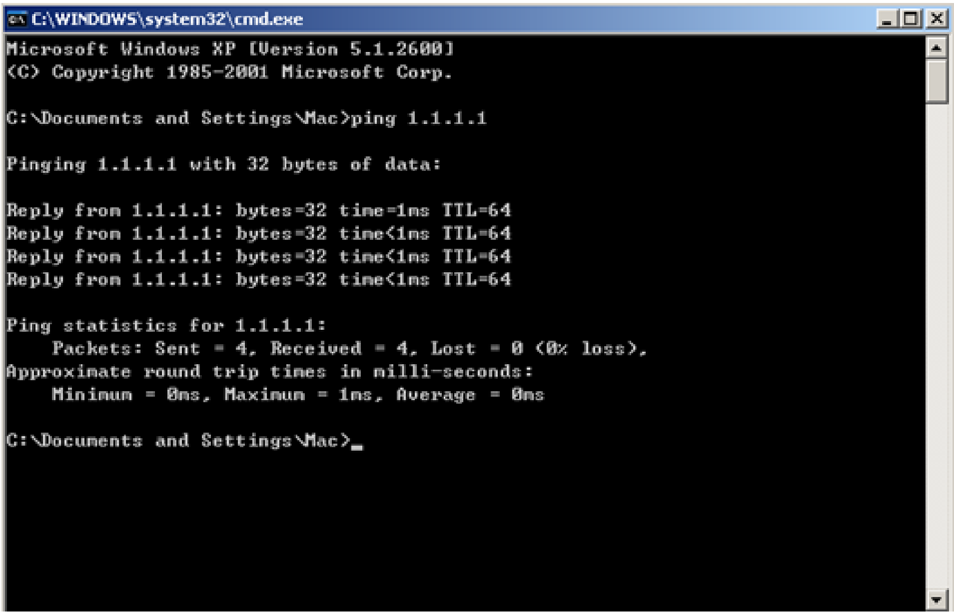

Figure 5-4 Connectivity test

Open winRadius:

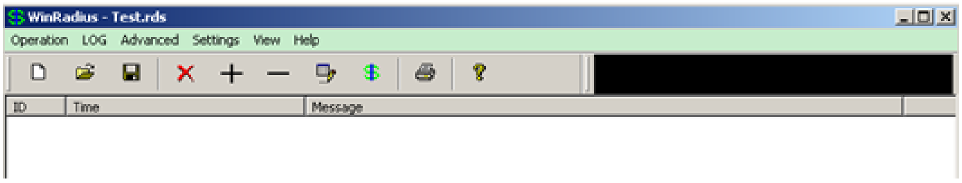

Figure 5-5 WinRadius

Configurations for winRadius:

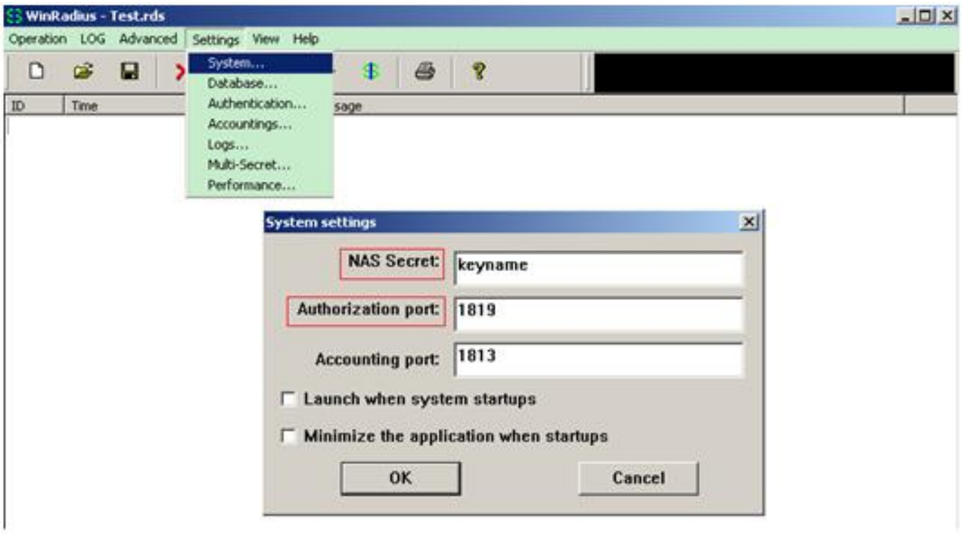

Figure 5-6 WinRadius

Add user and password:

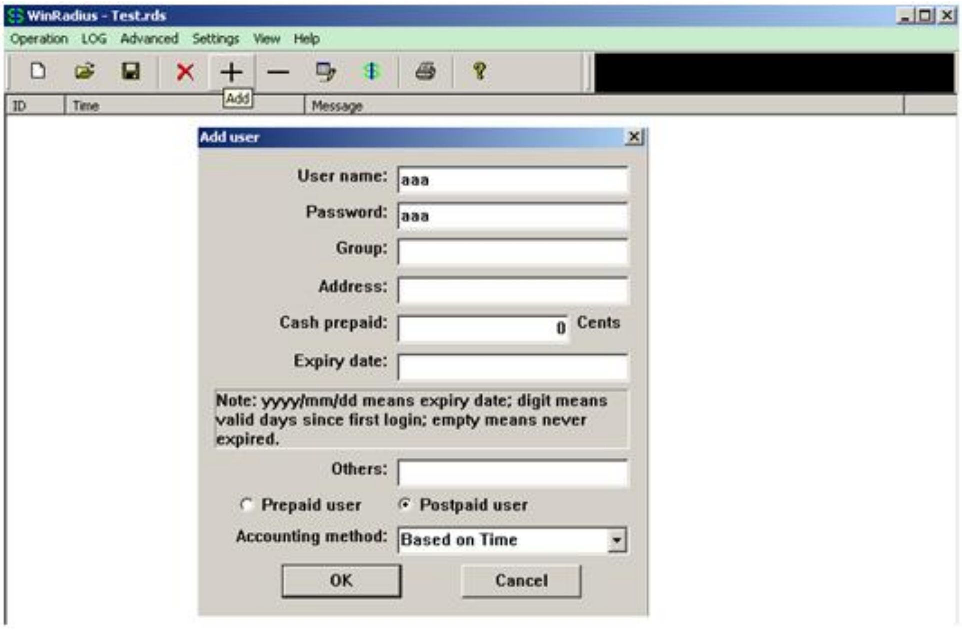

Figure 5-7 Add user and password

Connectivity test between client and switch:

```
C:\Documents and Settings\mac>ping 10.10.29.215

Pinging 10.10.29.215 with 32 bytes of data:

Reply from 10.10.29.215: bytes=32 time<1ms TTL=63
Reply from 10.10.29.215: bytes=32 time<1ms TTL=63
Reply from 10.10.29.215: bytes=32 time<1ms TTL=63
Reply from 10.10.29.215: bytes=32 time<1ms TTL=63

Ping statistics for 10.10.29.215:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

Figure 5-8 Connectivity test

5.3.4 Application cases

N/A

5.4 Configuring AAA and TACACS+

5.4.1 Overview

Function Introduction

Authentication verifies users before they are allowed access to the network and network services. System can use AAA authentication methods and Non-AAA authentication methods. TACACS+ Authentication is one of AAA authentication methods. TACACS+ is a distributed client/server system that secures networks against unauthorized access. TACACS+ is widely used protocol in network environments. It is commonly used for embedded network devices such as routers, modem servers, switches, etc. TACACS+ clients run on support routers and switches. Clients send authentication requests to a central TACACS+ server, which contains all user authentication and network service access information.

Principle Description

Reference to IETF RFC1492 for TACACS; Reference to IETF RFC1321 for MD5.

5.4.2 Configuration

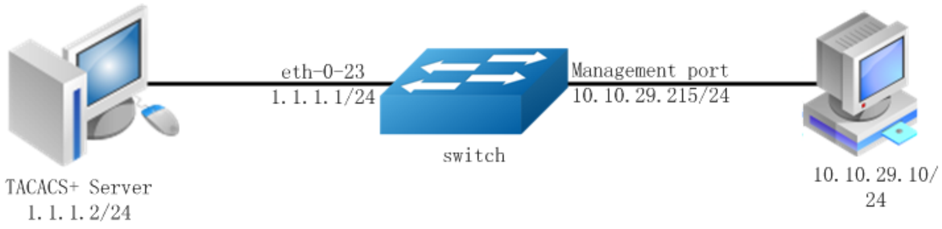

Figure 5-9 TACACS+ authentication application

The figure above is the networking topology for TACACS+ authentication functions. We need one Switch and two computers for this test. One computer as TACACS+ server, its ip address of the eth0 interface is 1.1.1.2/24. Switch has TACACS+ authentication function. The ip address of interface eth-0-23 is 1.1.1.1/24. The management ip address of switch is 10.10.29.215, management port is connected the PC for test login, PC's ip address is 10.10.29.10

AAA and TACACS+ Configuration

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enable AAA

```
Switch(config)# aaa new-model
Switch(config)# aaa authentication login tac-login tacplus local
Switch(config)# aaa authorization exec tac-author tacplus
Switch(config)# aaa accounting exec tac-acct start-stop tacplus
Switch(config)# aaa accounting commands taccmd-acct tacplus
```

step 3 Configure tacacs+ server

```
Switch(config)# tacacs-server host 1.1.1.2 auth-port 123 key keyname
```

step 4 Configure a layer 3 interface and set ip address

```
Switch(config)# interface eth-0-23
Switch(config-if-eth-0-23)# no shutdown
Switch(config-if-eth-0-23)# no switchport
Switch(config-if-eth-0-23)# ip address 1.1.1.1/24
Switch(config-if-eth-0-23)# quit
```


step 5 set authentication mode

```
Switch(config)# line vty 0 7
Switch(config-line)# login authentication tac-login
Switch(config-line)# authorization exec tac-author
Switch(config-line)# accounting exec tac-acct
Switch(config-line)# privilege level 4
Switch(config-line)# no line-password
```

step 6 Exit the configure mode

```
Switch(config-line)# end
```

step 7 Validation

You can use command show authentication status in switch：

```
Switch# show aaa status
AAA status:
     Authentication enable
     Authorization  enable
     Accounting     enable
```

You can use command show keys in switch:

```
Switch# show aaa method-lists all
Authen queue = AAA_ML_AUTHEN_LOGIN
   Name = default   state = ALIVE: local
   Name = tac-login   state = ALIVE: tacplus local
Author queue = AAA_ML_AUTHOR_SHELL
   Name = default   state = ALIVE: local
   Name = tac-author   state = ALIVE: tacplus
Account queue = AAA_ML_ACCT_SHELL
   Name = default   state = ALIVE: none
   Name = tac-acct   state = ALIVE: tacplus
Account queue = AAA_ML_ACCT_COMMAND
   Name = default   state = ALIVE: none
   Name = taccmd-acct   state = ALIVE: tacplus
```

Telnet output:

Figure 5-10 Telnet connecting test

Tacacs server configuration

Download TACACS+ server code, DEVEL.201105261843.tar.bz2.

Build the TACACS+ server.

Add username and password in configure file.

```
#!../obj.linux-2.6.9-89.29.1.elsmp-x86_64/tac_plus
id = spawnd {
        listen = { port = 49 }
        spawn = {
                instances min = 1
                instances max = 10
        }
        background = no
}
        user = aaa {
                password = clear bbb
                member = guest
        }
```

Run TACACS+ server:

```
[disciple: ~]$ ./tac_plus ./tac_plus.cfg.in -d 1
```

Use Ping command for test on PC:

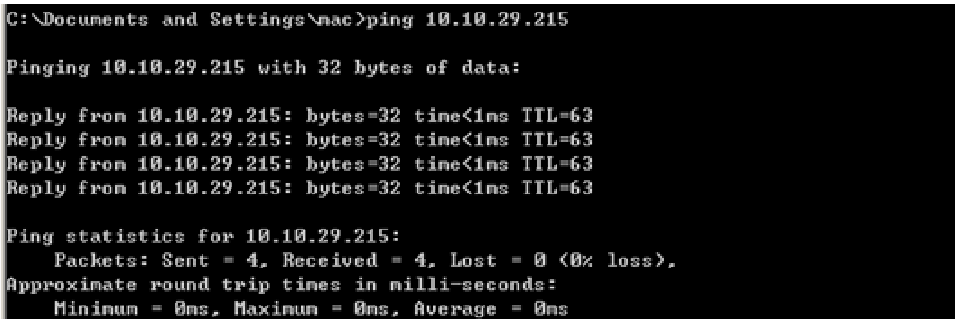

Figure 5-11 Connectivity test

5.4.3 Application cases

N/A

5.5 Configuring DDoS

5.5.1 Overview

Function Introduction

A denial-of-service attack (DoS attack) or distributed denial-of-service attack (DDoS attack) is an attempt to make a computer resource unavailable to its intended users. Although the means to carry out, motives for, and targets of a DoS attack may vary, it generally consists of the concerted efforts of a person or people to prevent an Internet site or service from functioning efficiently or at all, temporarily or indefinitely. Perpetrators of DoS attacks typically target sites or services hosted on high-profile web servers such as banks, credit card payment gateways, and even root name servers. The term is generally used with regards to computer networks, but is not limited to this field, for example, it is also used in reference to CPU resource management.

DDoS prevent is a feature which can protect our switch from follow kinds of denial-of-service attack and intercept the attack packets.

The flowing types are supported：

- **ICMP flood**: attackers overwhelm the victim with ICMP packets.
- **Smurf attack**: attackers flood a target system via spoofed broadcast ping messages.
- **SYN flood**: attackers send a succession of SYN requests to a target’s system.
- **UDP flood**: attackers send a large number of UDP packets to random ports on a remote host.
- **Fraggle attack**:attackers send a large number of UDP echo traffic to IP broadcast addresses, all fake source address.
- **Small-packet**: attackers send a large number of small packets to the system utill the resource exhaust.
- **bad mac intercept**: attackers send packets with same source and destination MAC address.
- **bad ip equal**: attackers send packets with same source and destination IP address.

Principle Description

N/A

5.5.2 Configuration

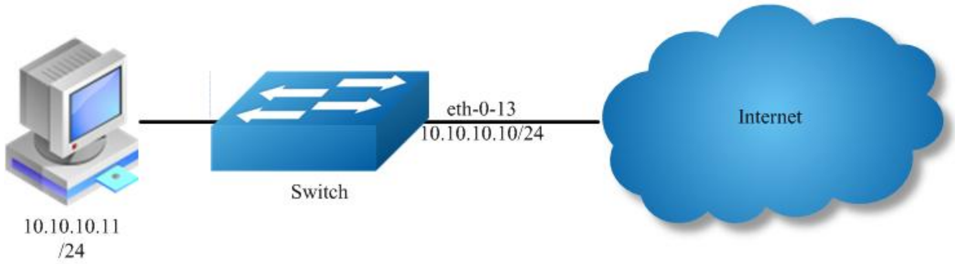

Figure 5-12 Topology for DDoS test

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set DDoS

Enable ICMP flood intercept and set the max received ICMP packet rate 100 packets per-second

```
Switch(config)# ip intercept icmp maxcount 100
```

Enable UDP flood intercept and set the max received UDP packet rate 100 packets per-second

```
Switch(config)# ip intercept udp maxcount 100
```

Enable Smurf attack intercept

```
Switch(config)# ip intercept smurf
```

Enable SYN flood intercept and set the max received SYN packet rate 100 packets per-second

```
Switch(config)# ip intercept tcp maxcount 100
```

Enable Fraggle attack intercept

```
Switch(config)# ip intercept fraggle
```

Enable Small-packet attack intercept and set the received packet length is be more than or equal to 32


```
Switch(config)# ip intercept small-packet length 32
```

Enable packet source IP equals destination IP intercept

```
Switch(config)# ip intercept ipeq
```

Enable packet source MAC equals destination MAC intercept

```
Switch(config)# ip intercept maceq
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

```
Switch# show ip-intercept config
Current DDoS Prevent configuration:
--------------------------------+--------+--------+---------
Fraggle Attack Intercept         :Enable
ICMP Flood Intercept             :Enable  Maxcount:100
IP Equal Intercept               :Enable
MAC Equal Intercept              :Enable
Small-packet Attack Intercept    :Enable  Packet Length:32
Smurf Attack Intercept           :Enable
SYN Flood Intercept              :Enable  Maxcount:100
UDP Flood Intercept              :Enable  Maxcount:100
DUT1# show ip-intercept statistics
Current DDoS Prevent statistics:
--------------------------------------------------+---------------
Resist Fraggle Attack packets number              :  0
Resist ICMP Flood packets number                  :  0
Resist Small-packet Attack packets number         :  0
Resist Smurf Attack packets number                :  0
Resist SYN Flood packets number                   :  0
Resist UDP Flood packets number                   :  0
mgmt-if Resist Fraggle Attack packets number      :  0
mgmt-if Resist ICMP Flood packets number          :  0
mgmt-if Resist Small-packet Attack packets number :  0
mgmt-if Resist Smurf Attack packets number        :  0
mgmt-if Resist SYN Flood packets number           :  0
mgmt-if Resist UDP Flood packets number           :  0
```

5.5.3 Application cases

N/A

6 Device Management Configuration Guide

6.1 Configuring STM

6.1.1 Overview

Function Introduction

Switch Table Management (STM) is used to configure system resources in the switch to optimize support for specific features, depending on how the switch is used in the network.

You can select a profile to provide maximum system usage for some functions; for example, use the default profile to balance resources and use vlan profile to obtain max MAC entries.

To allocate ternary content addressable memory (TCAM) resources for different usages, the switch STM profile prioritize system resources to optimize support for certain features. You can select STM templates to optimize these features:

- default: The default template gives balance to all functions.

When users configured a profile mode which is not exist in the next reboot image, then default hardware configure will be used when system up with the next image. The hardware configure may be different from the default profile.

Principle Description

N/A

6.1.2 Configuration

Follow these guidelines when selecting and configuring STM profiles.

You must reload the switch for the configuration to take effect.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set STM profile(use default for example)

```
Switch(config)# stm prefer default
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

This is an example of an output display:

```
Switch# show stm prefer current
default profile::
number of Ethernet features:
    VLAN forwarding instances               : 100/4094
    Ucast MAC addresses                     : 0/65536
        static MAC address                  : 0/1024
        dynamic MAC address                 : 0/65536
number of macfilter entry                   : 0/128
number of IP unicast features:
    IPv4 host routes                        : 0/4096
    Indirect IPv4 routes                    : 2/8192
    IPv4 ecmp groups                        : 0/8192
    IPv4 source guard entries               : 0/1024
number of Security features:
    ACL entries                             : 0/256
    System ACL confiure                     : 0/4096
    System ACE confiure                     : 0/8192
    System L4 Port confiure                 : 0/7
number of dot1x mac based                   : 0/256
number of L2 multicast features:
    Group Member                            : 0/2048
    L2 Mcast Entry                          : 0/2048
number of link aggregation(static & lacp)  : 0/55
```

step 5 Reboot the device

```
Switch# reload
```

6.1.3 Application cases

N/A

6.2 Configuring syslog

6.2.1 Overview

Function Introduction

The system message logging software can save messages in a log file or direct the messages to other devices. The system message logging facility has these features:

- Provides you with logging information for monitoring and troubleshooting.
- Allows you to select the types of logging information that is captured.
- Allows you to select the destination of the captured logging information.

By default, the switch logs normal but significant system messages to its internal buffer and sends these messages to the system console. You can specify which system messages should be saved based on the type of the severity level. The messages are time-stamped to enhance real-time debugging and management.

You can access the logged system messages using the switch command-line interface (CLI) or by saving them to a properly configured log server. The switch software saves the log messages in an internal buffer that can store up to 1000 messages. You can monitor the system messages remotely by accessing the switch through Telnet or the console port, or by viewing the logs on a log server.

Principle Description

Terminology:

Terminology	Description
Logging	Current logging configuration
Show	Show logging configuration
Levels	Severity level information
Enable	Enable write log to local file
Disable	Disable write log to local file

System Message Log Facility Types：

Facility Name	Definition
kern	kernel messages
user	random user-level messages
mail	mail system
daemon	system daemons
auth	security/authorization messages
syslog	messages generated internally by syslogd
lpr	line printer subsystem
news	network news subsystem
uucp	UUCP subsystem
cron	clock daemon
authpriv	security/authorization messages (private)
ftp	ftp daemon

Severity Level Definitions:

Severity Level	Definition
emergency	system is unusable
alert	action must be taken immediately
critical	critical conditions
error	error conditions
warning	warning conditions
notice	normal but significant condition
information	Informational
debug	debug-level messages

6.2.2 Configuration

Configuring Logging server

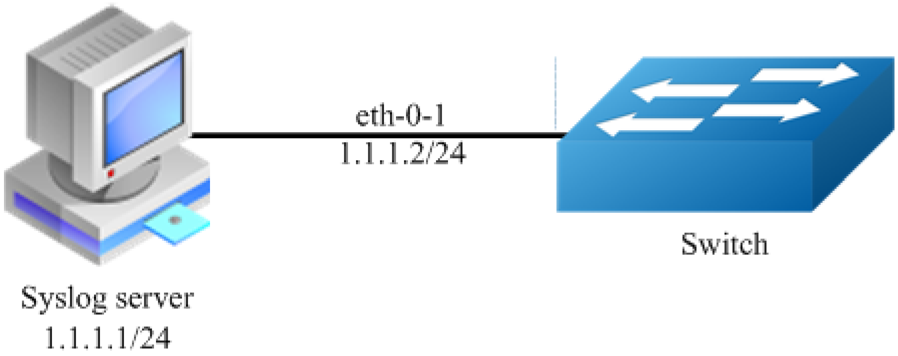

Figure 6-1 syslog server

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enable logging server and set the attributes

```
Switch (config)# logging server enable
Switch (config)# logging server address 1.1.1.1
Switch (config)# logging server severity debug
Switch (config)# logging server facility mail
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

```
Switch# show logging
Current logging configuration:
------------------------------------------------------------
logging buffer 700
logging timestamp bsd
logging file enable
logging level file warning
logging level module debug
logging server enable
logging server severity debug
logging server facility mail
logging server address 1.1.1.1 inband
logging merge enable
```



```
logging merge fifo-size 1024
logging merge timeout 10
```

Configuring Logging Buffer Size

By default, the number of messages to log to the logging buffer is 500. If desired, you can set the number between 10 and 1000.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set the logging Buffer Size

```
Switch(config)# logging buffer 700
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

```
Switch# show logging
Current logging configuration:
------------------------------------------------------------
logging buffer 700
logging timestamp bsd
logging file enable
logging level file warning
logging level module debug
logging server enable
logging server severity debug
logging server facility mail
logging merge enable
logging merge fifo-size 1024
logging merge timeout 10
Switch#
```

The following is the information of logging server:

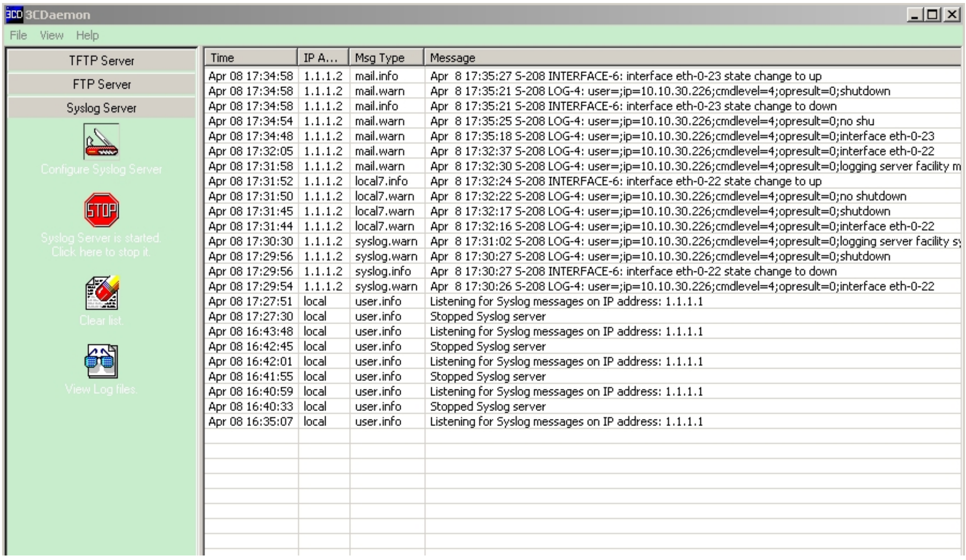

Figure 6-2 syslog on server

NOTE You can use command to check showing Logging Information. When configuring the syslog Servers, make sure the cables is linked correctly and two computers can ping each other. Before you can send the system log messages to a log server, you must configure Syslog Software, at the end you can see the log from your software.

6.2.3 Application cases

N/A

6.3 Configuring mirror

6.3.1 Overview

Function Introduction

Mirror function can send one or more copies of packets which are passing through the ports/vlans or sending and receiving by CPU to one or more specified destination ports. It can also send the copies to the CPU and keep in memory or flash files.

The copies of the packets are used for network analyze. The mirror function does not affect the original network traffic.

Principle Description

The following describes concepts and terminology associated with mirror configuration:

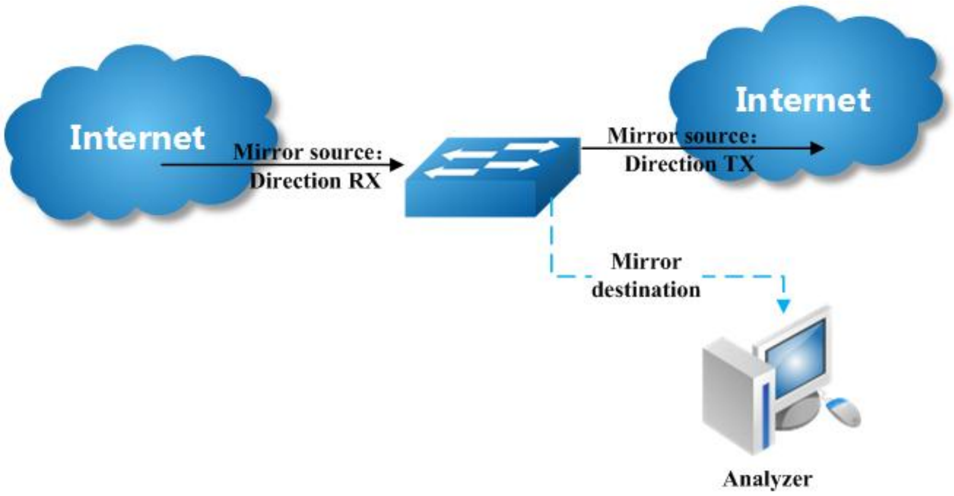

Figure 6-3 Mirror

1. Mirror session

A mirror session is an association of a mirror destination with one or more mirror source. The mirror destination and mirror source will describe later.

The device supports up to 4 mirror sessions.

Mirror sessions do not interfere with the normal operation of the switch. However, an oversubscribed mirror destination, for example, a 10-Gbps port monitoring a 100-Gbps port, results in dropped or lost packets.

You can configure mirror sessions on disabled ports; however, a mirror session does not become active unless you enable the destination port and at least one source port or source VLAN for that session.

A mirror session remains inactive after system power-on until the destination port is operational.

2. Mirror direction

The device supports to set the direction of the mirror source, there are 3 options for choose: TX/RX/BOTH.

- **Receive (RX) mirror**: The goal of receive (or ingress) mirror is to monitor as much as possible packets received by the source interface or VLAN before any modification or processing is performed by the switch. A copy of each packet received (except these packets: BPDU, LACPDU, BMGPDU, packets have been discarded by IP-MAC binding check for vlan_based mirror, CRC error packets for both Port_based and vlan_based mirror) by the source is sent to the destination port for that mirror session. You can monitor a series or range of ingress ports or VLANs in a mirror session. Packets that are modified because of routing are copied without modification; that is, the original packet is copied. Packets that are modified because of quality of service (QoS)—for example, modified Differentiated Services Code Point (DSCP)—are copied without modification. Packets that are modified because of VLAN translation or VLAN classification is copied without the modification. Some features that can cause a packet to be dropped during receive processing have no effect on mirror, the destination port can receive a copy of the packet even if the actual incoming packet is dropped. These features include ingress ACL, VLAN's ingress filter, MAC filter, STP, VLAN tag control, port security, unknown routing packets.
- **Transmit (TX) mirror**: The goal of transmit (or egress) mirror is to monitor as much as possible packets sent by the source interface after all modification and processing is performed by the switch. A copy of each packet (except these packets: packets from CPU port for vlan_based mirror, mirroring packets for both Port_based and vlan_based mirror) sent by the source is sent to the destination port for that mirror session. Some features that can cause a packet to be dropped during transmit processing might have affect on mirror.
- **Both**: In a mirror session, you can monitor a single port for both received and sent packets.

3. Mirror source

The Mirror source is the original traffic of the network.

A source port (also called a monitored port) is a switched or routed port that you monitor for network traffic analysis. In a single mirror session, you can monitor source port traffic such as received (Rx), transmitted (Tx), or bidirectional (both). The switch supports any number of source ports (up to the maximum number of available ports on the switch) and any number of source VLANs (up to the maximum number of VLANs supported).

A source port has these characteristics:

- It can be any port type (for example, EtherChannel).
- It can only be monitored in a single mirror session.
- It cannot be a destination port.
- Each source port can be configured with a direction (ingress, egress, or both) to monitor. For EtherChannel sources, the monitored direction would apply to all the physical ports in the group.
- Source ports can be in the same or different VLANs.
- For VLAN sources, user should create VLAN Interface before configure a vlan source. It can not be a physical port that is assigned to an EtherChannel group.

The types of source are described as following:

- **Source port**: A source port is a layer2 or layer 2 interface which need to be monitored. A physical port or link agg port can be a source port. The member of link agg port is not supported to be a mirror source.
- **Source VLAN:** A source vlan is a vlan which need to be monitored. User should create a vlan interface before set a vlan as mirror source.
- **CPU**:User can set CPU as mirror source to monitor the packets send to or receive from the CPU. The copies of packets send to the mirror destination are before copp process.

4. Mirror destination

Each mirror session must have a destination port (also called a monitoring port) that receives a copy of traffic from the source ports and VLANs.

The destination port has these characteristics:

- It must reside on the same switch as the source port.
- It can be any Ethernet physical port.
- It can not be physical port that is assigned to an EtherChannel group.
- It can participate in only one mirror session at a time (a destination port in one mirror session cannot be a destination port for a second mirror session).
- It cannot be a source port.

- The port does not transmit any traffic except that required for the mirror session.
- It does not participate in spanning tree while the mirror session is active.
- When it is a destination port, all other normal system function of this port should not work until mirror destination configure disabled on this port.
- No address learning occurs on the destination port.
- The real statues of the speed/duplex might not coincide with the values which are displayed.

The types of destination are described as following:

Local destination port: The destination port should be a physical port or link agg port, member of link agg port is not supported. The destination port has these characteristics:

- It must reside on the same switch as the source port.
- It should not be in "shutdown" state
- It can participate in only one mirror session at a time (a destination port in one mirror session cannot be a destination port for a second mirror session).
- It cannot be a source port.
- The port does not transmit any traffic except that required for the mirror session.
- It does not participate in spanning tree while the mirror session is active.
- When it is a destination port, all other normal system function of this port should not work until mirror destination configure disabled on this port.
- No address learning occurs on the destination port.
- The real statues of the speed/duplex might not coincide with the values which are displayed.

Multi-destination: The device supports to use a group of destination ports to receive several copies of the traffic. The characteristics of each member in the group of destination ports are same as single destination port.

Remote destination：A remote mirror destination is a remote destination vlan, which has a specified out-going port. The copies of the packets should send to the specified port and add the tag of the remote vlan. A remote destination has these characteristics:

- It is a vlan with a specified out going port.
- The remote VLAN range should be 2 to 4094. If the VLAN isn't created in system, user can not configure this VLAN as mirror remote vlan.
- The out going port should be a physical port. User should manually check if the out going port can transfer mirrored packets.
- Monitor traffic packets are inserted a tag with the remote VLAN ID and directed over the specified out going port to the mirror destination session device.

CPU destination: send the copies of packet to the CPU of current device. If there is no analyzer available, user can use CPU as mirror destination and save the result for user or developers analyze packets.

You can analyze network traffic passing through ports or vlans by using mirror function to send a copy of the traffic to another port on the switch that has been connected to a Switch Probe device or other Remote Monitoring (RMON) probe or security device. However, when there is no other monitoring device for capturing packets, normal mirror destination to ports doesn't work. So we can set CPU as mirror destination to send a copy of the traffic to CPU for storing packets. It supports the cli to display the packets of mirror CPU and write the packets in a text file. It is a very functional debug tool. Mirror does not affect the switching of network traffic on source ports or source vlans; a copy of the packets received or sent by the source interfaces are sent to the destination CPU. The cpu-traffic-limit rate can be configured. CPU can participate as a destination in only one mirror session.

6.3.2 Configuration

Configuring Local port mirror

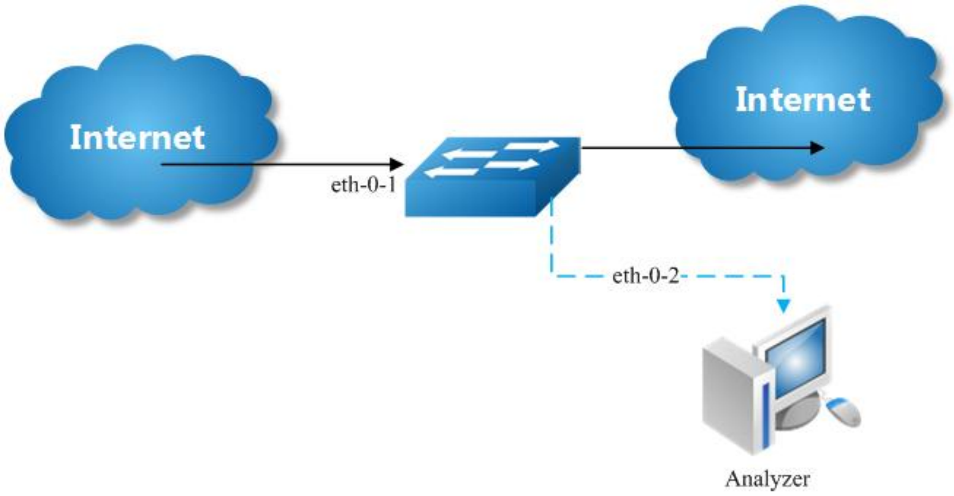

Figure 6-4 port Mirror

Copy the packets of eth-0-1 and send them to eth-0-2

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set the destination of mirror

```
Switch(config)# interface eth-0-2
Switch(config-if-eth-0-2)# no shutdown
Switch(config-if-eth-0-2)# exit
Switch(config)# monitor session 1 destination interface eth-0-2
```

step 3 Set the source of mirror

```
Switch(config)# monitor session 1 source interface eth-0-1 both
```

step 4 Exit the configure mode

```
Switch(config)# end
```

step 5 Validation

```
Switch# show monitor session 1
Session   1
----------
```



```
Status              : Valid
Type                : Local Session
Source Ports        :
  Receive Only      :
  Transmit Only     :
  Both              : eth-0-1
Source VLANs        :
  Receive Only      :
  Transmit Only     :
  Both              :
Destination Port    : eth-0-2
```

Configuring local vlan mirror

Copy the packets from vlan 10 and send them to eth-0-2

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set the destination of mirror

```
Switch(config)# interface eth-0-2
Switch(config-if-eth-0-2)# no shutdown
Switch(config-if-eth-0-2)# exit
Switch(config)# monitor session 1 destination interface eth-0-2
```

step 3 Create a vlan

```
Switch (config)#vlan 10
Switch (config-vlan10)# exit
```

step 4 Create a vlan interface

```
Switch (config)# interface vlan10
Switch (config-if-vlan10)# exit
```

step 5 Set the source of mirror

```
Switch(config)# monitor session 1 source vlan 10 rx
```

step 6 Exit the configure mode

```
Switch(config)# end
```


step 7 Validation

```
Switch# show monitor session 1
Session   1
----------
Status              : Valid
Type                : Local Session
Source Ports        :
  Receive Only      :
  Transmit Only     :
  Both              :
Source VLANs        :
  Receive Only      : 10
  Transmit Only     :
  Both              :
Destination Port    : eth-0-2
```

Configuring CPU as mirror source

Copy the packets from or to CPU and send them to eth-0-2

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set the destination of mirror

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)# exit
Switch(config)# monitor session 1 destination interface eth-0-1
```

step 3 Set the source of mirror

```
Switch(config)# monitor session 1 source cpu both
```

step 4 Exit the configure mode

```
Switch(config)# end
```

step 5 Validation

```
DUT1# show monitor session 1
Session   1
----------
Status              : Valid
Type                : Cpu Session
Source Ports        :
```



```
  Receive Only      :
  Transmit Only     :
  Both              : cpu
Source VLANs        :
  Receive Only      :
  Transmit Only     :
  Both              :
Destination Port    :eth-0-1
```

Configuring Multi-destination Mirror

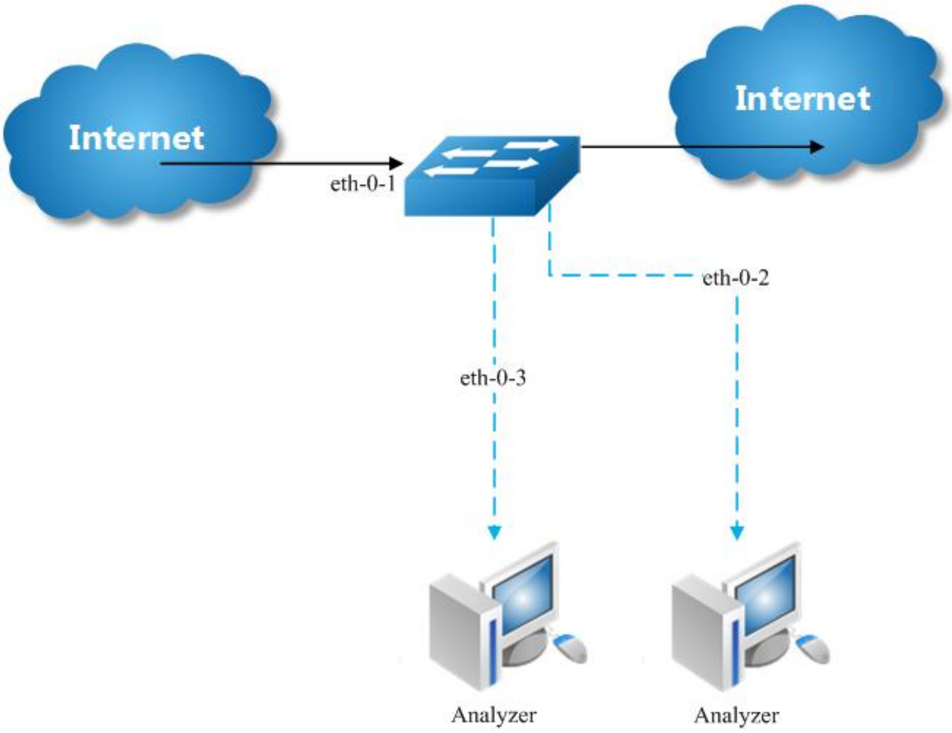

Figure 6-5 Multi-destination Mirror

Copy the packets of eth-0-1 and send them to eth-0-2 and eth-0-3

The rules of mirror source are same as single destination port. The following case use source port for example.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set the destination group of mirror

```
Switch (config)# interface range eth-0-2 - 3
Switch (config-if-range)# no shutdown
Switch (config-if-range)# exit
```



```
Switch (config)# monitor session 1 destination group 1 member eth-0-2
Switch (config)# monitor session 1 destination group 1 member eth-0-3
```

step 3 Set the source of mirror

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)# exit
Switch (config)# monitor session 1 source interface eth-0-1 both
```

step 4 Exit the configure mode

