Network Packet Brokers Configuration Guide

Models: T5850-32S2Q, T5850-48S6Q, T5850-48S2Q4C, T8050-20Q4C ,T5800-8TF12S

CONFIGURATION GUIDE

Contents

List of Tables

List of Figures

Revision History

Date	Version	Description
2019-09-24	R1.0	Initial Release for T5850, T8050 Network Packet Broker (separate from T5800 Network Packet Broker)
2019-11-05	R1.1	Update document for product upgrade
2020-03-09	R1.2	Update document for product upgrade
2020-06-06	R1.3	Update document for product upgrade
2020-07-08	R1.4	Update document for product upgrade
2020-08-12	R1.5	Update document for product upgrade
2020-11-12	R1.6	Update document for product upgrade
2021-02-19	R1.7	Update document for product upgrade

1 Preface

1.1 Declaration

This document updates at irregular intervals because of product upgrade or other reason.

This document is for your reference only.

1.2 Suggestion feedback

If you have any questions when using our product and reading this document, please contact us:

Email:

1.3 Audience

This document is for the following audiences:

System maintenance engineers

Debugging and testing engineers

Network monitoring engineers

Field maintenance engineers

2 Brief Introduction

This document describes the basic conceptions, applications and usages (include network topology, configuration examples and limitations) of Network Packet Brokers .

2.1 Network Packet Broker Group Introduction

A Network Packet Broker Group has at least one ingress port and one egress port. The ingress and egress ports should be link aggregation or physical ports. Network Packet Broker series switches support 2 modes: PORT and PORT WITH FLOW.

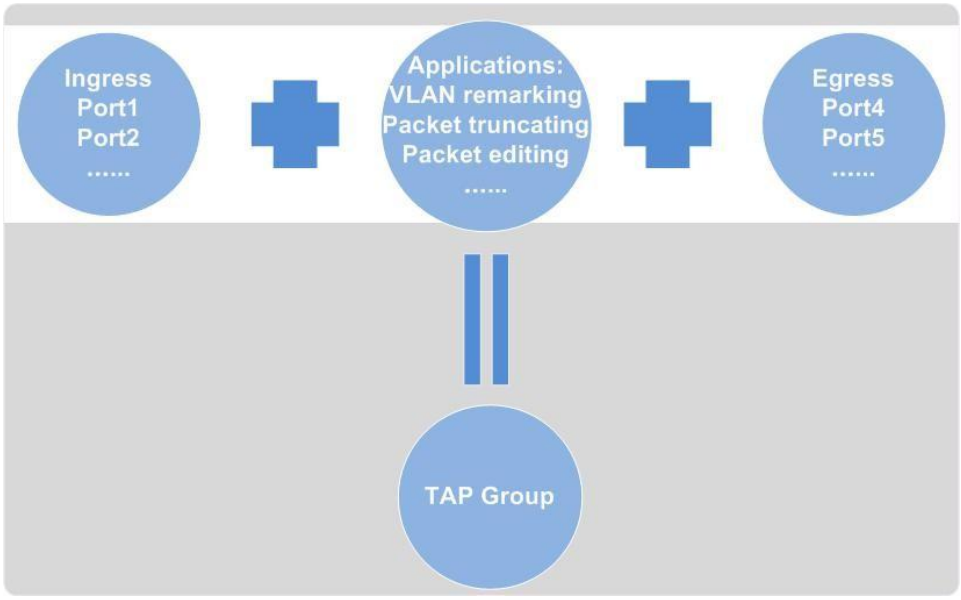

Figure 2-1 Composition of Network Packet Broker group

2.1.1 Port mode

Applications are taking effect on all packets which pass through the port.
One ingress port can only belong to one Network Packet Broker group. One Egress port can belong to several Network Packet Broker groups.
All packets enter the ingress port should be forward to the egress port.

2.1.2Port with flow mode

Applications are taking effect on packets which pass through the port and match the flow rule. One ingress port with different flow rules can join different Network Packet Broker Groups. One Egress port can join several Network Packet Broker groups.
Packets enter the ingress port should compare with the flow rule, only the packets matching the flow rule can be forward to the egress port.
E.g.: eth-0-1 with Flow A is the ingress member of Network Packet Broker group 1; eth-0-1 with Flow B is the ingress member of Network Packet Broker group 2. When the packets enter the port eth0-01, packets which match Flow A should forward to Network Packet Broker group1's egress port; packets which match Flow B should forward to Network Packet Broker group2's egress port.

2.2 FLOW types

Network Packet Brokers support 2 types of the flow: default (UDF) Flow; decap (inner-match) Flow. Default Flow is used for matching normal packets.
Decap Flow is used for matching the inner header of the packet which is encapsulated with GRE/NVGRE/VXLAN, etc.

2.3 Precondition

The following actions are supported for both PORT and PORT WITH FLOW mode:
VLAN remarking
VLAN heading stripping
Packet editing
Packet truncating
Time stamp

The following actions are only supported on PORT WITH FLOW mode:
GRE/NVGRE/VXLAN/IPIP/ERSPAN/MPLS/PPPOE/header stripping and UDF header
L2-GRE/L3-GRE/VXLAN/ERSPAN header adding
Inner header field matching

Table 1-1 Supported actionsfor different mode

Action Mode	PORT	PORT with FLOW
LAN remarking	√	√
VLAN heading stripping	√	√
Packet truncating	√	√
Packet editing	√	√
Inner header field matching	×	√
Packet header stripping	×	√
Inner VXLAN header stripping	×	√
Time STAMP (Apply to the egress port of Network Packet Broker Group)	√	√

2.4 Limitaions

Table 1-2 Mutual exclusion table

	VLAN header stripping	VLAN remarking	Packet truncating	Packet editing	Packet head stripping	Time stamp	Inner VXLAN header stripping
VLAN header stripping	N/A	×	×	√	×	√	×
VLAN remarking	×	N/A	×	√	√	√	√
Packet truncating	×	×	N/A	×	×	×	×
Packet editing	√	√	×	N/A	√	×	√
Packet head stripping	×	√	×	√	N/A	√	√

Time stamp	√	√	×	×	√	N/A	√
Inner VXLAN header stripping	×	√	×	√	√	√	N/A

√：These 2 actions can be configured together.
× : These 2 actions are mutual exclusive and cannot be configured together.

3 Device Management Configuration

Network Packet Brokers have 2 types of management ports: Ethernet port and console port. User can choose any of these management ports to manage the device.

3.1 Configuring console port for management

3.1.1 Configuration

Before you can assign 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.
The follow list describes the default value of console parameters for Network Packet Broker:
Baud rate default is 115200.
Data bits default is 8.
Stop bits default is 1.
Parity settings default is none.
User can modify the console parameters after login in the device. The following example shows how to set the baud rate as 9600:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# line console 0
Network Packet Broker(config-line)# speed 9600
```

3.1.2Validation

The following example shows how to display the configuration of the console port:

```
Network Packet Broker# show console
Current console configuration:
------------------------------------------------------------------------
line console 0
 speed 9600
 parity none
 databits 8
 stopbits 1
 exec-timeout 10 0
 privilege level 1
 no line-password
 no login
```

3.2 Configuring out band Ethernet port for management

User should set the management IP address by console port before managing the device by out band Ethernet port.

3.2.1 Configuration

Set the management IP address as 10.10.10.11/23:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# management ip address 10.10.10.11/23
```

(optional) Set the management gateway address:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# management route gateway 10.10.10.1
```

3.2.2Validation

The following example shows how to display the configuration:

```
Network Packet Broker# show management ip address
Management IPv4 address: 10.10.10.11/23
IPv4 Gateway: 10.10.10.1
```

3.3 Configuring Temperature

Network Packet Brokers support temperature alarm management.
User can configure three temperature thresholds: low, high and critical. When the temperature of the device is lower than low threshold or higher than higher threshold, the device will give an alarm. If the temperature of the device is higher than critical threshold, the device will cut off its power automatically.

NOTE:The critical threshold is not recommended to set too low, otherwise it may lead the device reboot unnecessary

3.3.1 Configuration

The following example shows how to set the low threshold of the device as 10°C; high threshold of the device as 70°C; critical threshold of the device as 85°C:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# temperature 10 70 85
```

NOTE: User can set the temperature of the board. The temperature of the chip cannot be changed.

3.3.2 Validation

The following example shows how to display the configuration of the temperature:

```
Network Packet Broker# show environment
Fan tray status:
Index     Status      Speed Rate  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        -           ALERT

Sensor status (Degree Centigrade):
Index     Temperature  Lower_alarm  Upper_alarm  Critical     Position
----------+------------+------------+------------+------------+----------
1         41           10           70           85           BEFORE_CHIP
2         43           10           70           85           BEHIND_CHIP
3         34           10           70           85           AROUND_FAN
4         41           10           70           85           AROUND_CPU
5         65           -10          100          110          SWITCH_CHIP0
```

3.4 Configuring Fan

Network Packet Brokers support to manage fan automatically according to the temperature of the board and chip.

Table 3-1 Correspondence of the chip temperature and the fan speed

Chip temperature (°C)	Work mode of the FAN	Speed rate of the FAN
≥100	Full	100%
90≤ Temperature < 100	High	80%
80≤ Temperature < 90	Low	60%
≤80	Bottom	40%

Table 3-2 Correspondence of the board temperature and the fan speed

Board temperature (°C)	Work mode of the FAN	Speed rate of the FAN
≥80	Full	100%
65 ≤ Temperature < 80	High	80%

50 ≤ Temperature < 65	Low	60%
≤50	Bottom	40%

NOTE: e.g. When the chip and the board are both 65 °C, according to Table 2-1 the FAN speed should be 40%, according to Table 2-2 the FAN speed should be 80%. The real speed should be according the higher one (80%).

3.4.1 Configuration

This application does not have any command line.

3.4.2 Validation

This application does not have any command line.

```
Network Packet Broker# 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       -          ALERT

Sensor status (Degree Centigrade):
Index     Temperature  Lower_alarm Upper_alarm  Critical     Position
----------+------------+----------+-----------+------------+----------
1         41           10         70          85          BEFORE_CHIP
2         43           10         70          85          BEHIND_CHIP
3         34           10         70          85          AROUND_FAN
4         41           10         70          85          AROUND_CPU
5         65           -10        100          110          SWITCH_CHIP0
```

3.5 Configuring Power

Network Packet Brokers support to manage power status automatically. When the power is failed or the fan is failed because of the power issue, the device should give an alarm. If power is removed or inserted, the switch should give an alarm too.

3.5.1 Configuration

This application does not have any command line.

3.5.2 Validation

The following example shows how to display the power information

```
Network Packet Broker# 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        -           ALERT

Sensor status (Degree Centigrade):
Index     Temperature  Lower_alarm  Upper_alarm  Critical     Position
----------+------------+------------+------------+------------+-----------
1         41           10           70           85           BEFORE_CHIP
2         43           10           70           85           BEHIND_CHIP
3         34           10           70           85           AROUND_FAN
4         41           10           70           85           AROUND_CPU
5         65           -10          100          110          SWITCH_CHIP0
```

3.6 Configuring Transceiver

Network Packet Brokers support to check up the information of the transceiver. 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. When the transceiver is inserted or removed or the real-time parameter is out of threshold, the switch should notice the users.

3.6.1 Configuration

This application does not have any command line.

3.6.2 Validation

The following example shows how to display the basic transceiver information:

```
Network Packet Broker# show transceiver

Port eth-0-1 transceiver info:
Transceiver Type: 1000BASE-SX
Transceiver Vendor Name : FINISAR CORP.
Transceiver PN          : FTLF8519P3BNL
Transceiver S/N         : PL36KUC
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
```

The following example shows how to display the detailed transceiver information:

```
Network Packet Broker# show transceiver detail eth-0-1

Port eth-0-1 transceiver info:
Transceiver Type: 1000BASE-SX
Transceiver Vendor Name : FINISAR CORP.
Transceiver PN          : FTLF8519P3BNL
Transceiver S/N         : PL36KUC
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
-------------------------------------------------------------
Transceiver is internally calibrated.
mA: milliamperes, dBm: decibels (milliwatts), NA or N/A: not applicable.
```

++ : high alarm, + : high warning, - : low warning, -- : low alarm.
The threshold values are calibrated.