```
Switch(config)# end
```

step 5 Validation

```
Session   1
----------
Status              :  Valid
Type                :  Local Session
Source Ports       :
  Receive Only      :
  Transmit Only     :
  Both              : eth-0-1
Source VLANs        :
  Receive Only      :
  Transmit Only     :
  Both              :
Destination Port    : eth-0-2 eth-0-3
```


Configuring Remote Mirror

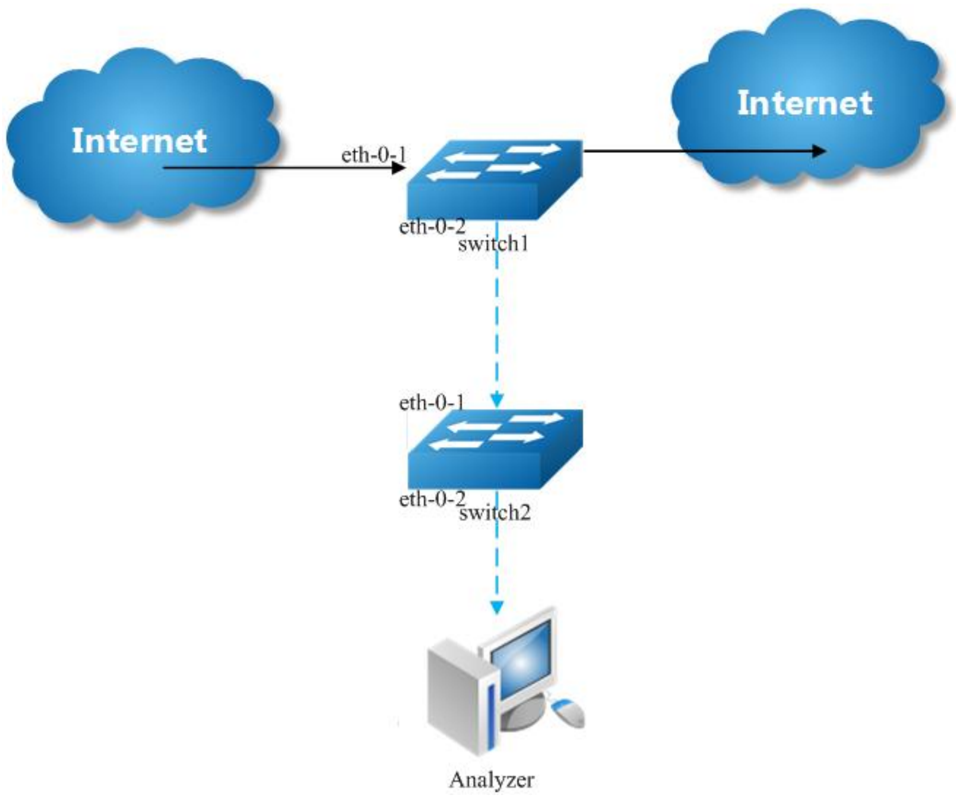

Figure 6-6 Remote Mirror

If local device cannot connect to an analyzer directly, User can choose remote mirror to send the copies of packets with specified vlan tag.

The remote device can pick out the packets with this vlan for analyze.

The following example copies the packets form Switch1's eth-0-1, and send them to Switch2 via Switch1's eth-0-2. Switch2 sends these packets to the analyzer.

The configuration of Switch1:

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set the destination of mirror

```
Switch1 (config)# vlan 15
Switch1 (config-vlan15)# exit

Switch(config)# interface eth-0-2
```

```
Switch(config-if-eth-0-2)# no shutdown
Switch(config-if-eth-0-2)# switchport mode trunk
Switch(config-if-eth-0-2)# switchport trunk allowed vlan add 15
Switch(config-if-eth-0-2)# exit

Switch(config)# monitor session 1 destination remote vlan 15 interface eth-0-2
```

step 3 Set the source of mirror

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)# exit
Switch(config)# monitor session 1 source interface eth-0-1 both
```

step 4 Exit the configure mode

```
Switch(config)# end
```

step 5 Validation

```
SwitchA# show monitor session 1
Session   1
----------
Status              : Valid
Type                 :  Remote Session
Source Ports        :
  Receive Only      :
  Transmit Only     :
  Both                : eth-0-1
Source VLANs        :
  Receive Only      :
  Transmit Only     :
  Both                :
Destination Port    : eth-0-2
Destination remote VLAN : 15
```

The configuration of Switch2:

Use these methods on Switch2 to send packets to analyzer via eth-0-2

method 1: use vlan 15 as mirror source， eth-0-2 as mirror destination

```
Switch# configure terminal
Switch(config)# vlan 15
Switch(config-vlan15)# exit

Switch(config)# interface vlan15
Switch(config-if)# exit

Switch(config)# interface eth-0-2
Switch(config-if-eth-0-2)# no shutdown
```



```
Switch(config-if-eth-0-2)# switchport mode access
Switch(config-if-eth-0-2)# switchport access vlan 15
Switch(config-if-eth-0-2)# exit

Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)# switchport mode trunk
Switch(config-if-eth-0-1)# switchport trunk allowed vlan add 15
Switch(config-if-eth-0-1)# exit
Switch(config)# monitor session 1 destination interface eth-0-2
Switch(config)# monitor session 1 source vlan 15 rx
Switch (config)# end
```

method 2: add both ports in to the same vlan (15), and make the packet flood in this vlan

```
Switch# configure terminal

Switch(config)# no spanning-tree enable

Switch(config)# vlan 15
Switch(config-vlan15)# exit

Switch(config)# interface eth-0-2
Switch(config-if-eth-0-2)# no shutdown
Switch(config-if-eth-0-2)# switchport mode access
Switch(config-if-eth-0-2)# switchport access vlan 15

Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)# switchport mode trunk
Switch(config-if-eth-0-1)# switchport trunk allowed vlan add 15
Switch(config-if-eth-0-1)# exit
```

NOTE In this configuration vlan tag is stripped because eth-0-2 is access port.

method 3: flood in vlan and keep vlan tag 15

If user needs to keep the vlan tag 15, eth-0-2 should be trunk port: (other configurations are same as method 2)

```
Switch(config)# interface eth-0-2
Switch(config-if-eth-0-2)# no shutdown
Switch(config-if-eth-0-2)# switchport mode trunk
Switch(config-if-eth-0-2)# switchport trunk allowed vlan add 15
```


Configuring CPU Mirror Dest

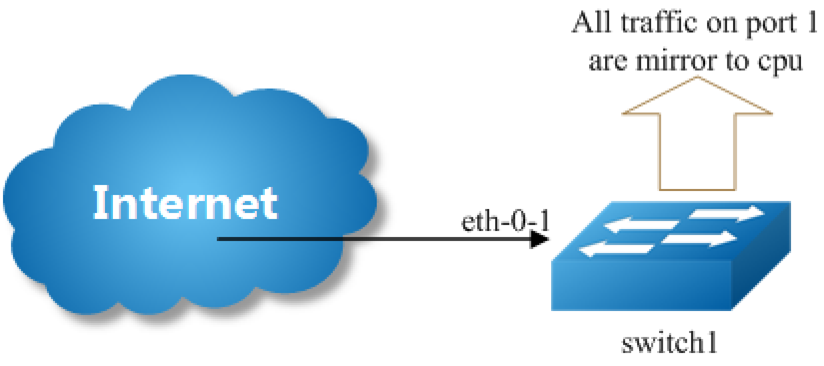

Figure 6-7 Mirror to cpu

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set the destination of mirror

```
Switch(config)# monitor session 1 destination cpu
Switch(config)# monitor cpu capture strategy replace
```

Set the buffer size and to cpu rate:

```
Switch(config)# monitor cpu set packet buffer 100
```

step 3 Set the source of mirror

```
Switch(config)# monitor session 1 source interface eth-0-1 both
```

step 4 Exit the configure mode

```
Switch(config)# end
```

Optional steps

Enable or disable to write the packets in to the flash files.

```
Switch# monitor cpu capture packet start
Switch# monitor cpu capture packet stop
```

Exchange the files from *.txt to *.pcap

```
Switch# pcap convert flash:/mirror/MirCpuPkt-2016-02-05-18-31-13.txt
flash:/MirCpuPkt-2016-02-05.pcap
```


Set the action after the packet buffer is exceeded: "drop" means discard the latest packet; "replace" means discard the oldest packet.

```
Switch(config)# monitor cpu capture strategy drop
Switch(config)# monitor cpu capture strategy replace
```

step 5 Validation

This example shows how to set up a mirror session, session 1, for monitoring source port traffic to a destination cpu. You can use show monitor session to see the configuration.

```
Switch# show monitor session 1
Session   1
----------
Status              : Valid
Type                : Cpu Session
Source Ports        :
  Receive Only      :
  Transmit Only     :
  Both              : eth-0-1
Source VLANs        :
  Receive Only      :
  Transmit Only     :
  Both              :
Destination Port    : cpu
```

This example shows how to display the mirror cpu packets

```
Switch# show monitor cpu packet all
-----------------show all mirror to cpu packet info-----------------
packet: 1
Source port: eth-0-1
MACDA:264e.ad52.d800, MACSA:0000.0000.1111
vlan tag:100
IPv4 Packet, IP Protocol is 0
IPDA:3.3.3.3, IPSA: 10.0.0.2
Data length: 47
Data:
 264e ad52 d800 0000 0000 1111 8100 0064
 0800 4500 001d 0001 0000 4000 6ad9 0a00
 0002 0303 0303 6365 6e74 6563 796f  75
```

This example shows how to display the mirror buffer size and the actions after the buffer is full:

```
Switch# show monitor cpu
Capture strategy : replace
Buffer size      : 0/100
```

This example shows how to display the files of the flash:


```
Switch# ls flash:/mirror
Directory of flash:/mirror

total 8
-rw-r-----  1 2287 Dec 23 01:16 MirCpuPkt-2016-12-23-01-15-54.txt
-rw-r-----  1 2568 Jan  3 11:41 MirCpuPkt-2017-01-03-11-41-33.txt
14.8T bytes total (7.9T bytes free)

Switch# more flash:/mirror/ MirCpuPkt-2017-01-03-11-41-33.txt
sequence  srcPort
1          eth-0-1
++++++++1483443444:648884
8c 1d cd 93 51 00 00 00 00 00 11 11 08 00 45 00
00 26 00 01 00 00 40 00 72 d0 01 01 01 01 03 03
03 03 63 65 6e 74 65 63 79 6f 75 63 65 6e 74 65
63 79 6f 75
--------
sequence  srcPort
2          eth-0-1
++++++++1483443445:546440
8c 1d cd 93 51 00 00 00 00 00 11 11 08 00 45 00
00 26 00 01 00 00 40 00 72 d0 01 01 01 01 03 03
03 03 63 65 6e 74 65 63 79 6f 75 63 65 6e 74 65
63 79 6f 75
```

This example shows how to display the files of the flash. *.pcap files can open with packets analyzer applications such as wireshark. Please referenc to the "ftp" and "tftp" part to download the files.

```
Switch#ls flash:/mirror
Directory of flash:/mirror

total 12
-rw-r-----  1 2287 Dec 23 01:16 MirCpuPkt-2016-12-23-01-15-54.txt
-rw-r-----  1 2568 Jan  3 11:41 MirCpuPkt-2017-01-03-11-41-33.txt
-rw-r--r--  1  704 Jan  3 13:07 test.pcap
14.8T bytes total (7.9T bytes free)
```

6.3.3 Application cases

N/A

6.4 Configuring Device Management

6.4.1 Overview

Function Introduction

User can manage the switch through the management port. The switch has two management ports: an Ethernet port and a console port.

Principle Description

N/A

6.4.2 Configuration

Configuring console port for management

The default console parameters of switch are:

- Baud rate default is 115200.
- Data bits default is 8.
- Stop bits default is 1.
- Parity settings default is none.

Before you can assign switch information, make sure you have connected a PC or terminal to the console port, and configured the PC or terminal software parameters to match the default console port parameters. After login in the switch, you can modify the console parameters.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enter line configuration mode and set the console speed

```
Switch(config)# line console 0
Switch(config-line)# speed 19200
```

step 3 Exit the configure mode

```
Switch(config-line)# end
```

step 4 Validation

After the above setting, console port parameter has been changed, and the PC or terminal can’t configure the switch by console port. You must update PC or terminal console speed from 115200 to 19200 to match the new console parameter and can continue configure the switch by console port.

Configuring out band Ethernet port for management

In order to manage device by out band Ethernet port, you should configure management ip address first by console port.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configure switch management address

IPv4 is supported, for example:

```
Switch(config)# management ip address 10.10.38.106/24
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

```
Switch# show management ip address
  Management IP address is:  10.10.38.106/24
  Gateway: 0.0.0.0

Switch# show management interface
Management Interface current state: UP
Description:
Link encap: Ethernet        HWaddr: 00:1E:08:0B:E6:C1
net addr: 10.10.39.104      Mask: 255.255.254.0
Bcast: 10.10.39.255         MTU: 1500
Speed: 1000Mb/s             Duplex: Full
Auto-negotiation: Enable

Received:            606800 Packets,        46870749 Bytes (44.6 MiB)
Transmitted:          46985 Packets,         4212579 Bytes (4.0 MiB)
```

Configuring Temperature

The switch supports temperature alarm management. You can configure three temperature thresholds: low, high and critical. When switch temperature is lower than low threshold or higher than higher threshold, the switch will be alarm. If the switch temperature is higher than critical threshold, the switch will cut off its power automatically.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configuring temperature threshold

5℃ for low, 70℃ for high, 85℃ for critical.

```
Switch(config)# temperature 5 70 85
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

```
Switch# show environment
Fan tray status:
Index      Status
1          PRESENT
FanIndex Status SpeedRate Mode
--------+------+---------+-----
1-1      OK      40%      Auto
1-2      OK      40%      Auto
1-3      OK      40%      Auto
1-4      OK      40%      Auto
---------------------------------------------------------
Power status:
Index Status    Power   Type    Alert
-----+---------+-------+-------+-------
1     PRESENT   FAIL    -       ALERT
2     PRESENT   OK      AC      NO
---------------------------------------------------------
Sensor status (Degree Centigrade):
Index Temperature Lower_alarm Upper_alarm Critical_limit Position
-----+-----------+-----------+-----------+--------------+----------
1     37          5           70          85             AROUND_CHIP
2     39          5           70          85             AROUND_CHIP
3     30          5           70          85             AROUND_FAN
4     34          5           70          85             AROUND_CPU
5     58          -10         100         110            SWITCH_CHIP
Switch# show environment
---------------------------------------------------------
Sensor status (Degree Centigrade):
Index Temperature  Lower_alarm  Upper_alarm Critical_limit
1     50           5            70          90
```


Configuring Fan

The switch supports to manage fan automatically. If the fan is fail or the fan tray is absent, the switch will be alarm. And if the fan tray supports speed-adjust, the switch can adjust the fan speed depending on the real-time temperature. The switch has three temperature thresholds: Tlow=50, Thigh=65 and Tcrit=80 Celsius scales. If Temperature<Tlow, the fan will stall; if Tlow<=Temperature<Thigh, the fan will run on 30% speed rate; if Thigh<=Temperature<Tcrit, the fan will run on 70% speed rate; if Tcrit<=Temperature, the fan will run on 100% speed rate. And there has a temperature hysteresis Thyst=2 Celsius scales. Assuming temperature has previously crossed above Tlow, Thigh or Tcrit, then the temperature must drop below the points corresponding Thyst(Tlow-Thyst, Thigh-Thyst or Tcrit-Thyst) in order for the condition to drive fan speed rate to lower level. For example:

- temperature is 58 Celsius scales, the fan speed rate is 30%, (Tlow<58<Thigh)
- temperature increases to 65 Celsius scales, the fan speed rate is 70%,(Thigh=65)
- temperature decreases to 63 Celsius scales, the fan speed rate is still 70%,(Thigh-Thyst =63)
- temperature decreases to 62 Celsius scales, the fan speed rate is 30%,(62<Thigh-Thyst)

The Tlow, Thigh, Tcrit, Thyst and fan speed rate for each temperature threshold are hard code, and couldn't be modified.

```
Switch# show environment
Fan tray status:
Index      Status      SpeedRate   Mode
----------+------------+------------+------------
1-1        OK          40%         AUTO
1-2        OK          40%         AUTO
1-3        OK          40%         AUTO
1-4        OK          40%         AUTO

Power status:
Index      Status      Power       Type        Alert
----------+------------+------------+------------+------------
1          PRESENT     OK          AC          NO
2          PRESENT     FAIL        AC          ALERT

Sensor status (Degree Centigrade):
Index      Temperature Lower_alarm Upper_alarm Critical    Position
----------+------------+------------+------------+------------+------------
1          41          5           65          80          AROUND_CHIP
2          43          5           65          80          AROUND_CHIP
3          32          5           65          80          AROUND_FAN
```



```
4          37          5          65          80          AROUND_CPU
5          61          -10        100         110         SWITCH_CHIP
```

Configuring Power

The switch supports to manage power status automatically. If the power is failed or the fan in power is failed, the switch will be alarm. If power is removed or inserted, the switch will notice user also.

User can show the power status to verify the power status.

```
Switch# show environment
Fan tray status:
Index      Status      SpeedRate   Mode
----------+------------+------------+------------
1-1        OK          40%         AUTO
1-2        OK          40%         AUTO
1-3        OK          40%         AUTO
1-4        OK          40%         AUTO

Power status:
Index      Status      Power       Type        Alert
----------+------------+------------+------------+------------
1          PRESENT     OK          AC          NO
2          PRESENT     FAIL        AC          ALERT

Sensor status (Degree Centigrade):
Index      Temperature  Lower_alarm  Upper_alarm  Critical    Position
----------+------------+------------+------------+------------+------------
1          41          5           65          80          AROUND_CHIP
2          43          5           65          80          AROUND_CHIP
3          32          5           65          80          AROUND_FAN
4          37          5           65          80          AROUND_CPU
5          61          -10         100         110         SWITCH_CHIP
```

Configuring Transceiver

The switch supports manage the transceiver information, and the transceiver information includes basic information and diagnostic information. The basic information includes transceiver type, vendor name, PN, S/N, wavelength and link length for supported type. The diagnostic information includes real-time temperature, voltage, current, optical transmit power, optical receive power and the threshold about these parameters. If the transceiver is inserted or removed, the real-time parameter is out of threshold, the switch will notice the users.

User can show the transceiver information to verify this function.


```
Switch# show transceiver

Port eth-0-49 transceiver info:
Transceiver Type: 1000BASE-SX
 Transceiver Vendor Name : FINISAR CORP.
 Transceiver PN          : FTLF8519P2BTL
 Transceiver S/N         : PLD3F3X
Transceiver Output Wavelength: 850 nm
Supported Link Type and Length:
      Link Length for 50/125um multi-mode fiber: 300 m
      Link Length for 62.5/125um multi-mode fiber: 150 m

Port eth-0-51 transceiver info:
Transceiver Type: 1000BASE-SX
 Transceiver Vendor Name : FINISAR CORP.
 Transceiver PN          : FTLF8519P2BTL
 Transceiver S/N         : PLD3F3X
Transceiver Output Wavelength: 850 nm
Supported Link Type and Length:
      Link Length for 50/125um multi-mode fiber: 300 m
      Link Length for 62.5/125um multi-mode fiber: 150 m
```

Upgrade bootrom

The switch supports to upgrade the bootrom image when system is running. And after upgrading, you must reboot the switch to take effect.

step 1 Copy bootrom image file to the flash

```
Switch# copy mgmt-if tftp://10.10.38.160/bootrom.bin flash:/boot/
```

step 2 Enter the configure mode

```
Switch# configure terminal
```

step 3 Upgrade the bootrom

```
Switch(config)# update bootrom flash:/boot/bootrom.bin
```

step 4 Exit the configure mode

```
Switch(config)# end
```

step 5 Reboot the system

```
Switch# reboot
```


step 6 Validation

After the above setting, you can show uboot version information of platform.

Before upgrade:

```
Switch# show version
CNOS Software, E580, Version 2.0.8
Copyright (C) 2004-2016 Centec Networks Inc. All rights reserved.
The current running image is: flash:/boot/CNOS-e580-v2.0.8.r.bin

Switch uptime is 1 days, 18 hours, 32 minutes
Hardware Type        : 20Q4Z
SDRAM size           : 1024M
Flash size           : 2048M
Hardware Version     : 1.0
EPLD Version         : 2.1
BootRom Version      : 8.0.1
System serial number : E130GD151005
```

After upgrade:

```
Switch# show version
CNOS Software, E580, Version 2.0.8
Copyright (C) 2004-2016 Centec Networks Inc. All rights reserved.
The current running image is: flash:/boot/CNOS-e580-v2.0.8.r.bin

Switch uptime is 0 days, 0 hours, 3 minutes
Hardware Type        : 20Q4Z
SDRAM size           : 1024M
Flash size           : 2048M
Hardware Version     : 1.0
EPLD Version         : 2.1
BootRom Version      : 8.1.1
System serial number : E130GD151005
```

6.4.3 Application cases

N/A

6.5 Configuring Bootrom

6.5.1 Overview

Function Introduction

The main function of Bootrom is to initialize the board simply and load the system image to boot. You can use some necessary commands in bootrom mode.

Bootrom can load the system image both from TFTP server and persistent storage like flash. Then you can configure the Switch and TFTP server IP address as environment variables in Bootrom mode for boot the system image.

Principle Description

N/A

6.5.2 Configuration

Configuring Boot from TFTP Server

Method 1: Boot the system from TFTP server

Save the configuration and reboot the system:

```
bootrom:> setenv bootcmd boot_tftp OS-ms-v3.1.9.it.r.bin
bootrom:> saveenv
bootrom:> reset
```

Method 2: Method 1:Boot the system from TFTP server without password

Save the configuration and reboot the system:

```
bootrom:> setenv bootcmd boot_tftp_nopass OS-ms-v3.1.9.it.r.bin
bootrom:> saveenv
bootrom:> reset
```

Method 3: Boot the system from TFTP server and reboot automatically

```
bootrom:> boot_tftp OS-ms-v3.1.9.it.r.bin
```

Method 4: Boot the system from TFTP server and reboot automatically without password

```
bootrom:> boot_tftp_nopass OS-ms-v3.1.9.it.r.bin
```

Validation

After the above setting, you can get show information:

```
bootrom:> reset
………………..
TFTP from server 10.10.29.160; our IP address is 10.10.29.118
Filename 'OS-ms-v3.1.9.it.r.bin'.
```



```
Load address: 0xaa00000
Loading: octeth0: Up 100 Mbps Full duplex (port  0)
#################################################################
         #####################
done
Bytes transferred = 12314539 (bbe7ab hex), 1829 Kbytes/sec
```

Configuring Boot from FLASH

Boot the system from FLASH

Save the configuration and reboot the system:

```
bootrom:> setenv bootcmd boot_flash OS-ms-v3.1.9.it.r.bin
bootrom:> saveenv
bootrom:> reset
```

Boot the system from without password

Save the configuration and reboot the system:

```
bootrom:> setenv bootcmd boot_flash_nopass OS-ms-v3.1.9.it.r.bin
bootrom:> saveenv
bootrom:> reset
Do you want to revert to the default config file ? [Y|N|E]:Y
```

Boot the system from FLASH and reboot automatically

```
bootrom:> boot_flash OS-ms-v3.1.9.it.r.bin
```

Boot the system from FLASH and reboot automatically without password

```
bootrom:> boot_flash_nopass OS-ms-v3.1.9.it.r.bin
Do you want to  revert  to the default config file ? [Y|N|E]:Y
```

Validation

After the above setting, you can get show information:

```
bootrom:> reset
……
Do you want to  revert  to the default config file ? [Y|N|E]:Y
### JFFS2 loading '/boot/OS-ms-v3.1.9.it.r.bin' to 0xaa00000
Scanning JFFS2 FS: . done.
### JFFS2 load complete: 12314539 bytes loaded to 0xaa00000
## Booting image at 0aa00000 ...
   Verifying Checksum ... OK
   Uncompressing Kernel Image ... OK
……
```


Set boot IP

step 1 Set Switch IP address , details information as follows

```
bootrom:> setenv ipaddr 10.10.29.101
bootrom:> saveenv
```

step 2 Set TFTP server IP address , details information as follows

```
bootrom:> setenv ipserver 10.10.29.160
bootrom:> saveenv
```

step 3 validation

After the above setting, you can get show information:

```
bootrom:> printenv
printenv
bootdelay=5
baudrate=9600
download_baudrate=9600
…………………
stderr=serial
ipaddr=10.10.29.101
ipserver=10.10.29.160
Environment size: 856/2044 bytes
```

Upgrade bootrom

step 1 upgrade the Bootrom image from TFTP server

```
bootrom:> upgrade_uboot bootrom.bin
```

step 2 validation

After the above setting, you can get show information:

```
bootrom:> version
version
Bootrom 3.0.3 (Development build) (Build time: Aug  4 2011 - 11:47:06)
```


Set gateway IP

step 1 Set Switch gateway IP address , details information as follows

```
bootrom:> setenv gatewayip 10.10.38.1
bootrom:> saveenv
```

step 2 Set network mask , details information as follows

```
bootrom:> setenv netmask 255.255.255.0
bootrom:> saveenv
```

step 3 validation

After the above setting, you can get show information:

```
bootrom:> printenv
printenv
bootdelay=5
baudrate=9600
download_baudrate=9600
…………………
stderr=serial
gatewayip=10.10.38.1
netmask=255.255.255.0
Environment size: 856/2044 bytes
```

6.5.3 Application cases

N/A

7 Network Management Configuration Guide

7.1 Configuring Network Diagnosis

7.1.1 Overview

Function Introduction

Ping is a computer network administration utility used to test the reachability of a host on an Internet Protocol (IP) network and to measure the round-trip time for messages sent from the originating host to a destination computer. The name comes from active sonar terminology.

Ping operates by sending Internet Control Message Protocol (ICMP) echo request packets to the target host and waiting for an ICMP response. In the process it measures the time from transmission to reception (round-trip time) and records any packet loss. The results of the test are printed in form of a statistical summary of the response packets received, including the minimum, maximum, and the mean round-trip times, and sometimes the standard deviation of the mean.

Traceroute is a computer network tool for measuring the route path and transit times of packets across an Internet Protocol (IP) network.

Traceroute sends a sequence of Internet Control Message Protocol (ICMP) packets addressed to a destination host. Tracing the intermediate routers traversed involves control of the time-to-live (TTL) Internet Protocol parameter. Routers decrement this parameter and discard a packet when the TTL value has reached zero, returning an ICMP error message (ICMP Time Exceeded) to the sender.

Principle Description

N/A

7.1.2 Configuration

Ping IP with in-band port

```
Switch# ping 10.10.29.247
```

Ping IP with management port

```
Switch# ping mgmt-if 10.10.29.247
```

Traceroute IP with inner port

```
Switch# traceroute 1.1.1.2
```

Traceroute IP with management port

```
Switch# traceroute mgmt-if 10.10.27.223
```

7.1.3 Application cases

Example for Ping

```
Switch# ping mgmt-if 10.10.27.223
PING 10.10.27.223 (10.10.27.223) 56(84) bytes of data.
64 bytes from 10.10.27.223: icmp_seq=1 ttl=63 time=1.14 ms
64 bytes from 10.10.27.223: icmp_seq=2 ttl=63 time=0.192 ms
64 bytes from 10.10.27.223: icmp_seq=3 ttl=63 time=0.198 ms
64 bytes from 10.10.27.223: icmp_seq=4 ttl=63 time=0.208 ms
64 bytes from 10.10.27.223: icmp_seq=5 ttl=63 time=0.281 ms

--- 10.10.27.223 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 3999ms
rtt min/avg/max/mdev = 0.192/0.404/1.145/0.372 ms
```

Example for traceroute

```
Switch# traceroute 10.10.29.247
traceroute to 10.10.29.247 (10.10.29.247), 30 hops max, 38 byte packets
 1  10.10.29.247 (10.10.29.247)  0.036 ms  0.034 ms  0.018 ms
Switch#
```


7.2 Configuring NTP

7.2.1 Overview

Function Introduction

NTP is a tiered time distribution system with redundancy capability. NTP measures delays within the network and within the algorithms on the machine on which it is running.

Using these tools and techniques, it is able to synchronize clocks to within milliseconds of each other when connected on a Local Area Network and within hundreds of milliseconds of each other when connected to a Wide Area Network.

The tiered nature of the NTP time distribution tree enables a user to choose the accuracy needed by selecting a level (stratum) within the tree for machine placement.

A time server placed higher in the tree (lower stratum number), provides a higher likelihood of agreement with the UTC time standard.

Some of the hosts act as time servers, that is, they provide what they believe is the correct time to other hosts. Other hosts act as clients, that is, they find out what time it is by querying a time server.

Some hosts act as both clients and time servers, because these hosts are links in a chain over which the correct time is forwarded from one host to the next.

As part of this chain, a host acts first as a client to get the correct time from another host that is a time server.

It then turns around and functions as a time server when other hosts, acting as clients, send requests to it for the correct time.

Principle Description

N/A

7.2.2 Configuration

Configuring Client/Server mode

Before configuring NTP client, make sure that NTP service is enabled on Server.

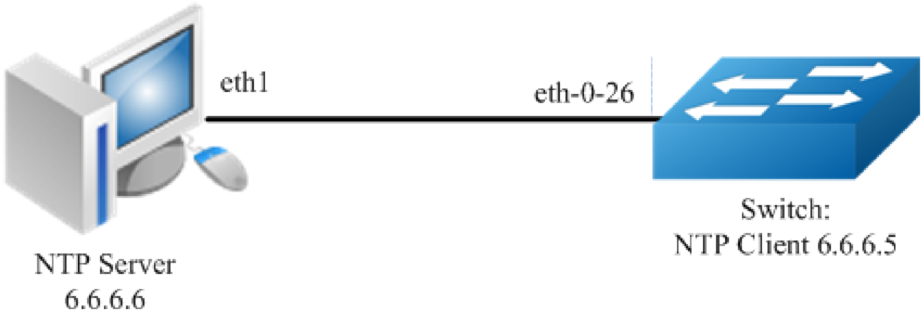

Figure 7-1 NTP

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create a vlan

```
Switch(config)# vlan 10
Switch(config-vlan10)# exit
```

step 3 Enter the interface configure mode and join the vlan

```
Switch(config)# interface eth-0-26
Switch(config-if-eth-0-10)# switchport access vlan 10
Switch(config-if-eth-0-10)# no shutdown
Switch(config-if-eth-0-10)# exit
```

step 4 create a vlan interface and set the IP address

```
Switch(config)# interface vlan10
Switch(config-if-vlan10)# ip address 6.6.6.5/24
Switch(config-if-vlan10)# exit
```

step 5 Set the attributes of NTP client

```
Switch(config)# ntp server 6.6.6.6
```


step 6 Exit the configure mode

```
Switch(config)# end
```

step 7 Validation

```
Switch# show ntp associations
Current NTP associations:
remote              refid      st   poll  reach  delay    offset   disp
======================================================================
*6.6.6.6         127.127.1.0    6    128   37     0.778    -0.234   71.945
* synchronized, + candidate, # selected, x falsetick, . excess, - outlier
```

Configuring symmetric mode

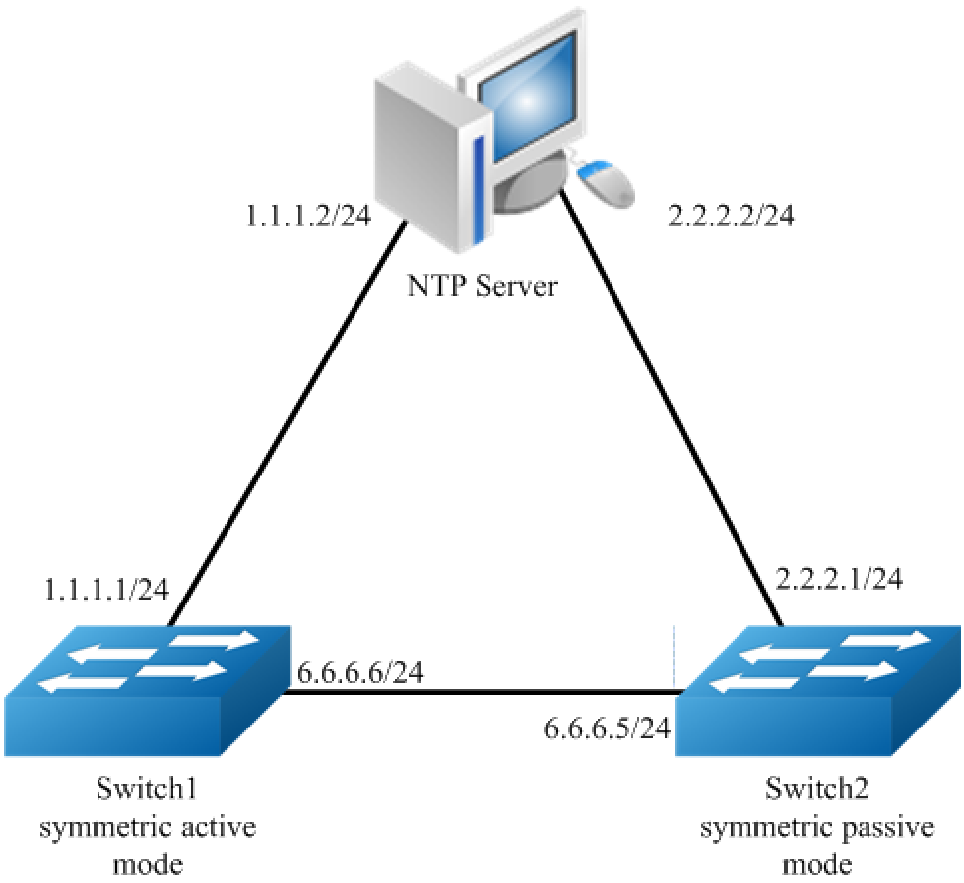

Figure 7-2 NTP symmetric mode

step 1 Enter the configure mode

```
Switch# configure terminal
```


step 2 Create a vlan

```
Switch(config)# vlan 10
Switch(config-vlan)# exit
Switch(config)# vlan 20
Switch(config-vlan20)# exit
```

step 3 Enter the interface configure mode and join the vlan

```
Switch(config)# interface eth-0-10
Switch(config-if-eth-0-10)# switch access vlan 10
Switch(config-if-eth-0-10)# no shutdown
Switch(config-if-eth-0-10)# exit
Switch(config)# interface eth-0-20
Switch(config-if-eth-0-20)# switch access vlan 20
Switch(config-if-eth-0-20)# no shutdown
Switch(config-if-eth-0-20)# exit
```

step 4 create a vlan interface and set the IP address

Switch1:

```
Switch(config)# interface vlan10
Switch(config-if-vlan10)# ip address 1.1.1.1/24
Switch(config-if-vlan10)# exit
Switch(config)# interface vlan20
Switch(config-if-vlan20)# ip address 6.6.6.6/24
Switch(config-if-vlan20)# exit
```

Switch2:

```
Switch(config)# interface vlan10
Switch(config-if-vlan10)# ip address 2.2.2.1/24
Switch(config-if-vlan10)# exit
Switch(config)# interface vlan20
Switch(config-if-vlan20)# ip address 6.6.6.5/24
Switch(config-if-vlan20)# exit
```

step 5 Set the attributes of NTP client

Switch1:

Set the active mode of symmetric.

```
Switch(config)# ntp server 1.1.1.2
```

Switch2:

```
Switch(config)# ntp server 2.2.2.2
```


Configuring Client/Server mode with authentication

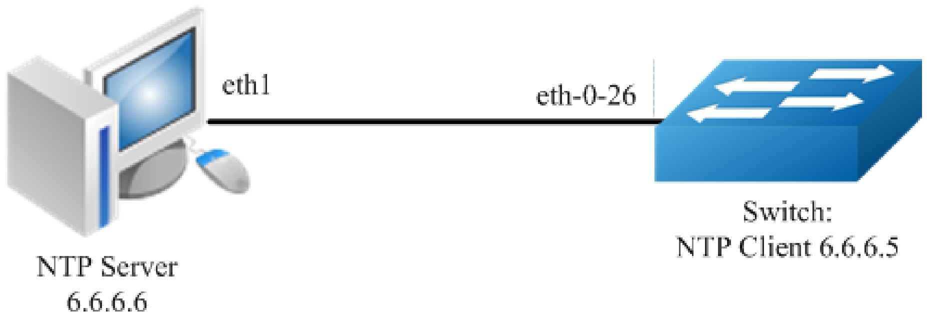

Figure 7-3 NTP

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create a vlan

```
Switch(config)# vlan 10
Switch(config-vlan10)# exit
```

step 3 Enter the interface configure mode and join the vlan

```
Switch(config)# interface eth-0-26
Switch(config-if-eth-0-10)# switch access vlan 10
Switch(config-if-eth-0-10))# no shutdown
Switch(config-if)# exit
```

step 4 create a vlan interface and set the IP address

```
Switch(config)# interface vlan10
Switch(config-if-vlan10)# ip address 6.6.6.5/24
Switch(config-if-vlan10)# exit
```

step 5 Set the attributes of NTP client

```
Switch(config)# ntp authentication enable
Switch(config)# ntp key 1 serverkey
Switch(config)# ntp trustedkey 1
Switch(config)# ntp server 6.6.6.6 key 1
```

step 6 Exit the configure mode

```
Switch(config)# end
```


step 7 Validation

```
Switch# show ntp associations
Current NTP associations:
remote              refid      st   poll  reach  delay    offset   disp
======================================================================
*6.6.6.6         127.127.1.0    6   128    37     0.778    -0.234   71.945
* synchronized, + candidate, # selected, x falsetick, . excess, - outlier
```

Configuring Client/Server mode with two servers

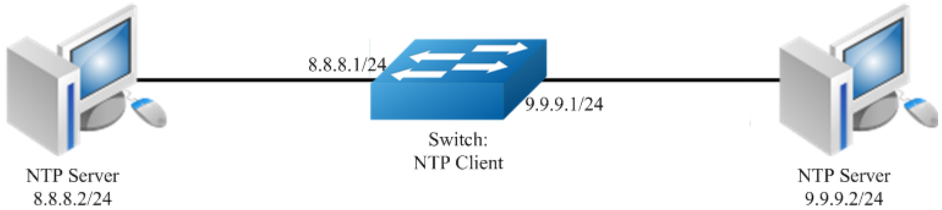

Figure 7-4 NTP

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create a vlan

```
Switch(config)# vlan 10
Switch(config-vlan10)# exit
Switch(config)# vlan 20
Switch(config-vlan20)# exit
```

step 3 Enter the interface configure mode and join the vlan

```
Switch(config)# interface eth-0-10
Switch(config-if-eth-0-10)# switch access vlan 10
Switch(config-if-eth-0-10)# no shutdown
Switch(config-if-eth-0-10)# exit
Switch(config)# interface eth-0-20
Switch(config-if-eth-0-20)# switch access vlan 20
Switch(config-if-eth-0-20)# no shutdown
Switch(config-if-eth-0-20)# exit
```

step 4 create a vlan interface and set the IP address

```
Switch(config)# interface vlan10
Switch(config-if-vlan10)# ip address 8.8.8.1/24
Switch(config-if-vlan10)# exit
Switch(config)# interface vlan20
```



```
Switch(config-if-vlan20)# ip address 9.9.9.1/24
Switch(config-if-vlan20)# exit
```

step 5 Set the attributes of NTP client

```
Switch(config)# ntp server 8.8.8.2
Switch(config)# ntp server 9.9.9.2 prefer
```

step 6 Exit the configure mode

```
Switch(config)# end
```

step 7 Validation

```
Switch# show ntp associations
Current NTP associations:
remote              refid       st   poll  reach  delay    offset   disp
======================================================================
 8.8.8.2        127.127.1.0    6   128    37     0.778    -0.234   71.945
*9.9.9.2        127.127.1.0    6   128    37     0.765    -0.211   69.446
* synchronized, + candidate, # selected, x falsetick, . excess, - outlier
```

Configuring NTP Server (Use the ntpd of linux system for example）

Step 1 Display eth1 ip address

```
[root@localhost octeon]# ifconfig eth1
eth1      Link encap:Ethernet  HWaddr 00:08:C7:89:4B:AA
          inet addr:6.6.6.6  Bcast:6.6.6.255  Mask:255.255.255.0
          inet6 addr: fe80::208:c7ff:fe89:4baa/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3453 errors:1 dropped:0 overruns:0 frame:1
          TX packets:3459 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:368070 (359.4 KiB)  TX bytes:318042 (310.5 KiB)
```

Step 2 Check networks via Ping

```
[root@localhost octeon]# ping 6.6.6.5
PING 6.6.6.5 (6.6.6.5) 56(84) bytes of data.
64 bytes from 6.6.6.5: icmp_seq=0 ttl=64 time=0.951 ms
64 bytes from 6.6.6.5: icmp_seq=1 ttl=64 time=0.811 ms
64 bytes from 6.6.6.5: icmp_seq=2 ttl=64 time=0.790 ms
```

Step 3 Configure ntp.conf

```
[root@localhost octeon]# vi /etc/ntp.conf
server  127.127.1.0 # local clock
```



```
fudge   127.127.1.0 stratum 5
#
# Drift file.  Put this in a directory which the daemon can write to.
# No symbolic links allowed, either, since the daemon updates the file
# by creating a temporary in the same directory and then rename()'ing
# it to the file.
#
driftfile /var/lib/ntp/drift
broadcastdelay  0.008
broadcast 6.6.6.255
#
# PLEASE DO NOT USE THE DEFAULT VALUES HERE. Pick your own, or remote
# systems might be able to reset your clock at will. Note also that
# ntpd is started with a -A flag, disabling authentication, that
# will have to be removed as well.
#
#disable auth
keys        /etc/ntp/keys
trustedkey 1
```

Step 4 Configure keys

```
[root@localhost octeon]# vi /etc/ntp/keys
#
# PLEASE DO NOT USE THE DEFAULT VALUES HERE. Pick your own, or remote
# systems might be able to reset your clock at will. Note also that
# ntpd is started with a -A flag, disabling authentication, that
# will have to be removed as well.
#
1 M serverkey
```

Step 5 Start ntpd service

```
[root@localhost octeon]# ntpd
```

7.2.3 Application cases

N/A

7.3 Configuring SNMP

7.3.1 Overview

Function Introduction

SNMP is an application-layer protocol that provides a message format for communication between managers and agents. The SNMP system consists of an SNMP manager, an SNMP agent, and a MIB. The SNMP manager can be part of a network management system (NMS). The agent and MIB reside on the switch. To configure SNMP on the switch, you define the relationship between the manager and the agent. The SNMP agent contains MIB variables whose values the SNMP manager can request or change. A manager can get a value from an agent or store a value into the agent. The agent gathers data from the MIB, the repository for information about device parameters and network data. The agent can also respond to a manager's requests to get or set data. An agent can send unsolicited traps to the manager. Traps are messages alerting the SNMP manager to a condition on the network. Error user authentication, restarts, link status (up or down), MAC address tracking, closing of a Transmission Control Protocol (TCP) connection, loss of connection to a neighbor, or other significant events may send a trap.