Port	Temperature (Celsius)	High Alarm Threshold (Celsius)	High Warn Threshold (Celsius)	Low Warn Threshold (Celsius)	Low Alarm Threshold (Celsius)
eth-0-1	39.10	110.00	93.00	-30.00	-40.00

Port	Voltage (Volts)	High Alarm Threshold (Volts)	High Warn Threshold (Volts)	Low Warn Threshold (Volts)	Low Alarm Threshold (Volts)
eth-0-1	3.32	3.60	3.50	3.10	3.00

Port	Current (milliamperes)	High Alarm Threshold (mA)	High Warn Threshold (mA)	Low Warn Threshold (mA)	Low Alarm Threshold (mA)
eth-0-1	6.56	13.00	12.50	2.00	1.00

Port	Optical Transmit Power (dBm)	High Alarm Threshold (dBm)	High Warn Threshold (dBm)	Low Warn Threshold (dBm)	Low Alarm Threshold (dBm)
eth-0-1	-5.11	0.00	-3.00	-9.50	-13.50

Port	Optical Receive Power (dBm)	High Alarm Threshold (dBm)	High Warn Threshold (dBm)	Low Warn Threshold (dBm)	Low Alarm Threshold (dBm)
eth-0-1	-6.15	0.50	-1.00	-16.99	-21.02

4 Interface configuration

4.1 Configuring Interface Split

4.1.1 Configuration

The following example shows how to split a 40G port into four 10G ports:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# split interface eth-0-1 10giga
```

NOTE: User must reboot the device to take effect.

4.1.2 Validation

The following example shows how to display the splitting information:

```
Network Packet Broker# show interface status
Name      Status     Duplex  Speed   Mode   Type         Description
----------+----------+-------+-------+------+------------+-----------------
eth-0-1/1 down       auto    auto    trunk  UNKNOWN
eth-0-1/2 down       auto    auto    trunk  UNKNOWN
eth-0-1/3 down       auto    auto    trunk  UNKNOWN
eth-0-1/4 down       auto    auto    trunk  UNKNOWN
```

4.2 Configuring Interface State

4.2.1 Configuration

The following example shows how to turn up eth-0-1 and turn down eth-0-2:

4.2.2 Validation

The following example shows how to display the interface information:

```
Network Packet Broker# show interface status
Name      Status     Duplex Speed Mode Type            Description
----------+----------+------+-------+------+------------+-----------------
eth-0-1   up         a-full a-1000 trunk  1000BASE_SX
eth-0-2   admin down  auto     a-1000 trunk  1000BASE_SX
```

4.3 Configuring Interface Duplex

4.3.1 Configuration

The following example shows how to set duplex of eth-0-1 to full and duplex of eth-0-2 to auto:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# interface eth-0-1
Network Packet Broker(config-if-eth-0-1)# duplex full
Network Packet Broker(config-if-eth-0-1)# exit
Network Packet Broker(config)# interface eth-0-2
Network Packet Broker(config-if-eth-0-2)# duplex auto
```

4.3.2 Validation

The following example shows how to display the duplex information:

```
Network Packet Broker# show interface status
Name      Status     Duplex Speed Mode Type            Description
----------+----------+------+-------+------+------------+-----------------
eth-0-1   up         full   a-1000 trunk 1000BASE_SX
eth-0-2   up         a-full a-1000 trunk  1000BASE_SX
```

4.4 Configuring Interface Speed

4.4.1 Configuration

The following example shows how to set speed of eth-0-1 to 1000M:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# interface eth-0-1
Network Packet Broker(config-if-eth-0-1)# speed 1000
```

4.4.2 Validation

The following example shows how to display the speed information:

```
Network Packet Broker# show interface status
Name      Status     Duplex   Speed    Mode    Type         Description
----------+----------+--------+--------+-------+------------+-------------------
eth-0-1    up         full     1000     trunk  1000BASE_SX
```

4.5 Configuring Unidirectional

4.5.1 Configuration

The following example shows how to set unidirectional of eth-0-1:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# interface eth-0-1
Network Packet Broker(config-if-eth-0-1)# unidirectional enable
Network Packet Broker(config-if-eth-0-1)# speed 1000
Network Packet Broker(config-if-eth-0-1)# duplex full
Network Packet Broker(config-if-eth-0-1)# end
```

The following example shows how to set unidirectional rx-only of eth-0-2:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# interface eth-0-2
Network Packet Broker(config-if-eth-0-1)# unidirectional rx-only
Network Packet Broker(config-if-eth-0-1)# speed 1000
Network Packet Broker(config-if-eth-0-1)# duplex full
Network Packet Broker(config-if-eth-0-1)# end
```

4.5.2 Validation

The following example shows how to display the unidirectional information:

```
Network Packet Broker# show interface status
Name Status Duplex Speed Mode Type Description
----------+----------+--------+--------+-------+------------+-------------------
eth-0-1    up         full     1000     trunk 1000BASE_SX
eth-0-2    up         full     1000     trunk  1000BASE_SX
```

NOTE: Interface state is always up when unidirectional is enabled. Duplex auto and speed auto are not supported when unidirectional is enabled, user should set proper duplex and speed value.

4.6 Configuring Interface Errdisable

4.6.1 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

4.6.2 Configuration

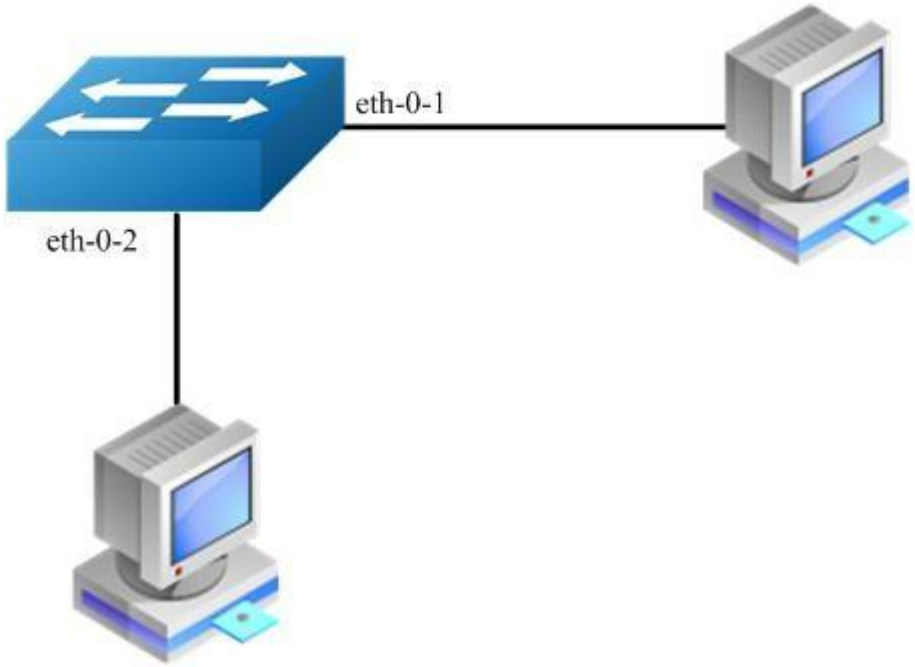

Figure 4-1 Errdisable topology

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
----------------------+----------------------
link-flap               Enabled
```

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
----------------------+----------------------
link-flap               Enabled

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
```

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

link-flap               Enabled
Timer interval: 300seconds
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
_________________       _____________

link-flap               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
```

4.6.3 Application cases

N/A

5 SSH configuration

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.

5.1.1 Configuration

The following example shows how to create a key which is named by "a":

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# rsa key a generate
```

The following example shows how to generate private key "a.pri" and public key "a.pub", then put them on the FTP server:

```
Network Packet Broker(config)# rsa key a export mgmt-if url ftp://username:password@host:port/a.pri private ssh2
Network Packet Broker(config)# rsa key a export mgmt-if url ftp://username:password@host:port/a.pub public ssh2
```

The following example shows how to download the public key from the FTP server and configure the user name of the device which need to login with SSH:

```
Network Packet Broker(config)# rsa key a.pub import mgmt-if url ftp:// username:password@host:port/a.pub public ssh2
Network Packet Broker(config)# username aaa privilege 4 password 123
Network Packet Broker(config)# username aaa assign rsa key a.pub
```

5.1.2 Validation

The following example shows how to download the private key on the client and login with SSH:

```
[Network Packet Broker@localhost]$ ssh -i a.pri aaa@10.10.33.122
```

6 Syslog configuration

System information can be saved in log file or be sent to other servers on the network. By default, The Network Packet Broker logs normal but significant system messages to its internal buffer and sends these messages to the system console.
User can check out the messages on the system console or the specified log server. The messages are time-stamped to enhance real-time debugging and management.

Table 6-1 System message types

Name	Definition
Kern	Kernel message
User	Random user level message
Mail	Mail system message
Daemon	System daemon message
Auth	Security/certification message
Syslog	Inner message generated by daemon “syslogd”
Lpr	Line printer message
News	Network news message
Uucp	UUCP message
Cron	Clock daemon message
Authpriv	Privacy security certification message
ftp	FTP message

6.2 Configuring log server

6.2.1 Configuration

The following shows how to enable the log server, how to set the IP address of the server and how to set the log level:
Network Packet Broker# configure terminal
Network Packet Broker(config)# logging server enable
Network Packet Broker(config)# logging server address mgmt-if 10.10.22.204
Network Packet Broker(config)# logging server severity debug

Table 6-2 Log level definition

Severity Level	Definition
emergency	system is unusable(0)
alert	action must be taken immediately(1)
critical	critical conditions(2)
error	error conditions(3)

warning	warning conditions(4)
notice	normal but significant condition(5)
information	Informational(6)
debug	debug-level messages(7)

6.2.2 Validation

The following example shows how to display the system log configuration information:

```
Network Packet Broker# show logging
Current logging configuration:
-------------------------------
logging buffer 500
logging timestamp bsd
logging file enable
logging level file warning
logging level module debug
logging server enable
logging server severity debug
logging server facility local4
logging server address 10.10.22.204
logging merge enable
logging merge fifo-size 1024
logging merge timeout 10
```

6.3 Configuring Logging Buffer Size

6.3.1 Configuration

The following example shows how to set the logging buffer size to 700 messages:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# logging buffer 700
```

6.3.2 Validation

The following example shows how to display the system log configuration information:

```
Network Packet Broker# 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 local4
logging server address 10.10.22.204
logging merge enable
logging merge fifo-size 1024
logging merge timeout 10
```

7 Time configuration

The devices need the correct system time in order to co-work with other devices. User can set the system date and time manually if there is no timer source outside.

7.1.1 Configuration

The following example shows how to set system time:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# clock set datetime 10:10:12 3 7 2017
```

The following example shows how to display the system time:

```
Network Packet Broker# show clock
10:10:16 Beijing Tue Mar 07 2017
Time Zone(Beijing) : UTC+08:00:00
```

8 User Management configuration

User management can improve the security level of the system. Only the authorized users can login to the system.

Table 8-1 Login modes for Network Packet Broker series switches

Mode	Definition
Login local	Login with the username and password configured in the system.
Login	Login with the password configured in the "line vty" mode.
No login	Login without password

8.2 Configuring the user management in login local mode

8.2.1 Configuration

The following example shows how to use the "login local" mode. Set username to "test", set password to "123", and choose "login local" mode:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# line vty 0 7
Network Packet Broker(config-line)# login local
Network Packet Broker(config-line)# exit
Network Packet Broker(config)# username test privilege 4 password 123
```

8.2.2 Validation

The following example shows how to login the device via Telnet:

```
Username: test
Password:
Network Packet Broker#
```

8.3 Configuring the user management in login mode

8.3.1 Configuration

The following example shows how to use the "login" mode. Set password to "123", and choose "login" mode:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# line vty 0 7
Network Packet Broker(config-line)# login
Network Packet Broker(config-line)# line-password 123
Network Packet Broker(config-line)# privilege level 4
```

8.3.2 Validation

The following example shows how to login the device via Telnet:

```
Password:
Network Packet Broker#
```

NOTE:The examples above show how to configure on Ethernet management port. The configuration of the console management port is similar as Ethernet port. Use "line console 0" to enter the console configuration mode.

8.4 Password recovery

8.4.1 Configuration

If the password is forgotten unfortunately, it can be recovered by following steps. Connect the device by console port.

Reset the system by plug out and plug in the power. The follow information will be printed on Console:

```
NAND read: device 0 offset 0x200000, size 0x400000
4194304 bytes read: OK
Press ctrl+b to stop autoboot: 5
```

Choose “no pass” mode in bootrom:

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

NOTE:After recover the password the configuration on the device may be lost. Please remember the password to avoid the service interruption.

8.4.2 Validation

Then system will reboot without loading startup-configuration. No password will be required. ## Configuring the user login with ACL ## set login acl ,and the acl name is loginACL

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# line vty 0 7
Network Packet Broker(config-line)# ip access-class loginAcl in
Notice: ACL applied on vty can only matching of source IP,destination IP,source port,or destination port for TCP packets, behaviour as WhiteList by default.
```

8.4.3 Validation

User can display the configuration files as below:

```
Network Packet Broker# show running-config

line vty 0 7
exec-timeout 0 0
privilege level 4
no line-password
ip access-class loginACL in
```

8.5 user login limit

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# login-security enable
Network Packet Broker(config)# login-security lock-duration 7
Network Packet Broker(config)# login-security max-fail-num 6 6
```

8.5.1 Validation

User can display the configuration files as below: Network Packet Broker# show running-config

```
Login Security:            Enable
Max Fail Number:           6
Fail Period:               6 min
Lock Duration:             7 min
Current Invalid Users:     0/5

Login Security Records:
User name                      Local   Locked     Resume Time(s)     Fail Count
-------------------------------------+--------+-----------+--------------------+-------------
```

9 Security Configuration Guide

9.1 Configuring Line VTY ACL

9.1.1 Overview

Function Introduction

Login through the user interface is restricted by reference to the access control list. IPv4 acls can be referenced, and login through the user interface is not restricted by default.

Currently, only matching of source IP, destination IP, source port, or destination port for TCP packets is supported, and the default is WhiteList.

Principle Description

N/A

9.1.2 Configuration

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Create ACL

```
Switch(config)# ip access-list a4
Switch(config-ip-acl-a4)# permit any src-ip host 10.0.0.1 dst-ip any
Switch(config-ip-acl-a4)# exit
```

step 3 Apply the ACL under Line VTY

```
Switch(config)# line vty 0 7
Switch(config-line)# ip access-class a4 in
Notice: ACL applied on vty can only matching of source IP,destination
IP,source port,or destination port for TCP packets, behaviour as WhiteList by
default.
Switch(config-line)# end
```

step 4 Validation

```
Switch# show vty
line vty maximum 8
line vty 0 7
 privilege level 4
 no line-password
 ip access-class a4 in
 no login
```

9.1.3 Application cases

When it is necessary to restrict the login through the user interface, that is, to control the source IP, destination IP, source port or destination port, the control action is to allow access or deny access, which can be achieved through this command.

10 SNMP configuration

SNMP is a communication protocol to connect a network management systems (NMS) and agents. It defines the standardized management frame work, common communication language, security and access control mechanism for monitoring and managing the devices in the network environment. Via SNMP, the administrator can connect to the device to query the information, modify the configuration, monitor the state, get the failures and generate a report automatically.

NOTE: Network Packet Brokers support SNMP V1/V2, Only part of the OID and trap are supported.

10.1 Configuring SNMP GET

10.1.1 Configuration

The following example shows how to set the SNMP community word:

```
Network Packet Broker(config)# snmp-server community test read-only
```

The following example shows how to enable SNMP service:

```
Network Packet Broker(config)# snmp-server enable
```

10.1.2 Validation

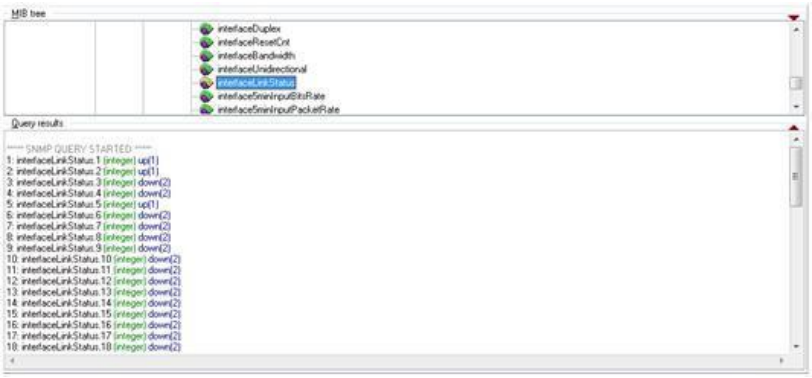

Figure 10-1 Display the OID interfaceLinkStatus by applications

10.2 Configuring SNMP TRAP

10.2.1 Configuration

The following example shows how to set the SNMP TRAP server IP and the SNMP community word:

```
Network Packet Broker(config)# snmp-server trap target-address mgmt-if 10.10.22.215 community public
```

The following example shows how to enable SNMP TRAP service:

```
Network Packet Broker(config)# snmp-server trap enable all
```

10.2.2 Validation

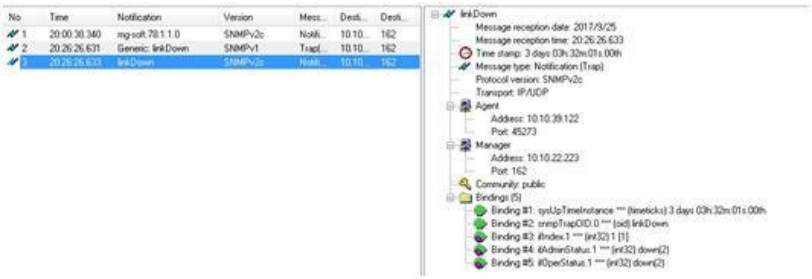

Figure 10-2 Display the Trap information of linkDown by applications

10.3 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.

10.3.1 Configuration

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set the globle configurations for SNMP

Set engineID; Set the user name, password, and authentication type; Create SNMP server; 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
```

10.3.2 Validation

```
Switch# show running-config
snmp-server engineID 8000123456
snmp-server usm-user usr1 authentication md5 mypassword privacy des yourpassword
snmp-server group grp1 user usr1 security-model usm
snmp-server access grp1 security-model usm noauth
```

10.4 SNMPv1 and SNMPv2 notifications configure

Beginning in privileged EXEC mode, follow these steps to configure SNMP on the switch.

10.4.1 Configuration

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 “10.0.0.2”; Configure a remote trap manager which IPv6 address is “2001:1000::1”.

```
Switch(config)# snmp-server trap enable all
Switch(config)# snmp-server trap target-address 10.0.0.2 community public
Switch(config)# snmp-server trap target-address 2001:1000::1 community public
```

step 3 Exit the configure mode

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

10.4.2 Validation

```
Switch# show running-config
snmp-server trap target-address 10.0.0.2 community public
```

```
snmp-server trap target-address 2001:1000::1 community public
snmp-server trap enable system
snmp-server trap enable coldstart
snmp-server trap enable warmstart
snmp-server trap enable linkdown
snmp-server trap enable linkup
```

10.5 Configuring SNMPv3 notifications

10.5.1 Configuration

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; Configure a remote trap manager's IP address; Configure a remote trap manager's IPv6 address; Add a local user to SNMPv3 notifications.

```
Switch(config)# snmp-server trap enable all
Switch(config)# snmp-server notify notif1 tag tmptag trap
Switch(config)# snmp-server target-address targ1 param parm1 10.0.0.2 taglist tmptag
Switch(config)# snmp-server target-address t1 param p1 2001:1000::1 taglist tag1
Switch(config)# snmp-server target-params parm1 user usr1 security-model v3 message-processing v3 noauth
```

step 3 Exit the configure mode

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

10.5.2 Validation

```
Switch# show running-config
snmp-server notify notif1 tag tmptag trap
snmp-server target-address t1 param p1 2001:1000::1 taglist tag1
snmp-server target-address targ1 param parm1 10.0.0.2 taglist tmptag
snmp-server target-params parm1 user usr1 security-model v3 message-processing v3 noauth
snmp-server trap enable system
snmp-server trap enable coldstart
snmp-server trap enable warmstart
snmp-server trap enable linkdown
snmp-server trap enable linkup
```

10.6 Configuring SNMP ACL

10.6.1 Configuration

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Configuring ACL

Either the acl is configured to continue to configure the ace before it is applied to SNMP, or the acl is configured to be applied to SNMP before it is configured to ace.

```
Switch(config)# ip access-list a4
Switch(config-ip-acl-a4)# permit src-ip host 10.10.25.25
Switch(config-ip-acl-a4)# exit
Switch(config)#
```

step 3 Apply ACL to SNMP

```
Switch(config)# snmp-server access-group a4 in
```

step 4 Exit the configure mode

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

10.6.2 Validation

```
Switch# show running-config
Building configuration...
version 2.1.9.8.1
!
!
snmp-server enable
snmp-server access-group a4 in
!
snmp-server community public read-write
!
ip access-list a4
 10 permit src-ip host 10.10.25.25
 exit
!
!
!
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
 privilege level 4
 no line-password
 no login
```

11 File Copy Configuration

11.1 Copy the file form the flash of device

The following example shows how to copy the file named “diagnostic-information.txt”.

11.1.1 Copy to TFTP server

```
Network Packet Broker# copy flash:/diagnostic-information.txt mgmt-if tftp://10.10.38.160
TFTP server [10.10.38.160]
Name of the TFTP file to access []diagnostic-information.txt
```

11.1.2 Copy to FTP server

```
Network Packet Broker# copy flash:/diagnostic-information.txt mgmt-if ftp://10.10.25.33
FTP server [10.10.25.33]
User name [] test
Password []
Name of the FTP file to access []diagnostic-information.txt
```

11.1.3 Copy to USB disk

```
Network Packet Broker# copy flash:/diagnostic-information.txt udisk:
```

11.2 Copy the file to the flash of device

11.2.1 Copy from TFTP server

```
Network Packet Broker# copy mgmt-if tftp://10.10.38.160/diagnostic-information.txt flash:
```

11.2.2 Copy from FTP server

```
Network Packet Broker# copy mgmt-if ftp://10.10.25.33/diagnostic-information.txt flash:/
FTP server [] 10.10.25.33
User name [] test
Password []
Name of the FTP file to access []diagnostic-information.txt
```

11.2.3 Copy from USB disk

```
Network Packet Broker# copy udisk:/diagnostic-information.txt flash:
```

12 M:N configuration

12.1 Networking requirements

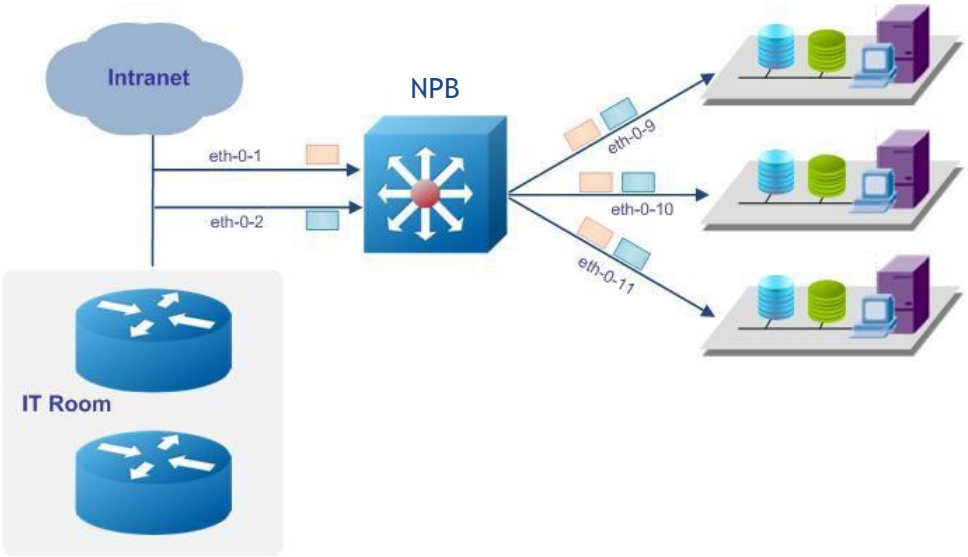

Figure 12-1 Topology of M:N networking:

12.2 Configuration Ideas

In some cases, packets enter the device from different port need to be sent to different monitors. Therefore Network Packet Broker Selective Any-to-Any Port Mapping is required. The packets enter the ingress ports will send copies to all egress ports. Reference to Figure 10-1: Packets enter eth-0-1 will send copies to eth-0-9/eth-0-10/eth-0-11. Packets enter eth-0-1 will also send copies to eth-0-9/eth-0-10/eth-0-11.