Principle Description

SNMP module is based on the following RFC draft:

- SNMPv1: Defined in RFC 1157.
- SNMPv2C: Defined in RFC 1901.
- SNMPv3: Defined in RFC 2273 to 2275.

Following is a brief description of terms and concepts used to describe the SNMP protocol:

- **Agent**: A network-management software module, an agent has local knowledge of management information and translates that information into a form compatible with SNMP.
- **Management Information Base (MIB)**: Management Information Base, collection of information is organized hierarchically.
- **Engine ID**: A unique ID for a network's node.
- **Trap**: Used by managed devices to asynchronously report events to the NMS.

7.3.2 Configuration

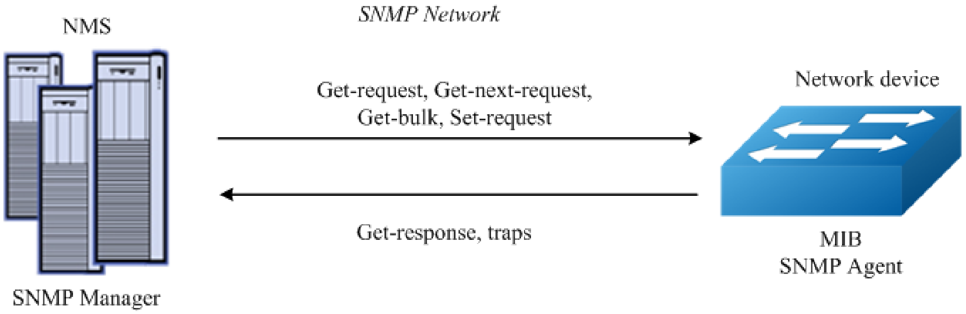

Figure 7-5 snmp

As shown in the figure SNMP agent gathers data from the MIB. The agent can send traps, or notification of certain events, to the SNMP manager, which receives and processes the traps. Traps alert the SNMP manager to a condition on the network such as improper user authentication, restarts, link status (up or down), MAC address tracking, and so forth. The SNMP agent also responds to MIB-related queries sent by the SNMP manager in get-request, get-next-request, and set-request format.

Enable SNMP

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enable SNMP globally

```
Switch(config)# snmp-server enable
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

```
Switch# show running-config
snmp-server enable
```


Configuring community string

You use the SNMP community string to define the relationship between the SNMP manager and the agent. The community string acts like a password to permit access to the agent on the switch. Optionally, you can specify one or more of these characteristics associated with the string:

- A MIB view, which defines the subset of all MIB objects accessible to the given community
- Read and write or read-only permission for the MIB objects accessible to the community

Beginning in privileged EXEC mode, follow these steps to configure a community string on the switch.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configuring community string

Configure a view named "DUT"(optional); Configure a community named "public" with read-write access and view "DUT".

```
Switch(config)# snmp-server view DUT included 1
Switch(config)# snmp-server community public read-write (view DUT)
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

```
Switch# show snmp-server community
Community-Access   Community-String     Security-name
------------------+--------------------+-------------------
read-only             public                   comm1
Switch# show running-config
Building configuration...
version v2.0.8
hostname DUT1
!
username admin privilege 4 password admin
!
!
```



```
snmp-server enable
!
snmp-server view DUT included .1
!
snmp-server community public read-only view DUT
!
management ip address 192.168.100.101/24
management route gateway 192.168.100.254
!
!
interface eth-0-1
!
interface eth-0-2
!
interface eth-0-3
!
interface eth-0-4
!
interface eth-0-5
!
interface eth-0-6
!
interface eth-0-7
!
interface eth-0-8
!
interface eth-0-9
!
interface eth-0-10
!
interface eth-0-11
!
interface eth-0-12
!
interface eth-0-13
!
interface eth-0-14
!
interface eth-0-15
!
interface eth-0-16
!
interface eth-0-17
!
interface eth-0-18
!
interface eth-0-19
!
interface eth-0-20
!
interface eth-0-21
!
interface eth-0-22
!
interface eth-0-23
```



```
!
interface eth-0-24
!
line console 0
 no line-password
 no login
line vty 0 7
 exec-timeout 0 0
 privilege level 4
 no line-password
 no login
```

Configuring SNMPv3 Groups, Users and Accesses

You can specify an identification name (engine ID) for the local SNMP server engine on the switch. You can configure an SNMP server group that maps SNMP users to SNMP views, you can add new users to the SNMP group, and you can add access for the SNMP group.

Beginning in privileged EXEC mode, follow these steps to configure SNMP on the switch.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set the global configurations for SNMP

Set engineID; Set the user name, password, and authentication type; Create SNMP group and set the authority for the group member.

```
Switch(config)# snmp-server engineID 8000123456
Switch(config)# snmp-server usm-user usr1 authentication md5 mypassword privacy des
yourpassword
Switch(config)# snmp-server group grp1 user usr1 security-model usm
Switch(config)# snmp-server access grp1 security-model usm noauth
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

```
Switch# show snmp-server usm-user usr1
User Name:     usr1
EnginedID:     8000123456
Auth Protocol: md5
```



```
Auth password: mypassword
Priv Protocol: des
Priv password: yourpassword
Storage Type:  nonvolatile
Row status:    active
DUT1# show snmp-server access grp1
Group name:     grp1
Context:
Security model: usm
Security level: noauth
Context Match:  exact
Read view:      _all_
Write view:     none
Notify view:    none
Storage Type:   permanent
Row status:     active
DUT1# show snmp-server engineID
Engine ID   : 8000123456
```

SNMPv1 and SNMPv2 notifications configure

Beginning in privileged EXEC mode, follow these steps to configure SNMP on the switch.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set the global configurations for SNMP

Enable all supported traps; Configure a remote trap manager which IP is "10.10.27.232";

```
DUT1(config)# snmp-server trap enable all
DUT1(config)# snmp-server trap target-address mgmt-if 10.10.27.232 community public
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

```
Switch#  show snmp-server trap-receiver
Target-ipaddress mgmt-if udpport version pdu-type community
----------------+-------+-------+-------+--------+-----------------------------
10.10.27.232     yes     162     v1      trap     public
10.10.27.232     yes     162     v2c     trap     public
```


Configuring SNMPv3 notifications

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set the global configurations for SNMP

Enable all supported traps; Configure a trap notify item for SNMPv3; Add a local user to SNMPv3 notifications. Configure a remote trap manager's IP address;

```
Switch(config)# snmp-server trap enable all
Switch(config)# snmp-server notify n1 tag t1 trap
Switch(config)# snmp-server target-params p1 user usr1 security-model v3 message-
processing v3 noauth
Switch(config)# snmp-server target-address t1 param p1 mgmt-if 10.10.27.232 taglist
t1
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

```
Switch# show snmp-server notify n1
Notify name:   n1
Notify tag:    t1
Notify type:   trap
Storage Type:  nonvolatile
Row status:    active
DUT1# show snmp-server target-params p1
Target parameter name:     p1
Message processing model:  v3
Security model:            v3
Security name:             usr1
Security level:            noauth
Storage Type:              nonvolatile
Row status:                active
DUT1# show snmp-server target-address t1
Targetaddr name:     t1
IP address:          10.10.27.232
Mgmt-If:             yse
UDP Port:            162
Timeout:             2
Retry count:         3
Tag List:            t1
Parameters:          p1
Storage Type:        nonvolatile
Row status:          active
```


7.3.3 Application cases

N/A

7.4 Configuring SFLOW

7.4.1 Overview

Function Introduction

sFlow is a technology for monitoring traffic in data networks containing switches and routers. In particular, it defines the sampling mechanisms implemented in a sFlow Agent for monitoring traffic, and the format of sample data used by the sFlow Agent when forwarding data to a central data collector.

The architecture and sampling techniques used in the sFlow monitoring system are designed to provide continuous site-wide (and network-wide) traffic monitoring for high speed switched and routed networks.

The sFlow Agent uses two forms of sampling: statistical packet-based sampling of switched flows, and time-based sampling of network interface statistics.

SFLOW Flow-sampling supported fields

- Raw packet Header：Intercepts all or part of the header of the original message.
- Ethernet Frame Data：Analyzes Ethernet header information.
- IPV4 Data：Analyzes IPV4 header information of messages.
- Extended Router Data：Records the routing and forwarding information of messages.
- Extended Switch Data：Records the vlan conversion of messages and the conversion of priorities.

SFLOW Counter-sampling supported fields

- Generic Interface Counters：Statistical interface traffic.
- Ethernet Interface Counters：Statistical Ethernet related traffic information.
- Processor Information：Statistical CPU usage and memory usage.

Principle Description

N/A

7.4.2 Configuration

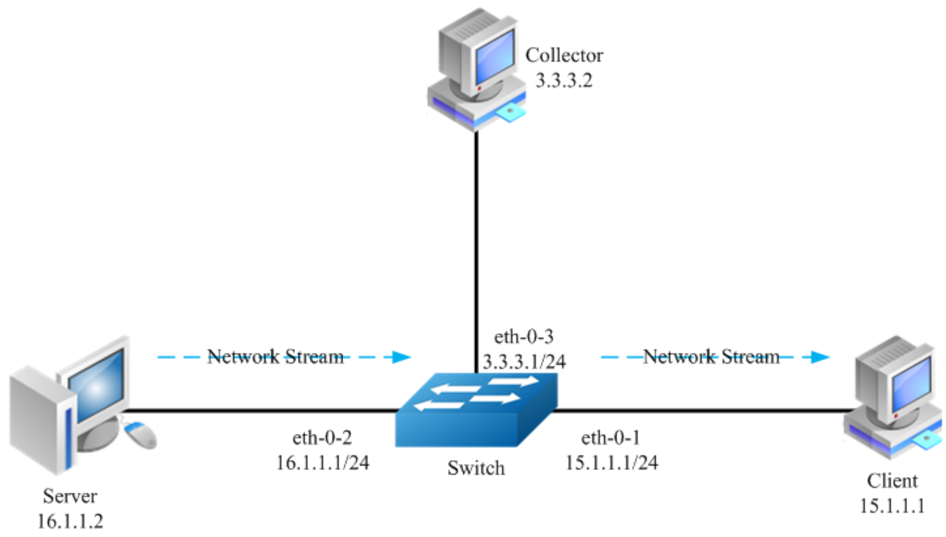

Figure 7-6 sflow

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enable sflow globally

```
Switch(config)# sflow enable
```

step 3 Set the global attribute for sflow

Set the agent IP address，set the collector IP address and udp port. If the udp port is not specified, it means default port 6343.

```
Switch(config)# sflow agent ip 3.3.3.1
Switch(config)# sflow collector 3.3.3.2 6342
```

NOTE At least one Agent and one collector must be configured for sflow.

Set the interval to send interface counter information (optional):

```
Switch(config)# sflow counter interval 15
```


step 4 Enter the interface configure mode and set the attributes of the interfaces

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no switchport
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)# ip address 15.1.1.1/24
Switch(config-if-eth-0-1)# exit

Switch(config)# interface eth-0-2
Switch(config-ifeth-0-2)#no switchport
Switch(config-ifeth-0-2)# no shutdown
Switch(config-ifeth-0-2)# ip address 16.1.1.1/24
Switch(config-ifeth-0-2)# exit

Switch(config)# interface eth-0-3
Switch(config-ifeth-0-2)# no switchport
Switch(config-ifeth-0-2)# no shutdown
Switch(config-ifeth-0-2)# ip address 3.1.1.1/24
Switch(config-ifeth-0-2)# exit
```

step 5 Enable sflow for input packets on eth-0-1

```
Switch(config)# interface eth-0-1
Switch(config-if)# sflow flow-sampling rate 8192
Switch(config-if)# sflow flow-sampling enable input
Switch(config-if)# sflow counter-sampling enable
Switch(config-if)# exit
```

step 6 Validation

To display the sflow configuration, use following command:

```
Switch# show sflow
sFlow Version: 5
sFlow Global Information:
Agent IPv4 address : 3.3.3.1
Counter Sampling Interval : 15 seconds
Collector 1:
IPv4 Address: 3.3.3.2
Port: 6342
sFlow Port Information:
Flow-Sample Flow-Sample
Port Counter Flow Direction Rate
--------------------------------------------------------
eth-0-1 Enable Enable Input 8192
```

7.4.3 Application cases

N/A

8 Traffic Managemant Configuration Guide

8.1 QoS Configuration Guidance

8.1.1 Overview

Function Introduction

Quality of Service (QoS) can be used to give certain traffic priority over other traffic. Without QoS, all traffic in a network has the same priority and chance of being delivered on time. If congestion occurs, all traffic has the same chance of being dropped. With QoS, specific network traffic can be prioritized to receive preferential treatment. In turn, a network performs more predictably, and utilizes bandwidth more effectively.

Classification information can be carried in the Layer-3 IP packet header or the Layer-2 frame. IP packet headers carry the information using 6 bits or 3 bits from the deprecated IP type of service (TOS) field. Layer-2 802.1Q frames carry the information using a 2-byte Tag Control Information field.

All switches and routers accessing the Internet depend on class information to give the same forwarding treatment to packets with the same class information, and give different treatment to packets with different class information. A packet can be assigned class information, as follows:

- End hosts or switches along a path, based on a configured policy
- Detailed packet examination, expected to occur nearer to the network edge, to prevent overloading core switches and routers
- A combination of the above two techniques

Class information can be used by switches and routers along a path to limit the amount of allotted resources per traffic class.

Per-hop behavior is an individual device's behavior when handling traffic in the DiffServ architecture. An end-to-end QoS solution can be created if all devices along a path have consistent per-hop behavior.

Principle Description

Terminology:

- SRTCM — Single Rate Three Color Marker
- TRTCM — Two Rate Three Color Marker
- CIR — Committed Information Rate
- CBS — Committed Burst Size
- EBS — Excess Burst Size
- PIR — Peak Information Rate

Following is a brief description of terms and concepts used to describe QoS:

CoS Value

Class of Service (CoS) is a 3-bit value used to classify the priority of Layer-2 frames upon entry into a network.

QoS classifies frames by assigning priority-indexed CoS values to them, and gives preference to higher-priority traffic.

Layer-2 802.1Q frame headers have a 2-byte Tag Control Information field that carries the CoS values in the 3 most significant bits, called the User Priority bits. On interfaces configured as Layer-2 802.1Q trunks, all traffic is in 802.1Q frames, except for traffic in the native VLAN.

Other frame types cannot carry Layer-2 CoS values. CoS values range from 0 to 7 and 7 is the highest priority.

DSCP Value

Differentiated Services Code Point (DSCP) is a 6-bit value used to classify the priority of Layer-3 packets upon entry into a network.

DSCP values range from 0 to 63, 63 being the highest priority, 0 being best-effort traffic.

EXP Value

The EXP field is located at the 20-22 bit of the MPLS tag,used to distinguish the priority of packets in MPLS networks. EXP values range from 0 to 7 and 7 is the highest priority.

Shaping

Shaping is to change the rate of incoming traffic flow to regulate the rate in such a way that the outgoing traffic flow behaves more smoothly. If the incoming traffic is highly bursty, it needs to be buffered so that the output of the buffer is less bursty and smoother.

Shaping has the following attributes:

- Shaping can be deployed base on physical port.
- Shaping can be deployed on queues of egress interface.

Policing

Policing determines whether a packet is in or out of profile by comparing the internal priority to the configured policer.

The policer limits the bandwidth consumed by a traffic flow. The result is given to the marker.

There are two types of policers:

- Port policer: Limits the bandwith of stream in port layer.
- Flow policer: Limits the bandwith of specified stream matched by acl.

Marking

Marking determines how to handle a packet when it is out of profile. It assesses the policer and the configuration information to determine the action required for the packet, and then handles the packet using one of the following methods:

- Let the packet through and mark color down

- Drop the packet

Marking can occur on ingress and egress interfaces.

Queuing

Queuing maps packets to a queue. Each egress port can accommodate up to 8 unicast queues, 4 multicast queues and 1 SPAN queue.

The packet internal priority can be mapped to one of the egress queues. The unit of queue depth is buffer cell. Buffer cell is the granularity, which is 288 bytes(Hybrid350 is 256 bytes), for packet storing.

After the packets are mapped to a queue, they are scheduled.

Tail Drop

Tail drop is the default congestion-avoidance technique on the interface. With tail drop, packets are queued until the thresholds are exceeded. The packets with color red are assigned to the first threshold for drop precedence 0, yellow are assigned to the second threshold for drop precedence 1, and green are assigned to the third threshold for drop precedence 2. You can modify the three tail-drop threshold to every egress queue by using the queue threshold interface configuration command. Each threshold value is packet number, which ranges from 0 to 12286.

WRED （Weighted Random Early Detection）

Weighted Random Early Detection (WRED) differs from other congestion-avoidance techniques because it attempts to anticipate and avoid congestion, rather than controlling congestion when it occurs.

WRED reduces the chances of tail drop by selectively dropping packets when the output interface begins to show signs of congestion. By dropping some packets early rather than waiting until the queue is full, WRED avoids dropping large numbers of packets at once. Thus, WRED allows the transmission line to be fully used at all times. WRED also drops more packets from large users than small. Therefore, sources that generate the most traffic are more likely to be slowed down versus sources that generate little traffic.

You can enable WRED and configure the two thresholds for a drop-precedence assigned to every egress queues. The WRED's color drop precedence map is the same as tail-drop's. Each min-threshold represents where WRED starts to randomly drop packets. After min-threshold is exceeded, WRED randomly begins to drop packets assigned to this threshold. As the queue max-threshold is approached, WRED continues to drop packets randomly with the rate of drop-probability. When the max-threshold is reached, WRED drops all packets assigned to the threshold. By default, WRED is disabled.

Scheduling

Scheduling forwards conditions packets using combination of WDRR and SP. Every queue belongs to a class. The class range from 0 to 7, and 7 is the highest priority. Several queues can be in a same class, or non queue in some class. Packets are scheduled by SP between classes and WDRR between queues in a class.

- Strict Priority-Based (SP), in which any high-priority packets are first transmitted. Lower-priority packets are transmitted only when the higher-priority queues are empty. A problem may occur when too many lower-priority packets are not transmitted.
- Weighted Deficit Round Robin (WDRR), in which each queue is assigned a weight to control the number of packets relatively sent from each queue.

Mapping Tables

During QoS processing, the switch represents the priority of all traffic (including non-IP traffic) with an internal priority value:

- During classification, QoS uses configurable mapping tables to derive the internal priority (a 6-bit value) from received CoS, EXP(3-bit), DSCP or IP precedence (3-bit) values. These maps include the CoS-to-priority-color/COS-to-PHB map, EXP-to-priority-color/EXP-to-PHB map, DSCP-to-priority-color/DSCP-to-PHB map and the IP-precedence-to- priority-color/IP-PREC-to-PHB map.
- During policing, QoS can assign another priority and color to an IP or non-IP packet (if the packet matches the class-map). This configurable map is called the policed-priority-color map.

- Before the traffic reaches the scheduling stage, and replace CoS or DSCP is set, QoS uses the configurable priority-color-to-CoS or priority-color-to-DSCP map to derive a CoS or DSCP value from the internal priority color.
- Each QoS domain has an independent set of map tables mentioned above.

The following provides information to consider before configuring QoS:

- QoS policing cannot be configured on Linkagg interface.
- All of QoS command cannot be done at the switch virtual interface level.

Enable QoS

QoS is enabled by default. QoS commands cannot be configured before QoS is enabled.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enter the QoS configure mode and enable QoS globally

```
Switch(config)# qos global
Switch(config-qos-global)# qos enable
```

step 3 Exit the configure mode

```
Switch(config)# end
```

Configure egress queue

Tail Drop Configurations

Tail drop is the default congestion-avoidance technique on every egress queue. With tail drop, packets are queued until the thresholds are exceeded.

The following example shows configuring tail drop threshold for queue 3. In this example, red-colored packet drop threshold is 2000, yellow-colored packet drop threshold is 3000, and green-colored packet drop threshold is 4000.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configuring WTD

```
Switch(config)# qos drop-profile p1
Switch(config-qos-drop-profile-p1)# green maximum 4000
Switch(config-qos-drop-profile-p1)# yellow maximum 3000
Switch(config-qos-drop-profile-p1)# red maximum 2000
Switch(config-qos-drop-profile-p1)# exit
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)# qos queue 3
Switch(config-if-eth-0-1-queue-3)# drop-profile p1
```

step 3 Exit the configure mode

```
Switch(config-if-eth-0-1-queue-3)# end
```

step 4 Validation

```
DUT1# show qos interface eth-0-1 queue drop
QUEUE DROP:
Queue Drop-mode      Color  Max_Tresh Min_Tresh Probability
-----+--------------+------+---------+---------+---------
0     tail-drop      green  600       -         -
                     yellow 600       -         -
                     red    600       -         -
1     tail-drop      green  600       -         -
                     yellow 600       -         -
                     red    600       -         -
2     tail-drop      green  600       -         -
                     yellow 600       -         -
                     red    600       -         -
3     tail-drop      green  4000      -         -
                     yellow 3000      -         -
                     red    2000      -         -
4     tail-drop      green  600       -         -
                     yellow 600       -         -
                     red    600       -         -
5     tail-drop      green  600       -         -
                     yellow 600       -         -
                     red    600       -         -
6     tail-drop      green  600       -         -
                     yellow 600       -         -
                     red    600       -         -
7     tail-drop      green  600       -         -
                     yellow 600       -         -
                     red    600       -         -
```


WRED Configurations

WRED reduces the chances of tail drop by selectively dropping packets when the output interface detects congestion.

By dropping some packets early rather than waiting until the queue is full, WRED avoids TCP synchronization dropping and thereafter improves the overall network throughput.

The min threshold is equal to max-threshold/8.

The following shows configuring WRED threshold for different color.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configuring WRED

```
Switch(config)# qos drop-profile p1
Switch(config-qos-drop-profile-p1)# green maximum 4000
Switch(config-qos-drop-profile-p1)# yellow maximum 3000
Switch(config-qos-drop-profile-p1)# red maximum 2000
Switch(config-qos-drop-profile-p1)# exit
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)# qos queue 3
Switch(config-if-eth-0-1-queue-3)# drop-profile p1
Switch(config-if-eth-0-1-queue-3)# random-detect enable
```

step 3 Exit the configure mode

```
Switch(config-if-eth-0-1-queue-3)# end
```

step 4 Validation

```
DUT1# show qos interface eth-0-1 queue drop
QUEUE DROP:
Queue Drop-mode      Color  Max_Tresh Min_Tresh Probability
-----+--------------+------+---------+---------+---------
0     tail-drop      green  600       -         -
                     yellow 600       -         -
                     red    600       -         -
1     tail-drop      green  600       -         -
                     yellow 600       -         -
                     red    600       -         -
2     tail-drop      green  600       -         -
                     yellow 600       -         -
```



```
                     red    600        -         -
3     random-detect  green  4000       500       19
                     yellow 3000       375       19
                     red    2000       250       19
4     tail-drop      green  600        -         -
                     yellow 600        -         -
                     red    600        -         -
5     tail-drop      green  600        -         -
                     yellow 600        -         -
                     red    600        -         -
6     tail-drop      green  600        -         -
                     yellow 600        -         -
                     red    600        -         -
7     tail-drop      green  600        -         -
                     yellow 600        -         -
                     red    600        -         -
```

Schedule Configurations

Packets are scheduled by SP between different queue which configured sp-mode.

The schedule priority of dwrr mode queues depend on the minimum queue id. for example, queue 0,1,3,6,7 configured mode sp and queue 2,4,5 configured mode dwrr, then the scheduler priority should be queue 7> queue 6> queue 3> (queue2, queue4, queue5)> queue 1>queue 0.

The following example shows configuring schedule parameters for egress queues. In this example, queue 0,1,3,6,7 belongs to sp mode, queue 2,4,5 belongs to dwrr mode. The DRR weight is 1:2:3.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configuring Schedule

```
Switch(config)# qos scheduler-profile p1
Switch(config-qos-scheduler-p1)# mode sp
Switch(config-qos-scheduler-p1)# exit
Switch(config)# qos scheduler-profile p2
Switch(config-qos-scheduler-p2)# mode dwrr
Switch(config-qos-scheduler-p2)# weight 1
Switch(config-qos-scheduler-p2)# exit
Switch(config)# qos scheduler-profile p3
Switch(config-qos-scheduler-p2)# mode dwrr
Switch(config-qos-scheduler-p2)# weight 2
Switch(config-qos-scheduler-p2)# exit
Switch(config)# qos scheduler-profile p4
Switch(config-qos-scheduler-p2)# mode dwrr
```



```
Switch(config-qos-scheduler-p2)# weight 3
Switch(config-qos-scheduler-p2)# exit
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)# qos queue 0
Switch(config-if-eth-0-1-queue-0)# scheduler-profile p1
Switch(config-if-eth-0-1-queue-0)# exit
Switch(config-if-eth-0-1)# qos queue 1
Switch(config-if-eth-0-1-queue-1)# scheduler-profile p1
Switch(config-if-eth-0-1-queue-1)# exit
Switch(config-if-eth-0-1)# qos queue 2
Switch(config-if-eth-0-1-queue-2)# scheduler-profile p2
Switch(config-if-eth-0-1-queue-2)# exit
Switch(config-if-eth-0-1)# qos queue 3
Switch(config-if-eth-0-1-queue-3)# scheduler-profile p1
Switch(config-if-eth-0-1-queue-3)# exit
Switch(config-if-eth-0-1)# qos queue 4
Switch(config-if-eth-0-1-queue-4)# scheduler-profile p3
Switch(config-if-eth-0-1-queue-4)# exit
Switch(config-if-eth-0-1)# qos queue 5
Switch(config-if-eth-0-1-queue-5)# scheduler-profile p4
Switch(config-if-eth-0-1-queue-5)# exit
Switch(config-if-eth-0-1)# qos queue 6
Switch(config-if-eth-0-1-queue-6)# scheduler-profile p1
Switch(config-if-eth-0-1-queue-6)# exit
Switch(config-if-eth-0-1)# qos queue 7
Switch(config-if-eth-0-1-queue-7)# scheduler-profile p1
```

step 3 Exit the configure mode

```
Switch(config-if-eth-0-1-queue-7)# end
```

step 4 Validation

```
Switch# show qos interface eth-0-1 queue schedule
QUEUE SCHEDULER:
Queue CIR(Kbps)     PIR(Kbps)      class      weight     mode
-----+--------------+-------------+----------+----------+---------
0     0              100000000     0          -          sp
1     0              100000000     1          -          sp
2     0              100000000     2          1          dwrr
3     0              100000000     3          -          sp
4     0              100000000     2          2          dwrr
5     0              100000000     2          3          dwrr
6     0              100000000     6          -          sp
7     0              100000000     7          -          sp
```


Configuring Shaping and policing

Port policing Configurations

All traffic received or transmitted in the physical interface can be limited rate, and all the exceeding traffic will be dropped.

The following example shows creating an ingress port policer. In this example, if the received traffic exceeds a 50000-kbps average traffic rate, it is dropped.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configuring port policing

```
Switch(config)# qos policer-profile pp
Switch(config-qos-policer-pp)# mode rfc2697 color-aware cir 50000 cbs 10000 ebs
16000 stats
Switch(config-qos-policer-pp)# exit
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# qos port-policer input pp
```

step 3 Exit the configure mode

```
Switch(config-if-eth-0-1)# end
```

step 4 Validation

```
Switch# show qos policer-profile pp
    POLICER-PROFILE-NAME: pp
    mode rfc2697, color aware mode, CIR 50000 Kbps, CBS 10000 Bytes, EBS 16000
Bytes,
    statistics enable
    Applied-interfaces:
             interface                direction
             ----------------------------------
             eth-0-1                  input
    Applied-policy-map:
             policy-map               class-map
             ----------------------------------
```

Configuring port shaping

All traffic transmitted in the physical interface can be shaped, and all the exceeding traffic will be buffered. If no buffer, it is dropped.

The following example shows creating a port shaping. In this example, if the received traffic exceeds a 100M, it is buffered.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configuring port shaping

```
Switch(config)# qos port-shape-profile p1
Switch(config-qos-port-shape-p1)# pir 100000
Switch(config-qos-scheduler-p3)# exit
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# qos port-shape-profile p1
```

step 3 Exit the configure mode

```
Switch(config-if-eth-0-1)# end
```

step 4 Validation

```
Switch# show qos interface eth-0-1 queue schedule
QUEUE SCHEDULER:
Queue CIR(Kbps)      PIR(Kbps)       class       weight      mode
-----+--------------+-------------+----------+----------+---------
0      0              100000000      0           -           sp
1      0              100000000      1           -           sp
2      0              100000000      2           -           sp
3      0              100000         3           -           sp
4      0              100000000      4           -           sp
5      0              100000000      5           -           sp
6      0              100000000      6           -           sp
7      0              100000000      7           -           sp
```

Configuring Mapping Tables

Configuring CoS-Tc-Color Map

The following shows modifying a CoS-Tc-Color map. This map is used to generate an internal priority color value from CoS during classification; this value determines the QoS action in the DUT, such as selecting one of the eight egress queues, etc. The CoS value can also came from the inner cos of incoming packets, if the port trusts inner cos.

- configure terminal.

- qos domain <0-6> map cos-tc-color cos <0-7> to TC COLOR to modify the CoS-Tc-Color Map. Tc = Tc value, range is 0-7. COLOR = color values, red, yellow or green.

The following example shows mapping cos 1 to tc 7 color green for QoS domain 1, and configure interface eth-0-1 to QoS domain 1.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configuring CoS-Tc-Color Map

```
Switch(config)# qos domain 1
Switch(config-qos-domain-1)# cos 1 to tc 7 color green
Switch(config-qos-domain-1)# exit
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# qos domain 1
```

step 3 Exit the configure mode

```
Switch(config-if-eth-0-1)# end
```

step 4 Validation

```
Switch# show qos domain 1 map-table ingress cos-tc-color

QoS DOMAIN 1, CFI Disable, COS map to TC & COLOR:
--------------------------------------------------------------
  COS     : 0      1      2      3      4      5      6      7
  TC      : 0      7      2      3      4      5      6      7
  color   : green  green  green  green  green  green  green  green
```

Configuring EXP-Tc-Color Map

The following shows modifying a EXP-Tc-Color map. This map is used to generate an internal priority color value from EXP during classification; this value determines the QoS action in the DUT, such as selecting one of the eight egress queues, etc. The EXP value can also came from the inner exp of incoming packets, if the port trusts inner exp.

- configure terminal.

- qos domain <0-6> map exp-tc-color exp <0-7> to TC COLOR to modify the EXP-Tc-Color Map. Tc = Tc value, range is 0-7. COLOR = color values, red, yellow or green.

The following example shows mapping exp 2 to tc 5 color green for QoS domain 1, and configure interface eth-0-1 to QoS domain 1.

步骤 1 进入配置模式

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configuring EXP-Tc-Color Map

```
Switch(config)# qos domain 1
Switch(config-qos-domain-1)# exp 2 to tc 5 color green
Switch(config-qos-domain-1)# exit
Switch(config)# interface eth-0-1
Switchconfig-if-eth-0-1)# qos domain 1
```

step 3 Exit the configure mode

```
Switch(config-if-eth-0-1)# end
```

step 4 Validation

```
Switch# show qos domain 1 map-table ingress exp-tc-color

QoS DOMAIN 1, EXP map to TC & COLOR:
-------------------------------------------------------------
  EXP      : 0      1      2      3      4      5      6      7
  TC       : 0      1      5      3      4      5      6      7
  color    : green  green  green  green  green  green  green  green
```

Configuring DSCP-Tc-Color Map

The following shows modifying a DSCP-Tc-Color map. This map is used to generate an internal tc color value from DSCP during classification; this value determines the QoS action in the DUT, such as selecting one of the eight egress queues, etc.

- configure terminal.
- qos domain 0 map dscp-tc-color DSCP to TC COLOR to modify the DSCP-Tc-Color Map. DSCP = DSCP value, range is 0-63. TC = tc value, range is 0-7. COLOR = color values, red, yellow or green

The following example shows mapping DSCP 34 to tc 7 color green for QoS domain 1, and configure interface eth-0-1 to QoS domain 1 with trust dscp.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configuring DSCP-Tc-Color Map

```
Switch(config)# qos domain 1
Switch(config-qos-domain-1)# dscp 34 to tc 7 color green
Switch(config-qos-domain-1)# exit
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# qos domain 1
```

step 3 Exit the configure mode

```
Switch(config-if-eth-0-1)# end
```

step 4 Validation

```
Switch# show qos domain 1 map-table ingress dscp-tc-color

QoS DOMAIN 1, DSCP map to TC & COLOR:
--------------------------------------------------------------
  DSCP      : 0      1      2      3      4      5      6      7
  TC        : 0      0      0      0      0      0      0      0
  color   : green  green  green  green  green  green  green  green
--------------------------------------------------------------
  DSCP      : 8      9      10     11     12     13     14     15
  TC        : 1      1      1      1      1      1      1      1
  color   : green  green  green  green  green  green  green  green
--------------------------------------------------------------
  DSCP      : 16     17     18     19     20     21     22     23
  TC        : 2      2      2      2      2      2      2      2
  color   : green  green  green  green  green  green  green  green
--------------------------------------------------------------
  DSCP      : 24     25     26     27     28     29     30     31
  TC        : 3      3      3      3      3      3      3      3
  color   : green  green  green  green  green  green  green  green
--------------------------------------------------------------
  DSCP      : 32     33     34     35     36     37     38     39
  TC        : 4      4      7      4      4      4      4      4
  color   : green  green  green  green  green  green  green  green
--------------------------------------------------------------
  DSCP      : 40     41     42     43     44     45     46     47
  TC        : 5      5      5      5      5      5      5      5
  color   : green  green  green  green  green  green  green  green
--------------------------------------------------------------
  DSCP      : 48     49     50     51     52     53     54     55
  TC        : 6      6      6      6      6      6      6      6
```



```
  color   : green  green  green  green  green  green  green  green
--------------------------------------------------------------
  DSCP      : 56     57     58     59     60     61     62     63
  TC        : 7      7      7      7      7      7      7      7
  color   : green  green  green  green  green  green  green  green
```

Configuring Tc-Color-CoS Map

The following shows modifying a Tc-Color-CoS map. This map is used to generate a new CoS from the internal tc color value in egress; This map is used if two domains have different CoS definitions; this map translates a set of one domain's CoS values to match the other domain's definition.

- configure terminal.
- qos domain <0-7> map tc-color-cos TC COLOR to COS to modify the Tc-Color-CoS Map.

The following example shows mapping tc 7 color green to CoS 6, and replace CoS in the interface eth-0-1 egress.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configuring Tc-Color-CoS Map

```
Switch(config)# qos domain 1
Switch(config-qos-domain-1)# tc 7 color green to cos 6
Switch(config-qos-domain-1)# exit
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# qos domain 1
Switch(config-if-eth-0-1)# replace cos
```

step 3 Exit the configure mode

```
Switch(config-if-eth-0-1)# end
```

step 4 Validation

```
Switch# show qos domain 1 map-table egress tc-color-cos

 QoS Domain 1, CFI Disable, TC & COLOR map to CoS:
             | COLOR:
             | red     yellow  green
-------------------------------------------
 TC       : 0  | 0       0       0
```



```
            1   | 1        1        1
            2   | 2        2        2
            3   | 3        3        3
            4   | 4        4        4
            5   | 5        5        5
            6   | 6        6        6
            7   | 7        7        6
```

Configuring Tc-Color-EXP Map

The following shows modifying a Tc-Color-EXP map. This map is used to generate a new EXP from the internal tc color value in egress; This map is used if two domains have different EXP definitions; this map translates a set of one domain's EXP values to match the other domain's definition.

- configure terminal.
- qos domain <0-7> map tc-color-exp TC COLOR to EXP to modify the Tc-Color-EXP Map.

The following example shows mapping tc 5 color green to EXP 4 in the interface eth-0-1 egress.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configuring Tc-Color-EXP Map

```
Switch# configure terminal
Switch(config)# qos domain 1
Switch(config-qos-domain-1)# tc 5 color green to exp 3
Switch(config-qos-domain-1)# exit
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# qos domain 1
Switch(config-if-eth-0-1)# end
```

step 3 Exit the configure mode

```
Switch(config-if-eth-0-1)# end
```

step 4 Validation

```
Switch# show qos domain 1 map-table egress tc-color-exp

 QoS Domain 1, TC & COLOR map to EXP:
             | COLOR:
```



```
               | red      yellow   green
-------------------------------------------
 TC        : 0  | 0        0        0
             1  | 1        1        1
             2  | 2        2        2
             3  | 3        3        3
             4  | 4        4        4
             5  | 5        5        3
             6  | 6        6        6
             7  | 7        7        7
```

Configuring Tc-Color-DSCP Map

The following shows modifying a Tc-Color-DSCP map. This map is used to generate a new DSCP from the internal tc color value in egress; This map is used if two domains have different DSCP definitions; this map translates a set of one domain's DSCP values to match the other domain's definition.

- configure terminal.
- qos domain <0-7> map tc-color-dscp TC COLOR to DSCP to modify the Tc-Color-DSCP Map.

The following example shows mapping tc 7 color green to DSCP 60, and replace DSCP in the interface eth-0-1 egress.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configuring Tc-Color-DSCP Map

```
Switch(config)# qos domain 1
Switch(config-qos-domain-1)# tc 7 color green to dscp 60
Switch(config-qos-domain-1)# exit
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# qos domain 1
Switch(config-if-eth-0-1)# replace dscp
```

step 3 Exit the configure mode

```
Switch(config-if-eth-0-1)# end
```

step 4 Validation

```
Switch# show qos domain 1 map-table egress tc-color-dscp
```



```
QoS Domain 1, TC & COLOR map to DSCP:
             | COLOR:
             | red     yellow  green
-------------------------------------------
TC       : 0  | 0       0       0
           1  | 8       8       8
           2  | 16      16      16
           3  | 24      24      24
           4  | 32      32      32
           5  | 40      40      40
           6  | 48      48      48
           7  | 56      56      60
```

Configuring QoS attributes of interface

Configuring QoS attributes of interface

Each domain has ingress and egress direction mapping, such as ingress direction include cos-tc-color, exptc-color, dscp-tc-color map, egress direction include tc-color-cos, tc-color-exp, tc-color-dscp map. To configure trust state on an interface to select specific mapping relationship for incoming packets, the trust state can be trust cos, trust dscp-exp and trust port. Similarly, users can configure replace cos, replace dscp in the egress direction of an interface.

Each interface has a default priority, and the default is 0. In port trust port mode, packets will use port default priority as internal priority. The command set cos to set port priority and port cos values range from 0-7.

The following example shows set interface eth-0-1 default priority 5 as well as set trust port mode, and replace DSCP in the interface eth-0-1 egress.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configuring QoS attributes of interface

```
Switch(config)# qos domain 1
Switch(config-qos-domain-1)# tc 5 color green to dscp 60
Switch(config-qos-domain-1)# exit
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# qos domain 1
Switch(config-if-eth-0-1)# set cos 5
Switch(config-if-eth-0-1)# trust port
Switch(config-if-eth-0-1)# replace dscp
```


step 3 Exit the configure mode

```
Switch(config-if-eth-0-1)# end
```

step 4 Validation

```
Switch# show qos interface eth-0-1 classify-remark
Interface   Domain       Trust        Replace-cos   Replace-dscp
-----------+------------+------------+-------------+-------------
eth-0-1     1            port(cos=5)  enable        enable
```

9 Reliability Configuration Guide

9.1 Configuring BHM

9.1.1 Overview

Function Introduction

BHM is a module which is used to monitor other PMs. The PMs monitor is implemented in user space. When a monitored PM is not send keepalive message in 5 minutes to BHM process, the BHM module will think this PM is uncontrolled and take measures, such as printing warning on screen, shutting all ports, or restarting the system, to help or remind users to recover the system. The monitored PMs include ccs, cds, chsm, fea, switch, routed, opm, authd, appcfg, dhcrelay, dhcsnooping, dhcpclient, ptp, ssm, ncs, pimd for CNOS product; and additional ovs-vswitchd for Openflow product. There are three activations of BHM, including "reload", "warning", "shutdown".

Principle Description

N/A

9.1.2 Configuration

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enable heart-beat-monitor globally

```
Switch(config)# heart-beat-monitor enable
```

step 3 Reload system if a monitored PM is uncontrolled

```
Switch(config)# heart-beat-monitor reactivate reload
```


NOTE There are three activations of BHM, including "reload system", "warning", "shutdown port".

step 4 Exit the configure mode

```
Switch(config)# end
```

step 5 Validation

```
Switch# show heart-beat-monitor
heart-beat-monitor enable
heart-beat-monitor reactivation: reload system
```

9.1.3 Application cases

N/A

9.2 Configuring Track

9.2.1 Overview

Function Introduction

Track is used for link the functional modules and monitor modules. Track builds a system structure with 3 levels: "functional modules -- Track -- monitor modules".

Track can shield the difference of the monitor modules and provide an unitized API for the functional modules.

The following monitor modules are supported:

- IP SLA
- interface states

The following functional modules are supported:

- Static route
- VRRP

Track makes a communication for the functional modules and monitor modules. When link states or network performance is changed, the monitor modules can detect the event and notify the track module; therefore track will change its owner states and notify the related functional modules.