12.3 Configuration

The following example shows to create a Network Packet Broker group with ingress port eth-0-1/eth-0-2, with egress port eth-0-9/eth-0-10/eth-0-11:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1
Network Packet Broker(config-tap-tap1)# ingress eth-0-2
Network Packet Broker(config-tap-tap1)# egress eth-0-9
Network Packet Broker(config-tap-tap1)# egress eth-0-10
Network Packet Broker(config-tap-tap1)# egress eth-0-11
```

12.4 Validation

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show tap-group

tap-group tap1
 ID: 10
 Ingress:
      eth-0-1
      eth-0-2
 egress:
      eth-0-9
      eth-0-10
      eth-0-11
```

12.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config

tap-group tap1 1
 ingress eth-0-1
 ingress eth-0-2
 egress eth-0-9
 egress eth-0-10
 egress eth-0-11
```

13 Load Balance Configuration(HASH)

13.1 Networking requirements

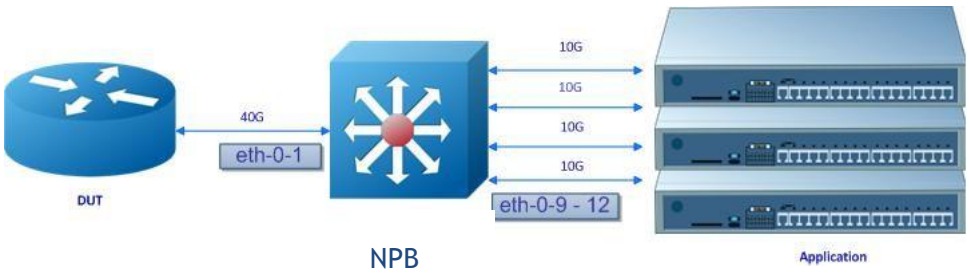

Figure 13-1 Topology of load balance:

13.2 Configuration Ideas

In some cases, the capability of the port is 40G/s, but the capability of the server or analyzer is 10G/s. Therefore, load balance is required to resolve this problem. Reference to Figure 11-1, eth-0-1 is a 40G port, Agg1 is a link aggregation port with four 10G members (eth-0-9/eth-0-10/eth-0-11/eth-0-12). Packets enter eth-0-1 should choose an outgoing port among eth-0-9/eth-0-10/eth-0-11/eth-0-12, according the load balance rule.

13.3 Configuration

The following example shows how to add eth-0-9/eth-0-10/eth-0-11/eth-0-12 into the link aggregation port Agg1:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# interface eth-0-9
Network Packet Broker(config-if-eth-0-9)# static-channel-group 1
Network Packet Broker(config-if-eth-0-9)# interface eth-0-10
Network Packet Broker(config-if-eth-0-10)# static-channel-group 1
Network Packet Broker(config-if-eth-0-10)# interface eth-0-11
Network Packet Broker(config-if-eth-0-11)# static-channel-group 1
Network Packet Broker(config-if-eth-0-11)# interface eth-0-12
Network Packet Broker(config-if-eth-0-12)# static-channel-group 1
```

The flowing example shows how to create a Network Packet Broker group with ingress port eth-0-1, egress port Agg1:

```
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1
Network Packet Broker(config-tap-tap1)# egress agg1
```

The flowing example shows how to set the load balance rule to hash by source MAC address (The default rule is hash by source IP, destination IP, source port, destination port):

```
Network Packet Broker(config)# port-channel load-balance set src-mac
Network Packet Broker(config)# end
```

(Optional) support detailed hash rule, e.g. inner IP/ inner MAC, .etc.

```
Network Packet Broker(config)# port-channel load-balance set inner-dst-ip
Network Packet Broker(config)# end
```

The follow command is necessary if user enable to load balance by inner fields:

```
Network Packet Broker(config)# port-channel load-balance tunnel-hash-mode both
```

13.4 Validation

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show tap-group

tap-group tap1
```

```
ID: 10
 Ingress:
      eth-0-1
 egress:
      agg1
```

The following example shows how to display the load balance rule:

```
Network Packet Broker# show port-channel load-balance
Port-channel load-balance hash fields:
-------------------------------------

src-mac
src-ip
dst-ip
src-port-l4
dst-port-l4
```

13.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
port-channel load-balance set src-mac
!
interface eth-0-9
 static-channel-group 1
!
interface eth-0-10
 static-channel-group 1
!
interface eth-0-11
 static-channel-group 1
!
interface eth-0-12
 static-channel-group 1
!
tap-group tap1 1
  ingress eth-0-1
  egress agg1
```

Table 13-1 load balance fields

Load balance field	Description
src-mac	Load balance by source MAC address
dst-mac	Load balance by destination MAC address
src-ip	Load balance by source IP address
dst-ip	Load balance by destination IP address
ip-protocol	Load balance by ip-protocol
src-port-l4	Load balance by source port

dst-port-l4	Load balance by destination port
vxlan-vni	Vni of vxlan
inner-dst-mac	Inner Source MAC address based load balancing
inner-src-mac	Inner Destination MAC address based load balancing
inner-src-ip	Inner Source IP address based load balancing
inner-dst-ip	Inner Destination IP address based load balancing
inner-src-port-l4	Inner Source Port based load balancing
inner-dst-port-l4	Inner Destination Port based load balancing
gre-key	Key of GRE
nvgre-vsid	Vsid of nvgre
nvgre-flow-id	Flow ID of GRE

14 Load Balance Configuration(RR)

14.1 Networking requirements

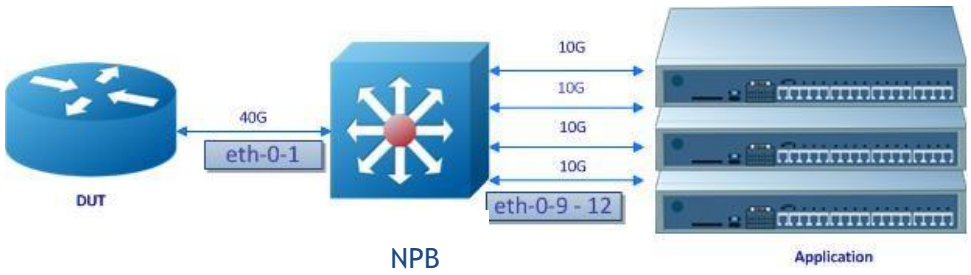

Figure 14-1 Topology of load balance

14.2 Configuration Ideas

In some cases, the capability of the port is 40G/s, but the capability of the server or analyzer is 10G/s. Therefore, load balance is required to resolve this problem. Reference to Figure 11-1, eth-0-1 is a 40G port, Agg1 is a link aggregation port with four 10G members (eth-0-9/eth-0-10/eth-0-11/eth-0-12). Packets enter eth-0-1 should choose an outgoing port among eth-0-9/eth-0-10/eth-0-11/eth-0-12, according the round-robin rule.

14.3 Configuration

The flowing example shows how to set the load balance mode to round-robin:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# port-channel 1 load-balance-mode round-robin
```

NOTE:Network Packet Broker series device supports at most 16 link aggregation ports to use round-robin mode. Round-robin mode must configure before ink aggregation port is created.

The following example shows how to add eth-0-9/eth-0-10/eth-0-11/eth-0-12 into the link aggregation port Agg1:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# interface eth-0-9
Network Packet Broker(config-if-eth-0-9)# static-channel- 1
Network Packet Broker(config)# interface eth-0-10
Network Packet Broker(config-if0)# static-channel-group 1
Network Packet Broker(config)# interface eth-0-11
Network Packet Broker(config-if1)# static-channel-group 1
Network Packet Broker(config)# interface eth-0-12
Network Packet Broker(config-if2)# static-channel-group 1
```

The flowing example shows how to create a Network Packet Broker group with ingress port eth-0-1, egress port Agg1:

```
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1
Network Packet Broker(config-tap-tap1)# egress agg1
```

14.4 Validation

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show tap-group
!
tap-group tap1
ID: 10
 Ingress:
      eth-0-1
 egress:
```

```
    agg1
```

14.5 Configuration file

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show running-config
!
port-channel 1 load-balance-mode round-robin
!
interface eth-0-9
 static-channel-group 1
!
interface eth-0-10
 static-channel-group 1
!
interface eth-0-11
 static-channel-group 1
!
interface eth-0-12
 static-channel-group 1
!
tap-group tap1 1
  ingress eth-0-1
  egress agg1
```

15 Ingress PORT with FLOW configuration

15.1 Configuring basic Flow

15.1.1 Networking requirements

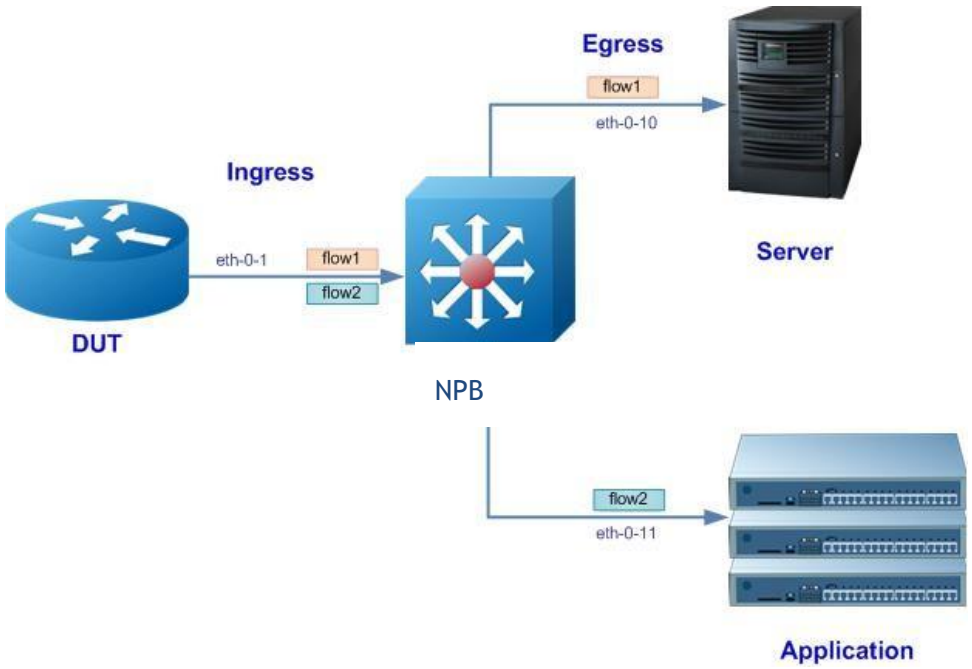

Figure 15-1 Topology of PORT with FLOW

15.1.2 Configuration Ideas

In some cases, packets from one interface need to copy to different outgoing ports. Use the PORT with FLOW Network Packet Broker groups can redirect the packets to different ports. Reference to Figure 13-1 packets with source IP address 1.1.1.0/24 or 2.2.2.0/24 should copy to eth-0-10. Packets with source IP address 10.1.1.0/24 or 20.1.1.0/24 should copy to eth-0-11 Packets with other source IP address should be discard.

15.1.3 Configuration

The follow example shows how to create a Flow rule:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# flowflow1
Network Packet Broker(config-flow-flow1)# permit any src-ip 1.1.1.0 0.0.0.255 dst-ip any
Network Packet Broker(config-flow-flow1)# permit any src-ip 2.2.2.0 0.0.0.255 dst-ip any
Network Packet Broker(config-flow-flow1)# exit
Network Packet Broker(config)# flow flow2
Network Packet Broker(config-flow-flow2)# permit any src-ip 10.1.1.0 0.0.0.255 dst-ip any
Network Packet Broker(config-flow-flow2)# permit any src-ip 20.1.1.0 0.0.0.255 dst-ip any
NOTE:The packets not matched by the flow rule should be discarded by default.
The following example shows how to create a Network Packet Broker group with flow1 and flow2:
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow flow1
Network Packet Broker(config-tap-tap1)# egress eth-0-10
Network Packet Broker(config-tap-tap1)# exit
Network Packet Broker(config)# tap-group tap2
Network Packet Broker(config-tap-tap2)# ingress eth-0-1 flow flow2
Network Packet Broker(config-tap-tap2)# egress eth-0-11
```

15.1.4 Validation

The following example shows how to display the flow rule information:

```
Network Packet Broker# show flow1
flow flow1
 sequence-num 10 permit any src-ip 1.1.1.0 0.0.0.255 dst-ip any
 sequence-num 20 permit any src-ip 2.2.2.0 0.0.0.255 dst-ip any
flow flow2
 sequence-num 10 permit any src-ip 10.1.1.0 0.0.0.255 dst-ip any
 sequence-num 20 permit any src-ip 20.1.1.0 0.0.0.255 dst-ip any
```

The following example shows how to display the Network Packet Broker group information:

```
Network Packet Broker# show tap-group

tap-group tap1
ID: 10
  Ingress:
      eth-0-1         flow flow1
  egress:
      eth-0-10
tap-group tap2
ID: 2
  Ingress:
      eth-0-1         flow flow2
  egress:
      eth-0-11
```

15.1.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
flow flow1
 sequence-num 10 permit any src-ip 1.1.1.0 0.0.0.255 dst-ip any
 sequence-num 20 permit any src-ip 2.2.2.0 0.0.0.255 dst-ip any
!
flow flow2
 sequence-num 10 permit any src-ip 10.1.1.0 0.0.0.255 dst-ip any
 sequence-num 20 permit any src-ip 20.1.1.0 0.0.0.255 dst-ip any
!
tap-group tap1 1
ingress eth-0-1 flow flow1
egress eth-0-10
!
tap-group tap2 2
ingress eth-0-1 flow flow2
egress eth-0-11
```