Principle Description

N/A

9.2.2 Configuration

Configuring track interface linkstate

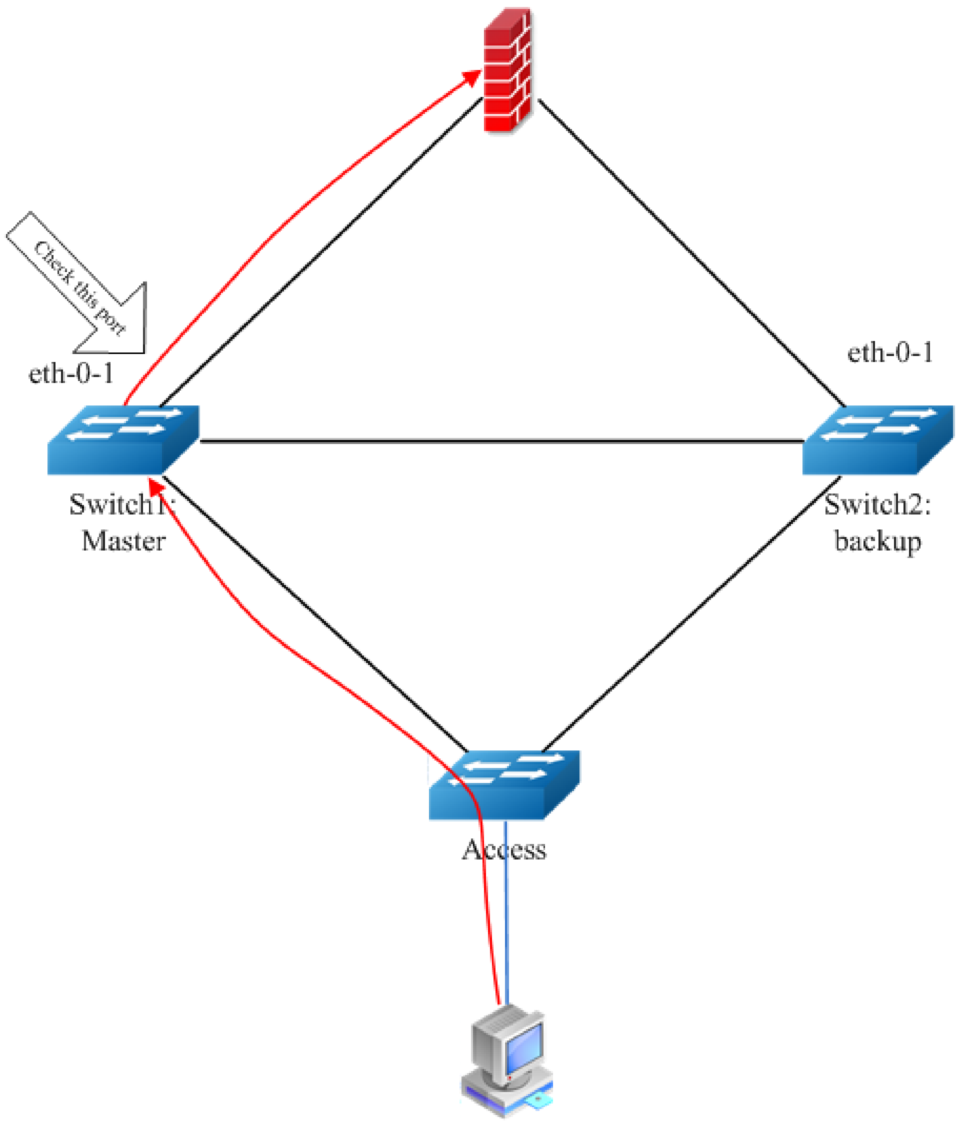

Figure 9-1 Track interface

Before the introduction of track feature, the topo has a simple tracking mechanism that allowed you to track the interface link state only. If the link state of the interface went down, the priority of the router was reduced, allowing another router with a higher priority to become active. The Track feature separates the tracking mechanism and creates a separate standalone tracking process that can be used by other processes in future. This feature allows tracking of other objects in addition to the interface link state. It can now register its interest in tracking objects and then be notified when the tracked object changes state. TRACK is a separate standalone tracking process that can be used by other processes. This feature allows tracking of other objects in addition to the interface link state.

Configuring on switch1:

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create track and set the attributes

```
Switch(config)# track 1 interface eth-0-1 linkstate
Switch(config-track-1)# delay up 30
Switch(config-track-1)# delay down 30
Switch(config-track-1)# exit
```

NOTE Parameters for track:

- delay up: After the interface states is up, the track will wait for a cycle before restore the states. Valid range is 1-180 second. The default configuration is restore without delay.
- delay down: After the interface states is down, the track will wait for a cycle before change the states. Valid range is 1-180 second. The default configuration is change without delay.

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

```
Switch#show track
Track 2
    Type                       : Interface Link state
```



```
    Interface                  : eth-0-1
    State                      : down
    Delay up                   : 30 seconds
Delay down                     : 30 seconds
```

Configuring track ip sla reachablity/state

Figure 9-2 IP SLA Track

The following configuration should be operated on all switches if the switch ID is not specified:

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create track and set the attributes

```
Switch(config)# ip sla monitor 1
Switch(config-ipsla-1)# exit
Switch(config)# track 2 ip sla 1 reachability
Switch(config-track-2)# track 2 ip sla 1 state
Switch(config-track-2)# exit
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

Display the result on Switch1.

```
Switch# show track 2
Track 2
    Type                    : Response Time Reporter(RTR) State
    IP-Sla entry number : 1
    State                   : down
```

9.2.3 Application cases

N/A

9.3 Configuring CoPP

9.3.1 Overview

Function Introduction

Control Plane Policing (CoPP) protects the control plane and separates it from the data plane, which ensures network stability, reachability, and packet delivery.

Speed limit means that the rate of sending packets to CPU is limited to a certain rate.

Filtering means that CoPP module provides a series of command that allow users to configure black-white list ACL to prohibit illegal access to this switch.

Schedule means that every kind of packet which send to CPU has one priority，the function of CoPP can ensure the higher priority packet to be processed firstly.

CoPP can configure some particular reason rate of PDU type which exception packet destined to CPU, the unit is kbps. We support reason mtu-fail，ttl-expire, arp, packet-in, dhcp, eapol, igmp, lldp, ptp, bpdu, erps, l2pro-group-mac, l2pro-protocol-mac, mlag, ospf, slow-protocol, vrrp, managment-traffic, ssh, telnet. In these reason mtu-fail, ttl-expired belongs to class 0, arp belongs to class 1, packet-in belongs to class 2, lldp, eapol, dhcp, igmp, ptp belong to class 3, slow-protocol, bpdu, erps, mlag, l2pro-group-mac, l2pro-protocol-mac, ospf, vrrp belong to class 4, management-traffic, ssh, telnet belongs class 5. The priority for 6 control plane class: class 5 > class 4 > class 3 > class 2 > class 1 > class 0. The other reason packets send to cpu belongs to class 0. CoPP support to configure control plane class rate and total rate. The class rate and total rate unit are pps. The aggregate rate of all streams destined to CPU should not exceed the total rate configured value.

Principle Description

N/A

9.3.2 Configuration

Configuring Control-plane Policy-map

By default, there is no control-plane policy map configured. With the configuration of policy-map, the all reason kinds of packet which send to CPU can be filter by ACL.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enable QoS globally

```
Switch(config)# qos global
Switch(config-qos-global)# qos enable
Switch(config-qos-global)# exit
```

step 3 Configuring class-map and policy-map

```
Switch(config)# ip access-list tt
Switch(config-ip-acl-tt)# deny
Switch(config-ip-acl-tt)# exit
Switch(config)# class-map tt
Switch(config-class-map-tt)# match access-list tt
Switch(config-class-map-tt)# exit
Switch(config)# policy-map tt
Switch(config-policy-map-tt)# class tt
Switch(config-pmap-tt-cmap-tt)# exit
Switch(config-policy-map-tt)# exit
```

step 4 Apply the policy-map

```
Switch(config)# control-plane
Switch(config-control-plane)# policy input tt
Switch(config-control-plane)# exit
```

step 5 Exit the configure mode

```
Switch(config)# end
```

step 6 Validation

Use lldp packets for example, enable lldp firstly:


```
Switch(config)# lldp enable
Switch(config)# interface eth-0-9
Switch(config-if-eth-0-9)# no shutdown
Switch(config-if-eth-0-9)# lldp enable txrx
```

Display the statistic information:

```
Switch# show control-plane statistics
Ingress service policy: tt
 Class name: tt
  access-group: tt
   10 deny any (2 match 822 bytes)
 total (2 match 822 bytes)
```

Configuring Control-plane Reason Rate

Use control-plane reason command to set reason rate. The default rate of mtu-fail is 64kbps，ttl-expired is 64kbps, arp is 160kbps，packet-in is 160kbps, dhcp is 320kbps, eapol is 64kbps，igmp is 128kbps, lldp is 64kbps，ptp is 128kbps，bpdu is 64kbps， erps is 64kbps， l2pro-group-mac is 256kbps, l2pro-protocol-mac is 256kbps, mlag is 256kbps, ospf is 32kbps, slow-protocol is 64kbps，vrrp is 256kbps, management-traffic is 1600kbps，ssh is 512kbps, telnet is 512kbps.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configuring reason rate

```
Switch(config)# control-plane
Switch(config-control-plane)# reason lldp rate 32
Switch(config-control-plane)# exit
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

```
Switch# show control-plane reason
Rate unit (kbps)
Reason                    Class      Rate       Default Rate
--------------------+----------+----------+------------
mtu-fail                  0          64         64
ttl-expired               0          64         64
arp                       1          160        160
icmpv6                    1          64         64
```



```
packet-in            2          160        160
dhcp                 3          320        320
eapol                3          64         64
igmp                 3          128        128
lldp                 3          32         64
ptp                  3          128        128
bpdu                 4          64         64
erps                 4          64         64
l2pro-group-mac      4          256        256
l2pro-protocol-mac   4          256        256
mlag                 4          256        256
ospf                 4          32         32
slow-protocol        4          64         64
vrrp                 4          256        256
management-traffic   5          1600       1600
ssh                  5          512        512
telnet               5          512        512
```

Configuring Control-plane class Rate

Use control-plane class command to set class rate, unit is pps. Class 1-5 default value is 2048pps, class 0 default value is 1024pps.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configuring class rate

```
Switch(config)# control-plane
Switch(config-control-plane)# class 1 rate 100
Switch(config-control-plane)# exit
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

```
Switch# show control-plane class
control-plane class        rate (pps)
--------------------------+------------
class 0                    1024
class 1                    100
class 2                    2048
class 3                    2048
class 4                    2048
class 5                    2048
--------------------------+------------
total rate                 2048
```



```
control-plane class information:
    class 5 is used for all managing packet and ssh telnet packet!
    class 4 is used for follow protocol packet!
       LACP, ESMC, STP, ERPS, OSPF, VRRP, MLAG, L2PROTOCOL!
    class 3 is used for follow protocol packet!
       LLDP, EAPOL, DHCP, IGMP, PTP!
    class 2 is used for packet-in!
    class 1 is used for ARP and ICMPv6 (include ND) packet!
    class 0 is used for other packet!
```

Configuring Control-plane Total Rate

Use control-plane total rate command to set total rate, unit is pps, default value is 2048pps. The aggregate rate of all streams destined to CPU should not exceed the configured value.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configuring class rate

```
Switch(config)# control-plane
Switch(config-control-plane)# total rate 1000
Switch(config-control-plane)# exit
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

```
Switch# show control-plane class
control-plane class        rate (pps)
--------------------------+------------
class 0                    1024
class 1                    2048
class 2                    2048
class 3                    2048
class 4                    2048
class 5                    2048
--------------------------+------------
total rate                 1000

control-plane class information:
    class 5 is used for all managing packet and ssh telnet packet!
    class 4 is used for follow protocol packet!
       LACP, ESMC, STP, ERPS, OSPF, VRRP, MLAG, L2PROTOCOL!
```



```
    class 3 is used for follow protocol packet!
       LLDP, EAPOL, DHCP, IGMP, PTP!
    class 2 is used for packet-in!
    class 1 is used for ARP and ICMPv6 (include ND) packet!
    class 0 is used for other packet!
```

9.3.3 Application cases

N/A

10 RPC API Configuration Guide

10.1 Configuring Service

10.1.1 Overview

Function Introduction

RPC API service allows user to configure and monitor the switch system through Remote Procedure Calls (RPC) from your program.

The service currently supports JSON-RPC over HTTP protocol together with HTTP Basic authentication.

Principle Description

RPC API service uses standard JSON-RPC over HTTP protocol to communicate the switch and your program. User may issue switch CLI commands through JSON-RPC method: 'executeCmds'. By default, the CLI mode is in privileged EXEC mode (#).

User could send JSON-RPC request via an HTTP POST request to URL: http://:/command-api. The detailed JSON-RPC request and response are show below:

JSON-RPC Request

```
{
    "params":[                                                  Parameters for command
        {
            "format":"text",                                    Expected response format,
can be 'text' or 'json',
the default format is 'text'
            "version":1,                                        The API version
            "cmds":[                                            List of CLI commands
                "show run",                                     CLI command 1
                "config t",                                     CLI command 2
                "vlan database",                                CLI command 3
                "vlan 1-8",                                     CLI command 4
                "interface eth-0-1",                            CLI command 5
```



```
                "switchport mode trunk",                CLI command 6
                "switchport trunk allowed vlan add 2",  CLI command 7
                "shutdown",                             CLI command 8
                "end",                                  CLI command 9
                "show interface switchport"             CLI command 10
            ]
        }
    ],
    "jsonrpc":"2.0",                                    JSON RPC protocol version.
Always 2.0.
    "method":"executeCmds",                             Method to run the switch
CLI commands
    "id":"70853aff-af77-420e-8f3c-fa9430733a19"         JSON RPC unique identifier
}
```

JSON-RPC Response

```
{
    "jsonrpc":"2.0",                                     JSON RPC protocol version.
Always 2.0.
    "id":"70853aff-af77-420e-8f3c-fa9430733a19",         JSON RPC unique identifier
    "result":[                                           Result list of objects
from each CLI command executed.
        {
            "sourceDetails":"version 5.1.6.fcs\n!\n …",  Output information of CLI
Command 1.
                                                         The Original ASCII output
information returned from CLI command if this command is successfully executed.
            "errorCode":-1003,                           Error code if it is
available.
            "errorDesc":"unsupported command…",          Error description if it is
available.
            "warnings":"% Invalid…",                     Warnings if it is
available.
                                                         Formatted JSON object will
also be returned if it is available.
        },
        { },                                             Output information of CLI
Command 2.
        { },                                             Output information of CLI
Command 3.
        { },                                             Output information of CLI
Command 4.
        { },                                             Output information of CLI
Command 5.
        { },                                             Output information of CLI
Command 6.
        { },                                             Output information of CLI
Command 7.
        { },                                             Output information of CLI
Command 8.
        { },                                             Output information of CLI
Command 9.
        {
            "sourceDetails":" Interface name          : eth-0-1\n Switchport
```



```
mode          : trunk\n …\n"
        }                                            Output information of CLI
Command 10.
    ]
}
```

Python Client Example Code

Here is an example code using ‘pyjsonrpc’ library:

```
import pyjsonrpc
import json

http_client = pyjsonrpc.HttpClient(
    url = "http://10.10.39.64:80/command-api",
    username = "username",
    password = "password"
)

cmds = {}
cmd_list = ["show run", "config t", "vlan database", "vlan 1-8", "interface eth-0-
1", "switchport mode trunk", "switchport trunk allowed vlan add 2", "shutdown",
"end", "show interface switchport"]

cmds['cmds'] = cmd_list
cmds['format'] = 'text'
cmds['version'] = 1

try:
    response = http_client.call("executeCmds", cmds)
    print("json response:");
    json_result = json.dumps(response, indent=4)
    print(json_result)
except Exception, e:
    if e.code == 401:
        print "Unauthorized user"
    else:
        print e.message
        print e.data
```

Error code

Here is a list of JSON-RPC 2.0 error code:

Error Code	Description
-32700	Parse error
-32600	Invalid Request
-32601	Method not found

-32602	Invalid param
-32603	Internal error

Here is a list of RPC-API error code:

Error Code	Description
-1000	General error
-2001	JSON RPC API Error: unsupported API version
-2002	JSON RPC API Error: must specify 'params' with 'cmds' in JSON RPC
-2003	JSON RPC API Error: unsupported command response format
-3001	Command execution failed: timed out
-3002	Command execution failed: unsupported command
-3003	Command execution failed: unauthorized command
-3004	Command execution failed: the string does not match any command in current mode
-3005	Command execution failed: can't convert to JSON format
-3006	Command execution failed: command list too short
-3007	Command execution failed: command list too long

10.1.2 Configuration

Configuring RPC API service

User could enable the RPC API service by the following steps.

The default port is 80.

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enable RPC API service

```
Switch(config)# service rpc-api enable port 1025
```

NOTE Use the following command to disable rpc-api service:

```
Switch(config)# service rpc-api disable
```

step 3 Exit the configure mode

```
Switch(config)# end
```

Configuring RPC API service with HTTP Authentication

User could configure the HTTP authentication mode of RPC API service.

Currently, only HTTP Basic authentication is supported. User will receive status code: 401 (Unauthorized access) if user provides invalid user name or password.

step 1 Enter the configure mode

```
Switch# configure terminal
```

Step 2 Set the username and password, then enable the rpc-api authentication

```
Switch(config)# username myuser password mypass
Switch(config)# service rpc-api auth-mode basic
```

NOTE Use the following command to disable authentication:

```
Switch(config)# no service rpc-api auth-mode
```


NOTE HTTP authentication settings of RPC API service will take effect after you restart this service or reboot the system.

step 3 Exit the configure mode

```
Switch(config)# end
```

step 4 Validation

```
Switch# show services rpc-api
RPC-API service configuration:
Server State        : enable
Port                : 1025
Authentication Mode : none
SSL State           : disable
Message Execute     : 0
Message Deny        : 0
```

10.1.3 Application cases

N/A

11 Openflow Configuration Guide

11.1 Hybrid

11.1.1 Overview

Centec Networks Hybrid switch is based on independent research and development system named CNOS(Centec Networks Open System) and combine with OVS(Openflow vswitch) open source code, which is a switch system gather tradition and openflow function.

Pure flow based openflow SDN network scenario usually faced below problem:

- Management link and business link separate wiring, which leads to hign operating costs;
- Management channel there is no redunancy backup, the device there is risk of out of controll when link down;
- Lack of link detection mechanism, such as link detection, packet loss, delay detection. If all of these rely on controller strategy, lead to problem that inaccuracy, high delay and occupy cpu resource;
- The inconsistency between switch CLI and north interface to send command. Business channel command is sent by controller based on openflow protocol, OVSDB DB infomation is rely on OVSDB channel or manual operation, switch itself command sent by switch CLI.

Against this background, Centec Hybrid switch system born as times require. Hybrid switch has a new definition about forward concept and CLI, especially the definition of hybrid interface. Hybrid system is a combination of traditional network technology and SDN network technology. Hybrid system there are many feature, such as support in-band management, flexible interface configuration and forwarding, unification of CLI and OVSDB configuration, plentiful north interface including RPC-API and Netconf.

11.1.2 Hybrid switch structure

The chapter aims at explain Hybrid switch forwarding structure in a brief word to help users to understand Hybrid switch forwarding process. Hybrid forwarding structure is consist of three part: Hybrid Global Forwarding Structure, Hybrid Inport Forwarding Structure, Hybrid Outport Forwarding Structure.

Hybrid Global Forwarding Structure

In_Port Module: Switch inport process

Hybrid Port: Interface enable/disable Hybrid mode(default disbale), for example:

```
interface eth-0-1
 [no]openflow enable
```

Protected-vlan: If Packet carried vlan within protected-vlan list, packet will forward with the L2/L3 rule, not follow openflow table anymore, for example:

```
interface eth-0-1
 openflow enable
 protected-vlan 4094
```

Flow Table: Forward with openflow table flow, for example:

```
ovs-ofctl add-flow br0 in_port=1,actions=output:2 -O openflow13
```

Bridge/Route, Traditional mode: Forward with the L2/L3 rule

Match Flow: Matched Openflow table flow

Action: Execute action defined in flow action

To normal: Packet match openflow flow entry, flow action output to normal logical port, packet will forward with the L2/L3 rule, for example:

```
ovs-ofctl add-flow br0 in_port=1,actions=normal -O openflow13
```

Traditional L2/L3 forwarding: Packet forward with FDB/ARP/ROUTE table

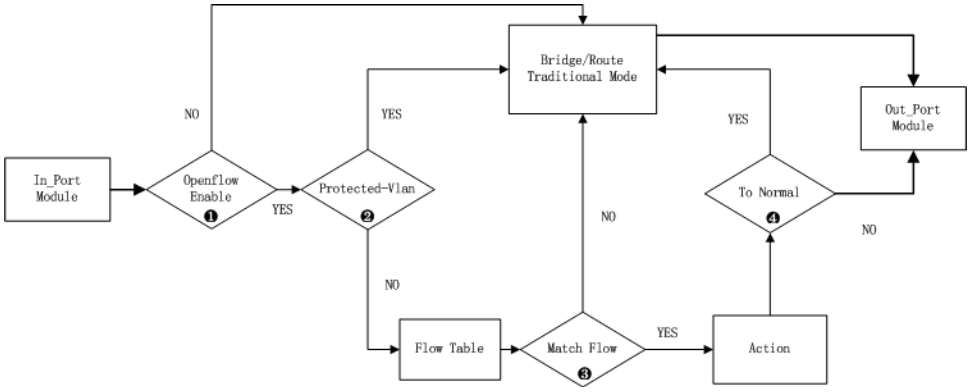

Figure 11-1 Hybrid Global Forwarding Structure topology

Hybrid Global Forwarding Structure Description:

1.When the packet is sended to one port, it will check the port is openflow enable first; If yes, it will goto the 2 step, if no, the packet will forward with the L2/L3 rule.

2.Check the packet is match the protected-vlan in the port; If yes, the packet will forward with the L2/L3 rule, if no, goto the step 3.

3.Match the openflow table, If yes, goto the step 4, if no, the packet will forward with the L2/L3 rule.

4.If the flow table action is "to normal", then the packet will forward with the L2/L3 rule.

Hybrid Inport Forwarding Structure

Vlan-filter(ingress direction): Enable/disable vlan filter behaviour of traditional port, for example: Eth-0-1 allow packet with vlan tag(default: vlan-filter enable).

Configuration:

```
interface eth-0-1
 openflow enable
 vlan-filter disable
```

Traditional mode Vlan-Filter(Ingress): Traditional behaviour to packet with vlan tag, for example: Eth-0-1 accept packet with vlan 200, but filter pcaket with vlan 100.

Configuration:


```
interface eth-0-1
 switchport mode trunk
 switchport trunk allowed vlan add 100
 openflow enable
```

Add Native Vlan: When forward with openflow flow entry, use this command to set oringal packet if affected with trational port vlan edit attribution that recieve untag packet if add port's native vlan. For example: Inport is eth-0-1, output port is eth-0-2, both of two port's native vlan is different. If incoming is untag packet, packet output from eth-0-2 will carry eth-0-1's native vlan 100, this explains how this button affect openflow flow forwarding. Default, ingress add native Vlan disable.

Configuration:

```
interface eth-0-1
  switchport mode trunk
  switchport trunk native vlan 100
  switchport trunk allowed vlan all
  openflow enable
  [no] ingress-add-native-vlan enable
!
interface eth-0-2
  switchport mode trunk
  switchport trunk native vlan 200
  switchport trunk allowed vlan all
  openflow enable
!
ovs-ofctl add-flow br0 in_port=1,actions=output:2 -O openflow13
```

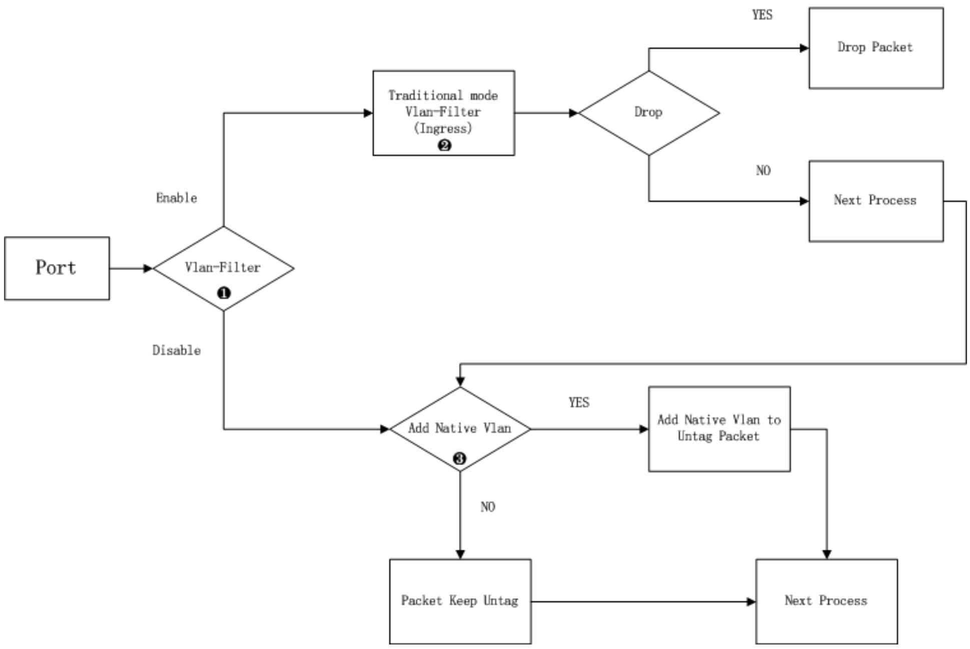

Figure 11-2 Hybrid inport forwarding structure topology

Hybrid Inport Forwarding Structure Description:

1.When the packet is sended to one port, it will check vlan filter is disable first; If no, it will goto the 2 step, if yes, it will goto the 2 step.

2.Check the packet with the L2/L3 rule, if the result is drop, the packet will drop and return; if the result is forward, goto the step 3.

3.If the port is openflow enable, the untag packet is not add the default vlan; If open the switch of adding default vlan in port, the packet will add the vlan tag and goto next step.

Note - Mac learning will still work although configurate vlan-filter disable on interface. In this situation, mac learning cannot go on because the interface doesn't belong to related vlan. And mac learning interruption will have a impact on cpu. So we advice you to disable mac learning on the interface which configruated vlan filter disable.

Hybrid Outport Forwarding Structure

Device Process: Packet processed by device, get to outport module

Bridge/Route, Traditional Mode: Packet forward with L2/L3 rule, get to outport module

Flow Table: Packet forward with openflow table, get to output module

Traditional mode, Vlan-Filter(Egress): Vlan filter attribution in egress direction affect whether packet with vlan tag could sent by outport. For example: after processed by switch device, packet with vlan 200 will be refused by eth-0-1 outport module, but packet with vlan 100 will be forwarded.

Configuration:

```
interface eth-0-1
 switchport mode trunk
 switchport trunk allowed vlan add 100
 openflow enable
```

Vlan-filter(Egress): Enable/disable vlan filter on egress port. For example: packet with vlan 200 forwarded with openflow table will ignore output port attribution. Configuration:


```
interface eth-0-1
 openflow enable
 vlan-filter disable
```

Tunnel Encapsulation: The output port type of flow action decide whether need to encapsulation.

- Simple stream: Flow content doesn't refer to any tunnel process, the flow action are just some simple edit process such as mac, ip, vlan edit action.
- Complex stream: Flow content refer to Mpls/Gre/Vxlan encapsulation.

The packet with vlan tag forwarded with simple stream ignore output port attribution while forwarded with complex stream is affected by output port attribution. For example, packet with vlan 100 forwarded with simple stream, will output with vlan 100 ignore access output port; packet with vlan 100 forwarded with complex stream, will output without vlan 100 that affected by access port.

Configuration:

```
interface eth-0-1
 switchport access vlan 100
 openflow enable
 vlan-filter disable
```

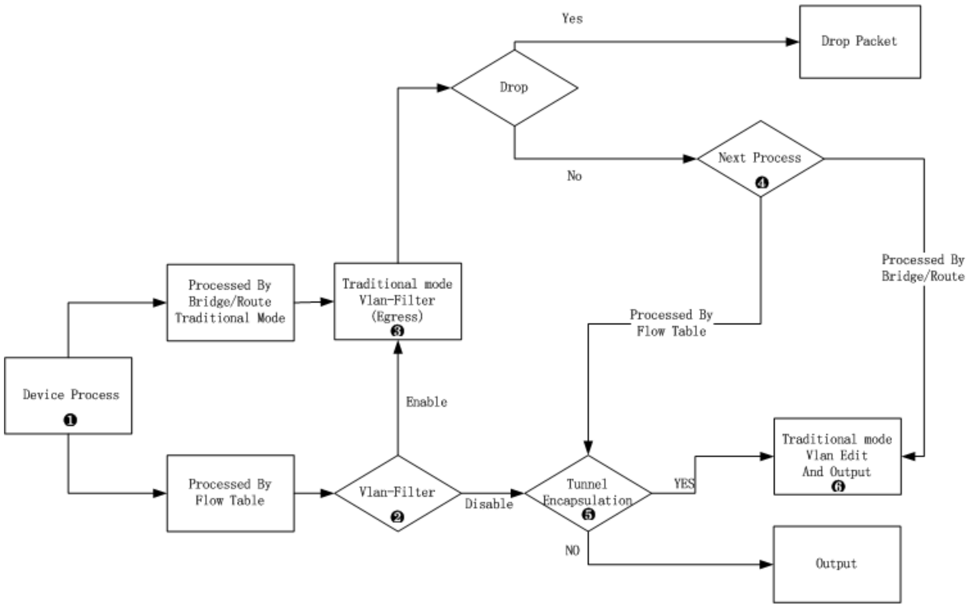

Figure 11-3 Hybrid Outport forwarding structure topology

Hybrid Outport Forwarding Structure Description:

1.If the packet is forwarded to the port with the L2/L3 rule, it will check vlan filter is disable first; If no, it will goto the 2 step, if yes, it will goto the 3 step.

2.If the packet is forwarded to the port with openflow table, it will check vlan filter is disable first; If no, it will goto the 2 step, if yes, it will goto the 3 step.

3.Check the packet with the L2/L3 rule, if the result is drop, the packet will drop and return; if the result is forward, goto the step 3.

4.The packet is edit with the port edit rule, the action of packet follow the the port form change.

5.If the packet is forwarded to the port with openflow table, check the sample flow action; If yes, the output edit will be not affected by port form, if no, the the output edit will be affected by port form.

6.The the sample flow action: the flowtable match one flow and not do the tunnel operation, not add/delete the L2 header, just do vlan operation, mac replace and forward.

11.1.3 Compatibility of Hybrid and OVS

Many features of pure openflow switch coupled with traditional, such as Gre/Vxlan tunnel, bond logical port, etc. All of these are defined in OVSDB. Hybrid switch do the things that fuse OVSDB features and traditional features together. Following are some examples of configurations.

Openflow API

Compatibility of Hybrid OVS and Openflow:

Physical port: Use system CLI to enable openflow function on physical port while a port will be added to OVSDB. For example:

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# openflow enable
Switch# ovs-ofctl dump-ports-desc br0 -O openflow13
OFPST_PORT_DESC reply (OF1.3) (xid=0x2):
 1(eth-0-1): addr:00:1e:08:0a:61:1b
     config:     0
     state:      LINK_DOWN
     speed: 0 Mbps now, 0 Mbps max
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no openflow enable
Switch# ovs-ofctl dump-ports-desc br0 -O openflow13
OFPST_PORT_DESC reply (OF1.3) (xid=0x2):
```


Gre/Vxlan logical port: Use system CLI to create Gre/Vxlan logcal port, when config openflow enable on Gre/Vxlan port, it is a hybrid mode, otherwise, it is a traditional tunnel port. For example:

```
Switch(config)# interface vxlan1
Switch(config-if-vxlan1)# tunnel-source-ip 1.1.1.1
Switch(config-if-vxlan1)# tunnel-remote-ip 2.2.2.2
Switch(config-if-vxlan1)# tunnel-bind-static bind-port eth-0-2 nexthop-mac 0.0.2
Switch(config-if-vxlan1)# openflow enable
Switch# ovs-ofctl dump-ports-desc br0  -O openflow13
OFPST_PORT_DESC reply (OF1.3) (xid=0x2):
2(eth-0-2): addr:00:1e:08:0a:54:4d
     config:     0
     state:      LINK_DOWN
     speed: 0 Mbps now, 0 Mbps max
2201(vxlan1): addr:00:1e:08:0a:54:4d
     config:     0
     state:      0
     speed: 0 Mbps now, 0 Mbps max
```

Bond/Agg logical port: Use system CLI to create agg logcal port, when config openflow enable on agg, it is a hybrid mode, otherwise it is a traditional agg port. For example:

```
Switch(config)# interface range eth-0-1 - 2
Switch(config-if-range)# static-channel-group 1
Switch(config)# interface agg1
Switch(config-if-agg1)# openflow enable
Switch# ovs-ofctl dump-ports-desc br0 -O openflow13
OFPST_PORT_DESC reply (OF1.3) (xid=0x2):
1221(agg1): addr:00:1e:08:0a:54:4c
     config:     0
     state:      LINK_DOWN
     speed: 0 Mbps now, 20000 Mbps max
```

Openflow attribute configuration

Openflow controller: System support multiple controller connection, the status and stats between switch and controller can be displayed.

```
Switch(config)# openflow set controller mgmt-if tcp 10.10.33.239 6692
Switch# show openflow controller status
Openflow controller-affect-flow: enable
Total Controllers: 1
Controller                : tcp:10.10.33.239:6692
--------------------------------------------------
status                    : online
online-time               : 5d 2h 55m 57s
role                      : other
mgmt-if                   : yes
bind_ip                   : none
```



```
max_backoff(sec)          : 8
inactivity_probe(sec)     : 5

Switch# show openflow controller stats
Openflow controller-affect-flow: enable
Total Controllers: 1
controller                : tcp:10.10.33.239:6692
--------------------------------------------------
connection attempts       : 1
successful  attempts      : 1
receive flow adds         : process:0 deny:0
receive flow mods         : process:0 deny:0
receive flow deletes      : 0
packet-in                 : 0
packet-out                : 0
echo-request on switch    : rx:0 tx:88512
echo-reply on switch      : rx:88512 tx:0
local
--------------------------------------------------
receive flow adds         : process:32 deny:6
receive flow mods         : process:0 deny:0
receive flow deletes      : 54
```

Openflow protocol version：System specify Openflow protocol version

```
Switch(config)# openflow set protocols openflow10 openflow13
Switch# show openflow protocol status
protocol support:
 OpenFlow10 OpenFlow13
```

11.2 Configuring Hybrid Controller

11.2.1 Overview

In Hybrid mode, the controller support the inband and outband, and the inband can specify the output address of the device.

The Hybrid device has been docked test with most manistream controller, for example, ODL,ONOS,RYU.

The Hybrid device most support for connecting five controllers at the same time.

The Hybrid device can connect the controller with tcp/ssl(tsl).

11.2.2 Configuration

TCP Controller Connecting

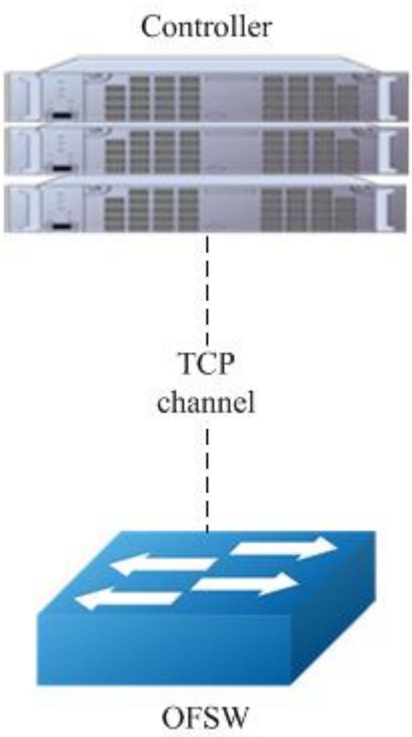

Figure 11-4 Hybrid Controller topology

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set the ip address and port of the controller

```
Switch(config)# openflow set controller tcp 1.1.1.1 6666
```

step 3 Exit the configure mode

```
Switch(config)# end
```

step4 Validation

```
Switch# show openflow controller status
Openflow controller-affect-flow: enable
Total Controllers: 1

Controller               : tcp:1.1.1.1:6666
```



```
------------------------------------------------------------
status                    : online
online-time               : 0d 0h 3m 25s
role                      : other
mgmt-if                   : yes
bind_ip                   : none
max_backoff(sec)          : 8
inactivity_probe(sec)     : 5
```

TLS Controller Connecting

If uses want to connect the controller with more safer method, we recommend that they can use TLS.

step 1 Create key

- Note: Hybrid580 does not support the ovs-pki.
- Users can install one openvswitch or ovs-pki sever to creat the key.
- Users can refer to the openvswitch ducumens.

Create a PKI by using ovs-pki script:

% ovs-pki init

(Default directory is /usr/local/var/lib/openvswitch/pki)

The pki directory consists of controllerca and switchca subdirectories. Each directory contains CA files.

Create a controller private key and certificate:

% ovs-pki req+sign ctl controller

ctl-privkey.pem and ctl-cert.pem are generated in the current directory.

Create a switch private key and certificate:

% ovs-pki req+sign sc switch

sc-privkey.pem and sc-cert.pem are generated in the current directory.

step 2 TLS connecting with switch configuration

Copy the key created by step1 to switch "flash:/" dir configure as follows.

```
Testing TLS Connection
Configuring ovs-vswitchd to use CA files using the system command "openflow set
```



```
ssl-key" ,e.g.:
Switch (config)# openflow set ssl-key flash:/sc-privkey.pem flash:/sc-cert.pem
flash:/controllerca_cacert.pem
Switch (config)# openflow set controller ssl 10.10.33.239 6695
```

step 3 TLS connecting with controller configuration

```
root@centec239-OptiPlex-380:/home/sdnlab/trunk/ryu-4.1/ryu/app# ryu-manager --ctl-
privkey key/ctl-privkey.pem --ctl-cert key/ctl-cert.pem --ca-certs
key/switchca/cacert.pem --verbose  --ofp-ssl-listen-port 6685 --wsapi-port 8182
ofctl_rest.py
```

step 4 Validation

```
Switch # show openflow controller status
Openflow controller-affect-flow: enable
Total Controllers: 1

Controller                : ssl:10.10.33.239:6695
--------------------------------------------------
status                    : online
online-time               : 0d 0h 5m 10s
role                      : other
mgmt-if                   : yes
bind_ip                   : none
max_backoff(sec)          : 8
inactivity_probe(sec)     : 5

Switch # show openflow ssl-key
private_key          : flash:/sc-privkey.pem
certificate          : flash:/sc-cert.pem
ca_cert              : flash:/controllerca_cacert.pem
```

11.3 Hybrid inband port management

11.3.1 Overview

Pure openflow switch doesn't support inband management because both of physical port and logical port don't support set the ip address. In Hybrid switch, users can set ip address on physical port, vlan interface, loopback logical port. After in combination with OSPF feature, Hybrid switch support inband management.

NOTE When use layer 3 interface for inband management, notice that if flow match key include 'in_port' field, it is better to match exact business packet to mismatch management packet.

11.3.2 Configuration

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enter the interface configure mode and enable openflow

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# openflow enable
```

step 3 Set the interface mode as layer 3 interface and configure the ip address

```
Switch(config-if-eth-0-1)# no switchport
Switch(config-if-eth-0-1)# ip address 1.1.1.1/24
Switch(config-if-eth-0-1)# exit
```

step 4 Exit the configure mode

```
Switch(config-if-eth-0-2)# end
```

step 5 Validation

```
[user1@systest ~]$ telnet 1.1.1.1
Trying 1.1.1.1...
Connected to 1.1.1.1.
Escape character is '^]'.

Switch #
```

11.4 Configuring Hybrid Interface

11.4.1 Overview

The openflow is disable on all type interface by default and the interface will not join to bridge. If users set the openflow enable on interface, the interface can be join to bridge and can be used by bridge.

The types of interface that supported configurate openflow enable can be physical, tunnel(such as vxlan, nvgre, l2gre), linkagg interface.

It will not support the operatation of set openflow enable/disable directly in ovsdb, it only can be set used by system CLI.