Table 15-1 Flow rule fields

Field	Description
➢ IP protocol[number| any| icmp|igmp|gre| nvgre| tcp |udp|]	➢ Specify the IP protocol number of the flow rule. Well known IP protocols can also be specified by name. e.g. IP protocol 1 = icmp, 2 = igmp, 6 = tcp, 17 = udp, 47 = gre/nvgre (gre protocol 0x0800 = gre, 0x6558 = nvgre). Parameter

	"any" indicates packets with any IP protocol can match this rule.
src-ip/src-ipv6	Source IPv4/IPv6 address
dst-ip/dst-ipv6	Destination IPv4/IPv6 address
flow-label	Flow label of IPv6
Inner-match	Specify the inner match profile of the flow rule. The inner-match profile is created by "inner-match" command in global configuration mode.
ip-precedence	IP precedence
src-port	Source layer 4 port
dst-port	Destination layer 4 port
first-fragment	Match packets with first fragment
non-first-fragment	Match packets with non first fragment
non-fragment	Match packets with non fragment
non-or-first-fragment	Match packets with non first fragment
small-fragment	Match packets with small fragment
any-fragment	Match packets with any fragment
options	Match packets with IP options
dscp	DSCP in IPv4 packets value
vxlan-vni	VNI of VXLAN
vlan	Vlan ID
inner-vlan	Inner vlan ID
cos	CoS value in vlan header
inner-cos	CoS value in inner vlan header
ether-type	Ether type
src-mac	Source mac address
dst-mac	Destination mac address
Ipv4-head	IPv4 packet header

l4-head	Layer 4 header

Table 15-2 Flow rule actions

Action	Description
un-tag/un-tag-outer-vlan/un-tag-inner-vlan	Remove vlan tags of the packets.
mark-source	Specify additional outer vlan id of the outgoing packets.
edit-macda	Edit the destination mac address of the outgoing packet.
edit-macsa	Edit the source mac address of the outgoing packet.
edit-ipda/edit-ipv6da	Edit the destination IPv4/IPv6 address of the outgoing packet.
edit-ipsa/edit-ipv6sa	Edit the source IPv4/IPv6 address of the outgoing packet.
edit-vlan	Edit the vlan tag of the outgoing packet
strip-header	Strip the gre/nvgre/vxlan/ipip/erspan/mpls/pppoe header
truncation	Truncate the packet

15.2 Configuring UDF Flow

15.2.1 Networking requirements

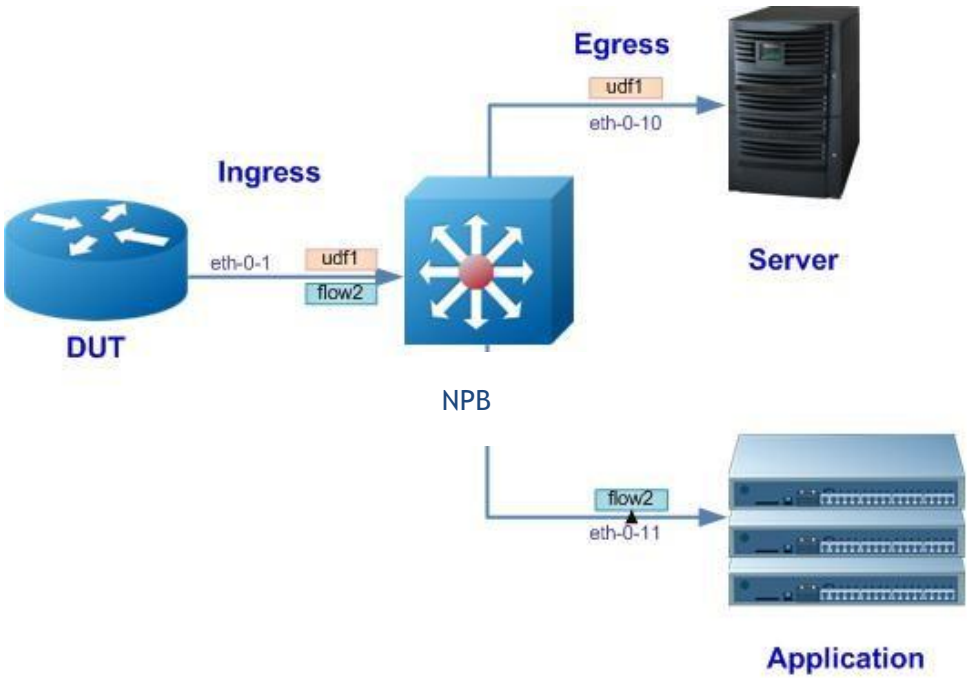

Figure 15-2 Topology of UDF FLOW

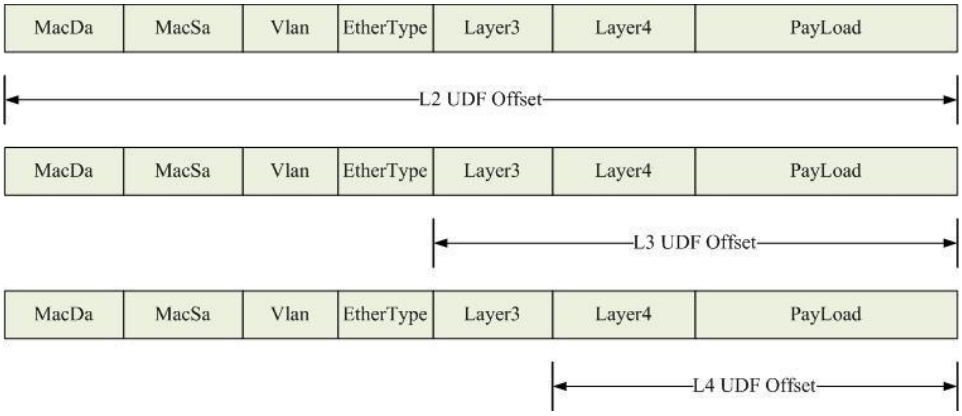

Figure 15-3 Packet structure for match the UDF flow rule

15.2.2 Configuration Ideas

Table 15-3 In some cases, user needs more detailed rules to filter the packets. The Network Packet Broker UDF (User defined format) can accurately match the specified field UDF use the specified value and the reversed wildcard bits to match the field which is concerned. An offset is needed to point out the position in the packet to match the UDF field.

15.2.3 Configuration

The following example shows how to create an IPv4 UDF rule, match field " 00 ab cd ef", the offset is 30 bytes starting at the layer 3 header.

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# flow udf1
Network Packet Broker(config-flow-udf1)# permit any src-ip any dst-ip any ipv4-head 0xabcdef 0x0 30
```

NOTE:These examples are based on IPv4 rules. The following example shows how to create basic flow rule:

```
Network Packet Broker(config)# flow flow2
Network Packet Broker(config-flow-map2)# permit any src-ip 10.1.1.0 0.0.0.255 dst-ip any
Network Packet Broker(config-flow-map2)# permit any src-ip 20.1.1.0 0.0.0.255 dst-ip any
```

The following example shows how to create a Network Packet Broker group with udf1 and flow2:

```
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow udf1
Network Packet Broker(config-tap-tap1)# egress eth-0-10
Network Packet Broker(config-tap-tap1)# exit
Network Packet Broker(config)# tap-group tap2
Network Packet Broker(config-tap-tap2)# ingress eth-0-1 flow flow2
Network Packet Broker(config-tap-tap2)# egress eth-0-11
```

NOTE:The Network Packet Broker series device supports one profile for IPv4 UDF rule, or three profiles for layer 4 UDF rule. The IPv4 UDF rule and layer 4 UDF rule are mutual exclusive. Layer 4 UDF rule has 2 key words: protocol & offset. If any of the key words is different, they belong to different profiles. IPv4 UDF rue has 1 key word: offset. (One profile for IPv4 UDF rule means all IPv4 UDF rule should use same offset)

15.2.4 Validation

The following example shows how to display the IPv4 UDF rules:

```
Network Packet Broker# show flow
flow udf1
 sequence-num 10 permit any src-ip any dst-ip any ipv4-head 0x00abcdef 0x00000000 30
flow flow2
 sequence-num 10 permit any src-ip 10.1.1.0 0.0.0.255 dst-ip any
 sequence-num 20 permit any src-ip 20.1.1.0 0.0.0.255 dst-ip any
```

The following example shows how to display the Network Packet Broker group:

```
Network Packet Broker# show group

tap-group 1
ID: 10
 Ingress:
      eth-0-1         flow udf1
 egress:
      eth-0-10

tap-group 2
ID: 2
 Ingress:
      eth-0-1         flow flow2
 egress:
      eth-0-11
```

15.2.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
flow udf1
 sequence-num 10 permit any src-ip any dst-ip any ipv4-head 0x00abcdef 0x00000000 30
!
flow flow2
 sequence-num 10 permit any src-ip 10.1.1.0 0.0.0.255 dst-ip any
 sequence-num 20 permit any src-ip 20.1.1.0 0.0.0.255 dst-ip any
```

```
!
tap-group tap1 1
ingress eth-0-1 flow udf1
egress eth-0-10
!
tap-group tap2 2
ingress eth-0-1 flow flow2
egress eth-0-11
```

15.3 Configuring Inner-match

15.3.1 Networking requirements

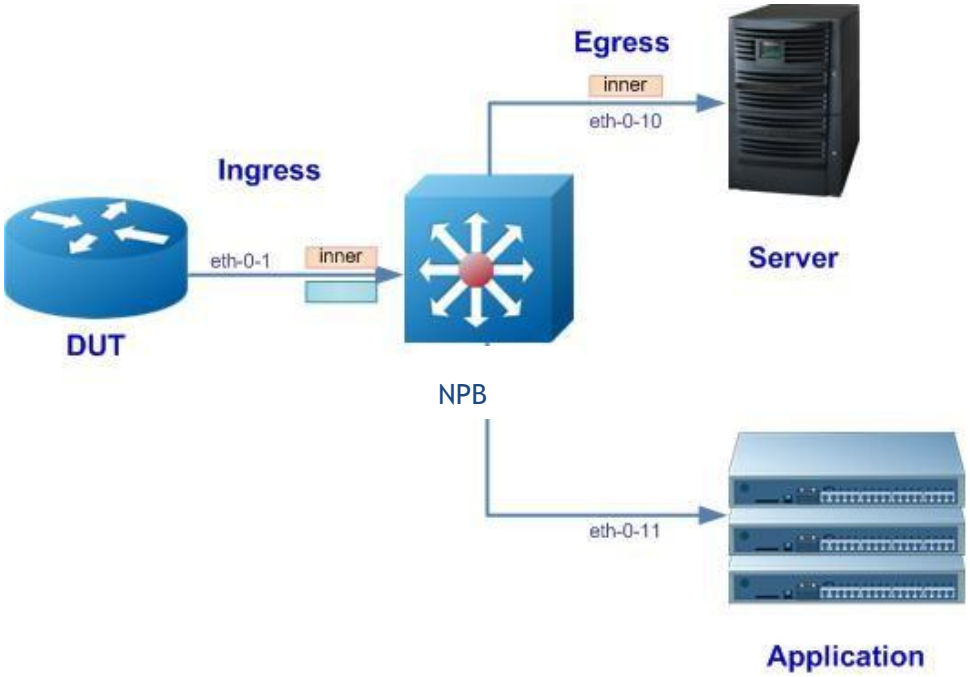

Figure 15-4 Topology of Inner match

Figure 15-5 Packet for inner-match

15.3.2 Configuration Ideas

In some cases, user needs to match the inner field of GRE/NVGRE/VXLAN packets. To meet the requirement, use the inner-match configuration.

15.3.3 Configuration

The following example shows how to create a inner-match profile, matching the destination IP address 1.1.1.1 or 1.1.1.2:

```
Network Packet Broker(config)# inner-match imf
Network Packet Broker(config-inner-match-imf)# match any src-ip any dst-ip 1.1.1.1 0.0.0.0
Network Packet Broker(config-inner-match-imf)# match any src-ip any dst-ip 1.1.1.2 0.0.0.0
Network Packet Broker(config-inner-match-imf)# exit
```

The following example shows how to create a Flow with decap enabled, matching the GRE packets with destination IP address 11.1.1.1, NVGRE packets with the destination IP address 12.1.1.1, VXLAN packets with the destination IP address 13.1.1.1, and apply the inner-match imf to this flow:

```
Network Packet Broker(config)# flow inner type decap
Network Packet Broker(config-flow-inner)# permit gre src-ip any dst-ip 11.1.1.1 0.0.0.0 inner-match imf
```

```
Network Packet Broker(config-flow-inner)# permit nvgre src-ip any dst-ip 12.1.1.1 0.0.0.0 inner-match imf
Network Packet Broker(config-flow-inner)# permit udp dst-port eq 4789 src-ip any dst-ip 13.1.1.1 0.0.0.0 inner-match imf
```

NOTE:To match the VXLAN packets, set the type to UDP and set the destination port to 4789.

Create a Network Packet Broker Group and apply the flow inner match to the ingress interface:

15.3.4 Validation

The following example shows how to display the inner-match rule and the flow rule:

```
Network Packet Broker# show inner-match
inner-match imf
 sequence-num 1 match any src-ip any dst-ip host 1.1.1.1
 sequence-num 2 match any src-ip any dst-ip host 1.1.1.2

Network Packet Broker# show flow
flow inner type decap
 sequence-num 10 permit gre src-ip any dst-ip host 11.1.1.1 inner-match imf
 sequence-num 20 permit nvgre src-ip any dst-ip host 12.1.1.1 inner-match imf
 sequence-num 30 permit udp dst-port eq 4789 src-ip any dst-ip host 13.1.1.1 inner-match imf
```

NOTE: Flows with decap enabled and disabled cannot bind to the same interface. E.g. eth-0-1 with decap flow inner is the ingress of Network Packet Broker Group 1, so eth-0-1 cannot bind with other flows without decap in any other Network Packet Broker groups.

The following example shows the error notification when configure different types of flow:

```
DUT1(config)# flow flow1 type decap
DUT1(config-flow-flow1)# exit
DUT1(config)# flow flow2
DUT1(config-flow-flow2)# exit
DUT1(config)# Network Packet Broker-group tap1
DUT1(config-Network Packet Broker-tap1)# ingress eth-0-1 flow flow1
DUT1(config-Network Packet Broker-tap1)# exit
DUT1(config)#Network Packet Broker-group tap2
DUT1(config-Network Packet Broker-tap2)# ingress eth-0-1 flow flow2
% Interface mode conflict
```

Reference to the Topology of Inner match, packets remark with blue rectangle is not matched by any flow rule so they should be discarded.

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show Network Packet Broker-group

tap-group tap1
ID: 10
 Ingress:
      eth-0-1         flow inner
 egress:
      eth-0-10
```

15.3.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
inner-match imf
 sequence-num 1 match any src-ip any dst-ip host 1.1.1.1
 sequence-num 2 match any src-ip any dst-ip host 1.1.1.2
!
flow inner type decap
 sequence-num 10 permit gre src-ip any dst-ip host 11.1.1.1 inner-match imf
 sequence-num 20 permit nvgre src-ip any dst-ip host 12.1.1.1 inner-match imf
 sequence-num 30 permit udp dst-port eq 4789 src-ip any dst-ip host 13.1.1.1 inner-match imf
!
tap-group tap1 1
```

```
ingress eth-0-1 flow inner
egress eth-0-10
```

15.4 Configuring L2 UDF Flow

15.4.1 Networking requirements

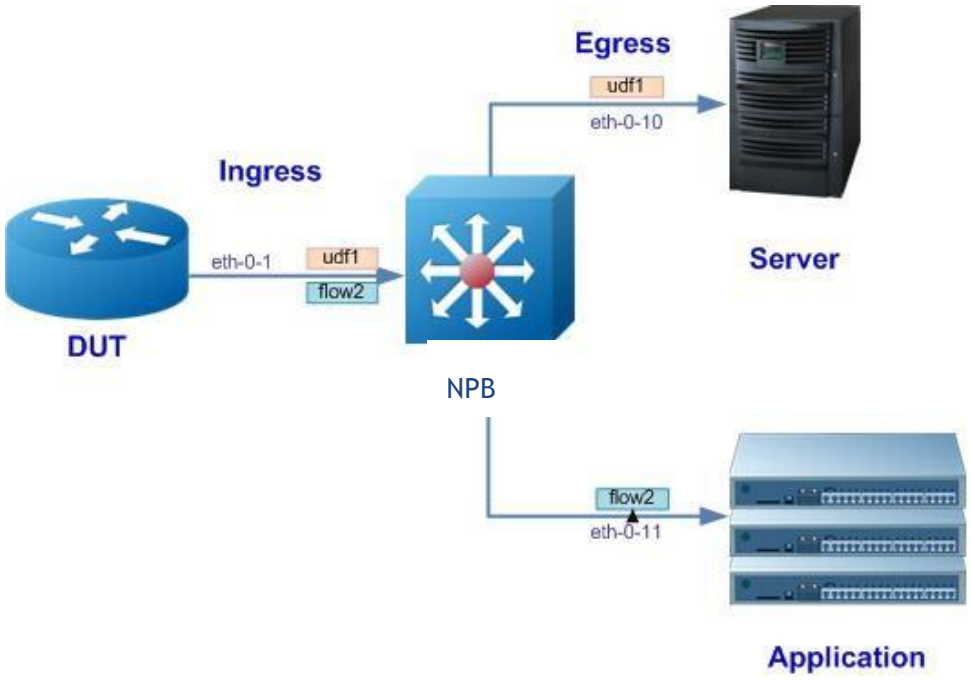

Figure 15-6 Topology of UDF FLOW

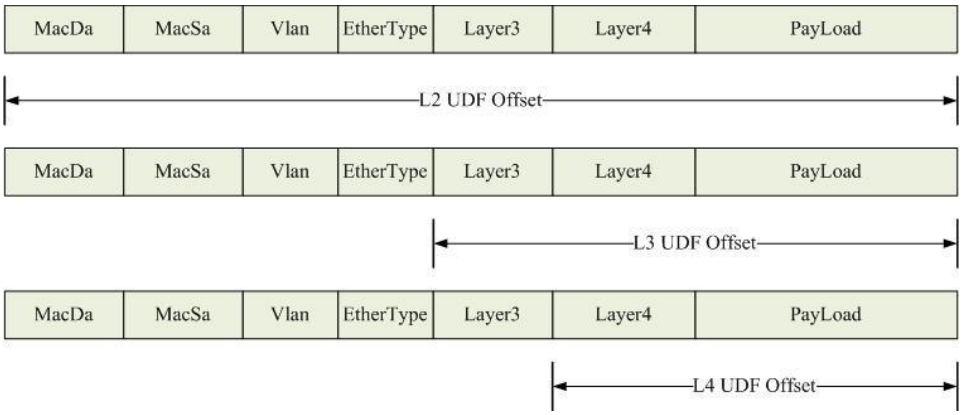

Figure 15-7 Packet structure for match the UDF flow rule

Figure 15-8 L2-L4 header for common packets

type	l2-head offset	l3-head offset	l4-head offset
TCP	Ethernet header	IP header	TCP header
UDP	Ethernet header	IP header	UDP header
ICMP	Ethernet header	IP header	ICMP header

GRE	Ethernet header	Outer IP header	GRE header
VXLAN	Ethernet header	Outer IP header	Outer UDP header
MPLS	Ethernet header	Outer MPLS label	IP header
VPLS	Ethernet header	Outer MPLS label	Inner Ethernet header

15.4.2 Configuration Ideas

In some cases, user needs more detailed rules to filter the packets. The Network Packet Broker UDF (User defined format) can accurately match the specified field UDF use the specified value and the reversed wildcard bits to match the field which is concerned. An offset is needed to point out the position in the packet to match the UDF field.

15.4.3 Configuration

The UDF function is enhanced on Network Packet Broker product and configured by new CLI. UDF support get maximum 4 bytes from 4 separated offset position from packets' L2 header.

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# udf 0 offset-type l2-header
Network Packet Broker(config-udf-5)# match ether-type 0x8926
Network Packet Broker(config-udf-5)# offset offset0 30 offset1 31
```

The following example shows how to create UDF flow rule:

```
Network Packet Broker(config)# flow udf
Network Packet Broker(config-flow-udf)# sequence-num 10 permit any src-ip any dst-ip any ether-type 0x8926 0x0 udf udf-id 0 udf0 0x50 0x00 udf1 0x60 0x00
```

The following example shows how to create a Network Packet Broker group with UDF applied on ingress port:

```
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow udf
Network Packet Broker(config-tap-tap1)# egress eth-0-2
```

NOTE:The maximum number of UDF entry on system is 4.

15.4.4 Validation

The following example shows how to display the UDF flow configuration:

```
Network Packet Broker# show flow
flow udf
 sequence-num 10 permit any src-ip any dst-ip any ether-type 0x8926 0x0 udf udf-id 0 udf0 0x50 0x00 udf1 0x60 0x00
```

The following example shows how to display the Network Packet Broker group:

```
Network Packet Broker# showtap-group
truncation            144
timestamp-over-ether : 0000.0000.0000 0000 0000.0000 0x0000

tap-group tap1
  ID: 10
  Ingress:
      eth-0-1          flow udf
  Egress:
      eth-0-2
```

The following example shows how to display the UDF entry configuration:

```
Network Packet Broker# show udf
Udf Global Information:
```

```
 Offset Unit : 1 Bytes

Udf Index 0
 Udf Type : l2 header
 Udf Match-Field:
   ether-type 0x8926 0x0
 Offset : 30|31|n/a|n/a
```

15.4.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# showrunning-config
udf 0 offset-type l2-header
 match ether-type 0x8926
 offset offset0 30 offset1 31
!
flow udf
 sequence-num 10 permit any src-ip any dst-ip any ether-type 0x8926 0x0 udf udf-id 0 udf0 0x50 0x00 udf1 0x60 0x00
 exit
!
tap-group tap1 1
 ingress eth-0-1 flow udf
 egress eth-0-2
```

16 Egress Port Filter configuration

16.1 Networking requirements

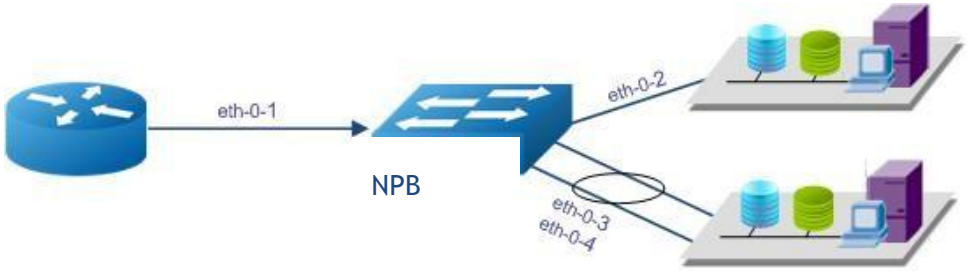

Figure 16-1 Topology of port filter usage

16.2 Configuration Ideas

In some cases, after packets forward to the destination port, a filter is required to discard some unneeded packets. Reference to The Figure, packets with source IP address 1.0.0.0/24 from eth-0-1 should forward to eth-0-2 and Agg1(with two members eth-0-3/eth-0-4). Eth-0-3 need to monitor the web packets, Agg1 need to monitor all packets.

16.3 Configuration

The following example shows how to add eth-0-3/eth-0-4 into the link aggregation port Agg1:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# interface eth-0-3
Network Packet Broker(config-if-eth-0-3)# static-channel-group 1
Network Packet Broker(config-if-eth-0-3)# interface eth-0-4
Network Packet Broker(config-if-eth-0-4)# static-channel-group 1
```

The following example shows how to create the filter:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# ip access-list filter1
Network Packet Broker(config-acl-filter1)# permit tcp dst-port eq 80 src-ip any dst-ip any
Network Packet Broker(config-acl-filter1)# exit
Network Packet Broker(config)# ip access-list filter2
Network Packet Broker(config-acl-filter2)# deny tcp dst-port eq 80 src-ip any dst-ip any
Network Packet Broker(config-acl-filter2)# permit any src-ip any dst-ip any
Network Packet Broker(config-acl-filter2)# end
```

NOTE: After apply the filter to the egress port, Packets which not matched by any filter rule should be discard by default.