11.4.2 Configuration

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enable openflow for physical interface

```
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# openflow enable
Switch(config-if-eth-0-1)# exit
```

step 3 Enable openflow for linkagg interface

```
Switch(config)# interface eth-0-2
Switch(config-if-eth-0-2)# static-channel-group 1
Switch(config-if-eth-0-2)# exit
Switch(config)# interface agg1
Switch(config-if-agg1)# openflow enable
Switch(config-if-agg1)# exit
```

step 4 Exit the configure mode

```
Switch(config)# end
```

step 5 Validation

```
Switch# ovs-ofctl show br0
OFPT_FEATURES_REPLY (xid=0x2): dpid:0000001e080bce01
n_tables:1, n_buffers:256
capabilities: FLOW_STATS TABLE_STATS PORT_STATS ARP_MATCH_IP
actions: OUTPUT SET_VLAN_VID SET_VLAN_PCP STRIP_VLAN SET_DL_SRC SET_DL_DST
SET_NW_SRC SET_NW_DST SET_NW_TOS SET_TP_SRC SET_TP_DST
 1(eth-0-1): addr:00:1e:08:0b:ce:1f
     config:     0
     state:      LINK_DOWN
     speed: 0 Mbps now, 0 Mbps max
 1221(agg1): addr:08:63:19:6c:e1:02
         config:     0
         state:      0
         speed: 0 Mbps now, 0 Mbps max
OFPT_GET_CONFIG_REPLY (xid=0x4): frags=normal miss_send_len=0
```


11.5 Configuring Hybrid openflow ipv4/ipv6 tranform flow

11.5.1 Overview

Hybrid Switch support V4/V6 tranform function when flow format satisfy follow format:

IPV4 convert into IPV6 flow format:

- match key: ip+tcp/udp
- action: macda+ipv6_da+ipv6_sa
- optional action: macsa/push_vlan+set_vlan/ip_dscp/ip_ttl/flow_label

IPV6 convert into IPV4 flow format:

- match key: ipv6+tcp6/udp6
- action: macda+ip_da+ip_sa
- optional action: macsa/push_vlan+set_vlan/ip_dscp/ip_ttl

V4/V6 tranform flow only support output to agg or pyhiscal port or logical ALL.

In Hybrid default profile, support IPV4 convert into IPV6 function; in Hybrid ipv6 profile, both IPV4 convert into Ipv6 and IPV6 convert into IPV4 function are supported.

11.5.2 Configuration

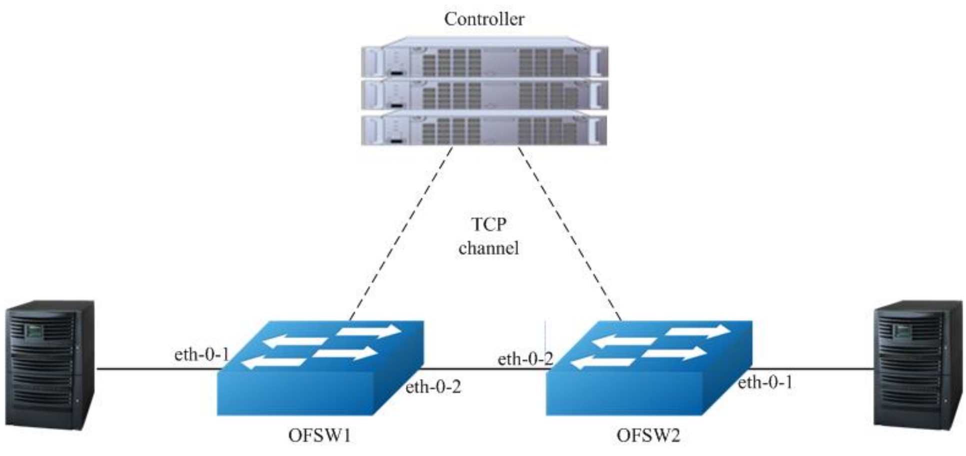

Figure 11-5 OpenFlow network topology

Configuring ipv4 tranform into ipv6 flow

Ipv4 convert into ipv6, the L2 and L3(ipv4) header will be striped, and then push new L2(use output port mac for new macsa if not specify macsa in flow action) and L3(ipv6) header, field after ip header should not changed.

step 1 Create a flow

```
Switch# ovs-ofctl add-flow br0 "ip,tcp,action=set_field:00:00:00:00:00:02-
>eth_dst,set_field:00:00:00:00:00:01->eth_src,push_vlan:0x8100,set_field:4199-
>vlan_vid,set_field:300::1->ipv6_dst,set_field:100::1->ipv6_src,output:2" -O
openflow13
```

step 2 Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=217.335s, table=0, n_packets=0, n_bytes=0, tcp
actions=set_field:00:00:00:00:00:02->eth_dst,set_field:00:00:00:00:00:01-
>eth_src,push_vlan:0x8100,set_field:4199->vlan_vid,set_field:300::1-
>ipv6_dst,set_field:100::1->ipv6_src,output:2
```

Configuring ipv6 tranform into ipv4 flow

Ipv6 convert into ipv4, the L2 and L3(ipv6) header will be striped, and then push new L2(use output port mac for packet macsa if not specify macsa in flow action) and L3(ipv4) header, field after ip header should not changed.

step 1 Create a flow

```
Switch# ovs-ofctl add-flow br0 "ipv6,tcp6,action=set_field:00:00:00:00:00:02-
>eth_dst,set_field:10.1.1.1->ip_dst,set_field:20.1.1.1-
>ip_src,set_nw_ttl:40,output:2" -O openflow13
```

step 2 Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=7.113s, table=0, n_packets=0, n_bytes=0, tcp6
actions=set_field:00:00:00:00:00:02->eth_dst,set_field:10.1.1.1-
>ip_dst,set_field:20.1.1.1->ip_src,mod_nw_ttl:40,output:2
```


11.6 Configuring Hybrid openflow table

11.6.1 Overview

Hybrid switch support comprehensive flow matching fields and actions, which can help user build flexibile SDN applications.

Hybrid580 at most support 4000 flows, Hybrid350 at most support 1200 flows.

Here are the match fields supported by Hybrid switch.

Match Fields supported	Notes
Ingress Port	
Eth SRC Address	Support mask
Eth DST Address	Support mask
Eth type	
VLAN id	Support mask(vlan_tci form)
VLAN PCP	
Inner VLAN id	Support mask(inner_vlan_tci form)
Inner VLAN PCP	
IPv4 DSCP	
IPv4 ECN	
IPv4 Protocol	
IPv4 SRC Address	Support mask
IPv4 DST Address	Support mask
L4 SRC Port	
L4 DST Port	
ICMP Type	
ICMP Code	
ARP OpCode	

ARP SPA	Support mask
ARP TPA	Support mask
ARP SHA	Support mask
ARP THA	Support mask
IPv6 DSCP	
IPv6 ECN	
IPv6 SRC Address	Support mask
IPv6 DST Address	Support mask
IPv6 Flow Lable	
IPv6 L4 SRC Port	
IPv6 L4 DST Port	
ICMP6 Type	
ICMP6 Code	
Tunnel Id	
Mpls_label	
mpls_label_num	Nicira extended match field 49
mpls_label0	Nicira extended match field 50
mpls_label1	Nicira extended match field 51
mpls_label2	Nicira extended match field 52
oam_session	Nicira extended match field 57
udf_id	Nicira extended match field 59
udf	Nicira extended match field 60 (Support mask)

Here are the actions supported by Hybrid switch.

Actions supported	Notes
Output	Physical Ports/ Logical Ports/ Reserved Ports, details see ourspec
Meter	
Set-queue	
Drop	
Group	
Push/Pop tags	Vlan/mpls
Set Field	Eth_dst/Eth_src/Eth_type/Vlan_id/Vlan_pcp/Ip_dscp/Ip_ecn /Ip_protocol/Ip_src/Ip_dst /Tcp_src/Tcp_dst/Udp_src/Udp_dst/Sctp_src/Sctp_dst /IcmpV4_type/IcmpV4_code/Arp_op/Arp_spa/Arp_tpa/Arp_sha/Arp_tha /Mpls_label/Mpls_tc/Tunnel_id
Set Field(IPv6)	Ipv6_dscp/Ipv6_ecn/Ipv6_src/Ipv6_dst/Ipv6_label /Tcp6_src/Tcp6_dst/Udp6_src/Udp6_dst /Sctp6_src/Sctp6_dst /IcmpV6_type/IcmpV6_code
Change TTL	Set MPLS TTL/ Set IP TTL/Decrement IP TTL
Extended-Push_L2	Nicira extended action subtype 25 （NX_VENDOR_ID 0x00002320）
Extended-Pop L2	Nicira extended action subtype 26 （NX_VENDOR_ID 0x00002320）
Extended-Pop_all_mpls	Nicira extended action subtype 28 （NX_VENDOR_ID 0x00002320）

Extended-oam_inlabel	Nicira extended action subtype 10001 （NX_VENDOR_ID 0x00002320）
Extended-oam_poplabel	Nicira extended action subtype 10002 （NX_VENDOR_ID 0x00002320）
Actions supported	Notes

11.6.2 Configuration

Add Different Fields To Match Openflow Table

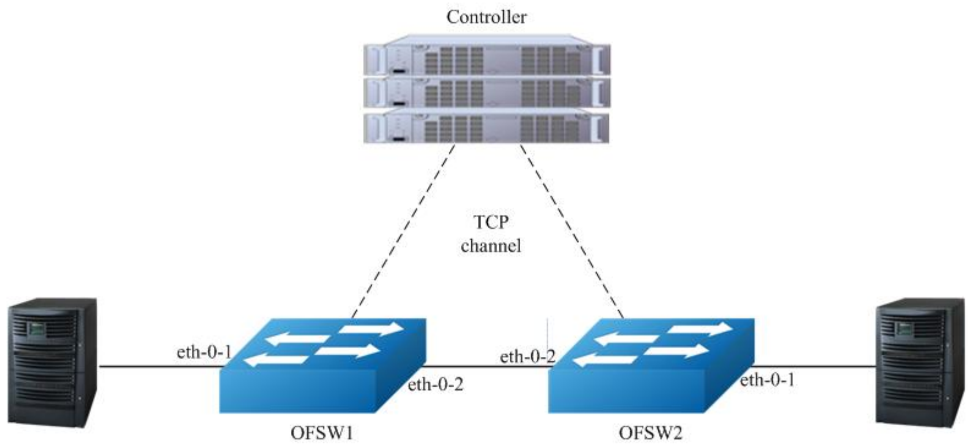

Figure 11-6 Hybrid Openflow Table topology

InPort Match Configuration and Validation

Configuration:

```
Switch# ovs-ofctl add-flow br0 in_port=1,actions=output:2 -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=164.933s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=output:2
```

Mac Address Match Configuration and Validation

Configuration:

```
Switch# ovs-ofctl add-flow br0
dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:02,actions=output:2 -O openflow13
```


Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=4.535s, table=0, n_packets=0, n_bytes=0,
dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:02 actions=output:2
```

Mask Address Match Configuration and Validation

Configuration:

```
Switch# ovs-ofctl add-flow br0
dl_src=00:00:00:00:01:00/00:00:00:00:ff:00,dl_dst=00:00:00:00:02:00/00:00:00:00:ff:
00,actions=output:2 -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=5.219s, table=0, n_packets=0, n_bytes=0,
dl_src=00:00:00:00:01:00/00:00:00:00:ff:00,dl_dst=00:00:00:00:02:00/00:00:00:00:ff:
00 actions=output:2
```

Ether_type Match Configuration and Validation

Configuration:

```
Switch# ovs-ofctl add-flow br0 dl_type=0x8847,actions=output:2 -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=6.421s, table=0, n_packets=0, n_bytes=0, mpls
actions=output:2
```

Vlan id/pcp Match Configuration and Validation

Configuration:

```
Switch# ovs-ofctl add-flow br0 dl_vlan=100,dl_vlan_pcp=3,actions=output:2 -O
openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=4.385s, table=0, n_packets=0, n_bytes=0,
dl_vlan=100,dl_vlan_pcp=3 actions=output:2
```


Vlan_tci Match Configuration and Validation

Match the Packet with Stag Configuration:

```
Switch# ovs-ofctl add-flow br0 vlan_tci=0x1000/0x1000,actions=2 -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=4.385s, table=0, n_packets=0, n_bytes=0,
vlan_tci=0x1000/0x1000 actions=output:2
```

Match the Packet without Stag Configuration:

```
Switch# ovs-ofctl add-flow br0 vlan_tci=0x0000/0x1000,actions=2 -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=12.534s, table=0, n_packets=0, n_bytes=0,
vlan_tci=0x0000/0x1000 actions=output:2
```

Inner Vlan id/pcp Match Configuration and Validation

Configuration:

```
Switch# ovs-ofctl add-flow br0
inner_dl_vlan=100,inner_dl_vlan_pcp=3,actions=output:2 -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=4.385s, table=0, n_packets=0, n_bytes=0,
inner_dl_vlan=100,inner_dl_vlan_pcp=3 actions=output:2
```

Inner Vlan_tci Match Configuration and Validation

Match the Packet with Ctag Configuration:

```
Switch# ovs-ofctl add-flow br0 inner_vlan_tci=0x1000/0x1000,actions=2 -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=12.534s, table=0, n_packets=0, n_bytes=0,
inner_vlan_tci=0x1000/0x1000 actions=output:2
```

Match the Packet without Ctag Configuration:


```
Switch# ovs-ofctl add-flow br0 inner_vlan_tci=0x0000/0x1000,actions=2 -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=12.534s, table=0, n_packets=0, n_bytes=0,
inner_vlan_tci=0x0000/0x1000 actions=output:2
```

Ip_dscp Match Configuration and Validation

Configuration:

```
Switch# ovs-ofctl add-flow br0 ip,ip_dscp=1,actions=output:2 -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=5.609s, table=0, n_packets=0, n_bytes=0, ip,nw_tos=4
actions=output:2
```

Ip_ecn Match Configuration and Validation

Configuration:

```
Switch# ovs-ofctl add-flow br0 ip,ip_ecn=1,actions=output:2 -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=7.275s, table=0, n_packets=0, n_bytes=0, ip,nw_ecn=1
actions=output:2
```

Ip Match Configuration and Validation

Configuration:

```
Switch# ovs-ofctl add-flow br0 ip,nw_src=1.1.1.1,nw_dst=2.2.2.2,actions=output:2 -O
openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=4.555s, table=0, n_packets=0, n_bytes=0,
ip,nw_src=1.1.1.1,nw_dst=2.2.2.2 actions=output:2
```

Ip Mask Match Configuration and Validation

Configuration:


```
Switch# ovs-ofctl add-flow br0
ip,nw_src=1.1.1.0/24,nw_dst=2.2.2.0/24,actions=output:2 -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=5.182s, table=0, n_packets=0, n_bytes=0,
ip,nw_src=1.1.1.0/24,nw_dst=2.2.2.0/24 actions=output:2
```

Layer 4 Port Match Configuration and Validation

TCP port match configuration:

```
Switch# ovs-ofctl add-flow br0 ip,tcp,tcp_src=1000,tcp_dst=2000,actions=output:2 -O
openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=1.689s, table=0, n_packets=0, n_bytes=0,
tcp,tp_src=1000,tp_dst=2000 actions=output:2
```

UDP port match configuration:

```
Switch# ovs-ofctl add-flow br0 ip,udp,udp_src=1000,udp_dst=2000,actions=output:2 -O
openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=6.523s, table=0, n_packets=0, n_bytes=0,
udp,tp_src=1000,tp_dst=2000 actions=output:2
```

SCTP port match configuration:

```
Switch# ovs-ofctl add-flow br0 ip,sctp,sctp_src=1000,sctp_dst=2000,actions=output:2
-O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=1.531s, table=0, n_packets=0, n_bytes=0,
sctp,tp_src=1000,tp_dst=2000 actions=output:2
```

Icmp Match Configuration and Validation

Configuration:

```
Switch# ovs-ofctl add-flow br0 ip,icmp,icmp_type=8,icmp_code=0,actions=output:2 -O
openflow13
```


Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=2.191s, table=0, n_packets=0, n_bytes=0,
icmp,icmp_type=8,icmp_code=0 actions=output:2
```

Arp Match Configuration and Validation

Match the Arp Request of Packet Configuration:

```
Switch# ovs-ofctl add-flow br0
arp,arp_op=1,arp_tpa=1.1.1.1,arp_spa=2.2.2.2,actions=output:2 -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=2.184s, table=0, n_packets=0, n_bytes=0,
arp,arp_spa=2.2.2.2,arp_tpa=1.1.1.1,arp_op=1 actions=output:2
```

Match the Arp Request Mask of Packet:

```
Switch# ovs-ofctl add-flow br0
arp,arp_op=1,arp_tpa=1.1.1.0/24,arp_spa=2.2.2.0/24,actions=output:2 -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=1.499s, table=0, n_packets=0, n_bytes=0,
arp,arp_spa=2.2.2.0/24,arp_tpa=1.1.1.0/24,arp_op=1 actions=output:2
```

Udf Match Configuration and Validation

System doesn't support udf flow only udf enable globally. Before enable or disable udf function, should delete all flow firstly in system.

```
Switch(config)# openflow udf enable
Switch(config)# no openflow udf enable
```

NOTE

Something notice for udf usage:

- For hybrid580:udf function is conflict with tunnel(vxlan/gre/nvgre). If udf enable and tunnel decap flow isn't supported.
- For hybrid550:when udf function enable, the maximum flow number will cut in halt.

- udf match field: udf_id, udf.
- udf_id:range is <0-15>
- udf_id is related with system udf parser module's udf_id, which should be created firstly when referred.
- udf:at most 16 bytes supported.
- For hybrid580:previous 4 bytes in udf field is valid, from left to right, each byte oppsite to offset0/offset1/offset2/offset3.
- For hybrid550:support 16 bytes, from left to right, each 4 bytes oppsite to offset0/offset1/offset2/offset3.

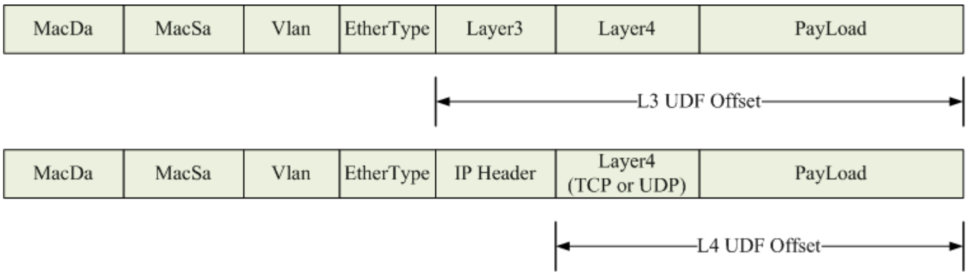

Figure 11-7 L3/L4 offset type in packet

L3 UDF Configuration:

```
DUT2(config)# udf 1 offset-type l3-header
DUT2(config-udf-1)# match ether-type 0x0800 0x0
DUT2(config-udf-1)# offset offset0 20 offset1 21 offset2 22 offset3 23
```

Validation:

```
DUT2# show udf
Udf Global Information:
  Offset Unit : 1 Bytes

Udf Index 1
  Udf Type : l3 header
  Udf Match-Field:
    ether-type 0x800 0x0
  Offset : 20|21|22|23
```

L4 UDF Configuration:

```
DUT2(config)# udf 2 offset-type l4-header
DUT2(config-udf-2)# match ip-protocol tcp
DUT2(config-udf-2)# offset offset0 30 offset1 31 offset2 32 offset3 33
```

Validation:

```
DUT2# show udf
Udf Global Information:
```



```
  Offset Unit : 1 Bytes

Udf Index 2
  Udf Type : l4 header
  Udf Match-Field:
    ip-protocol tcp
  Offset : 30|31|32|33
```

UDF Match Flow Configuration:

```
Switch# ovs-ofctl add-flow br0 udf_id=1,udf=12345678/ffffffff,actions=output:2 -O
openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
  cookie=0x0, duration=6.684s, table=0, n_packets=6, n_bytes=494,
udf_id=1,udf=12345678/ffffffff actions=output:2
```

Tunnel/Mpls Configuration

Configure the flow of Tunnel/Mpls, please refer to the Tunnel/Mpls module.

Add Different Fields To Edit Openflow Table Configurations

Output Configuration and Validation

Configuration of flow table forward to interface 2:

```
Switch# ovs-ofctl add-flow br0 in_port=1,actions=output:2 -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=3.181s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=output:2
```

Configuration of flow table forward to all logical interfaces:

```
Switch# ovs-ofctl add-flow br0 in_port=1,actions=output:All -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=4.719s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=ALL
```

Configuration of flow table forward to controller:


```
Switch# ovs-ofctl add-flow br0 in_port=1,actions=output:controller -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=2.056s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=CONTROLLER:65535
```

Configuration of flow table forward to in_port:

```
Switch# ovs-ofctl add-flow br0 in_port=1,actions=output:in_port -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=4.590s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=IN_PORT
```

Configuration of flow table forward to normal:

```
Switch# ovs-ofctl add-flow br0 in_port=1,actions=output:normal -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=1.539s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=NORMAL
```

Add/Strip Vlan Configuration and Validation

Configure the flow table to edit outer vlan tag, real vlan_id=set_field_vlan_id - 4096.

```
Switch# ovs-ofctl add-flow br0 "ip,dl_vlan=100,actions=set_field:4296-
>vlan_vid,output:2" -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=4.623s, table=0, n_packets=0, n_bytes=0, ip,dl_vlan=100
actions=push_vlan:0x8100,set_field:4196->vlan_vid,output:2
```

Configure the flow table to push one layer vlan tag, real vlan_id=set_field_vlan_id - 4096.

```
Switch# ovs-ofctl add-flow br0 "ip,actions=push_vlan:0x8100,set_field:4196-
>vlan_vid,output:2" -O openflow13
```

Validation:


```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=4.623s, table=0, n_packets=0, n_bytes=0, ip
actions=push_vlan:0x8100,set_field:4196->vlan_vid,output:2
```

Configure the flow table to pop outer vlan tag.

```
Switch# ovs-ofctl add-flow br0 "ip,dl_vlan=100,actions=pop_vlan,output:2" -O
openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=4.623s, table=0, n_packets=0, n_bytes=0, ip,dl_vlan=100
actions=pop_vlan,output:2
```

Configure the flow table to pop outer vlan tag and edit inner vlan tag, real vlan_id=set_field_vlan_id - 4096.

```
Switch# ovs-ofctl add-flow br0
"ip,dl_vlan=100,inner_dl_vlan=200,actions=pop_vlan,set_field:4196-
>vlan_vid,output:2" -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=4.623s, table=0, n_packets=0, n_bytes=0,
ip,dl_vlan=100,inner_dl_vlan=200 actions=pop_vlan,set_field:4196->vlan_vid,output:2
```

Configure the flow table to push two layer vlan tag, real vlan_id=set_field_vlan_id - 4096.

```
Switch# ovs-ofctl add-flow br0 "ip,actions=push_vlan:0x8100,set_field:4196-
>vlan_vid,push_vlan:0x8100,set_field:4296->vlan_vid output:2" -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=4.623s, table=0, n_packets=0, n_bytes=0, ip
actions=push_vlan:0x8100,set_field:4196->vlan_vid,push_vlan:0x8100,set_field:4296-
>vlan_vid output:2
```

Configure the flow table to pop two layer vlan tag.

```
Switch# ovs-ofctl add-flow br0 "ip,
dl_vlan=100,inner_dl_vlan=200actions=pop_vlan,pop_vlan,output:2" -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=4.623s, table=0, n_packets=0, n_bytes=0,
ip,dl_vlan=100,inner_dl_vlan=200 actions=pop_vlan,pop_vlan,output:2
```


Source/Dest MAC Edit Configuration and Validation

Configuration:

```
Switch# ovs-ofctl add-flow br0 "ip,actions=set_field:00:00:00:00:00:02-
>eth_dst,set_field:00:00:00:00:00:01->eth_src,output:2" -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=1.979s, table=0, n_packets=0, n_bytes=0, ip
actions=set_field:00:00:00:00:00:02->eth_dst,set_field:00:00:00:00:00:01-
>eth_src,output:2
```

Ip_dscp/Ip_ecn Edit Configuration and Validation

Configuration:

```
Switch# ovs-ofctl add-flow br0 "ip,actions=set_field:4->ip_dscp,set_field:1-
>ip_ecn,output:2" -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=1.672s, table=0, n_packets=0, n_bytes=0, ip
actions=set_field:4->ip_dscp,set_field:1->nw_ecn,output:2
```

Ip_ttl Edit Configuration and Validation

Configuration:

```
Switch# ovs-ofctl add-flow br0 "ip,actions=set_nw_ttl:64,output:2" -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=13.679s, table=0, n_packets=0, n_bytes=0, ip
actions=mod_nw_ttl:64,output:2
```

ip_src/ip_dst Edit Configuration and Validation

Configuration:

```
Switch# ovs-ofctl add-flow br0 "ip,actions=set_field:1.1.1.1-
>ip_src,set_field:2.2.2.2->ip_dst,output:2" -O openflow13
```

Validation:


```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=4.864s, table=0, n_packets=0, n_bytes=0, ip
actions=set_field:1.1.1.1->ip_src,set_field:2.2.2.2->ip_dst,output:2
```

配置编辑四层端口

Layer 4 port Edit Configuration and Validation

Configuration of Edit tcp port number:

```
Switch# ovs-ofctl add-flow br0 "ip,tcp,actions=set_field:100-
>tcp_dst,set_field:200->tcp_src,output:2" -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=1.502s, table=0, n_packets=0, n_bytes=0, tcp
actions=set_field:100->tcp_dst,set_field:200->tcp_src,output:2
```

Configuration of Edit udp port number:

```
Switch# ovs-ofctl add-flow br0 "ip,udp,actions=set_field:100-
>udp_dst,set_field:200->udp_src,output:2" -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=1.307s, table=0, n_packets=0, n_bytes=0, udp
actions=set_field:100->udp_dst,set_field:200->udp_src,output:2
```

Configuration of Edit sctp port number:

```
Switch# ovs-ofctl add-flow br0 "ip,sctp,actions=set_field:100-
>sctp_dst,set_field:200->sctp_src,output:2" -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=1.456s, table=0, n_packets=0, n_bytes=0, sctp
actions=set_field:100->sctp_dst,set_field:200->sctp_src,output:2
```

Arp Edit Configuration and Validation

Configuration:

```
Switch# ovs-ofctl add-flow br0
"arp,arp_op=1,arp_spa=1.1.1.1,arp_tpa=1.1.1.254,actions=set_field:00:00:00:00:00:01
->arp_tha,set_field:00:00:00:00:00:02->arp_sha,set_field:2-
```



```
>arp_op,set_field:1.1.1.1->arp_tpa,set_field:1.1.1.254->arp_spa,output:1" -O
openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=3.936s, table=0, n_packets=0, n_bytes=0,
arp,arp_spa=1.1.1.1,arp_tpa=1.1.1.254,arp_op=1 actions=set_field:00:00:00:00:00:01-
>arp_tha,set_field:00:00:00:00:00:02->arp_sha,set_field:2-
>arp_op,set_field:1.1.1.1->arp_tpa,set_field:1.1.1.254->arp_spa,output:1
```

strip_header action for overlay packets Configuration and Validation

Configuration:

```
Switch# ovs-ofctl add-flow br0
udp,tp_dst=4789,tun_id=100/0xfffff0,actions=strip_header:16,output:2 -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=15.555s, table=0, n_packets=0, n_bytes=0,
udp,tun_id=0x60/0xfffff0,tp_dst=4789 actions=strip_header:16,output:2
```

Update The Existed Flow Table and Validation

Update Fuzzy Flow Table

Add flow:

```
Switch# ovs-ofctl add-flow br0 "in_port=1,ip,actions=output:2" -O   openflow13
Switch# ovs-ofctl add-flow br0 "ip,actions=output:2" -O openflow13
```

Update flow:

```
Switch# ovs-ofctl mod-flows br0 "ip,actions=output:3" -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=43.059s, table=0, n_packets=0, n_bytes=0, ip actions=output:3
 cookie=0x0, duration=62.028s, table=0, n_packets=0, n_bytes=0, ip,in_port=1
actions=output:3
```

Update Exact Flow Table

Add flow:


```
Switch# ovs-ofctl add-flow br0 "in_port=1,ip,actions=output:2" -O openflow13
Switch# ovs-ofctl add-flow br0 "ip,actions=output:2" -O openflow13
```

Update flow:

```
Switch# ovs-ofctl mod-flows br0 " ip,actions=output:3" --strict -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=43.059s, table=0, n_packets=0, n_bytes=0, ip actions=output:3
 cookie=0x0, duration=62.028s, table=0, n_packets=0, n_bytes=0, ip,in_port=1
actions=output:2
```

Delete The Existed Flow Table and Validation

Delete Fuzzy Match Flow Table

Add flow:

```
Switch# ovs-ofctl add-flow br0 "in_port=1,ip,actions=output:2" -O openflow13
Switch# ovs-ofctl add-flow br0 "ip,actions=output:2" -O openflow13
```

Delete flow:

```
Switch# ovs-ofctl del-flows br0 "ip" -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
```

Delete Exact Match Flow Table

Add flow:

```
Switch# ovs-ofctl add-flow br0 "in_port=1,ip,actions=output:2" -O openflow13
Switch# ovs-ofctl add-flow br0 "ip,actions=output:2" -O openflow13
```

Delete flow:

```
Switch# ovs-ofctl del-flows br0 "ip" --strict -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=10.755s, table=0, n_packets=0, n_bytes=0, ip,in_port=1
actions=output:2
```


Flow Table Statistics Configuration and Validation

Configuration:

```
Switch# ovs-ofctl add-flow br0 "in_port=1,ip,actions=output:2" -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-tables br0 -O openflow13
OFPST_TABLE reply (OF1.3) (xid=0x2): 1 tables
  0: active=1, lookup=0, matched=0
```

Flow Table Priority Configuration and Validation

Flow table is distinguish the priority of diferent flow tables priority. For example, the priority of flow table A is higher than flow table B, A matchs ip address, B matchs tcp, then a packet with tcp will match the flow table A, because tcp packet belongs to ip packet and the priority of A is higher, so matchs A. If priority of A is equals to B and the packet matchs A and B, the packet will forward with the first one.

Configuration:

```
Switch# ovs-ofctl add-flow br0 priority=65535,ip,actions=2 -O openflow13
Switch# ovs-ofctl add-flow br0 priority=1,ip,tcp,actions=3 -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=4.310s, table=0, n_packets=0, n_bytes=0, priority=1,tcp
actions=output:3
 cookie=0x0, duration=52.887s, table=0, n_packets=0, n_bytes=0, priority=65535,ip
actions=output:2
```

Flow Table Timeout Configuration and Validation

Openflow protocol defines two timeout mechanism, idle_timeout and hard_timeout. idle_timeout means the time that from the packet match the table to present, hard_timeout means the time that from the flow table set to chip to present.

Configuration:

```
Switch# ovs-ofctl add-flow br0
in_port=1,idle_timeout=10,hard_timeout=100,actions=output:2 -O openflow13
```

Validation:


```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=8.920s, table=0, n_packets=0, n_bytes=0, idle_timeout=10,
hard_timeout=100, in_port=1 actions=output:2
```

Flow Table FLAG field Configuration and Validation

Hybrid flow-mod support three flags:

The first is SEND_FLOW_REM, the flag can report a message to controller actively if the flow is deleted.

The second is CHECK_OVERLAP, the flag will check the other flows, if other flows have conflicts with the flow table, return error.

The Third is RESET_COUNTS, the flag will clear the flow table statistics if the flows have modified.

SEND_FLOW_REM configuration

Configuration:

```
Switch# ovs-ofctl add-flow br0 "send_flow_rem,in_port=1,actions=2" -O openflow13
```

Validation:

```
Switch# ovs-ofctl snoop br0
OFPT_ECHO_REQUEST (OF1.3) (xid=0x0): 0 bytes of payload
OFPT_ECHO_REPLY (OF1.3) (xid=0x0): 0 bytes of payload
OFPT_FLOW_REMOVED (OF1.3) (xid=0x0): in_port=1 reason=delete table_id=0
duration46.567s idle0 pkts0 bytes0
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
```

Check_Overlap configuration

Configuration:

```
Switch# ovs-ofctl add-flow br0 "in_port=1,actions=2" -O openflow13
```

Validation:

```
Switch# ovs-ofctl add-flow br0 "check_overlap,in_port=1,ip,actions=2" -O openflow13
OFPT_ERROR (OF1.3) (xid=0x2): OFPFMFC_OVERLAP
OFPT_FLOW_MOD (OF1.3) (xid=0x2):
(***truncated to 64 bytes from 96***)
00000000  04 0e 00 60 00 00 00 02-00 00 00 00 00 00 00 00 |...`............|
00000010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 80 00 |................|
00000020  ff ff ff ff ff ff ff ff-ff ff ff ff 00 02 00 00 |................|
00000030  00 01 00 12 80 00 00 04-00 00 00 01 80 00 0a 02 |................|
```



```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=57.497s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=output:2
```

Reset_Counts configuration

Configuration:

```
Switch# ovs-ofctl add-flow br0 "in_port=1,actions=2" -O openflow13
```

Validation:

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=8.879s, table=0, n_packets=1, n_bytes=128, in_port=1
actions=output:2
Switch# ovs-ofctl mod-flows br0 "reset_counts,in_port=1,actions=2" -O openflow13
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=40.406s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=output:2
```

11.7 Configuring Hybrid Openflow Group

11.7.1 Overview

The abstraction group is introduced in OpenFlow 1.1 specification firstly. Flow entries can point to a group, which enables OpenFlow to represent additional methods of forwarding. Because of the ASIC limitation, not all buckets in a group entry will be installed to ASIC. The system will install buckets at most as possible to ASIC.

Notice：

- Hybrid system just support meter action and push/pop/set(vlan id and pcp) vlan action before group action in flow entry.
- Hybrid system just support at most two tier group chained, and just support all group nested all group or ff group. For example: all->all, all->ff, all->all+ff...

11.7.2 Configuration

All Type Group

This type group is a sample broadcast group, if packets match this group flow table, the packets will copy to every bucket and forward. The system support 160 groups of type all with statistics enabled, and maximum bucket is 288 for per all group.

step 1 Configuration of All type Group

```
Switch# ovs-ofctl add-group br0
group_id=1,type=all,bucket=output:2,bucket=mod_vlan_vid:10,output:3,bucket=mod_dl_s
rc:01:02:03:04:05:06,output:4 -O openflow13
```

step 2 Configuration of a flow point to group

```
Switch# ovs-ofctl add-flow br0 in_port=1,actions=group:1 -O openflow13
```

step3 Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=2.127s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=group:1
Switch# ovs-ofctl dump-groups br0 -O openflow13
OFPST_GROUP_DESC reply (OF1.3) (xid=0x2):
group_id=1,type=all,bucket=output:2,bucket=push_vlan:0x8100,set_field:4106-
>vlan_vid,output:3,bucket=set_field:01:02:03:04:05:06->eth_src,output:4
```

Select Type Group

This type group is used for load balance between per bucket member, the hash keys are fixable configurable chosen. The system support 63 groups of type select with statistics enabled, and only support 16 buckets for per select group. Select group bucket can only contain one physical, linkagg or tunnel output port.

step1 Configuration of select type Group

```
Switch# ovs-ofctl add-group br0
group_id=1,type=select,bucket=output:1,bucket=output:2,bucket=output:3 -O
openflow13
```


step2 Configuration of a flow point to group

```
Switch# ovs-ofctl add-flow br0 in_port=1,actions=group:1 -O openflow13
```

step3 Validation

```
Switch# ovs-ofctl dump-groups br0 -O openflow13
OFPST_GROUP_DESC reply (OF1.3) (xid=0x2):
 group_id=1,type=select,bucket=output:1,bucket=output:2,bucket=output:3
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=40.063s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=group:1
```

step 4 Select Group key(Optional)

```
Switch# configure terminal
Switch(config)# ecmp hash-field-select macsa macda inner-macsa inner-ipsa
```

step5 Validation(Select Group key)

```
Switch# show ecmp information
ECMP load balance enable mode: Static
ECMP hash-field-select:
    macsa macda inner-macsa inner-ipsa
```

FAST-FAILOVER Type Group

Execute the first live bucket. Each action bucket is associated with a specific port that controls its liveness. The buckets are evaluated in the order defined by the group, and the first bucket which is associated with a live port is selected. This group type enables the switch to change forwarding without requiring a round trip to the controller. If no buckets are live, packets are dropped. This group type must implement a liveness mechanism. The system support 63 groups of type fast failover with statistics enabled, and only support 16 buckets for per fast failover group.

step1 Configuration of FF type Group

```
Switch# ovs-ofctl add-group br0
group_id=1,type=ff,bucket=watch_port=2,output:2,bucket=watch_port=3,output:3 -O
openflow13
```

step2 Configuration of a flow point to group

```
Switch# ovs-ofctl add-flow br0 in_port=1,actions=group:1 -O openflow13
```

step3 Validation

```
Switch# ovs-ofctl dump-groups br0 -O openflow13
OFPST_GROUP_DESC reply (OF1.3) (xid=0x2):
 group_id=1,type=ff,bucket=watch_port:2,output:2,bucket=watch_port:3,output:3
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=13.710s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=group:1
```

Indirect Type Group

Execute the one defined bucket in this group. This group supports only a single bucket. Allows multiple flow entries to point to a common group identifier, supporting faster, more efficient convergence (e.g. next hops for IP forwarding). This group type is effectively identical to an all group with one bucket. The system support 63 groups of type indirect with statistics enabled, and only a single bucket for per indirect group.

step1 Configuration of indirect type Group

```
Switch# ovs-ofctl add-group br0 group_id=1,type=indirect,bucket=output:2 -O
openflow13
```

step2 Configuration of a flow point group

```
Switch# ovs-ofctl add-flow br0 in_port=1,actions=group:1 -O openflow13
```

step3 Validation

```
Switch# ovs-ofctl dump-groups br0 -O openflow13
OFPST_GROUP_DESC reply (OF1.3) (xid=0x2):
 group_id=1,type=indirect,bucket=output:2
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=12.407s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=group:1
```


Group Chain Configuration

Hybrid system support group chain: all or ff group can be nested in a all type group as a bucket member, while can coexist with other general edit bucket.

step 1 Configuration of All type Group and FF type group

```
Switch# ovs-ofctl add-group br0 group_id=1,type=all,bucket=output:2 -O openflow13
Switch# ovs-ofctl add-group br0
group_id=2,type=ff,bucket=watch_port:3,output:3,bucket=watch_port:4,output:4 -O
openflow13
```

step 2 Configuration of All type group refer to group1 and group2

```
Switch# ovs-ofctl add-group br0
group_id=3,type=all,bucket=group:1,bucket=group:2,bucket=output:5 -O openflow13
```

step 3 Configuration of a flow point to group3

```
Switch# ovs-ofctl add-flow br0 in_port=1,actions=group:3
```

step 4 Validation

```
Switch# ovs-ofctl dump-groups br0 -O openflow13
OFPST_GROUP_DESC reply (OF1.3) (xid=0x2):
 group_id=1,type=all,bucket=output:2
 group_id=2,type=ff,bucket=watch_port:3,output:3,bucket=watch_port:4,output:4
 group_id=3,type=all,bucket=group:1,bucket=group:2,bucket=output:5
jinluser129# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=8.996s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=group:3
```

Dump Group Statistic

```
Switch# ovs-ofctl dump-group-stats br0 -O openflow13
OFPST_GROUP reply (OF1.3) (xid=0x2):
 group_id=1,duration=141.534s,ref_count=1,packet_count=0,byte_count=0
```

Update Group

step1 Configuration of Group

```
Switch# ovs-ofctl add-group br0 group_id=1,type=all,bucket=output:2 -O openflow13
Switch# ovs-ofctl add-group br0
group_id=2,type=select,bucket=output:2,bucket=output:3 -O openflow13
```


step2 Configuration of updating group

```
Switch# ovs-ofctl add-group br0 group_id=1,type=all,bucket=output:2 -O openflow13
Switch# ovs-ofctl add-group br0
group_id=2,type=select,bucket=output:2,bucket=output:3 -O openflow13
```

step3 Validation

```
Switch# ovs-ofctl dump-groups br0 -O openflow13
OFPST_GROUP_DESC reply (OF1.3) (xid=0x2):
 group_id=1,type=all,bucket=output:3
 group_id=2,type=select,bucket=output:2,bucket=output:3
```

Delete Group

step1 Configuration of Group

```
Switch# ovs-ofctl add-group br0 group_id=1,type=all,bucket=output:2 -O openflow13
Switch# ovs-ofctl add-group br0
group_id=2,type=select,bucket=output:2,bucket=output:3 -O openflow13
```

step2 Configuration of deleting group

```
Switch# ovs-ofctl del-groups br0 group_id=1 -O openflow13
```

step3 Validation

```
Switch# ovs-ofctl dump-groups br0 -O openflow13
OFPST_GROUP_DESC reply (OF1.3) (xid=0x2):
 group_id=2,type=select,bucket=output:2,bucket=output:3
```

11.8 Configuring Hybrid Openflow Meter

11.8.1 Overview

A meter measures the rate of packets assigned to it and enables controlling the rate of those packets. Meters are attached directly to flow entries (as opposed to queues which are attached to ports). Any flow entry can specify a meter in its instructions set, the meter measures and controls the rate of the aggregate of all flow entries to which it is attached.

Each meter can have one band, specifying a rate, which is used to limit the flow-matched traffic. All measured traffic that exceeds the band rate should be dropped(only support drop). Meter statistics quality the traffic that input into the meter, and band statistics quality the traffic dropped due to exceeding band rate.

The system can support at most 1100 meters and every meter can count the statistics, Hybrid350 at most support 1300 Meters.