The following example shows how to apply the filter:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# interface eth-0-2
Network Packet Broker(config-if-eth-0-2)# egress filter1
Network Packet Broker(config-if-eth-0-2)# exit
Network Packet Broker(config)# interface agg1
Network Packet Broker(config-if-agg1)# egress filter2
```

The following example shows to create a Network Packet Broker group with ingress port eth-0-1, with egress port eth-0-2/Agg1:

```
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1
Network Packet Broker(config-tap-tap1)# egress agg1
Network Packet Broker(config-tap-tap1)# egress eth-0-2
```

16.4 Validation

The following example shows how to display the filter rules:

```
Network Packet Broker# show ipaccess-list
ip access-list filter1
 sequence-num 10 permit tcp dst-port eq 80 src-ip any dst-ip any
ip access-list filter2
 sequence-num 10 deny tcp dst-port eq 80 src-ip any dst-ip any
 sequence-num 20 permit any src-ip any dst-ip any
```

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show tap-group

tap-group tap1
ID: 10
 Ingress:
      eth-0-1
 egress:
      eth-0-2
      agg1
```

16.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
ip access-list filter1
 sequence-num 10 permit tcp dst-port eq 80 src-ip any dst-ip any
!
ip access-list filter2
 sequence-num 10 deny tcp dst-port eq 80 src-ip any dst-ip any
 sequence-num 20 permit any src-ip any dst-ip any
!
interface eth-0-2
 egress filter1
!
interface eth-0-3
 static-channel-group 1
!
interface eth-0-4
 static-channel-group 1
!
interface agg1
 egress filter2
!
tap-group tap1 1
ingress eth-0-1
egress eth-0-2
egress agg1
```

Table 16-1 Network Packet Broker Filter fields

Field	Description
IP protocol[number| any| icmp| igmp|gre| nvgre| tcp |udp|]	Specify the IP protocol number of the flow rule. Well known IP protocols can also be specified by name. e.g. IP protocol 1 = icmp, 2 = igmp, 6 =tcp, 17 = udp, 47 = gre/nvgre (gre protocol 0x0800 = gre, 0x6558 = nvgre). Parameter “any” indicates

	packets with any IP protocol can match this rule.
src-ip/src-ipv6	Source IPv4/IPv6 address
dst-ip/dst-ipv6	Destination IPv4/IPv6 address
flow-label	Flow label of IPv6
ip-precedence	IP precedence
first-fragment	Match packets with first fragment
non-first-fragment	Match packets with non first fragment
non-fragment	Match packets with non fragment
non-or-first-fragment	Match packets with non first fragment
small-fragment	Match packets with small fragment
any-fragment	Match packets with any fragment
options	Match packets with IP options
dscp	DSCP in IPv4 packets value
vlan	Vlan ID
inner-vlan	Inner vlan ID
cos	CoS value in vlan header
inner-cos	CoS value in inner vlan header
ether-type	Ether type
src-mac	Source mac address
dst-mac	Destination mac address
Ipv4-head	IPv4 packet header
l4-head	Layer 4 header

17 VLAN Remarking Configuration

17.1 Networking requirements

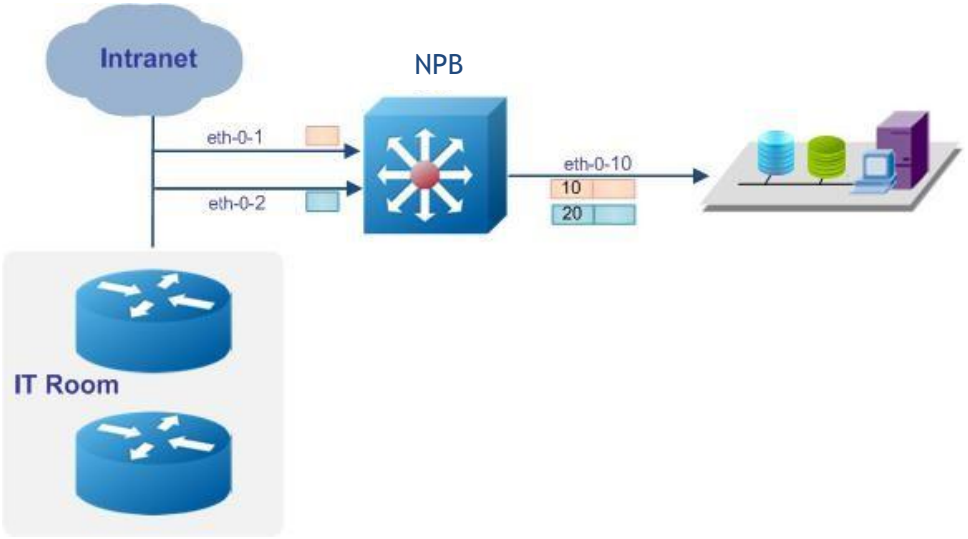

Figure 17-1 Topology of VLAN Remarking

17.2 Configuration Ideas

Figure 17-2 In some cases, the server and analyzer need to separate different packets. The VLAN Remarking application can meet the requirement. Reference to the Figure Packets from eth-0-1 should add VLAN tag 10. Packets from eth-0-2 should add VLAN tag 20.

17.3 Configuration

PORT mode and PORT WITH FLOW mode both support VLAN remarking.

17.3.1 VLAN Remarking for PORT mode

The following example shows how to create Network Packet Broker group, and remark the VLAN tag to 10 for the packets form eth-0-1, remark the VLAN tag to 20 for the packets form eth-0-2:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 mark-source 10
Network Packet Broker(config-tap-tap1)# ingress eth-0-2 mark-source 20
Network Packet Broker(config-tap-tap1)# egress eth-0-10
```

17.3.2 VLAN Remarking for PORT WITH FLOW mode

The following example shows how to create Network Packet Broker group, and remark the VLAN tag to 10 for the packets with destination IP 1.1.1.1 form eth-0-1, remark the VLAN tag to 20 for the packets with destination IP 1.1.1.2 form eth-0-2:

```
Network Packet Broker(config)# flowflow1
Network Packet Broker(config-flow-map1)# permit any src-ip any dst-ip 1.1.1.1 0.0.0.0 mark-source 10
Network Packet Broker(config)# flow flow2
Network Packet Broker(config-flow-map1)# permit any src-ip any dst-ip 1.1.1.2 0.0.0.0 mark-source 20
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow flow1
Network Packet Broker(config-tap-tap1)# ingress eth-0-2 flow flow2
Network Packet Broker(config-tap-tap1)# egress eth-0-10
```

17.4 Validation

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show tap-group

tap-group tap1
```

```
ID: 10
Ingress:
    eth-0-1         mark-src 10
    eth-0-2         mark-src 20
egress:
    eth-0-10
```

NOTE:The result above shows the Network Packet Broker group for PORT mode.

17.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
tap-group tap1 1
 ingress eth-0-1 mark-source 10
 ingress eth-0-2 mark-source 20
 egress eth-0-10
```

NOTE:The result above shows the Network Packet Broker group for PORT mode.

18 VLAN Stripping Configuration

18.1 Networking requirements

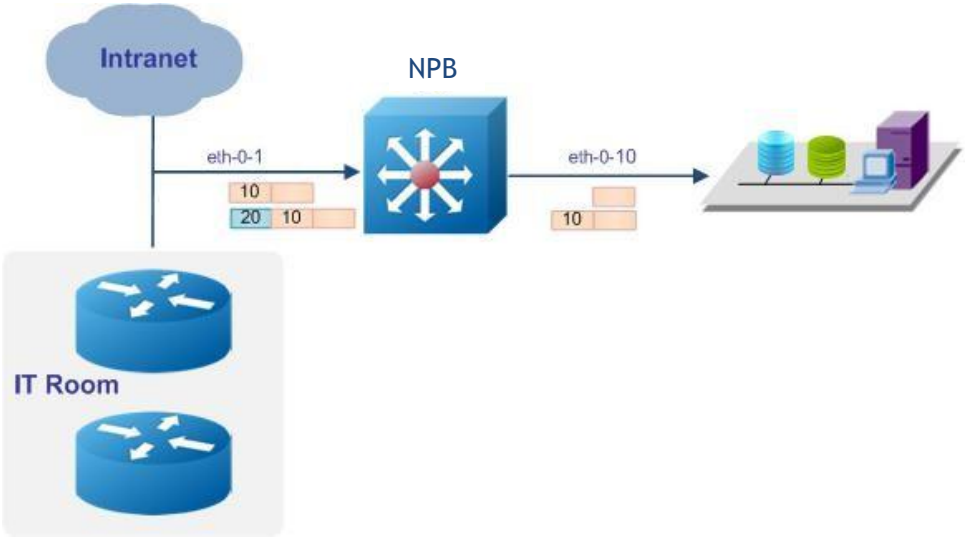

Figure 18-1 Topology of VLAN stripping

18.2 Configuration Ideas

In some cases server or analyzer cannot deal with the packets with VLAN tag or double VLAN tags. The VLAN stripping application can resolve the problem.

Reference to the Figure, Packets from eth-0-1 with VLAN 10 should be stripped the VLAN tag, Packets from eth-0-1 with S-VLAN 20 C-VLAN 10 should be stripped the outer VLAN tag S-VLAN 20.

VLAN stripping application should do nothing to untagged packets.

18.3 Configuration

PORT mode and PORT WITH FLOW mode both support VLAN stripping.

18.3.1 VLAN Stripping for PORT mode

The following example shows how to create Network Packet Broker group, strip the VLAN for the packets form eth-0-1, and send a copy to eth-0-10:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 un-tag-outer-vlan
Network Packet Broker(config-tap-tap1)# egress eth-0-10
```

18.3.2 VLAN Stripping for PORT WITH FLOW mode

The following example shows how to create Network Packet Broker group, strip the VLAN for the packets with destination IP address 1.1.1.1form eth-0-1, and send a copy to eth-0-2:

```
Network Packet Broker(config)# flow flow1
Network Packet Broker(config-flow-map1)# permit any src-ip any dst-ip 1.1.1.1 0.0.0.0 un-tag-outer-vlan
Network Packet Broker(config-flow-map1)# permit any src-ip any dst-ip any
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow flow1
Network Packet Broker(config-tap-tap1)# egress eth-0-2
```

18.4 Validation

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show tap-group

tap-group tap1
 ID: 10
```

```
Ingress:
    eth-0-1         un-tag-outer-vlan
egress:
    eth-0-10
```

NOTE:The result above shows the Network Packet Broker group for PORT mode.

18.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
tap-group tap1 1
 ingress eth-0-1 un-tag-outer-vlan
egress eth-0-10
```

NOTE:The result above shows the Network Packet Broker group for PORT mode.

19 Packet Editing Configuration

19.1 Networking requirements

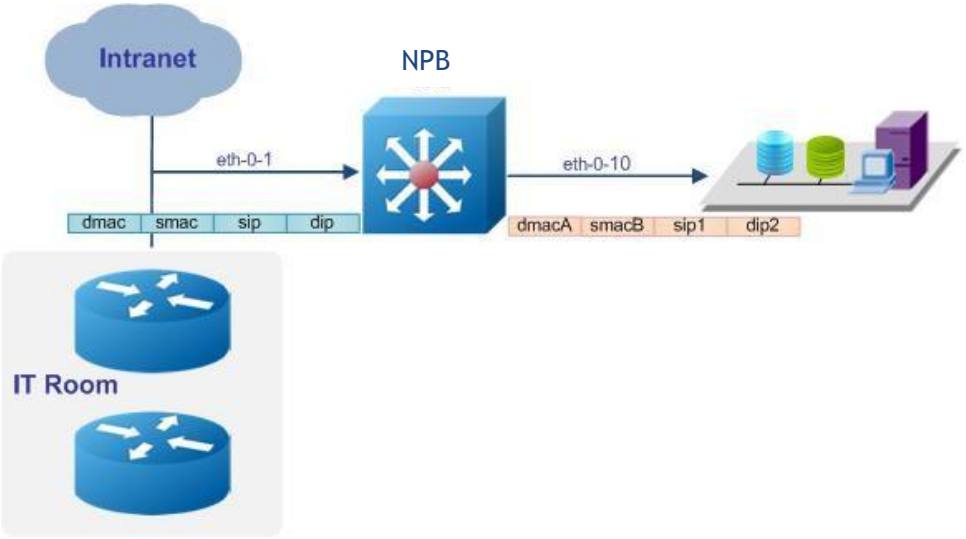

Figure 19-1 Topology of packet editing

19.2 Configuration Ideas

Figure 19-2 In some cases, the server or analyzer can only receive the packets with the destination address equal to its own address The packet editing application can meet the requirement. Source and destination MAC address, Source and destination IP address of the packets can be modified when enter the ingress port. Reference to the Figure, the device should modify the source and destination MAC address, Source and destination IP address of the packets from eth-0-1 and send a copy to eth-0-10.

19.3 Configuration

PORT mode and PORT WITH FLOW mode both support packet editing.

19.3.1 Packet editing for PORT mode

The following example shows how to create Network Packet Broker group, edit the source and destination IP/MAC address of the packets form eth-0-1, and send a copy to eth-0-10:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 edit-macsa a.a.a edit-macda b.b.b edit-ipda 1.1.1.1 edit-ipsa 2.2.2.2
Network Packet Broker(config-tap-tap1)# egress eth-0-10
```

19.3.2 Packet editing for PORT WITH FLOW mode

The following example shows how to create Network Packet Broker group with flow rule, and edit the destination IP address to 100.100.100.1 for the packets with destination IP address 1.1.1.1, edit the destination IP address to 100.100.100.2 for the packets with destination IP address 1.1.1.2:

```
Network Packet Broker(config)# flow flow1
Network Packet Broker(config-flow-map1)# permit any src-ip any dst-ip 1.1.1.1 0.0.0.0 edit-ipda 100.100.100.1
Network Packet Broker(config-flow-map1)# permit any src-ip any dst-ip 1.1.1.2 0.0.0.0 edit-ipda 100.100.100.2
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow flow1
Network Packet Broker(config-tap-tap1)# egress eth-0-10
```

19.4 Validation

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show tap-group

tap-group tap1
```

```
ID: 10
Ingress:
    eth-0-1      edit-macda 000B.000B.000B
                 edit-macsa 000A.000A.000A
                 edit-ipda 1.1.1.1
                 edit-ipsa 2.2.2.2
egress:
    eth-0-10
```

NOTE:The result above shows the Network Packet Broker group for PORT mode.

19.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
tap-group tap1 1
 ingress eth-0-1 edit-macda 000B.000B.000B edit-macsa 000A.000A.000A edit-ipda 1.1.1.1 edit-ipsa 2.2.2.2
 egress eth-0-10
```

NOTE:The result above shows the Network Packet Broker group for PORT mode.