11.8.2 Configuration

Create Meter Configuring

In the following configuration, the system can create a meter entry. The valid meter id is from 1 to 1100(Hybrid350 1300), meter flag is kbps/burst/stats, and kbps must be configured, burst and stats are optional. Band type support drop only, and the valid rate is from 0 to 100000000(kbps), burst is from 0 to 52400(kbps). The burst is advised to set to 52400(kbps).

step 1 Configuration of Meter

```
Switch# ovs-ofctl add-meter br0
meter=1,kbps,stats,burst,band=type=drop,rate=1000,burst_size=52400 -O openflow13
```

step 2 Configuration of a flow point to meter

```
Switch# ovs-ofctl add-flow br0 in_port=1,actions=meter:1,output:2 -O openflow13
```

step3 Validation

```
Switch# ovs-ofctl dump-meters br0 -O openflow13
OFPST_METER_CONFIG reply (OF1.3) (xid=0x2):
meter=1 kbps burst stats bands=type=drop rate=1000 burst_size=52400
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=54.741s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=meter:1,output:2
```

Update Meter Configuring

Meter support updating, but the flag must include kbps.

step 1 Configuration of meter

```
Switch# ovs-ofctl add-meter br0
meter=1,kbps,stats,burst,band=type=drop,rate=1000,burst_size=52400 -O openflow13
```


step 2 Configuration of a flow point to meter

```
Switch# ovs-ofctl add-flow br0 in_port=1,actions=meter:1,output:2 -O openflow13
```

step 3 Configuration of updating meter

```
Switch# ovs-ofctl mod-meter br0 meter=1,kbps,band=type=drop,rate=2000 -O openflow13
```

step4 Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=151.588s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=meter:1,output:2
```

Delete Meter Configuring

step 1 Configuration of meter

```
Switch# ovs-ofctl add-meter br0
meter=1,kbps,stats,burst,band=type=drop,rate=1000,burst_size=52400 -O openflow13
```

step 2 Configuration of a flow point to meter

```
Switch# ovs-ofctl add-flow br0 in_port=1,actions=meter:1,output:2 -O openflow13
```

step 3 Configuration of deleting meter

```
Switch# ovs-ofctl del-meter br0 meter=1 -O openflow13
```

step4 Validation

```
Switch# ovs-ofctl dump-meters br0 -O openflow13
OFPST_METER_CONFIG reply (OF1.3) (xid=0x2):
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
```

step4 Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=151.588s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=meter:1,output:2
```


Dump Meter Statistic

step 1 Configuration of meter

```
Switch# ovs-ofctl add-meter br0
meter=1,kbps,stats,burst,band=type=drop,rate=1000,burst_size=52400 -O openflow13
```

step 2 Configuration of a flow point to meter

```
Switch# ovs-ofctl add-flow br0 in_port=1,actions=meter:1,output:2 -O openflow13
```

step 3 Check the statistic result of meter

```
Switch# ovs-ofctl meter-stats br0 -O openflow13
```

step4 Validation

The statistics of meter stats include two part, the front part is the statistics of all packet processed by the meter, the second part is the statistics of dropped by band-drop.

```
Switch# ovs-ofctl meter-stats br0 -O openflow13
OFPST_METER reply (OF1.3) (xid=0x2):
meter:1 flow_count:0 packet_in_count:0 byte_in_count:0 duration:9.454s
bands:packet_count:0 byte_count:0
```

11.9 Configuring Hybrid Openflow Tunnel

11.9.1 Overview

Hybrid system support three type of Tunnel, vxlan/l2gre/nvgre.

The tunnel ports should be named as: "l2gre1 ~ l2gre200", "nvgre1 ~ nvgre 200", "vxlan1 ~ vxlan 200". Each tunnel port has a openflow port number: 201~400 is for l2gre, 1301~1500 is for nvgre, 2201~2400 is for vxlan.

If an output port number in a flow is a tunnel port, such as "output:201", all packets that matching this flow should be encapsulated with a new GRE or VXLAN header, new IP header and new mac header, according to the configurations of the tunnel port. The destination IP address of the new IP header is the tunnel's remote IP address, and the source IP address is the local vtep IP address. The destination mac address of the new layer2 header is the tunnel's nexthop-mac address, and the source mac address is bind-port's mac address.

If an incoming packet matches a flow which its destination IP address matches the local vtep IP address, and source IP address matches tunnel's remote IP, this packet will be decapsulated. If the decapsulated packet matches a flow with in_port or in_port + tun_id, the packet will be processed according to this flow's actions.

The openflow tunnel port number at most support 500, the local vtep of tunnel at most support 4.

11.9.2 Configuration

Add Tunnel Logic Port

Configuring static Vxlan Tunnel

Configuration:

```
Switch# configure terminal
Switch(config)# interface vxlan1
Switch(config-if-vxlan1)# tunnel-source-ip 1.1.1.1
Switch(config-if-vxlan1)# tunnel-remote-ip 2.2.2.2
Switch(config-if-vxlan1)# tunnel-bind-static bind-port eth-0-2 nexthop-mac 0.0.1
bind-vlan 100
Switch(config-if-vxlan1)# openflow enable
```

Validation:

```
Switch# show openflow interface tunnel brief
The Maximum of tunnel ports is 500, currently 1 tunnel ports is valid.

Default tunnel_type of bind_port is vxlan, if you want to use other type of
tunnel, modify tunnel mode on the bind_port.

Vxlan source port is dynamic
Vxlan dest port is default: 4789

Decap mode: ipda + ipsa + vni [default]
----------------------------------------
index 1
    type:            vxlan
    name:            vxlan1
    port:            2201
    source-ip:       1.1.1.1
    remote_ip:       2.2.2.2
    link:            UP
    dynamic:         FALSE
    bind_port:       eth-0-2
    remote_mac:      00:00:00:00:00:01
    vlan_id:         100
----------------------------------------
```


Configuring dynamic Vxlan Tunnel

Configuration:

```
Switch# configure terminal
Switch(config)# interface vxlan1
Switch(config-if-vxlan1)# tunnel-source-ip 2.2.2.1
Switch(config-if-vxlan1)# tunnel-remote-ip 2.2.2.2
Switch(config-if-vxlan1)# openflow enable
```

Validation:

```
Switch# show openflow interface tunnel brief
The Maximum of tunnel ports is 500, currently 1 tunnel ports is valid.

Default tunnel_type of bind_port is vxlan, if you want to use other type of
tunnel, modify tunnel mode on the bind_port.

Vxlan source port is dynamic
Vxlan dest port is default: 4789

Decap mode: ipda + ipsa + vni [default]
----------------------------------------
index 1
    type:            vxlan
    name:            vxlan1
    port:            2201
    source-ip:       2.2.2.1
    remote_ip:       2.2.2.2
    link:            DOWN
    dynamic:         TRUE
----------------------------------------
```

Configuring static l2gre Tunnel

Configuration:

```
Switch# configure terminal
Switch(config)# interface l2gre1
Switch(config-if-l2gre1)# tunnel-source-ip 1.1.1.1
Switch(config-if-l2gre1)# tunnel-remote-ip 3.3.3.3
Switch(config-if-l2gre1)# tunnel-bind-static bind-port eth-0-3 nexthop-mac 0.0.2
bind-vlan 200
Switch(config-if-l2gre1)# openflow enable
```

Validation:

```
Switch# show openflow interface tunnel brief
The Maximum of tunnel ports is 500, currently 1 tunnel ports is valid.

Default tunnel_type of bind_port is vxlan, if you want to use other type of
tunnel, modify tunnel mode on the bind_port.

Vxlan source port is dynamic
```



```
Vxlan dest port is default: 4789

Decap mode: ipda + ipsa + vni [default]
----------------------------------------
index 1
    type:             l2gre
    name:             l2gre1
    port:             201
    source-ip:        1.1.1.1
    remote_ip:        3.3.3.3
    link:             UP
    dynamic:          FALSE
    bind_port:        eth-0-3
    remote_mac:       00:00:00:00:00:02
    vlan_id:          200
----------------------------------------
```

Configuring dynamic l2gre Tunnel

Configuration:

```
Switch# configure terminal
Switch(config)# interface l2gre1
Switch(config-if-l2gre1)# tunnel-source-ip 2.2.2.1
Switch(config-if-l2gre1)# tunnel-remote-ip 2.2.2.2
Switch(config-if-l2gre1)# openflow enable
```

Validation:

```
Switch# show openflow interface tunnel brief
The Maximum of tunnel ports is 500, currently 1 tunnel ports is valid.

Default tunnel_type of bind_port is vxlan, if you want to use other type of
tunnel, modify tunnel mode on the bind_port.

Vxlan source port is dynamic
Vxlan dest port is default: 4789

Decap mode: ipda + ipsa + vni [default]
----------------------------------------
index 1
    type:             l2gre
    name:             l2gre1
    port:             201
    source-ip:        2.2.2.1
    remote_ip:        2.2.2.2
    link:             DOWN
    dynamic:          TRUE
----------------------------------------
```

Configuring static nvgre Tunnel

Configuration:


```
Switch# configure terminal
Switch(config)# interface nvgre1
Switch(config-if-nvgre1)# tunnel-source-ip 1.1.1.1
Switch(config-if-nvgre1)# tunnel-remote-ip 4.4.4.4
Switch(config-if-nvgre1)# tunnel-bind-static bind-port eth-0-4 nexthop-mac 0.0.3
bind-vlan 300
Switch(config-if-nvgre1)# openflow enable
```

Validation:

```
Switch# show openflow interface tunnel brief
The Maximum of tunnel ports is 500, currently 1 tunnel ports is valid.

Default tunnel_type of bind_port is vxlan, if you want to use other type of
tunnel, modify tunnel mode on the bind_port.

Vxlan source port is dynamic
Vxlan dest port is default: 4789

Decap mode: ipda + ipsa + vni [default]
----------------------------------------
index 1
    type:             nvgre
    name:             nvgre1
    port:             1301
    source-ip:        1.1.1.1
    remote_ip:        4.4.4.4
    link:             UP
    dynamic:          FALSE
    bind_port:        eth-0-4
    remote_mac:       00:00:00:00:00:03
    vlan_id:          300
----------------------------------------
```

Configuring dynamic nvgre Tunnel

Configuration:

```
Switch# configure terminal
Switch(config)# interface nvgre1
Switch(config-if-nvgre1)# tunnel-source-ip 1.1.1.1
Switch(config-if-nvgre1)# tunnel-remote-ip 4.4.4.4
Switch(config-if-nvgre1)# openflow enable
```

Validation:

```
Switch# show openflow interface tunnel brief
The Maximum of tunnel ports is 500, currently 1 tunnel ports is valid.

Default tunnel_type of bind_port is vxlan, if you want to use other type of
tunnel, modify tunnel mode on the bind_port.

Vxlan source port is dynamic
Vxlan dest port is default: 4789
```



```
Decap mode: ipda + ipsa + vni [default]
----------------------------------------
index 1
    type:             nvgre
    name:             nvgre1
    port:             1301
    source-ip:        2.2.2.1
    remote_ip:        2.2.2.2
    link:             DOWN
    dynamic:          TRUE
----------------------------------------
```

Configuring static Vxlan Tunnel Service

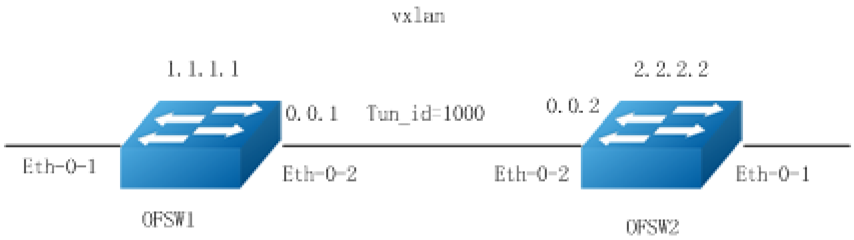

Figure 11-8 Vxlan Tunnel topology

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create vxlan interface and enter the configure mode, specify the local and remote ip, bind the remote mac address, and enable openflow

OFSW1:

```
Switch(config)# interface vxlan1
Switch(config-if-vxlan1)# tunnel-source-ip 1.1.1.1
Switch(config-if-vxlan1)# tunnel-remote-ip 2.2.2.2
Switch(config-if-vxlan1)# tunnel-bind-static bind-port eth-0-2 nexthop-mac 0.0.2
Switch(config-if-vxlan1)# openflow enable
```

OFSW2:

```
Switch(config)# interface vxlan1
Switch(config-if-vxlan1)# tunnel-source-ip 2.2.2.2
Switch(config-if-vxlan1)# tunnel-remote-ip 1.1.1.1
Switch(config-if-vxlan1)# tunnel-bind-static bind-port eth-0-2 nexthop-mac 0.0.1
Switch(config-if-vxlan1)# openflow enable
```

step 3 Create the flow for encapsulate and decapsulate

OFSW1:


```
Switch#ovs-ofctl add-flow br0 in_port=1,actions=set_field:1000->tun_id,output:2201
-O openflow13
```

OFSW2:

```
Switch#ovs-ofctl add-flow br0 in_port=2201,tun_id=1000,actions=output:1 -O
openflow13
```

step 4 Validation

OFSW1:

```
Switch# show openflow interface tunnel brief
The Maximum of tunnel ports is 500, currently 1 tunnel ports is valid.

Default tunnel_type of bind_port is vxlan, if you want to use other type of
tunnel, modify tunnel mode on the bind_port.

Vxlan source port is dynamic
Vxlan dest port is default: 4789

Decap mode: ipda + ipsa + vni [default]
----------------------------------------
index 1
    type:             vxlan
    name:             vxlan1
    port:             2201
    source-ip:        1.1.1.1
    remote_ip:        2.2.2.2
    link:             UP
    dynamic:          FALSE
    bind_port:        eth-0-2
    remote_mac:       00:00:00:00:00:02
    vlan_id:          None
----------------------------------------
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=12.869s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=set_field:0x3e8->tun_id,output:2201
```

OFSW2:

```
Switch# show openflow interface tunnel brief
The Maximum of tunnel ports is 500, currently 1 tunnel ports is valid.

Default tunnel_type of bind_port is vxlan, if you want to use other type of
tunnel, modify tunnel mode on the bind_port.

Vxlan source port is dynamic
Vxlan dest port is default: 4789

Decap mode: ipda + ipsa + vni [default]
----------------------------------------
index 1
    type:             vxlan
```



```
    name:             vxlan1
    port:             2201
    source-ip:        2.2.2.2
    remote_ip:        1.1.1.1
    link:             UP
    dynamic:          FALSE
    bind_port:        eth-0-2
    remote_mac:       00:00:00:00:00:01
    vlan_id:          None
----------------------------------------
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=12.869s, table=0, n_packets=0, n_bytes=0,
in_port=2201,tun_id=1000 actions=output:1
```

step 5 Configuring Vxlan Tunnel L4 Source Port and Dest Port(Optional)

Hybrid system support to modify vxlan outer header udp soure port and dest port, but before modification we should delete vxlan tunnel flow and vxlan interface.

```
Switch(config)# vxlan-tunnel dest-port 1111
Switch(config)# vxlan-tunnel src-port 2222
```

step 6 Configuring Vxlan Tunnel-id Match Field

Hybrid system support match vxlan outer tunnel id to shunt the stream. Configuration:

- Flow match vxlan tun_id=1000, output to port 2;

- Flow match vxlan tun_id=1001, output to port 3;

```
Switch# ovs-ofctl add-flow br0 udp,udp_dst=4789,tun_id=1000,actions=output:2 -O
openflow13
Switch# ovs-ofctl add-flow br0 udp,udp_dst=4789,tun_id=1001,actions=output:3 -O
openflow13
```

step 7 Validation(Vxlan Tunnel-id Match Field)

```
DUT1# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=4.972s, table=0, n_packets=0, n_bytes=0,
udp,tun_id=0x3e9,tp_dst=4789 actions=output:3
 cookie=0x0, duration=132.216s, table=0, n_packets=0, n_bytes=0,
udp,tun_id=0x3e8,tp_dst=4789 actions=output:2
```


Configuring dynamic Vxlan Tunnel Service

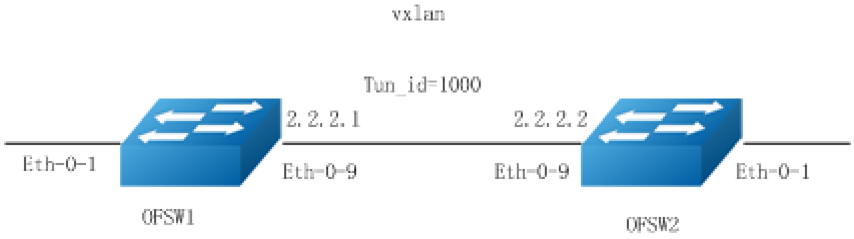

Figure 11-9 Vxlan Tunnel topology

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create vxlan interface and enter the configure mode, specify the local and remote ip, and enable openflow

OFSW1:

```
Switch(config)# interface eth-0-9
Switch(config-eth-0-9)# openflow enable
Switch(config-eth-0-9)# no switchport
Switch(config-eth-0-9)# ip address 2.2.2.1/24
Switch(config-eth-0-9)# no shutdown
Switch(config-eth-0-9)# exit
Switch(config)# interface vxlan1
Switch(config-if-vxlan1)# tunnel-source-ip 2.2.2.1
Switch(config-if-vxlan1)# tunnel-remote-ip 2.2.2.2
Switch(config-if-vxlan1)# openflow enable
```

OFSW2:

```
Switch(config)# interface eth-0-9
Switch(config-eth-0-9)# openflow enable
Switch(config-eth-0-9)# no switchport
Switch(config-eth-0-9)# ip address 2.2.2.2/24
Switch(config-eth-0-9)# no shutdown
Switch(config-eth-0-9)# exit
Switch(config)# interface vxlan1
Switch(config-if-vxlan1)# tunnel-source-ip 2.2.2.2
Switch(config-if-vxlan1)# tunnel-remote-ip 2.2.2.1
Switch(config-if-vxlan1)# openflow enable
```

step 3 Create the flow for encapsulate and decapsulate

OFSW1:


```
Switch#ovs-ofctl add-flow br0 in_port=1,actions=set_field:1000->tun_id,output:2201
-O openflow13
```

OFSW2:

```
Switch#ovs-ofctl add-flow br0 in_port=2201,tun_id=1000,actions=output:1 -O
openflow13
```

step 4 Validation

OFSW1:

```
Switch# show openflow interface tunnel brief
The Maximum of tunnel ports is 500, currently 1 tunnel ports is valid.

Default tunnel_type of bind_port is vxlan, if you want to use other type of
tunnel, modify tunnel mode on the bind_port.

Vxlan source port is dynamic
Vxlan dest port is default: 4789

Decap mode: ipda + ipsa + vni [default]
----------------------------------------
index 1
    type:             vxlan
    name:             vxlan1
    port:             2201
    source-ip:        2.2.2.1
    remote_ip:        2.2.2.2
    link:             UP
    dynamic:          TRUE
----------------------------------------
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=178.701s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=set_field:0x3e8->tun_id,output:2201
```

OFSW2:

```
Switch# show openflow interface tunnel brief
The Maximum of tunnel ports is 500, currently 1 tunnel ports is valid.

Default tunnel_type of bind_port is vxlan, if you want to use other type of
tunnel, modify tunnel mode on the bind_port.

Vxlan source port is dynamic
Vxlan dest port is default: 4789

Decap mode: ipda + ipsa + vni [default]
----------------------------------------
index 1
    type:             vxlan
    name:             vxlan1
    port:             2201
    source-ip:        2.2.2.2
```

```
    remote_ip:        2.2.2.1
    link:             UP
    dynamic:          TRUE
----------------------------------------
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=285.408s, table=0, n_packets=0, n_bytes=0,
tun_id=0x3e8,in_port=2201 actions=output:1
```

Configuring static l2gre Tunnel Service

In l2gre tunnel flow, the tunnel-id is optional.

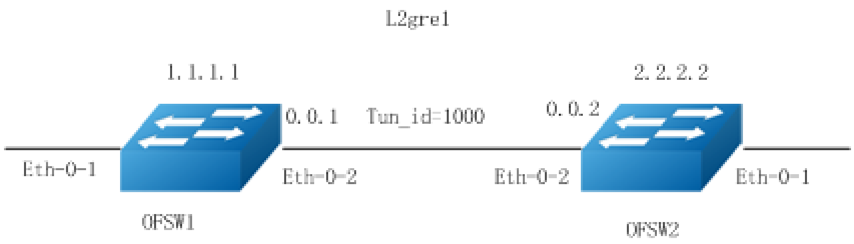

Figure 11-10 l2gre Tunnel topology

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create vxlan interface and enter the configure mode, specify the local and remote ip, bind the remote mac address， and enable openflow

OFSW1:

```
Switch(config)# interface l2gre1
Switch(config-if-l2gre1)# tunnel-source-ip 1.1.1.1
Switch(config-if-l2gre1)# tunnel-remote-ip 2.2.2.2
Switch(config-if-l2gre1)# tunnel-bind-static bind-port eth-0-2 nexthop-mac 0.0.2
Switch(config-if-l2gre1)# openflow enable
```

OFSW2:

```
Switch(config)# interface l2gre1
Switch(config-if-l2gre1)# tunnel-source-ip 2.2.2.2
Switch(config-if-l2gre1)# tunnel-remote-ip 1.1.1.1
Switch(config-if-l2gre1)# tunnel-bind-static bind-port eth-0-2 nexthop-mac 0.0.1
Switch(config-if-l2gre1)# openflow enable
```

step 3 Create the flow for encapsulate and decapsulate

OFSW1:


```
Switch#ovs-ofctl add-flow br0 in_port=1,actions=output:201 -O openflow13
```

OFSW2:

```
Switch#ovs-ofctl add-flow br0 in_port=201,actions=output:1 -O openflow13
```

step 4 Validation

OFSW1:

```
Switch# show openflow interface tunnel brief
The Maximum of tunnel ports is 500, currently 1 tunnel ports is valid.

Default tunnel_type of bind_port is vxlan, if you want to use other type of
tunnel, modify tunnel mode on the bind_port.

Vxlan source port is dynamic
Vxlan dest port is default: 4789

Decap mode: ipda + ipsa + vni [default]
----------------------------------------
index 1
    type:             l2gre
    name:             l2gre1
    port:             201
    source-ip:        1.1.1.1
    remote_ip:        2.2.2.2
    link:             UP
    dynamic:          FALSE
    bind_port:        eth-0-2
    remote_mac:       00:00:00:00:00:02
    vlan_id:          None
----------------------------------------
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=14.971s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=output:201
```

OFSW2:

```
Switch# show openflow interface tunnel brief
The Maximum of tunnel ports is 500, currently 1 tunnel ports is valid.

Default tunnel_type of bind_port is vxlan, if you want to use other type of
tunnel, modify tunnel mode on the bind_port.

Vxlan source port is dynamic
Vxlan dest port is default: 4789

Decap mode: ipda + ipsa + vni [default]
----------------------------------------
index 1
    type:             l2gre
    name:             l2gre1
    port:             201
```

```
    source-ip:        2.2.2.2
    remote_ip:        1.1.1.1
    link:             UP
    dynamic:          FALSE
    bind_port:        eth-0-2
    remote_mac:       00:00:00:00:00:01
    vlan_id:          None
----------------------------------------
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=11.315s, table=0, n_packets=0, n_bytes=0, in_port=201
actions=output:1
```

Configuring dynamic l2gre Tunnel Service

In l2gre tunnel flow, the tunnel-id is optional.

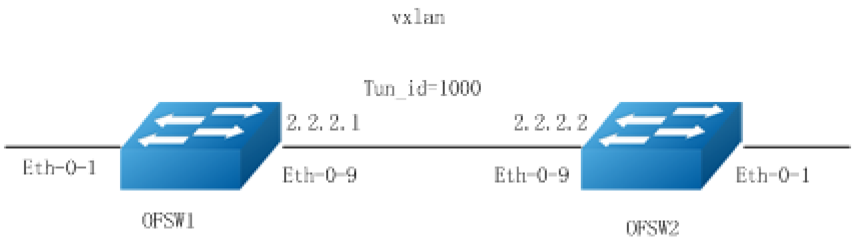

Figure 11-11 l2gre Tunnel topology

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create vxlan interface and enter the configure mode, specify the local and remote ip, and enable openflow

OFSW1:

```
Switch(config)# interface eth-0-9
Switch(config-eth-0-9)# openflow enable
Switch(config-eth-0-9)# no switchport
Switch(config-eth-0-9)# ip address 2.2.2.1/24
Switch(config-eth-0-9)# no shutdown
Switch(config-eth-0-9)# exit
Switch(config)# interface l2gre1
Switch(config-if-vxlan1)# tunnel-source-ip 2.2.2.1
Switch(config-if-vxlan1)# tunnel-remote-ip 2.2.2.2
Switch(config-if-vxlan1)# openflow enable
```

OFSW2:


```
Switch(config)# interface eth-0-9
Switch(config-eth-0-9)# openflow enable
Switch(config-eth-0-9)# no switchport
Switch(config-eth-0-9)# ip address 2.2.2.2/24
Switch(config-eth-0-9)# no shutdown
Switch(config-eth-0-9)# exit
Switch(config)# interface l2gre1
Switch(config-if-vxlan1)# tunnel-source-ip 2.2.2.2
Switch(config-if-vxlan1)# tunnel-remote-ip 2.2.2.1
Switch(config-if-vxlan1)# openflow enable
```

step 3 Create the flow for encapsulate and decapsulate

OFSW1:

```
Switch#ovs-ofctl add-flow br0 in_port=1,actions=output:201 -O openflow13
```

OFSW2:

```
Switch#ovs-ofctl add-flow br0 in_port=201,actions=output:1 -O openflow13
```

step 4 Validation

OFSW1:

```
Switch# show openflow interface tunnel brief
The Maximum of tunnel ports is 500, currently 1 tunnel ports is valid.

Default tunnel_type of bind_port is vxlan, if you want to use other type of
tunnel, modify tunnel mode on the bind_port.

Vxlan source port is dynamic
Vxlan dest port is default: 4789

Decap mode: ipda + ipsa + vni [default]
----------------------------------------
index 1
    type:            l2gre
    name:            l2gre1
    port:            201
    source-ip:       2.2.2.1
    remote_ip:       2.2.2.2
    link:            UP
    dynamic:         TRUE
----------------------------------------
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=243.086s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=set_field:0x3e8->tun_id,output:201
```

OFSW2:

```
Switch# show openflow interface tunnel brief
The Maximum of tunnel ports is 500, currently 1 tunnel ports is valid.
```



```
Default tunnel_type of bind_port is vxlan, if you want to use other type of
tunnel, modify tunnel mode on the bind_port.

Vxlan source port is dynamic
Vxlan dest port is default: 4789

Decap mode: ipda + ipsa + vni [default]
----------------------------------------
index 1
    type:            l2gre
    name:            l2gre1
    port:            201
    source-ip:       2.2.2.2
    remote_ip:       2.2.2.1
    link:            UP
    dynamic:         TRUE
----------------------------------------
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=98.231s, table=0, n_packets=0, n_bytes=0,
tun_id=0x3e8,in_port=201 actions=output:1
```

Configuring static nvgre Tunnel Service

In nvgre tunnel flow, the tunnel-id is must specified.

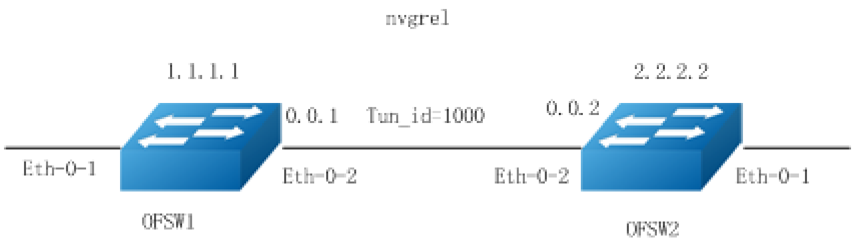

Figure 11-12 nvgre Tunnel topology

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create vxlan interface and enter the configure mode, specify the local and remote ip, bind the remote mac address， and enable openflow

OFSW1:

```
Switch(config)# interface nvgre1
Switch(config-if-nvgre1)# tunnel-source-ip 1.1.1.1
Switch(config-if-nvgre1)# tunnel-remote-ip 2.2.2.2
```



```
Switch(config-if-nvgre1)# tunnel-bind-static bind-port eth-0-2 nexthop-mac 0.0.2
Switch(config-if-nvgre1)# openflow enable
```

OFSW2:

```
Switch(config)# interface nvgre1
Switch(config-if-nvgre1)# tunnel-source-ip 2.2.2.2
Switch(config-if-nvgre1)# tunnel-remote-ip 1.1.1.1
Switch(config-if-nvgre1)# tunnel-bind-static bind-port eth-0-2 nexthop-mac 0.0.1
Switch(config-if-nvgre1)# openflow enable
```

step 3 Create the flow for encapsulate and decapsulate

OFSW1:

```
Switch#ovs-ofctl add-flow br0 in_port=1,actions=set_field:1000->tun_id,output:1301
-O openflow13
```

OFSW2:

```
Switch#ovs-ofctl add-flow br0 in_port=1301,tun_id=1000,actions=output:1 -O
openflow13
```

step 4 Validation

OFSW1:

```
Switch# show openflow interface tunnel brief
The Maximum of tunnel ports is 500, currently 1 tunnel ports is valid.

Default tunnel_type of bind_port is vxlan, if you want to use other type of
tunnel, modify tunnel mode on the bind_port.

Vxlan source port is dynamic
Vxlan dest port is default: 4789

Decap mode: ipda + ipsa + vni [default]
----------------------------------------
index 1
    type:             nvgre
    name:             nvgre1
    port:             1301
    source-ip:        1.1.1.1
    remote_ip:        2.2.2.2
    link:             UP
    dynamic:          FALSE
    bind_port:        eth-0-2
    remote_mac:       00:00:00:00:00:02
    vlan_id:          None
----------------------------------------
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=16.454s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=set_field:0x3e8->tun_id,output:1301
```


OFSW2:

```
Switch# show openflow interface tunnel brief
The Maximum of tunnel ports is 500, currently 1 tunnel ports is valid.

Default tunnel_type of bind_port is vxlan, if you want to use other type of
tunnel, modify tunnel mode on the bind_port.

Vxlan source port is dynamic
Vxlan dest port is default: 4789

Decap mode: ipda + ipsa + vni [default]
----------------------------------------
index 1
    type:             nvgre
    name:             nvgre1
    port:             1301
    source-ip:        2.2.2.2
    remote_ip:        1.1.1.1
    link:             UP
    dynamic:          FALSE
    bind_port:        eth-0-2
    remote_mac:       00:00:00:00:00:01
    vlan_id:          None
----------------------------------------
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=8.423s, table=0, n_packets=0, n_bytes=0,
tun_id=0x3e8,in_port=1301 actions=output:1
```

Configuring dynamic nvgre Tunnel Service

In nvgre tunnel flow, the tunnel-id is must specified.

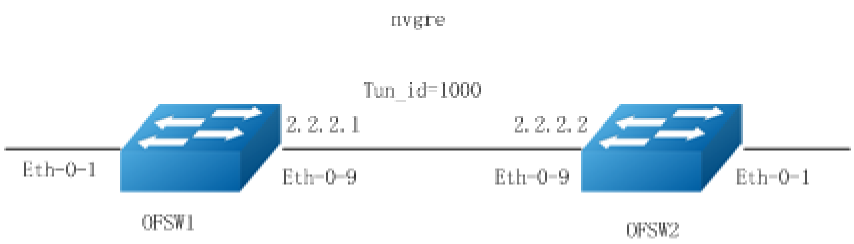

Figure 11-13 nvgre Tunnel topology

step 1 Enter the configure mode

```
Switch# configure terminal
```


step 2 Create vxlan interface and enter the configure mode, specify the local and remote ip, and enable openflow

OFSW1:

```
Switch(config)# interface eth-0-9
Switch(config-eth-0-9)# openflow enable
Switch(config-eth-0-9)# no switchport
Switch(config-eth-0-9)# ip address 2.2.2.1/24
Switch(config-eth-0-9)# no shutdown
Switch(config-eth-0-9)# exit
Switch(config)# interface nvgre1
Switch(config-if-vxlan1)# tunnel-source-ip 2.2.2.1
Switch(config-if-vxlan1)# tunnel-remote-ip 2.2.2.2
Switch(config-if-vxlan1)# openflow enable
```

OFSW2:

```
Switch(config)# interface eth-0-9
Switch(config-eth-0-9)# openflow enable
Switch(config-eth-0-9)# no switchport
Switch(config-eth-0-9)# ip address 2.2.2.2/24
Switch(config-eth-0-9)# no shutdown
Switch(config-eth-0-9)# exit
Switch(config)# interface nvgre1
Switch(config-if-vxlan1)# tunnel-source-ip 2.2.2.2
Switch(config-if-vxlan1)# tunnel-remote-ip 2.2.2.1
Switch(config-if-vxlan1)# openflow enable
```

step 3 Create the flow for encapsulate and decapsulate

OFSW1:

```
Switch#ovs-ofctl add-flow br0 in_port=1,actions=set_field:1000->tun_id,output:1301
-O openflow13
```

OFSW2:

```
Switch#ovs-ofctl add-flow br0 in_port=1301,tun_id=1000,actions=output:1 -O
openflow13
```

step 4 Validation

OFSW1:

```
Switch# show openflow interface tunnel brief
The Maximum of tunnel ports is 500, currently 1 tunnel ports is valid.

Default tunnel_type of bind_port is vxlan, if you want to use other type of
tunnel, modify tunnel mode on the bind_port.

Vxlan source port is dynamic
```



```
Vxlan dest port is default: 4789

Decap mode: ipda + ipsa + vni [default]
----------------------------------------
index 1
    type:            nvgre
    name:            nvgre1
    port:            1301
    source-ip:       2.2.2.1
    remote_ip:       2.2.2.2
    link:            UP
    dynamic:         TRUE
----------------------------------------
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=155.543s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=set_field:0x3e8->tun_id,output:1301
```

OFSW2:

```
Switch# show openflow interface tunnel brief
The Maximum of tunnel ports is 500, currently 1 tunnel ports is valid.

Default tunnel_type of bind_port is vxlan, if you want to use other type of
tunnel, modify tunnel mode on the bind_port.

Vxlan source port is dynamic
Vxlan dest port is default: 4789

Decap mode: ipda + ipsa + vni [default]
----------------------------------------
index 1
    type:            nvgre
    name:            nvgre1
    port:            1301
    source-ip:       2.2.2.2
    remote_ip:       2.2.2.1
    link:            UP
    dynamic:         TRUE
----------------------------------------
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=617.327s, table=0, n_packets=0, n_bytes=0,
tun_id=0x3e8,in_port=1301 actions=output:1
```

11.10 Openflow BondPort Configuration

11.10.1 Overview

Hybrid system support creating linkagg port and set openflow enable on linkagg port, does not support creat bond port in ovsdb.

The linkagg port in Hybrid most support 55, from 1221 to 1275. The linkagg port not only can be inport, but also can be output.

Now Hybrid deivce only can support static agg，channel-group agg is not support.

11.10.2 Configuration

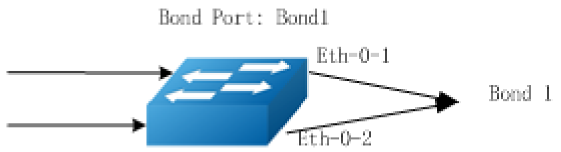

Figure 11-14 Hybrid Bond topology

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enter the interface mode, disable openflow and join the static linkagg interface

```
Switch(config)# interface range eth-0-1 - 2
Switch(config-if-range)# no openflow enable
Switch(config-if-range)# static-channel-group 1
Switch(config-if-range)# exit
```

step 3 Enable openflow on linkagg interface

```
Switch(config)# interface agg1
Switch(config-if-agg1)# openflow enable
Switch(config-if-agg1)# end
```

step 4 Add a flow to match the ip packets and forward to the linkagg interface

```
Switch# ovs-ofctl add-flow br0 "ip,actions=output:1221" -O openflow13
```

step 5 Validation

```
Switch# show channel-group 1 summary
port-channel load-balance hash-arithmetic: xor
Port-channel load-balance hash-field-select:
    src-mac dst-mac src-ip dst-ip
Flags:  s - suspend           T - standby
        w - wait              B - in Bundle
        R - Layer3            S - Layer2
        D - down/admin down   U - in use
```



```
Mode:  SLB   - static load balance
       DLB   - dynamic load balance
       RR    - round robin load balance

Aggregator Mode Protocol Ports
----------+----+--------+---------------------------------------------------
agg1(SU)   SLB  Static   eth-0-1(B)    eth-0-2(B)
Switch# ovs-ofctl show br0
1221(agg1): addr:00:1e:08:0b:ce:1f
     config:     0
     state:      0
     speed: 0 Mbps now, 0 Mbps max
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=119.899s, table=0, n_packets=0, n_bytes=0, ip
actions=output:1221
```

11.11 Configuring Hybrid Openflow Normal-Mpls

11.11.1 Overview

Function Introduction

MPLS is a highly scalable, protocol agnostic, data-carrying mechanism. In a traditional MPLS network, data packets are assigned labels. Packet forwarding decisions are made solely on the contents of this label, without the need to examine the packet itself. This allows one to create end-to-end circuits across any type of transport medium, using any protocol.

With OpenFlow and SDN, the control-plane and MPLS data-plane can be de-coupled; network applications on top of the SDN controller can create MPLS services globally, and optimize MPLS services more flexibly.

Hybrid system software have extended OpenFlow specification 1.3.1 to support programming MPLS flow tables, it supports the following features:

- IPv4 over MPLS
- Virtual Pseudo-Wire Service
- Virtual Private LAN Service (Note: currently only supports broadcast packets to remote PEs or ACs)
- Sub-Path-Maintenance Elements(FF-Spme)

In this configuration guide, we will describe how to define flow tables for IP over MPLS and VPWS/VPLS services using CLI interface.

Extended actions based on Nicira Extension for the extended matching fields and actions based on Nicira extension.

Principle Description

References:

- OpenFlow 1.3.1 specification
- RFC 3031
- RFC 3032
- RFC 4447
- RFC 4762
- Open vSwitch

11.11.2 Configuration

Configuring IP over MPLS

Figure 11-15 IP over MPLS Topology

Hybrid system supports encapsulating IPv4 packets in MPLS, and tunnels the IPv4 packets to remote devices.

In this configuration example, a LSP (Label Switching Path) is created. Along the path switch1, switch2 and switch3, which is used to encapsulate IPv4 packets received from eth-0-1 of Switch 1 in MPLS packets, and tunnel it to Switch 3, after Switch 3 decapsulates the MPLS packets, it sends the original IPv4 packets out from eth-0-2.

Here is a brief description of MPLS label operation in each switch:

- Switch1: push MPLS label 16
- Switch2: swap MPLS label 16 with MPLS label 17

- Switch3: pop MPLS label 17

Configuring Switch 1

IPv4 packets received from eth-0-1 will be pop ethernet header, and then pushed mpls label 16, pushed a new ethernet header which new dst_mac is switch2 eth-0-1's mac address, sent out from eth-0-2 to switch2.

```
Switch# ovs-ofctl add-flow br0
"in_port=1,dl_type=0x0800,actions=pop_l2,push_mpls:0x8847,set_field:16-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:2" -O openflow13
```

Configuring Switch 2

For L2VPN, the ether type for pop_mpls action is not definite, according to this, for all MPLS flows, we do not check the ether-type in the pop_mpls action, user can specify any ether-types.

Mpls packets received from eth-0-1 will be pop ethernet header, and then swap mpls 16 label with mpls 17, pushed a new ethernet header which new dst_mac is switch3 eth-0-1's mac address, sent out from eth-0-2 to switch3.

```
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=16,actions=pop_l2,pop_mpls:0x800,push_mpls:0x8847,set_fi
eld:17->mpls_label,push_l2,set_field:00:1e:08:00:03:01->eth_dst,output:2" -O
openflow13
```

Configuring Switch 3

After decapsulating MPLS packets, Hybrid device ASIC does not support matching IP flow tables again currently, so user must specify a valid output interface and next hop MAC address for the MPLS pop flow table in Switch 3.

Mpls packets received from eth-0-1 will be pop ethernet header and mpls label, pushed new ethernet header, the original IPv4 packets are sent out from eth-0-2.

```
Switch# ovs-ofctl add-flow br0
"in_port=1,dl_type=0x8847,mpls_label=17,actions=pop_l2,pop_mpls:0x0800,push_l2,set_
field:00:00:00:00:04:01->eth_dst,output:2" -O openflow13
```

Validation

Validation of switch 1


```
Switch1# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x4):
 cookie=0x0, duration=4.085s, table=0, n_packets=0, n_bytes=0, ip,in_port=1
actions=pop_l2,push_mpls:0x8847,set_field: 16-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:2
```

Validation of switch 2

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x4):
 cookie=0x0, duration=3.113s, table=0, n_packets=0, n_bytes=0,
dl_type=0x8847,mpls_label=16
actions=pop_l2,pop_mpls:0x0800,push_mpls:0x8847,set_field:17-
>mpls_label,push_l2,set_field:00:1e:08:00:03:01->eth_dst,output:2
```

Validation of switch 3

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x4):
 cookie=0x0, duration=4.273s, table=0, n_packets=0, n_bytes=0,
dl_type=0x8847,mpls_label=17
actions=pop_l2,pop_mpls:0x0800,push_l2,set_field:00:00:00:00:04:01-
>eth_dst,output:2
```

Configuring VPWS

Figure 11-16 VPWS Topology

Hybrid system supports encapsulating Ethernet packets in MPLS, and tunnels the MPLS packets to remote devices, this feature can be used to create VPWS services. In this configuration example, a PW (Pseudo Wire) is created.

Along the path switch1, switch 2 and switch3, which is used to encapsulate Ethernet packets received from AC (Attach Circuit) eth-0-1 of Switch 1 in MPLS packets, and tunnel it to Switch 3, after Switch 3 decapsulates the MPLS packets, it sends the original Ethernet packets out from eth-0-2.

Here is a brief description of MPLS label operation in each switch:

- Switch1: push PW label 16, push Tunnel label 300
- Switch2: swap Tunnel label 300 with Tunnel label 400
- Switch3: pop Tunnel label 400, pop PW label 16

Configuring Switch 1

Ethernet packets received from eth-0-1 will be pushed PW label 16 and tunnel label 300, pushed a new ethernet header which new dst_mac is switch2 eth-0-1's mac address, sent out from eth-0-2 to switch2.

```
Switch# ovs-ofctl add-flow br0 "in_port=1,actions=push_mpls:0x8847,set_field:16->mpls_label,push_mpls:0x8847,set_field:300->mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:2" -O openflow13
```

Configuring Switch 2

Mpls packets received from eth-0-1 will be pop ethernet header, swap mpls label 300 with mpls 400, pushed a new ethernet header which new dst_mac is switch3 eth-0-1's mac address, sent out from eth-0-2 to switch3.

```
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=300,actions=pop_l2,pop_mpls:0x8847,push_mpls:0x8847,set_field:400->mpls_label,push_l2,set_field:00:1e:08:00:03:01->eth_dst,output:2" -O openflow13
```

Configuring Switch 3

After Switch 3 received the MPLS packets with two labels, the first tunnel label will be pop(the flow use a special port 0xfffffff7(PW_FWD) to pop tunnel label), however there is still one PW label present, so the packet will be forwarded according to the flow table corresponding to the PW label, and the action pop_l2 in the flow table corresponding to the PW label will be ignored because it have been done in the first flow corresponding to the first tunnel label.

Pop mpls packet ethernet header, pop tunnel label 400, output to PW_FWD

Pop PW label, send out the packet from eth-0-2

```
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=400,actions=pop_l2,pop_mpls:0x8847,PW_FWD "  -O openflow13
Switch# ovs-ofctl add-flow br0
dl_type=0x8847,mpls_label=16,actions=pop_l2,pop_mpls:0x800,output:2 -O openflow13
```

Validation

Validation of switch 1

```
Switch1# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x4):
```



```
 cookie=0x0, duration=1.592s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=push_mpls:0x8847,set_field:16->mpls_label,push_mpls:0x8847,set_field:300-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:2
```

Validation of switch 2

```
OFPST_FLOW reply (OF1.3) (xid=0x4):
 cookie=0x0, duration=1.082s, table=0, n_packets=0, n_bytes=0,
dl_type=0x8847,mpls_label=300
actions=pop_l2,pop_mpls:0x8847,push_mpls:0x8847,set_field:400-
>mpls_label,push_l2,set_field:00:1e:08:00:03:01->eth_dst,output:2
```

Validation of switch 3

```
Switch3# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (xid=0x4):
 cookie=0x0, duration=27.027s, table=0, n_packets=0, n_bytes=0,
dl_type=0x8847,mpls_label=400 actions=pop_l2,pop_mpls:0x8847,PW_FWD
 cookie=0x0, duration=3.824s, table=0, n_packets=0, n_bytes=0,
dl_type=0x8847,mpls_label=16 actions=pop_l2,pop_mpls:0x0800,output:2
```

Configuring VPLS

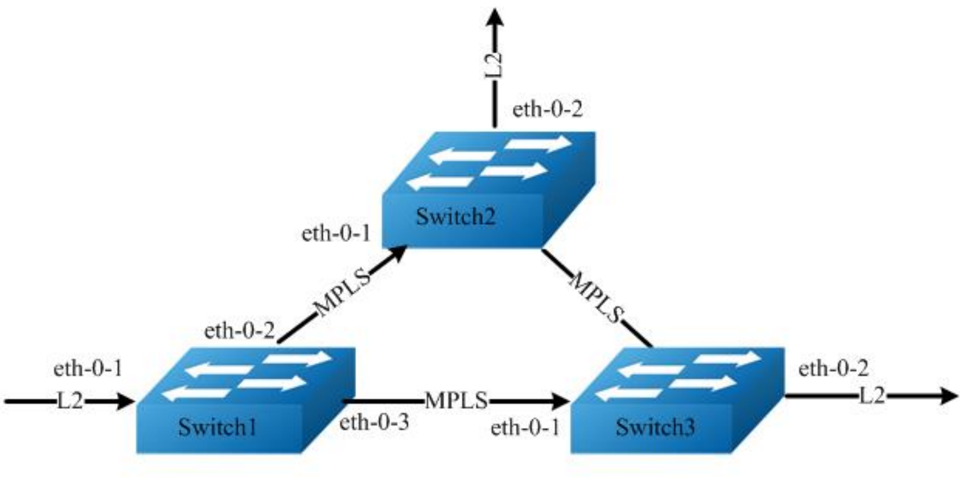

Figure 11-17 VPLS Topology

OpenFlow does not support define VSI (Virtual Switch Instance) currently, so the VPLS feature of Hybrid is implemented by creating flow table that broadcasts Ethernet packets to remote PEs or ACs.

In this configuration example, switches are connected by PW in full-mesh topology, for Ethernet packets received from AC of each switch; they will be encapsulated in MPLS and sent to other switches.

Here is a brief description of MPLS label operations in each switch:

- Switch1: for Switch 2, push PW label 16, push Tunnel label 300, for Switch 3 push PW label 16, push Tunnel label 400
- Switch2: pop Tunnel label 300, pop PW label 16
- Switch3: pop Tunnel label 400, pop PW label 16

Configuring Switch 1

Ethernet packets received from eth-0-1 will be pushed PW label 16 and tunnel label 300, pushed a new ethernet header which new dst_mac is switch2 eth-0-1's mac address, sent out from eth-0-2 to switch2; similarly, pushed PW label 16 and tunnel label 400, pushed a new ethernet header which new dst_mac is switch3 eth-0-1's mac address, sent out from eth-0-3 to switch3.

```
Switch# ovs-ofctl add-flow br0 "in_port=1, actions= push_mpls:0x8847,set_field:16-
>mpls_label,push_mpls:0x8847,set_field:300-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:2,
push_mpls:0x8847,set_field:16->mpls_label,push_mpls:0x8847,set_field:400-
>mpls_label,push_l2,set_field:00:1e:08:00:03:01->eth_dst,output:3" -O openflow13
```

Configuring Switch 2

Pop mpls packet ethernet header, pop tunnel label 300, output to PW_FWD;

Pop PW label, send out the packet from eth-0-2.

```
Switch# ovs-ofctl add-flow br0 "dl_type=0x8847,mpls_label=300,
actions=pop_l2,pop_mpls:0x8847,PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=16,actions=pop_l2,pop_mpls:0x800,output:2" -O openflow13
```

Configuring Switch 3

Pop mpls packet ethernet header, pop tunnel label 400, output to PW_FWD;

Pop PW label, send out the packet from eth-0-2.

```
Switch# ovs-ofctl add-flow br0 "dl_type=0x8847,mpls_label=400,
actions=pop_l2,pop_mpls:0x8847,PW_FWD " -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=16,actions=pop_l2,pop_mpls:0x800,output:2" -O openflow13
```

Validation

Validation of switch 1


```
Switch1# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x4):
 cookie=0x0, duration=2.69s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=push_mpls:0x8847,set_field:16->mpls_label,push_mpls:0x8847,set_field:300-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01-
>eth_dst,output:2,push_mpls:0x8847,set_field:16-
>mpls_label,push_mpls:0x8847,set_field:400-
>mpls_label,push_l2,set_field:00:1e:08:00:03:01->eth_dst,output:3
```

Validation of switch 2

```
Switch2# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x4):
 cookie=0x0, duration=11.7s, table=0, n_packets=0, n_bytes=0,
dl_type=0x8847,mpls_label=300 actions=pop_l2,pop_mpls:0x8847,PW_FWD
 cookie=0x0, duration=3.233s, table=0, n_packets=0, n_bytes=0,
dl_type=0x8847,mpls_label=16 actions=pop_l2,pop_mpls:0x0800,output:2
```

Validation of switch 3

```
Switch3# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x4):
 cookie=0x0, duration=12.732s, table=0, n_packets=0, n_bytes=0,
dl_type=0x8847,mpls_label=400 actions=pop_l2,pop_mpls:0x8847,PW_FWD
 cookie=0x0, duration=2.877s, table=0, n_packets=0, n_bytes=0,
dl_type=0x8847,mpls_label=16 actions=pop_l2,pop_mpls:0x0800,output:2
```

Configuring Spme

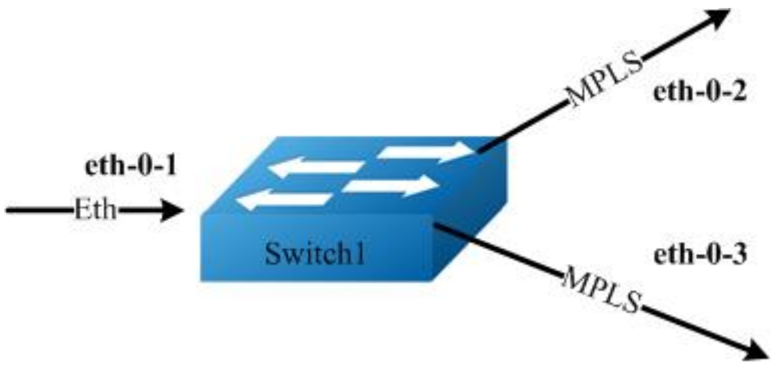

Figure 11-18 Spme Topology

Hybrid device support configuring Spme.

When the switch revieves the mpls packet with label, switch will take off this tag and L2 header, then the packet will de added new two tags and header.

- Switch1 revieves the mpls packet with label100, takes off the label100 and add new label1000 and add label2000 again, the packet will output from eth-0-2 with new L2 header. This is one bucket of ffgroup;
- Switch1 revieves the mpls packet with label100, takes off the label100 and add new label1001 and add label2001 again, the packet will output from eth-0-3 with new L2 header. This is other bucket of ffgroup;

Add ff-group with type spme, add a flow point to the group

```
Switch# ovs-ofctl add-group br0
"group_id=1,type=ff,bucket=watch_port:2,pop_l2,pop_mpls:0x0800,push_mpls:0x8847,set
_field:1000->mpls_label,push_mpls:0x8847,set_field:2000-
>mpls_label,push_l2,set_field:00:00:00:01:02:04-
>eth_dst,output:2,bucket=watch_port:3,pop_l2,pop_mpls:0x0800,push_mpls:0x8847,set_f
ield:1001->mpls_label,push_mpls:0x8847,set_field:2001-
>mpls_label,push_l2,set_field:00:00:00:01:02:05->eth_dst,output:3" -O openflow13
Switch# ovs-ofctl add-flow br0 mpls,mpls_label=100,actions=group:1 -O openflow13
```

Validation

```
Switch# ovs-ofctl dump-groups br0 -O openflow13
OFPST_GROUP_DESC reply (OF1.3) (xid=0x2):

group_id=1,type=ff,bucket=watch_port:2,pop_l2,pop_mpls:0x0800,push_mpls:0x8847,set_
field:1000->mpls_label,push_mpls:0x8847,set_field:2000-
>mpls_label,push_l2,set_field:00:00:00:01:02:04-
>eth_dst,output:2,bucket=watch_port:3,pop_l2,pop_mpls:0x0800,push_mpls:0x8847,set_f
ield:1001->mpls_label,push_mpls:0x8847,set_field:2001-
>mpls_label,push_l2,set_field:00:00:00:01:02:05->eth_dst,output:3
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=129.591s, table=0, n_packets=0, n_bytes=0,
mpls,mpls_label=100 actions=group:1
```

Openflow-Spec1.3 modification

The following list describes the modification on top of OpenFlow Spec 1.3, the medication is in C source code.

Modification Description	Modifcation in C source code

Define a special fake port to abstract all the MPLS tunnels which are used to carry L2VPN traffic, because Hybrid device will do next MPLS label lookup(corresponding to VPWS/VPLS PW label)after the MPLS Tunnel label have been popped, and a physical port is nonsense for the MPLS tunnel egress flow.	In enum ofp_port_no, add a new fake port: OFPP_PW_FWD = 0xfffffff7

Extended actions based on Nicira Extension

The following list describes the extended actions on Hybrid which are based on Nicira extension to Open vSwitch, the action type definitions are in C source code.

Extension flow action	Action definition	Ovs-ofctl command syntax
PUSH_L2=25	struct nx_action_push_l2 { ovs_be16 type;/* OFPAT_VENDOR.*/ ovs_be16 len;/* Length is 8. */ ovs_be32 vendor;/* NX_VENDOR_ID.*/ ovs_be16 subtype;/* PUSH_L2 */ uint8_t pad[6]; };	push_l2
POP_L2 = 26	struct nx_action_pop_l2 { ovs_be16 type;/* OFPAT_VENDOR. */ ovs_be16 len;/* Length is 8. */ ovs_be32 vendor;/* NX_VENDOR_ID.*/ ovs_be16 subtype;/* POP_L2 */ uint8_t pad[6]; };	pop_l2

Complete action list to create MPLS service

The following list describes all the valid action lists to create MPLS service. Here is the syntax description in the action list:

- [] in the action list means the element in the [] is optional.
- () is used to group related actions, the grouped actions must be both present.
- in the [] stands for alternative action.

Service Type	Action list	Output port restriction	Note
IP over MPLS ingress	NXAST_POP_L2 PUSH_MPLS [SET_FIELD->MPLS_TC | SET_MPLS_TTL] SET_FIELD->MPLS_LABEL PUSH_L2 SET_FIELD->DL_DST [SET_FIELD->VLAN_ID] OFPAT_OUTPUT	physical port only	TC and TTL actions are optional
IP over MPLS transit	NXAST_POP_L2 POP_MPLS PUSH_MPLS| [SET_FIELD->MPLS_TC | SET_MPLS_TTL] SET_FIELD->MPLS_LABEL PUSH_L2 SET_FIELD->DL_DST [SET_FIELD->VLAN_ID] OFPAT_OUTPUT	physical port only	TC and TTL actions are optional

IP over MPLS egress	NXAST_POP_L2 POP_MPLS NXAST_PUSH_L2 SET_FIELD->DL_DST [SET_FIELD- >VLAN_ID] OFPAT_OUTPU	physical port only	
MPLS tunnel egress	NXAST_POP_L2 POP_MPLS OFPAT_OUTPUT	OFPP_PW_FWD fake port only	This flow pattern is only used to decapsulate the tunnel label in L2VPN packet.
VPWS ingress	[POP_VLAN | (PUSH_VLAN SET_FIELD- >VLAN_ID) | SET_FIELD- >VLAN_ID] PUSH_MPLS [SET_FIELD- >MPLS_TC | SET_MPLS_TTL] SET_FIELD- >MPLS_LABEL PUSH_MPLS [SET_FIELD- >MPLS_TC | SET_MPLS_TTL] SET_FIELD- >MPLS_LABEL NXAST_PUSH_L2 SET_FIELD->DL_DST [SET_FIELD- >VLAN_ID] OFPAT_OUTPUT	physical port only	1. Only one optional action in [POP_VLAN | (PUSH_VLAN SET_FIELD->VLA N_ID) | SET_FIELD->VLA N_ID]can be present

VPLS ingress	Multiple VPWS ingress action list combined	physical port only	
VPLS egress	Multiple VPWS egress action list combined	physical port only	

11.12 Configuring Hybrid Openflow Flex-Mpls

11.12.1 Overview

Function Introduction

For network TAP usecase which uses OpenFlow as packet filtering and redirecting technology, there are two typical scenarios where MPLS technology is needed.

Firstly, there is a need to match MPLS packet and strip MPLS L2VPN encapsulation in order to send the inner Ethernet packet to the monitoring tools that are not capable of analyzing MPLS packet.

Secondly, there is a need to match Ethernet packet and push a MPLS label and a new outer Ethernet header using MPLS L2VPN encapsulation to help the monitoring tools that are capable of analyzing MPLS packet to decide which port the the monitored packet received on the OpenFlow device , in this case, the MPLS label is used to identify the ingress port where the monitored packet coming from.

Hybrid flexible MPLS feature addresses the two needs and can support both use cases well.

Principle Description

References:

- OpenFlow 1.3.1 specification
- RFC 3031
- RFC 3032
- RFC 4447
- RFC 4762

➢ Open vSwitch

11.12.2 Configuration

Strip MPLS L2VPN Encapsulation

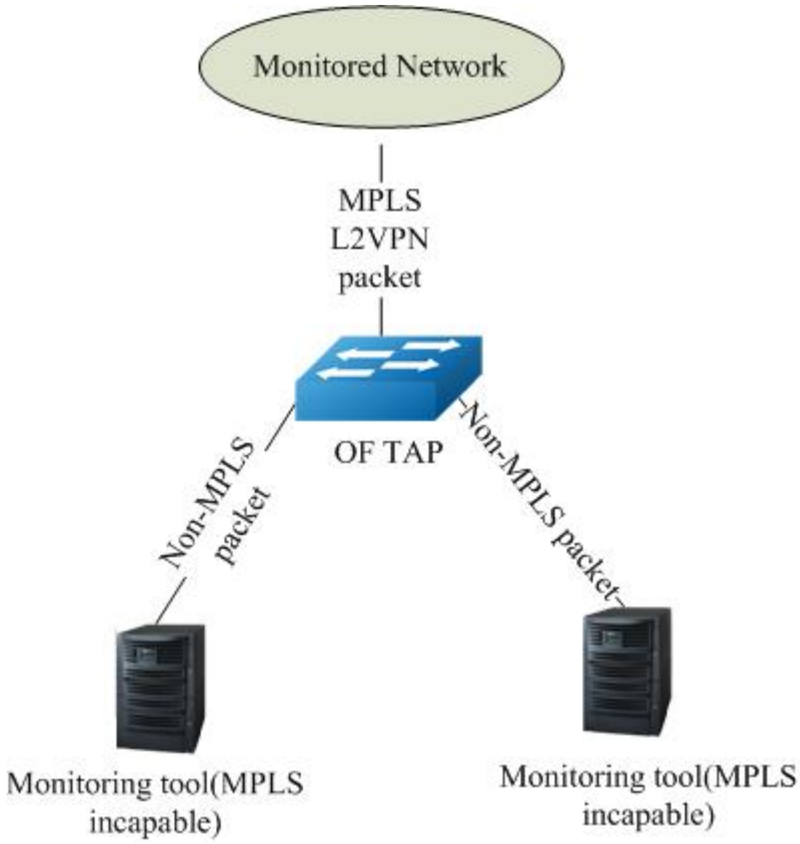

Figure 11-19 Strip Flex-Mpls Topology

step 1 Add a flow match the packet with 4 layer mpls label, label0=400, label1=300, execute the actions that pop all mpls label, strip L2 header, output to analyser

```
Switch# ovs-ofctl add-flow br0
mpls,mpls_label_num=4,mpls_label0=400,mpls_label1=300,actions=pop_l2,pop_all_mpls,o
utput:2,pop_l2,pop_all_mpls,output:3 -O openflow13
```

step 2 Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=42.247s, table=0, n_packets=0, n_bytes=0,
```



```
mpls,mpls_label_num=4,mpls_label0=400,mpls_label1=300
actions=pop_l2,pop_all_mpls,output:2,pop_l2,pop_all_mpls,output:3
```

Strip MPLS L2VPN Encapsulation and Add New L2header

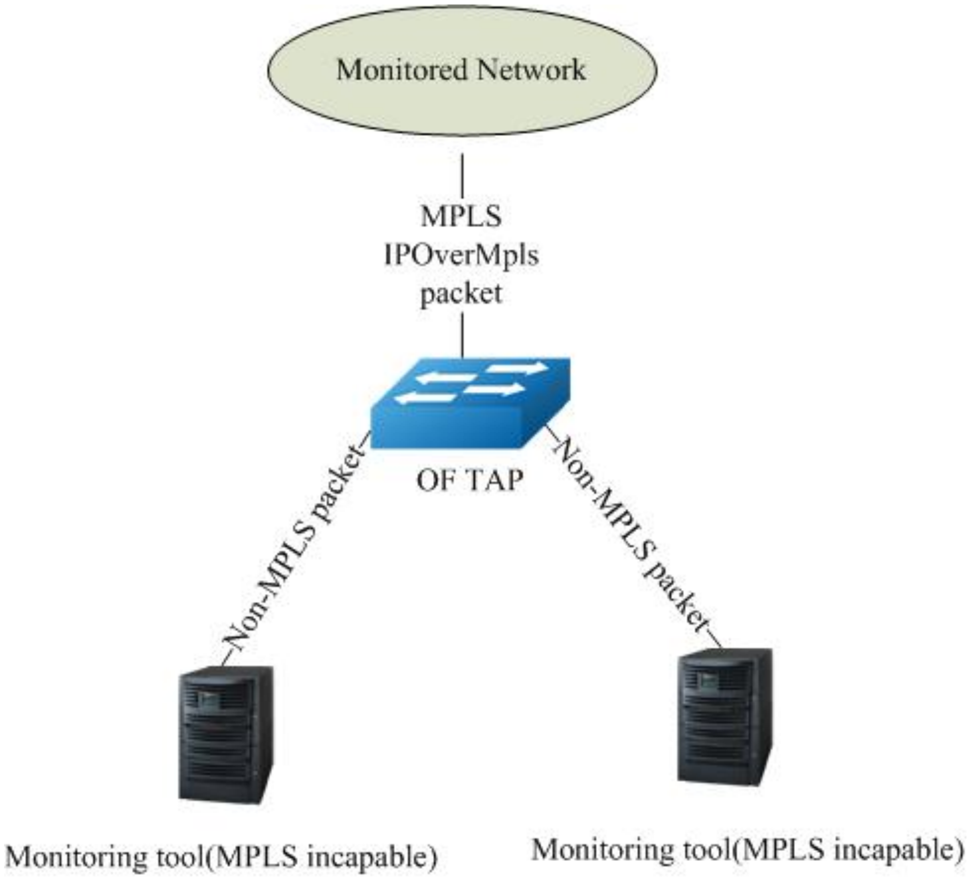

Figure 11-20 Strip Flex-Mpls and Add Header Topology

step 1 Add a flow match the packet with 4 layer mpls label, label0=400, label1=300, execute the actions pop all mpls label, strip L2 header, push a new specified L2 header, output to analyser

```
Switch# ovs-ofctl add-flow br0
"mpls,mpls_label_num=4,mpls_label0=400,mpls_label1=300,actions=pop_l2,pop_all_mpls,
push_l2,set_field:00:00:00:33:33:33-
>eth_dst,output:2,pop_l2,pop_all_mpls,push_l2,set_field:00:00:00:44:44:44-
>eth_dst,output:3" -O openflow13
```

step 2 Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=1.433s, table=0, n_packets=0, n_bytes=0,
```



```
mpls,mpls_label_num=4,mpls_label0=400,mpls_label1=300
actions=pop_l2,pop_all_mpls,push_l2,set_field:00:00:00:33:33:33-
>eth_dst,output:2,pop_l2,pop_all_mpls,push_l2,set_field:00:00:00:44:44:44-
>eth_dst,output:3
```

Add MPLS L2VPN Encapsulation

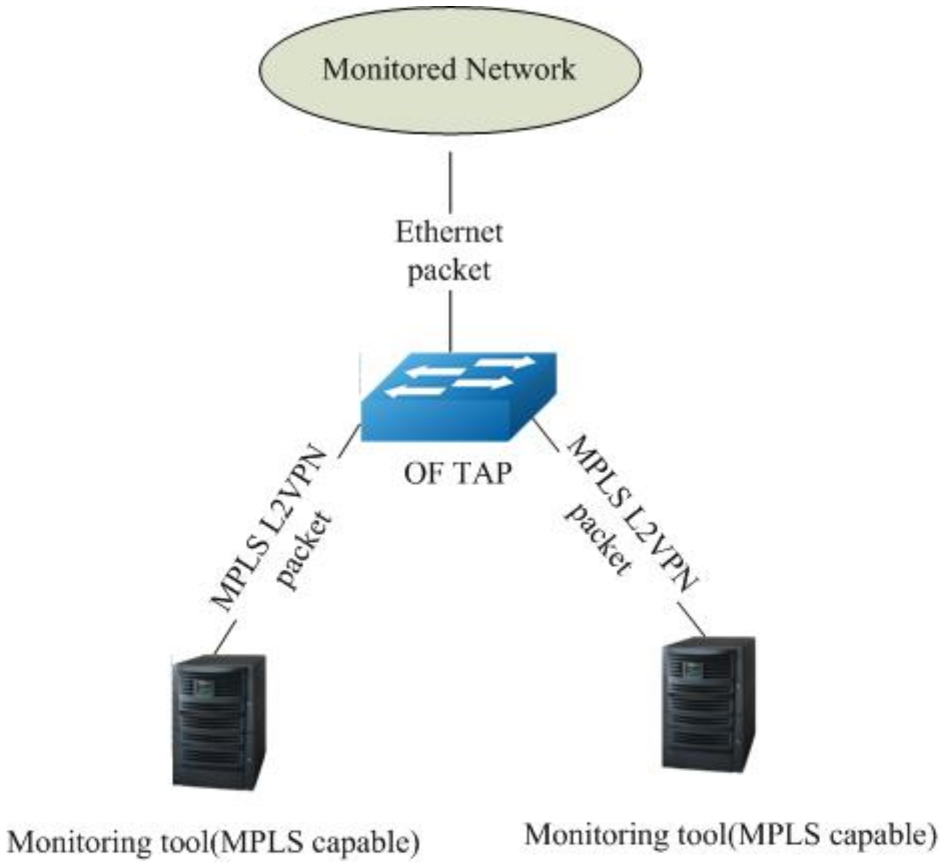

Figure 11-21 Add Flex-Mpls Header Topology

step 1 Add a flow match the packet input port 1, execute the actions push a mpls lable, push a new specified L2 header, output to analyser

```
Switch# ovs-ofctl add-flow br0 in_port=1,actions="push_mpls:0x8847,set_field:100-
>mpls_label,push_l2,set_field:00:00:00:22:22:22-
>eth_dst,output:2,push_mpls:0x8847,set_field:100-
>mpls_label,push_l2,set_field:00:00:00:33:33:33->eth_dst,output:3" -O openflow13
```

step 2 Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=1.345s, table=0, n_packets=0, n_bytes=0, in_port=1
actions=push_mpls:0x8847,set_field:100-
```



```
>mpls_label,push_l2,set_field:00:00:00:22:22:22-
>eth_dst,output:2,push_mpls:0x8847,set_field:100-
>mpls_label,push_l2,set_field:00:00:00:33:33:33->eth_dst,output:3
```

Match MPLS Packet and Redirect

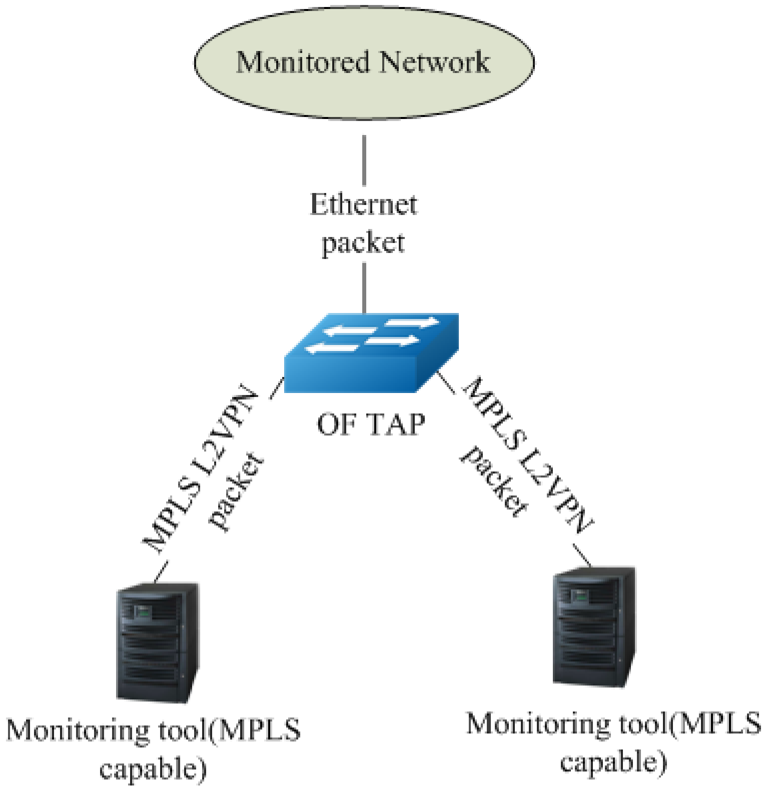

Figure 11-22 MPLS Flow Match and Redirect Topology

step 1 Add flows match the packet with one layer mpls label, mpls_label0=200/300, execute the action output to port 2/3

```
Switch# ovs-ofctl add-flow br0
mpls,mpls_label_num=1,mpls_label0=200,actions=output:2 -O openflow13
Switch# ovs-ofctl add-flow br0
mpls,mpls_label_num=1,mpls_label0=300,actions=output:3 -O openflow13
```

step 2 Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=39.302s, table=0, n_packets=0, n_bytes=0,
mpls,mpls_label_num=1,mpls_label0=200 actions=output:2
 cookie=0x0, duration=32.454s, table=0, n_packets=0, n_bytes=0,
mpls,mpls_label_num=1,mpls_label0=300 actions=output:3
```


11.13 Configuring G.8131

11.13.1 Overview

Function Introduction

G.8131/Y.1382 provides requirements and mechanisms for end-to-end trail and SNC protection switching for MPLS transport profile (MPLS-TP) networks. It describes the trail protection and SNC protection architectures types, the unit and bidirectional switching types and the revertive/non-revertive operation types. It defines the automatic protection switching (APS) protocol used to align both ends of the protected domain.

It specifies linear protection switching mechanisms to be applied to MPLS-TP layer networks as described in G.8110.1/Y.1370.1. Protection switching is a fully allocated survivability mechanism. It is fully allocated in the sense that the route and bandwidth of the protection entity is reserved for a selected working entity. It provides a fast and simple survivability mechanism. It is easier for the network operator to grasp the status of the network (e.g., active network topology) with a protection switching than with other survivability mechanisms.

The 1+1 architecture operates with unidirectional switching. The 1:1 architecture operates with bidirectional switching.

In the 1+1 architecture, a protection transport entity is dedicated to each working transport entity. The normal traffic is copied and fed to both working and protection transport entities with a permanent bridge at the source of the protected domain. The traffic on working and protection transport entities is transmitted simultaneously to the sink of the protected domain, where a selection between the working and protection transport entities is made based on some predetermined criteria, such as server defect indication.

In the 1:1 architecture, the protection transport entity is dedicated to the working transport entity. However, the normal traffic is transported either on the working transport entity or on the protection transport entity using a selector bridge at the source of the protected domain. The selector at the sink of the protected domain selects the entity that carries the normal traffic. Since source and sink need to be coordinated to ensure that the selector bridge at the source and the selector at the sink select the same entity, an APS protocol is necessary.

In the UG document, it only support 1:1 architecture.

Principle Description

References

ITU-T Recommendation G.8131.1/Y.1382.1

Terminology

- APS : Automatic Protection Switching
- DNR : Do Not Revert
- EXER : Exercise
- FS : Forced Switch
- MPLS : Multiprotocol Label Switching
- MS : Manual Switch
- OAM : Operation, Administration and Maintenance
- MPLS-TP : MPLS transport profile
- WTR : Wait-to-Restore
- SF : Signal Fail

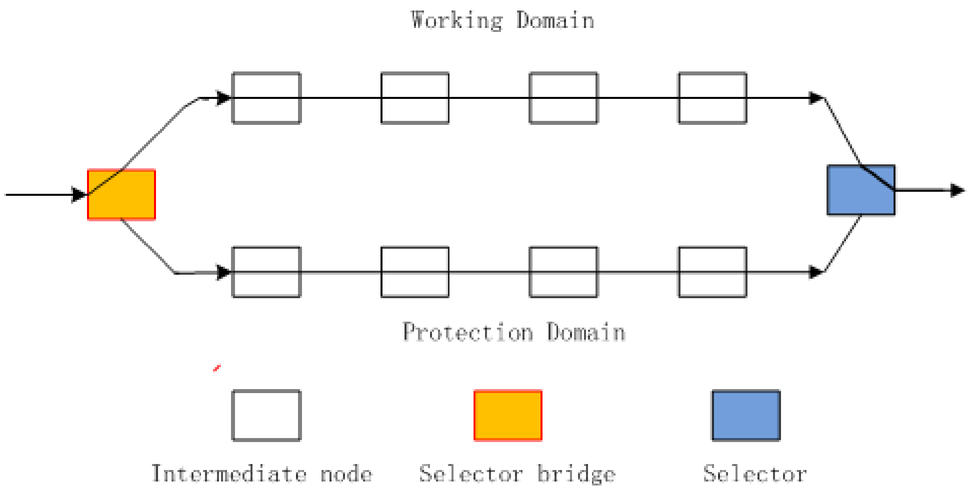

Figure 11-23 G8131 basic topology

11.13.2 Configuration

LSP APS

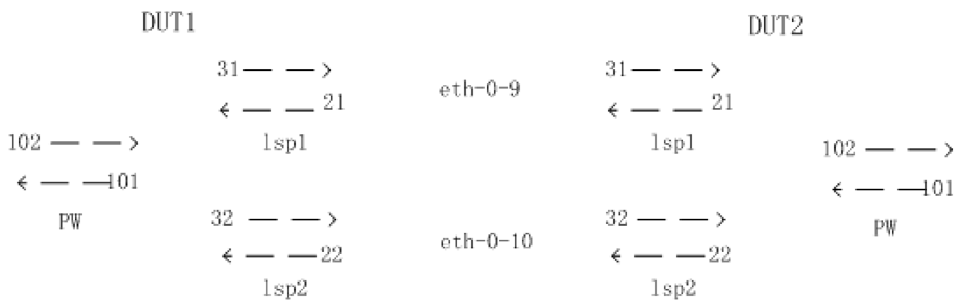

Figure 11-24 LSP APS topology

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enable openflow on interfaces

```
Switch(config)# interface range eth-0-1 - 10
Switch(config-if-range)# openflow enable
Switch(config-if-range)# no shutdown
Switch(config-if-range)# end
```

step 3 Configuring groups and flows

Switch1:

- Add group 3 of lsp_aps group type, configure lsp1 and lsp2
- Add a flow point to group 3
- Pop the mpls packets received from eth-0-1 ethernet header, and pop outer mpls label 21
- Pop the mpls packets received from eth-0-1 ethernet header, and pop outer mpls label 22
- Pop inner PW label 101 after outer label pop, and output to eth-0-1
- Configurate pw with aps oam
- Configurate lsp1 oam
- Configurate lsp2 oam


```
Switch# ovs-ofctl add-group br0
"group_id=3,type=lsp_aps,bucket=push_mpls:0x8847,set_field:31-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01-
>eth_dst,output:9,bucket=push_mpls:0x8847,set_field:32-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:10" -O openflow13
Switch# ovs-ofctl add-flow br0 "in_port=1,actions=push_mpls:0x8847,set_field:102-
>mpls_label,group:3" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=21,actions=pop_l2,pop_mpls:0x8847,PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=22,actions=pop_l2,pop_mpls:0x8847,PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=101,actions=pop_l2,pop_mpls:0x800,output:1" -O
openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=1,actions=oam_inlabel=101,push_mpls:0x8847,set_field:102-
>mpls_label,group:3" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=2,actions=oam_inlabel=21,push_mpls:0x8847,set_field:31-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:9" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=3,actions=oam_inlabel=22,push_mpls:0x8847,set_field:32-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:10" -O openflow13
```

Switch2:

- Add group 3 of lsp_aps group type, configure lsp1 and lsp2
- Add a flow point to group 3
- Pop the mpls packets received from eth-0-1 ethernet header, and pop outer mpls label 31
- Pop the mpls packets received from eth-0-1 ethernet header, and pop outer mpls label 32
- Pop inner PW label 102 after outer label pop, and output to eth-0-1
- Configurate pw with aps oam
- Configurate lsp1 oam
- Configurate lsp2 oam

```
Switch# ovs-ofctl add-group br0
"group_id=3,type=lsp_aps,bucket=push_mpls:0x8847,set_field:21-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01-
>eth_dst,output:9,bucket=push_mpls:0x8847,set_field:22-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:10" -O openflow13
Switch# ovs-ofctl add-flow br0 "in_port=1,actions=push_mpls:0x8847,set_field:101-
>mpls_label,group:3" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=31,actions=pop_l2,pop_mpls:0x8847,PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=32,actions=pop_l2,pop_mpls:0x8847,PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0
```



```
"dl_type=0x8847,mpls_label=102,actions=pop_l2,pop_mpls:0x800,output:1" -O
openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=1,actions=oam_inlabel=102,push_mpls:0x8847,set_field:101-
>mpls_label,group:3" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=2,actions=oam_inlabel=31,push_mpls:0x8847,set_field:21-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:9" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=3,actions=oam_inlabel=32,push_mpls:0x8847,set_field:22-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:10" -O openflow13
```

step 4 Configuring the revertive mode and restore time for g8131

```
Switch# configure terminal
Switch(config)# lsp-aps-group 3
Switch(lsp-aps-group-3)# g8131 time wait-to-restore 1
Switch(lsp-aps-group-3)# g8131 mode revertive
```

step 5 Validation

```
Switch# show g8131
     CS - Current State, LS - Last State, CE - Current Event,
    FE - Far end last Event, RS - Request Signal, WRSF - Working recovers from SF,
    PRSF - Protecting recovers from SF, DFOP - Failure of protocol defects
    A - APS protocol type (No APS Channel, APS Channel)
    B - Local protection architecture type (1+1, 1:1)
    D - Local protection switching type (Unidirectional, Bidirectional)
    R - Local protection operation type (Non-revertive, Revertive)
    T - Local Bridge Type (Selector, Broadcast)
    ====================================================================
     CS       LS       CE       FE       RS      A      B      D      R      T
    ---------------+-------------+------------+-------------+----------+--------+--
-----+---------+-------+----------
     NR_W     NR_W     N/A       N/A       NULL    APS    1:1    BI      REV   BR
    LSP Group ID     : 3
    Working Info      : lsp_outlabel = 31 ofport = 9
    Protection Info    : lsp_outlabel = 32 ofport = 10
    Active-Path       : Working
    WTR-Timer       : -/60(s)
    HOLD OFF-Timer  : -/0(ms)
    DFOP State       : Not in defect mode
```


PW APS Without LSP APS

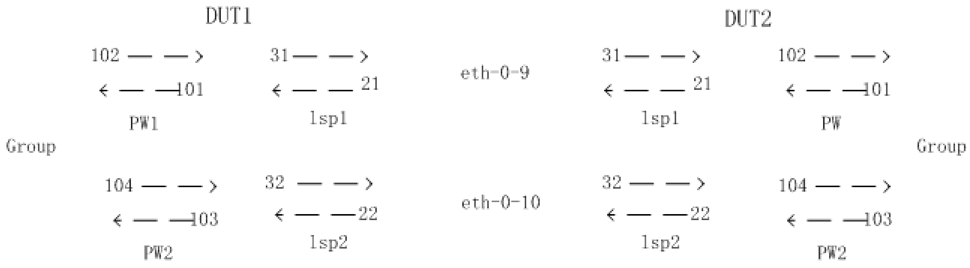

Figure 11-25 PW APS Without LSP APS topology

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enable openflow on interfaces

```
Switch(config)# interface range eth-0-1 - 10
Switch(config-if-range)# openflow enable
Switch(config-if-range)# no shutdown
Switch(config-if-range)# end
```

step 3 Configuring groups and flows

Switch1:

- Add group 4 of pw_aps group type, configure lsp1 and lsp2, pw1 and pw2
- Add a flow point to group 4
- Pop the mpls packets received from eth-0-1 ethernet header, and pop outer mpls label 21
- Pop the mpls packets received from eth-0-1 ethernet header, and pop outer mpls label 22
- Pop inner PW label 101 after outer label pop, and output to eth-0-1
- Pop inner PW label 103 after outer label pop, and output to eth-0-1
- Configurate lsp1 oam
- Configurate lsp2 oam
- Configurate pw1 without lsp oam
- Configurate pw2 without lsp oam


```
Switch# ovs-ofctl add-group br0
"group_id=4,type=pw_aps,bucket=push_mpls:0x8847,set_field:102-
>mpls_label,push_mpls:0x8847,set_field:31-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01-
>eth_dst,output:9,bucket=push_mpls:0x8847,set_field:104-
>mpls_label,push_mpls:0x8847,set_field:32-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:10" -O openflow13
Switch# ovs-ofctl add-flow br0 "in_port=1,actions=group:4" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=21,actions=pop_l2,pop_mpls:0x8847,PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=22,actions=pop_l2,pop_mpls:0x8847,PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=101,actions=pop_l2,pop_mpls:0x800,output:1" -O
openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=103,actions=pop_l2,pop_mpls:0x800,output:1" -O
openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=3,actions=oam_inlabel=21,push_mpls:0x8847,set_field:31-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:9" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=4,actions=oam_inlabel=22,push_mpls:0x8847,set_field:32-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:10" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=1,actions=oam_inlabel=101,push_mpls:0x8847,set_field:102-
>mpls_label,push_mpls:0x8847,set_field:31-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:9" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=2,actions=oam_inlabel=103,push_mpls:0x8847,set_field:104-
>mpls_label,push_mpls:0x8847,set_field:32-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:10" -O openflow13
```

Switch2:

- Add group 4 of pw_aps group type, configure lsp1 and lsp2, pw1 and pw2
- Add a flow point to group 4
- Pop the mpls packets received from eth-0-1 ethernet header, and pop outer mpls label 31
- Pop the mpls packets received from eth-0-1 ethernet header, and pop outer mpls label 32
- Pop inner PW label 102 after outer label pop, and output to eth-0-1
- Pop inner PW label 104 after outer label pop, and output to eth-0-1
- Configurate lsp1 oam
- Configurate lsp2 oam
- Configurate pw1 without lsp aps oam
- Configurate pw2 without lsp aps oam


```
Switch# ovs-ofctl add-group br0
"group_id=4,type=pw_aps,bucket=push_mpls:0x8847,set_field:101-
>mpls_label,push_mpls:0x8847,set_field:21-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01-
>eth_dst,output:9,bucket=push_mpls:0x8847,set_field:103-
>mpls_label,push_mpls:0x8847,set_field:22-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:10" -O openflow13
Switch# ovs-ofctl add-flow br0 "in_port=1,actions=group:4" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=31,actions=pop_l2,pop_mpls:0x8847,PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=32,actions=pop_l2,pop_mpls:0x8847,PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=102,actions=pop_l2,pop_mpls:0x800,output:1" -O
openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=104,actions=pop_l2,pop_mpls:0x800,output:1"-O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=3,actions=oam_inlabel=31,push_mpls:0x8847,set_field:21-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:9" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=4,actions=oam_inlabel=32,push_mpls:0x8847,set_field:22-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:10" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=1,actions=oam_inlabel=102,push_mpls:0x8847,set_field:101-
>mpls_label,push_mpls:0x8847,set_field:21-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:9" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=2,actions=oam_inlabel=104,push_mpls:0x8847,set_field:103-
>mpls_label,push_mpls:0x8847,set_field:22-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:10" -O openflow13
```

step 4 Configuring the revertive mode and restore time for g8131

```
Switch# configure terminal
Switch(config)# lsp-aps-group 3
Switch(lsp-aps-group-3)# g8131 time wait-to-restore 1
Switch(lsp-aps-group-3)# g8131 mode revertive
```

step 5 Validation

```
Switch# show g8131
 CS - Current State, LS - Last State, CE - Current Event,
FE - Far end last Event, RS - Request Signal, WRSF - Working recovers from SF,
PRSF - Protecting recovers from SF, DFOP - Failure of protocol defects
A - APS protocol type (No APS Channel, APS Channel)
B - Local protection architecture type (1+1, 1:1)
D - Local protection switching type (Unidirectional, Bidirectional)
R - Local protection operation type (Non-revertive, Revertive)
T - Local Bridge Type (Selector, Broadcast)
=======================================================================
 CS        LS        CE        FE        RS      A      B      D      R      T
--------------+--------------+------------+-------------+----------+-------+-------
-+--------+---------+------------
```



```
 NR_W   NR_W      N/A       N/A       NULL   APS  1:1    BI     REV   BR
PW Group ID       : 4
Working Info        : pw_outlabel = 102 lsp_outlabel = 31 ofport = 9
Protection Info      : pw_outlabel = 104 lsp_outlabel = 32 ofport = 10
Active-Path         : Working
WTR-Timer         : -/60(s)
HOLD OFF-Timer  : -/0(ms)
DFOP State        : Not in defect mode
```

PW APS With LSP APS

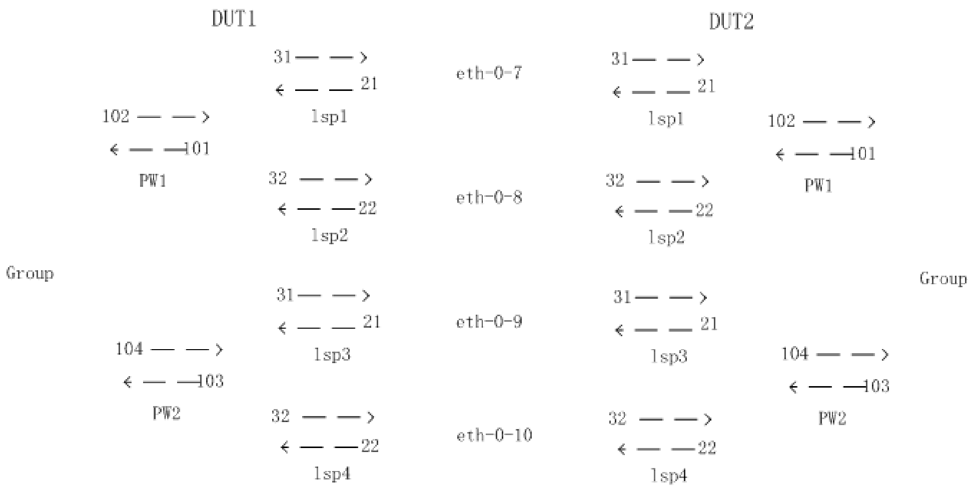

Figure 11-26 PW APS With LSP APS topology

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Enable openflow on interfaces

```
Switch(config)# interface range eth-0-1 - 10
Switch(config-if-range)# openflow enable
Switch(config-if-range)# no shutdown
Switch(config-if-range)# end
```

step 3 Configuring groups and flows

Switch1:

- Add group 1 of lsp_aps group type, configure lsp1 and lsp2
- Add group 2 of lsp_aps group type, configure lsp3 and lsp4
- Add group 3 of pw_aps group type, bind group 1 and group 2
- Add a flow point to group 3

- Pop the mpls packets received from eth-0-1 ethernet header, and pop outer mpls label 21
- Pop the mpls packets received from eth-0-1 ethernet header, and pop outer mpls label 22
- Pop the mpls packets received from eth-0-1 ethernet header, and pop outer mpls label 23
- Pop the mpls packets received from eth-0-1 ethernet header, and pop outer mpls label 24
- Pop inner PW label 101 after outer label pop, and output to eth-0-1
- Pop inner PW label 103 after outer label pop, and output to eth-0-1
- Configurate pw1 without aps oam
- Configurate pw2 without aps oam
- Configurate lsp1 oam
- Configurate lsp2 oam
- Configurate lsp3 oam
- Configurate lsp4 oam

```
Switch# ovs-ofctl add-group br0
"group_id=1,type=lsp_aps,bucket=push_mpls:0x8847,set_field:31-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01-
>eth_dst,output:6,bucket=push_mpls:0x8847,set_field:32-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:7" -O openflow13
Switch# ovs-ofctl add-group br0
"group_id=2,type=lsp_aps,bucket=push_mpls:0x8847,set_field:33-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01-
>eth_dst,output:8,bucket=push_mpls:0x8847,set_field:34-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:9" -O openflow13
Switch# ovs-ofctl add-group br0
"group_id=3,type=pw_aps,bucket=push_mpls:0x8847,set_field:102-
>mpls_label,group:1,bucket=push_mpls:0x8847,set_field:104->mpls_label,group:2" -O
openflow13
Switch# ovs-ofctl add-flow br0 "in_port=1,actions=group:3" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=21,actions=pop_l2,pop_mpls:0x8847,PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=22,actions=pop_l2,pop_mpls:0x8847,PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=23,actions=pop_l2,pop_mpls:0x8847,PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=24,actions=pop_l2,pop_mpls:0x8847,PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=101,actions=pop_l2,pop_mpls:0x800,output:1" -O
openflow13
Switch# ovs-ofctl add-flow br0
```



```
"dl_type=0x8847,mpls_label=103,actions=pop_l2,pop_mpls:0x800,output:1" -O
openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=1,actions=oam_inlabel=101,push_mpls:0x8847,set_field:102-
>mpls_label,group:1" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=2,actions=oam_inlabel=103,push_mpls:0x8847,set_field:104-
>mpls_label,group:2" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=3,actions=oam_inlabel=21,push_mpls:0x8847,set_field:31-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:6" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=4,actions=oam_inlabel=22,push_mpls:0x8847,set_field:32-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:7" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=5,actions=oam_inlabel=23,push_mpls:0x8847,set_field:33-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:8" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=6,actions=oam_inlabel=24,push_mpls:0x8847,set_field:34-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:9" -O openflow13
```

Switch2:

- Add group 1 of lsp_aps group type, configure lsp1 and lsp2
- Add group 2 of lsp_aps group type, configure lsp3 and lsp4
- Add group 3 of pw_aps group type, bind group 1 and group 2
- Add a flow point to group 3
- Pop the mpls packets received from eth-0-1 ethernet header, and pop outer mpls label 31
- Pop the mpls packets received from eth-0-1 ethernet header, and pop outer mpls label 32
- Pop the mpls packets received from eth-0-1 ethernet header, and pop outer mpls label 33
- Pop the mpls packets received from eth-0-1 ethernet header, and pop outer mpls label 34
- Pop inner PW label 102 after outer label pop, and output to eth-0-1
- Pop inner PW label 104 after outer label pop, and output to eth-0-1
- Configurate pw1 without aps oam
- Configurate pw2 without aps oam
- Configurate lsp1 oam
- Configurate lsp2 oam
- Configurate lsp3 oam

➢ Configurate lsp4 oam

```
Switch# ovs-ofctl add-group br0
"group_id=1,type=lsp_aps,bucket=push_mpls:0x8847,set_field:21-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01-
>eth_dst,output:6,bucket=push_mpls:0x8847,set_field:22-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:7" -O openflow13
Switch# ovs-ofctl add-group br0
"group_id=2,type=lsp_aps,bucket=push_mpls:0x8847,set_field:23-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01-
>eth_dst,output:8,bucket=push_mpls:0x8847,set_field:24-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:9" -O openflow13
Switch# ovs-ofctl add-group br0
"group_id=3,type=pw_aps,bucket=push_mpls:0x8847,set_field:101-
>mpls_label,group:1,bucket=push_mpls:0x8847,set_field:103->mpls_label,group:2" -O
openflow13
Switch# ovs-ofctl add-flow br0 "in_port=1,actions=group:3" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=31,actions=pop_l2,pop_mpls:0x8847,PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=32,actions=pop_l2,pop_mpls:0x8847,PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=33,actions=pop_l2,pop_mpls:0x8847,PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=34,actions=pop_l2,pop_mpls:0x8847,PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=102,actions=pop_l2,pop_mpls:0x800,output:1" -O
openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=104,actions=pop_l2,pop_mpls:0x800,output:1" -O
openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=1,actions=oam_inlabel=102,push_mpls:0x8847,set_field:101-
>mpls_label,group:1" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=2,actions=oam_inlabel=104,push_mpls:0x8847,set_field:103-
>mpls_label,group:2" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=3,actions=oam_inlabel=31,push_mpls:0x8847,set_field:21-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:6" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=4,actions=oam_inlabel=32,push_mpls:0x8847,set_field:22-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:7" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=5,actions=oam_inlabel=33,push_mpls:0x8847,set_field:23-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:8" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=6,actions=oam_inlabel=34,push_mpls:0x8847,set_field:24-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:9" -O openflow13
```

step 4 Configuring the revertive mode and restore time for g8131

```
Switch# configure terminal
Switch(config)# lsp-aps-group 1
Switch(lsp-aps-group-1)# g8131 time wait-to-restore 1
```



```
Switch(lsp-aps-group-1)# g8131 mode revertive
Switch(lsp-aps-group-1)# exit
Switch(config)# lsp-aps-group 2
Switch(lsp-aps-group-2)# g8131 time wait-to-restore 2
Switch(lsp-aps-group-2)# g8131 mode revertive
Switch(lsp-aps-group-2)# exit
Switch(config)# pw-aps-group 3
Switch(pw-aps-group-3)# g8131 time wait-to-restore 3
Switch(pw-aps-group-3)# g8131 mode revertive
```

step 5 Validation

```
Switch# show g8131
CS - Current State, LS - Last State, CE - Current Event,
FE - Far end last Event, RS - Request Signal, WRSF - Working recovers from SF,
PRSF - Protecting recovers from SF, DFOP - Failure of protocol defects
A - APS protocol type (No APS Channel, APS Channel)
B - Local protection architecture type (1+1, 1:1)
D - Local protection switching type (Unidirectional, Bidirectional)
R - Local protection operation type (Non-revertive, Revertive)
T - Local Bridge Type (Selector, Broadcast)
==================================================================
 CS        LS        CE        FE        RS      A     B     D     R     T
---------------+-------------+------------+-------------+----------+--------+------
-+---------+--------+-------
 NR_W     NR_W     N/A       N/A       NULL    APS  1:1    BI      REV  BR
LSP Group ID     : 1
Working Info      : lsp_outlabel = 31 ofport 6
Protection Info     : lsp_outlabel = 32 ofport 7
Active-Path        : Working
WTR-Timer        : -/60(s)
HOLD OFF-Timer  : -/0(ms)
DFOP State       : Not in defect mode

===================================================================
 CS        LS        CE        FE        RS      A     B     D     R     T
---------------+-------------+------------+-------------+----------+--------+------
--+--------+-------+--------
 NR_W     NR_W     N/A       N/A       NULL    APS  1:1    BI      REV  BR
LSP Group ID     : 2
Working Info      : lsp_outlabel = 33 ofport 8
Protection Info     : lsp_outlabel = 34 ofport 9
Active-Path        : Working
WTR-Timer        : -/120(s)
HOLD OFF-Timer  : -/0(ms)
DFOP State       : Not in defect mode

===================================================================
 CS        LS        CE        FE        RS      A     B     D     R     T
---------------+-------------+------------+-------------+----------+--------+------
--+--------+-------+--------
 NR_W     NR_W     N/A       N/A       NULL    APS  1:1    BI      REV  BR
PW Group ID      : 3
Working Info       : pw_outlabel = 102 lsp_aps_group = 1
Protection Info     : pw_outlabel = 104 lsp_aps_group = 2
```

```
Active-Path         : Working
WTR-Timer          : -/180(s)
HOLD OFF-Timer   : -/0(ms)
DFOP State         : Not in defect mode
```

11.14 Configuring TPOAM

11.14.1 Overview

Function Introduction

This chapter describes how to configure the Y.1731 based MPLS-TP OAM.

OAM is an important and fundamental functionality in MPLS-TP transport networks. OAM contributes to:

- The reduction of operational complexity and costs, by allowing for efficient and automatic detection, localization, and handling and diagnosis of defects, as well as by minimizing service interruptions and operational repair times.
- The enhancement of network availability, by ensuring that defects (for example, those resulting in misdirected customer traffic) and faults are detected, diagnosed, and dealt with before a customer reports the problem.
- Meeting service and performance objectives, as the OAM functionality allows for SLA verification in a multi-maintenance domain environment and allows for the determination of service degradation due, for example, to packet delay or packet loss.
- Although Y.1731 is focused on Ethernet OAM, the definition of OAM PDUs and procedures are technology independent and can also be used in MPLS-TP provided that the technology specific encapsulation is defined.

Hybrid system supports CCM, DM, LM features of Y.1731 MPLS-TP OAM.

Principle Description

References

[1] ITU-T Y.1731: OAM functions and mechanisms for Ethernet based networks [2] draft-bhh-mpls-tp-oam-y1731-07.txt

Terminology

- OAM operations, maintenance, and administration

- GACH Generic Associated Channel
- MEG Maintenance Entity Group
- MEP Maintenance End Point
- MIP Maintenance Intermediate Point
- CCM Continuity Check Message
- LSP Label Switched Path
- PW Pseudo wire

11.14.2 Configuration

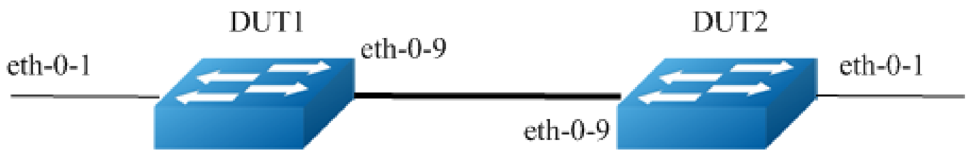

Figure 11-27 Tpoam Topology

Configuring SECTION OAM

Configuring Switch 1

Interface attribute:

```
Switch# configure terminal
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)#openflow enable
Switch(config-if-eth-0-1)# vlan-filter disable
Switch(config-if-eth-0-1)# interface eth-0-9
Switch(config-if-eth-0-9)# no shutdown
Switch(config-if-eth-0-9)#openflow enable
Switch(config-if-eth-0-9)# vlan-filter disable
Switch(config-if-eth-0-9)# end
```

Flow:

```
Switch# ovs-ofctl add-flow br0 "oam_session=4,actions=output:9" -O openflow13
```

Configuring Switch 2

Interface attribute:

```
Switch# configure terminal
Switch(config)# interface eth-0-1
```



```
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)#openflow enable
Switch(config-if-eth-0-1)# vlan-filter disable
Switch(config-if-eth-0-1)# interface eth-0-9
Switch(config-if-eth-0-9)# no shutdown
Switch(config-if-eth-0-9)#openflow enable
Switch(config-if-eth-0-9)# vlan-filter disable
Switch(config-if-eth-0-9)# end
```

Flow:

```
Switch# ovs-ofctl add-flow br0 "oam_session=4,actions=output:9" -O openflow13
```

Validation

RMEP state is OK

Switch1：

```
Switch# show mpls-tp oam-y1731 mp
SessID Type          MEG            LVL MP   LMEP CCM INTVL RMEP State
------+------------+-------------+---+---+----+---+-----+----+----------
4      Section       megdefault     7   MEP 1    En  3.3ms 1    OK
```

Switch2：

```
Switch# show mpls-tp oam-y1731 mp
SessID Type          MEG            LVL MP   LMEP CCM INTVL RMEP State
------+------------+-------------+---+---+----+---+-----+----+----------
4      Section       megdefault     7   MEP 1    En  3.3ms 1    OK
```

Switch1:

Shut down interface eth-0-9 on switch 1 and rmep state turns to fail

```
Switch# configure terminal
Switch(config)# interface eth-0-9
Switch(config-if-eth-0-9)# shutdown
Switch(config-if-eth-0-9)# end
Switch# show mpls-tp oam-y1731 mp
SessID Type          MEG            LVL MP   LMEP CCM INTVL RMEP State
------+------------+-------------+---+---+----+---+-----+----+----------
4      Section       megdefault     7   MEP 1    En  3.3ms 1    Fail
```

Switch2:

```
Switch# show mpls-tp oam-y1731 mp
SessID Type          MEG            LVL MP   LMEP CCM INTVL RMEP State
------+------------+-------------+---+---+----+---+-----+----+----------
4      Section       megdefault     7   MEP 1    En  3.3ms 1    Fail
```


Configuring LSP OAM

Configuring Switch 1

Interface attribute:

```
Switch# configure terminal
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)#openflow enable
Switch(config-if-eth-0-1)# vlan-filter disable
Switch(config-if-eth-0-1)# interface eth-0-9
Switch(config-if-eth-0-9)# no shutdown
Switch(config-if-eth-0-9)#openflow enable
Switch(config-if-eth-0-9)# vlan-filter disable
Switch(config-if-eth-0-9)# end
```

Flow:

```
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=51,actions=pop_l2,pop_mpls:0x8847, PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0 "in_port=1,actions=push_mpls:0x8847, set_field:61-
>mpls_label, push_l2,set_field:d8:44:ae:77:5e:00->eth_dst,output:9" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=3,actions=oam_inlabel=51,push_mpls:0x8847,
set_field:61->mpls_label, push_l2,set_field:d8:44:ae:77:5e:00->eth_dst,output:9" -O
openflow13
```

Configuring Switch 2

Interface attribute:

```
Switch# configure terminal
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)#openflow enable
Switch(config-if-eth-0-1)# vlan-filter disable
Switch(config-if-eth-0-1)# interface eth-0-9
Switch(config-if-eth-0-9)# no shutdown
Switch(config-if-eth-0-9)#openflow enable
Switch(config-if-eth-0-9)# vlan-filter disable
Switch(config-if-eth-0-9)# end
```

Flow:

```
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=61,actions=pop_l2,pop_mpls:0x8847, PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0 "in_port=1,actions=push_mpls:0x8847, set_field:51-
>mpls_label, push_l2,set_field:5a:eb:cd:7d:62:00->eth_dst,output:9" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=3,actions=oam_inlabel=61,push_mpls:0x8847,
```



```
set_field:51->mpls_label, push_l2,set_field:5a:eb:cd:7d:62:00->eth_dst,output:9" -O
openflow13
```

Validation

RMEP state is OK

Switch1：

```
Switch# show mpls-tp oam-y1731 mp
SessID Type          MEG            LVL MP   LMEP CCM INTVL RMEP State
------+------------+-------------+---+---+----+---+-----+----+----------
3      LSP-PE       megdefault     7   MEP 1    En  3.3ms 1    OK
```

Switch2：

```
Switch# show mpls-tp oam-y1731 mp
SessID Type          MEG            LVL MP   LMEP CCM INTVL RMEP State
------+------------+-------------+---+---+----+---+-----+----+----------
3      LSP-PE       megdefault     7   MEP 1    En  3.3ms 1    OK
```

Switch1:

Shut down interface eth-0-9 on switch 1 and rmep state turns to fail

```
Switch# configure terminal
Switch(config)# interface eth-0-9
Switch(config-if-eth-0-9)# shutdown
Switch(config-if-eth-0-9)# end
Switch# show mpls-tp oam-y1731 mp
SessID Type          MEG            LVL MP   LMEP CCM INTVL RMEP State
------+------------+-------------+---+---+----+---+-----+----+----------
3      LSP-PE       megdefault     7   MEP 1    En  3.3ms 1    Fail
```

Switch2:

```
Switch# show mpls-tp oam-y1731 mp
SessID Type          MEG            LVL MP   LMEP CCM INTVL RMEP State
------+------------+-------------+---+---+----+---+-----+----+----------
3      LSP-PE       megdefault     7   MEP 1    En  3.3ms 1    Fail
```

Configuring PW OAM

Configuring Switch 1

Interface attribute:

```
Switch# configure terminal
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)#openflow enable
Switch(config-if-eth-0-1)# vlan-filter disable
```



```
Switch(config-if-eth-0-1)# interface eth-0-9
Switch(config-if-eth-0-9)# no shutdown
Switch(config-if-eth-0-9)#openflow enable
Switch(config-if-eth-0-9)# vlan-filter disable
Switch(config-if-eth-0-9)# end
```

Flow:

```
Switch# ovs-ofctl add-flow br0 "in_port=1,actions=push_mpls:0x8847,set_field:102-
>mpls_label, push_mpls:0x8847,set_field:61-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst,output:9" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=51,actions=pop_l2,pop_mpls:0x8847, PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=101,actions=pop_l2,pop_mpls:0x800, output:1" -O
openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=4,actions=oam_inlabel=101,push_mpls:0x8847, set_field:102->mpls_label,
push_mpls:0x8847,set_field:61->mpls_label,push_l2, set_field:00:1e:08:00:02:01-
>eth_dst,output:9" -O openflow13
```

Configuring Switch 2

Interface attribute:

```
Switch# configure terminal
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)#openflow enable
Switch(config-if-eth-0-1)# vlan-filter disable
Switch(config-if-eth-0-1)# interface eth-0-9
Switch(config-if-eth-0-9)# no shutdown
Switch(config-if-eth-0-9)#openflow enable
Switch(config-if-eth-0-9)# vlan-filter disable
Switch(config-if-eth-0-9)# end
```

Flow:

```
Switch# ovs-ofctl add-flow br0 "in_port=1,actions=push_mpls:0x8847,set_field:101-
>mpls_label, push_mpls:0x8847,set_field:51-
>mpls_label,push_l2,set_field:00:1e:08:00:02:01->eth_dst, output:9" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=61,actions=pop_l2,pop_mpls:0x8847, PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=102,actions=pop_l2,pop_mpls:0x800, output:1" -O
openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=4,actions=oam_inlabel=102,push_mpls:0x8847, set_field:101->mpls_label,
push_mpls:0x8847,set_field:51->mpls_label,push_l2, set_field:00:1e:08:00:02:01-
>eth_dst,output:9" -O openflow13
```

Validation

RMEP state is OK

Switch1：

```
Switch# show mpls-tp oam-y1731 mp
SessID Type         MEG           LVL MP  LMEP CCM INTVL RMEP State
------+------------+-------------+---+---+----+---+-----+----+----------
4      PW TPE       megdefault    7   MEP 1    En  10ms 1    OK
```

Switch2：

```
Switch# show mpls-tp oam-y1731 mp
SessID Type         MEG           LVL MP  LMEP CCM INTVL RMEP State
------+------------+-------------+---+---+----+---+-----+----+----------
4      PW TPE       megdefault    7   MEP 1    En  10ms 1    OK
```

Switch1:

Shut down interface eth-0-9 on switch 1 and rmep state turns to fail

```
Switch# configure terminal
Switch(config)# interface eth-0-9
Switch(config-if-eth-0-9)# shutdown
Switch(config-if-eth-0-9)# end
Switch# show mpls-tp oam-y1731 mp
SessID Type         MEG           LVL MP  LMEP CCM INTVL RMEP State
------+------------+-------------+---+---+----+---+-----+----+----------
4      PW TPE       megdefault    7   MEP 1    En  10ms 1    Fail
```

Switch2:

```
Switch# show mpls-tp oam-y1731 mp
SessID Type         MEG           LVL MP  LMEP CCM INTVL RMEP State
------+------------+-------------+---+---+----+---+-----+----+----------
4      PW TPE       megdefault    7   MEP 1    En  10ms 1    Fail
```

Configuring LOOPBACK

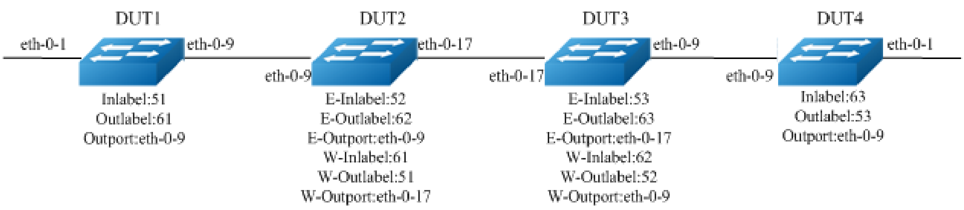

Figure 11-28 Tpoam loopback Topology

Configuring Switch 1

Interface attribute:

```
Switch# configure terminal
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
```



```
Switch(config-if-eth-0-1)#openflow enable
Switch(config-if-eth-0-1)# vlan-filter disable
Switch(config-if-eth-0-1)# interface eth-0-9
Switch(config-if-eth-0-9)# no shutdown
Switch(config-if-eth-0-9)#openflow enable
Switch(config-if-eth-0-9)# vlan-filter disable
Switch(config-if-eth-0-9)# end
```

Get route mac:

```
Switch# show route-mac
    Route MAC is: 0ca3.859f.be00
```

Flow:

```
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=51,actions=pop_l2,pop_mpls:0x8847,
    PW_FWD" -O openflow13
Switch# ovs-ofctl add-flow br0 "in_port=1,actions=push_mpls:0x8847, set_field:61-
>mpls_label,
    push_l2,set_field:68:c2:26:0d:00:00->eth_dst,output:9" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=3,actions=oam_inlabel=51,push_mpls:0x8847,
    set_field:61->vs-ofctl add-flow br0
"oam_session=4,actions=oam_inlabel=101,push_mpls:0x8847, set_field:102->mpls_label,
push_mpls:0x8847,set_field:61->mpls_label,push_l2, set_field:00:1e:08:00:02:01-
>eth_dst,output:9" -O openflow13
```

Configuring Switch 2

Interface attribute:

```
Switch# configure terminal
Switch(config)# mpls-tp node-id 1.1.1.2
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)#openflow enable
Switch(config-if-eth-0-1)# vlan-filter disable
Switch(config-if-eth-0-1)# interface eth-0-9
Switch(config-if-eth-0-9)# no shutdown
Switch(config-if-eth-0-9)#openflow enable
Switch(config-if-eth-0-9)# vlan-filter disable
Switch(config-if-eth-0-9)# interface eth-0-17
Switch(config-if-eth-0-17)# no shutdown
Switch(config-if-eth-0-17)#openflow enable
Switch(config-if-eth-0-17)# vlan-filter disable
Switch(config-if-eth-0-17)# end
```

Get route mac:

```
Switch# show route-mac
    Route MAC is: 68c2.260d.0000
```

Flow:


```
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=61,actions=pop_l2,pop_mpls:0x800,
    push_mpls:0x8847,set_field:62->mpls_label,push_l2,set_field:ac:f5:8f:7b:83:00-
>eth_dst,output:
    17" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=52,actions=pop_l2,pop_mpls:0x800,
    push_mpls:0x8847,set_field:51->mpls_label,push_l2,set_field:0c:a3:85:9f:be:00-
>eth_dst,output:
9" -O openflow13
Switch# ovs-ofctl add-flow br0 "oam_session=6,actions=oam_poplabel:61,
push_mpls:0x8847,
    set_field:51->mpls_label,push_l2,set_field:0c:a3:85:9f:be:00->eth_dst,output:9,
    oam_poplabel:52, push_mpls:0x8847,set_field:62->mpls_label,push_l2,
    set_field:ac:f5:8f:7b:83:00->eth_dst,output:17" -O openflow13
```

Configuring Switch 3

Interface attribute:

```
Switch# configure terminal
Switch(config)# mpls-tp node-id 1.1.1.3
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)#openflow enable
Switch(config-if-eth-0-1)# vlan-filter disable
Switch(config-if-eth-0-1)# interface eth-0-9
Switch(config-if-eth-0-9)# no shutdown
Switch(config-if-eth-0-9)#openflow enable
Switch(config-if-eth-0-9)# vlan-filter disable
Switch(config-if-eth-0-9)# interface eth-0-17
Switch(config-if-eth-0-17)# no shutdown
Switch(config-if-eth-0-17)#openflow enable
Switch(config-if-eth-0-17)# vlan-filter disable
Switch(config-if-eth-0-17)# end
```

Get route mac:

```
Switch# show route-mac
    Route MAC is: acf5.8f7b.8300
```

Flow:

```
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=62,actions=pop_l2,pop_mpls:0x800,
    push_mpls:0x8847,set_field:63->mpls_label,push_l2,set_field:e0:97:8e:c8:ae:00-
>eth_dst,output:
9" -O openflow13
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=53,actions=pop_l2,pop_mpls:0x800,
    push_mpls:0x8847,set_field:52->mpls_label,push_l2,set_field:68:c2:26:0d:00:00-
>eth_dst,output:
    17" -O openflow13
Switch# ovs-ofctl add-flow br0 "oam_session=6,actions=oam_poplabel:62,
push_mpls:0x8847,
```

```
    set_field:52->mpls_label,push_l2,set_field:68:c2:26:0d:00:00->eth_dst,output:17,
    oam_poplabel:53, push_mpls:0x8847,set_field:63->mpls_label,push_l2,
    set_field:e0:97:8e:c8:ae:00->eth_dst,output:9" -O openflow13
```

Configuring Switch 4

Interface attribute:

```
Switch# configure terminal
Switch(config)# interface eth-0-1
Switch(config-if-eth-0-1)# no shutdown
Switch(config-if-eth-0-1)#openflow enable
Switch(config-if-eth-0-1)# vlan-filter disable
Switch(config-if-eth-0-1)# interface eth-0-9
Switch(config-if-eth-0-9)# no shutdown
Switch(config-if-eth-0-9)#openflow enable
Switch(config-if-eth-0-9)# vlan-filter disable
Switch(config-if-eth-0-9)# end
```

Get route mac:

```
Switch# show route-mac
    Route MAC is: e097.8ec8.ae00
```

Flow:

```
Switch# ovs-ofctl add-flow br0
"dl_type=0x8847,mpls_label=63,actions=pop_l2,pop_mpls:0x8847,PW_FWD"
    -O openflow13
Switch# ovs-ofctl add-flow br0 "in_port=1,actions=push_mpls:0x8847, set_field:53-
>mpls_label,push_l2,
    set_field:ac:f5:8f:7b:83:00->eth_dst,output:9" -O openflow13
Switch# ovs-ofctl add-flow br0
"oam_session=3,actions=oam_inlabel=63,push_mpls:0x8847,set_field:53->
    mpls_label,push_l2,set_field:ac:f5:8f:7b:83:00->eth_dst,output:9" -O openflow13
```

Validation

LOOPBACK RMEP:

```
Switch# mpls-tp oam-y1731 loopback rmep 1 session 3

    Sending MPLS-TP OAM Y.1731 loopback MEP messages
    Remote MEP     :   1
    Timeout        :   5
    Repeat Count   :   1
    EXP            :   6
    (! Pass . Wait)
    !
     Loopback completed, takes 2.43 seconds.
     ----------------------------------
     Success rate is 100 percent(1/1)
```

LOOPBACK MIP:


```
Switch# mpls-tp oam-y1731 loopback mip session 3 node-id 1.1.1.3 ttl 2
    Sending MPLS-TP OAM Y.1731 loopback MIP messages
    TTL           :   2
    Timeout       :   5
    Repeat Count  :   1
    EXP           :   6
    (! Pass . Wait)
    !
     Loopback completed, takes 1.34 seconds.
     -----------------------------------
     Success rate is 100 percent(1/1)
```

LOOPBACK discovery:

```
Switch# mpls-tp oam-y1731 loopback discovery session 3 ttl from 1 to 3
    Sending MPLS-TP OAM Y.1731 loopback discovery messages
    TTL           :   [1-3]
    Timeout       :   5
    Repeat Count  :   1
    EXP           :   6
    TTL Reply    MEPID    ICC           NodeID
    ---+--------+--------+-------------+---------------
    1   MIP               megdef        1.1.1.2
    2   MIP               megdef        1.1.1.3
    3   MEP      1

     Loopback completed, takes 4.68 seconds.
     -----------------------------------
     Success rate is 100 percent(3/3)
```

11.15 Configuring Hybrid openflow ipv6 profile

11.15.1 Overview

Function Introduction

Hybrid580 switch support comprehensive flow matching ipv6 fields, and editing ipv6 fields.

Hybrid580 most can support 1400 flows in ipv6 profile.

Table 11-1 Ipv6 match fields table

Match Fields supported	Notes
Ingress Port	
Eth SRC Address	Support mask

Eth DST Address	Support mask
Eth type	
VLAN id	Support mask(vlan_tci form)
VLAN PCP	
Ipv6_src	Support mask
Ipv6_dst	Support mask
DSCP	
Ecn	
ipv6_label	Support mask
Tcp6	
Udp6	
Icmp6_type	
Icmp6_code	

Table 11-2 Ipv6 edit fields table

Actions supported	Notes
L2 field set	src_mac, dst_mac, vlan_id, etc.
Ipv6_src	
Ipv6_dst	
Ipv6_label	
Set ttl	
Ip_dscp	
Ip_ecn	
Tcp_src	
Tcp_dst	

Udp_src	
Udp_dst	
Icmp6_type	
Icmp6_code	

11.15.2 Configuration

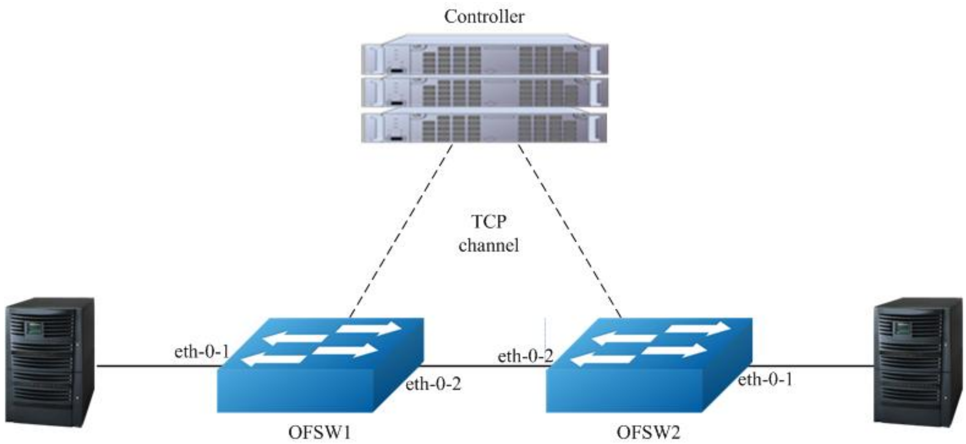

Figure 11-29 OpenFlow network topology

Switch to ipv6 profile

```
Switch# configure terminal
Switch(config)# stm prefer openflow-ipv6
```

Configuring Flow with ipv6 match key

Add a flow to filter ipv6 src-ip

Add flow

```
Switch# ovs-ofctl add-flow br0
ipv6,ipv6_src=FE80:0000:0000:0000:0200:01FF:FE00:0000,actions=output:2 -O
openflow13
```

Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=5.645s, table=0, n_packets=0, n_bytes=0,
ipv6,ipv6_src=fe80::200:1ff:fe00:0 actions=output:2
```


Add a flow to filter ipv6 dst-ip

Add flow

```
Switch# ovs-ofctl add-flow br0
ipv6,ipv6_dst=3555:5555:6666:6666:7777:7777:8888:8888,actions=output:2 -O
openflow13
```

Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=13.174s, table=0, n_packets=0, n_bytes=0,
ipv6,ipv6_dst=3555:5555:6666:6666:7777:7777:8888:8888 actions=output:2
```

Add a flow to filter ipv6 label

Add flow

```
Switch# ovs-ofctl add-flow br0 "ipv6,ipv6_label=1,actions=output:2" -O openflow13
```

Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=4.358s, table=0, n_packets=0, n_bytes=0,
ipv6,ipv6_label=0x00001 actions=output:2
```

Add a flow to filter ipv6 Traffic class

Add flow

```
Switch# ovs-ofctl add-flow br0 "ipv6,ip_dscp=1,ip_ecn=1,,actions=output:2" -O
openflow13
```

Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=63.027s, table=0, n_packets=0, n_bytes=0,
ipv6,nw_tos=4,nw_ecn=1 actions=output:2
```

Add a flow to filter ipv6 tcp6

Add flow

```
Switch# ovs-ofctl add-flow br0 "ipv6,tcp6,tp_src=100,tp_dst=200,actions=output:2" -
O openflow13
```

Validation


```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=8.707s, table=0, n_packets=0, n_bytes=0,
tcp6,tp_src=100,tp_dst=200 actions=output:2
```

Add a flow to filter ipv6 udp6

Add flow

```
Switch# ovs-ofctl add-flow br0 "ipv6,udp6,tp_src=100,tp_dst=200,actions=output:2" -
O openflow13
```

Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=29.217s, table=0, n_packets=0, n_bytes=0,
udp6,tp_src=100,tp_dst=200 actions=output:2
```

Add a flow to filter ipv6 icmp6_type,icmp6_code

Add flow

```
Switch# ovs-ofctl add-flow br0
"ipv6,icmp6,icmp_type=1,icmp_code=0,actions=output:2" -O openflow13
```

Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=5.186s, table=0, n_packets=0, n_bytes=0,
icmp6,icmp_type=1,icmp_code=0 actions=output:2
```

Configuring Flow with ipv6 edit

Edit ipv6 address

Add flow

```
Switch# ovs-ofctl add-flow br0 "ipv6,actions=set_field:201:100::1-
>ipv6_dst,output:2" -O openflow13
```

Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=5.186s, table=0, n_packets=0, n_bytes=0,
ipv6 actions=set_field:201:100::1->ipv6_dst,output:2
```


Edit ipv6 label

Add flow

```
Switch# ovs-ofctl add-flow br0 "ipv6,actions=set_field:12->ipv6_label,output:2" -O
openflow13
```

Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=5.186s, table=0, n_packets=0, n_bytes=0, ipv6
actions=set_field:12->ipv6_label,output:2
```

Edit ipv6 traffic class

Add flow

```
Switch# ovs-ofctl add-flow br0 "ipv6,actions=set_field:28->ip_dscp,set_field:1-
>ip_ecn,output:2"-O openflow13
```

Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=5.186s, table=0, n_packets=0, n_bytes=0, ipv6
actions=set_field:28->ip_dscp,set_field:1->ip_ecn,output:2
```

Edit ipv6 tcp6

Add flow

```
Switch# ovs-ofctl add-flow br0 "ipv6,tcp6,actions=set_field:100-
>tcp_dst,set_field:90->tcp_src,output:2" -O openflow13
```

Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=5.186s, table=0, n_packets=0, n_bytes=0, ipv6,tcp6
actions=set_field:100->tcp_dst,set_field:90->tcp_src,output:2
```

Edit ipv6 udp6

Add flow

```
Switch# ovs-ofctl add-flow br0 "ipv6,udp6,actions=set_field:100-
>udp_dst,set_field:200->udp_src,output:2" -O openflow13
```

Validation


```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=5.186s, table=0, n_packets=0, n_bytes=0, ipv6,udp6
actions=set_field:100->udp_dst,set_field:200->udp_src,output:2
```

Edit ipv6 icmp6

Add flow

```
Switch# ovs-ofctl add-flow br0 "ipv6,icmp6,actions=set_field:0-
>icmpv6_type,set_field:2->icmpv6_code,output:2" -O openflow13
```

Validation

```
Switch# ovs-ofctl dump-flows br0 -O openflow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=5.186s, table=0, n_packets=0, n_bytes=0, ipv6,icmp6
actions=set_field:0->icmpv6_type,set_field:2->icmpv6_code,output:2
```