20 Time Stamp Configuration

20.1 Overview

To monitor the outgoing traffic of the data center is a common application scenario of Network Packet Broker. With the increase of data center scale and the improvement of the performance requirements, user need to monitor the inner traffic of the data center and get more detailed information. Network Packet Broker series device provides flexible packet remarking applications, which can insert an additional header before the original packet header. The additional header use an ether-type defined by private protocol, which can carry 16 bytes private data.

flowid 16bit	Srcport 16bit	ResidenceTime
Timestamp 64bit		

Packet structure
Flowid: ID of the flow. Default value is 0x1000, cannot be modified.
Srcport: source port ID of the packet. (The port ID is assigned by chip which is not same as the ID on the device panel)
Time stamp: add the time stamp before the chip processing the packet. The duration of packets stay in the chip should not record by the timestamp.
Note: The timestamp function needs to be used in conjunction with the timestamp sync systime command.
Timestamp use standard Time of Day format. The high 32 bits record seconds (since 1970-01-01), the low 32 bits record nanosecond.
The analyzer can recognize the time stamp packets by ether header, and analyze the TCP traffic by the information carried in the packets.

20.2 Networking requirements

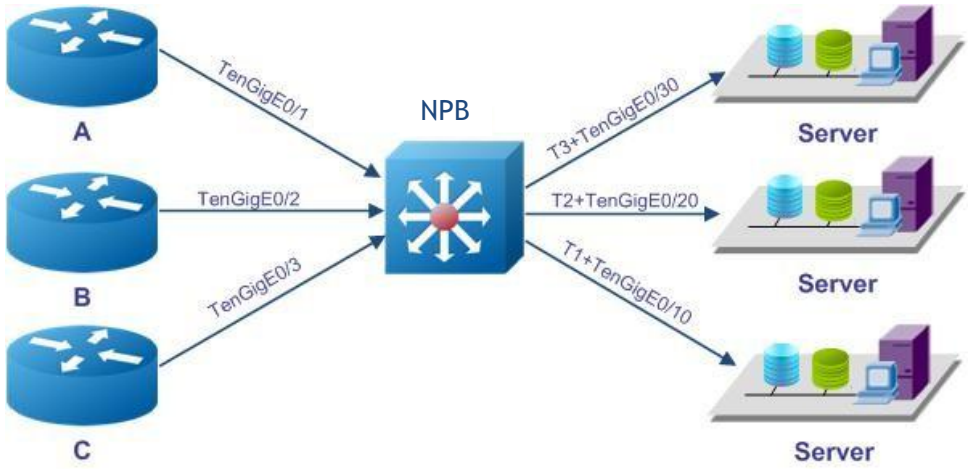

Figure 20-1 Topology of Time stamp

20.3 Configuration Ideas

Reference to the Figure, the cluster of the server can get the accurate duration the packet spent on each node of the data center by the source port and timestamp information. Use the source port to identify different devices, use the information in timestamp to get the latency.

20.4 Configuration

The following example shows how to set private ether-type to 0xFF12, and set the destination MAC address to 1.1.1, set the source MAC address to 2.2.2; use the system time as time source for time-stamp:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# timestamp-over-ether 1.1.1 2.2.2 0xff12
Network Packet Broker(config)# timestamp sync systime
```

The following example shows how to create 3 Network Packet Broker groups, with 3 source ports eth-0-1/eth-0-2/eth-0-3, and with 3 destination ports eth-0-10/eth-0-20/eth-0-30 which enabled time stamp:

```
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1
Network Packet Broker(config-tap-tap1)# egress eth-0-10 timestamp
Network Packet Broker(config-tap-tap1)# exit
```

```
Network Packet Broker(config)# tap-group tap2
Network Packet Broker(config-tap-tap2)# ingress eth-0-2
Network Packet Broker(config-tap-tap2)# egress eth-0-20 timestamp
Network Packet Broker(config-tap-tap2)# exit
Network Packet Broker(config)# tap-group tap3
Network Packet Broker(config-tap-tap3)# ingress eth-0-3
Network Packet Broker(config-tap-tap3)# egress eth-0-30 timestamp
Network Packet Broker(config-tap-tap3)# exit
```

20.5 Validation

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show tap-group

tap-group tap1
ID: 10
 Ingress:
      eth-0-1
 egress:
      eth-0-10        time-stamp
tap-group tap2
ID: 20
 Ingress:
      eth-0-2
 egress:
      eth-0-20        time-stamp
tap-group tap3
ID: 30
 Ingress:
      eth-0-3
 egress:
      eth-0-30        time-stamp
```

20.6 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
timestamp-over-ether 0001.0001.0001 0002.0002.0002 0xff12
!
timestamp sync systime
!
tap-group tap1 1
 ingress eth-0-1
egress eth-0-10 timestamp
!
tap-group tap2 2
 ingress eth-0-2
 egress eth-0-20 timestamp
!
tap-group tap3 3
 ingress eth-0-3
 egress eth-0-30 timestamp
```

21 Packet truncation Configuration

21.1 Overview

Figure 21-1 sketch map of packet truncation

21.2 Configuration Ideas

In some cases, packets need to be truncated in order to reduce the pressure of the server or in order to protect privacy. The packet truncation application can meet the requirement. E.g. the size of packet enters the Network Packet Broker device from eth-0-1 is 1518 bytes. The size of packet leaves destination port eth-0-10 is 64 byte.

21.3 Configuration

PORT mode and PORT WITH FLOW mode both support packet truncation.

21.3.1 Packet Truncation for PORT mode

The following example shows how to set the packet length after truncated to 64 byte:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# truncation 64
```

The follow example shows how to create Network Packet Broker group with ingress port eth-0-1 and enable packet truncation:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 truncation
Network Packet Broker(config-tap-tap1)# egress eth-0-10
```

21.3.2 Packet Truncation for PORT WITH FLOW mode

The following example shows how to set a flow rule to match the packets with destination IP address 1.1.1.0/24 and enable truncation. Packets with other destination IP address should not be truncated:

```
Network Packet Broker(config)# flow flow1
Network Packet Broker(config-flow-flow1)# permit any src-ip any dst-ip 1.1.2.0 0.0.0.255 truncation
Network Packet Broker(config-flow-flow1)# permit any src-ip any dst-ip any
Network Packet Broker(config-flow-flow1)# exit
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow flow1
Network Packet Broker(config-tap-tap1)# egress eth-0-10
Network Packet Broker(config-tap-tap1)# end
```

21.4 Validation

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show tap-group

tap-group tap1
ID: 10
 Ingress:
     eth-0-1         truncation
 egress:
     eth-0-10
```

NOTE:he result above shows the Network Packet Broker group for PORT mode.

21.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
truncation 64
```

```
!
tap-group tap1 1
 ingress eth-0-1 truncation
egress eth-0-10
```

NOTE:Packet truncation is mutual exclusive to other actions. E.g. Only Packet truncation is effective and all other configuration(egress-filter/time stamp etc.) is invalid in the following configuration:

```
ip access-list filter1
sequence-num 10 deny any src-ip any dst-ip any
!
interface eth-0-2
  egress filter1
!
timestamp-over-ether 000A.000A.000A 000B.000B.000B 0xff12
!
tap-group tap1
 ingress eth-0-1 truncation
 egress eth-0-2 timestamp
```

22 Packet header stripping Configuration

22.1 Configuring strip the VXLAN header

22.1.1 Networking requirements

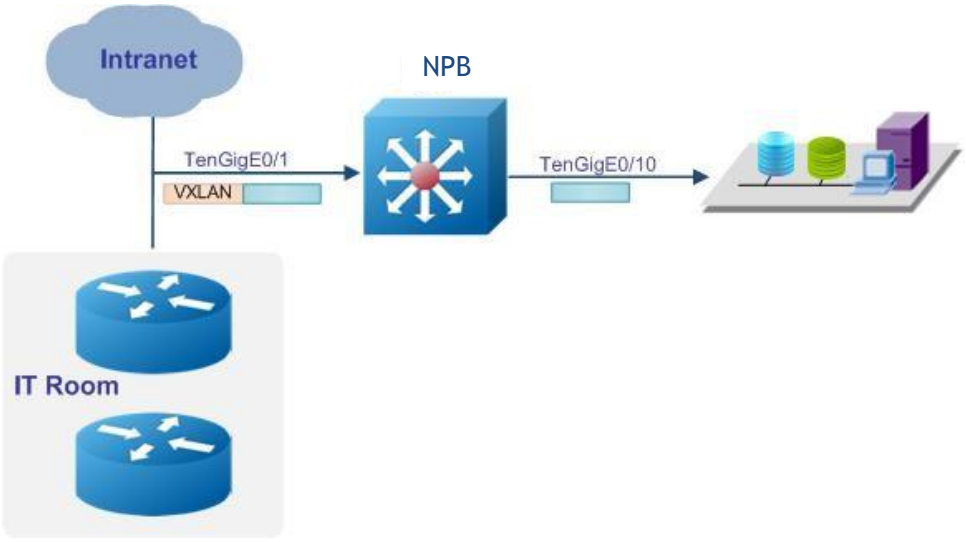

Figure 22-1 Topology of stripping VXLAN header

22.1.2 Configuration Ideas

In some cases, server or analyzer cannot parse the packet with VXLAN/NVGRE/GRE header. The packet header stripping application can resolve the problem.

Reference to the Figure the packet enter eth-0-1, the VLAN header should be stripped

22.1.3 Configuration

The following example shows how to create a flow rule the match the VXLAN packets and strip the header:

```
Network Packet Broker(config)# flow flow1
Network Packet Broker(config-flow-flow1)# permit udp dst-port eq 4789 vxlan-vni any src-ip any dst-ip any strip-header
Network Packet Broker(config-flow-flow1)# exit
```

The following example shows how to create a Network Packet Broker group with ingress port eth-0-1 and flow1:

```
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow flow1
Network Packet Broker(config-tap-tap1)# egress eth-0-10
Network Packet Broker(config-tap-tap1)# end
```

22.1.4 Validation

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show tap-group

tap-group tap1
ID: 10
 Ingress:
     eth-0-1         flow flow1
 egress:
     eth-0-10
```

22.1.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
flow flow1
 sequence-num 10 permit udp dst-port eq 4789 vxlan-vni any src-ip any dst-ip any strip-header
!
tap-group tap1 1
 ingress eth-0-1 flow flow1
egress eth-0-10
```

```
Network Packet Broker(config)# flow flow1
Network Packet Broker(config-flow-map1)# permit udp dst-port eq 4789 vxlan-vni 1000 0x0 src-ip any dst-ip any strip-header
Network Packet Broker(config-tap-tap1)# end
```

NOTE:Network Packet Brokers support to match the specified VNI. E.g. match VNI 1000 and strip the VXLAN header.you can configure flow udp dst-port not 4789 to match vxlan,but now you just can configure same global vxlan dst-port .

```
Network Packet Broker(config)# flow flow1
Network Packet Broker(config-flow-map1)# permit udp dst-port eq 1234 vxlan-vni 1000 0x0 src-ip any dst-ip any strip-header
Network Packet Broker(config-flow-map1)# permit udp dst-port eq 1234 vxlan-vni 1200 0x0 src-ip any dst-ip any strip-header
Network Packet Broker(config-tap-tap1)# end
```

22.2 Configuring strip the NVGRE header

22.2.1 Networking requirements

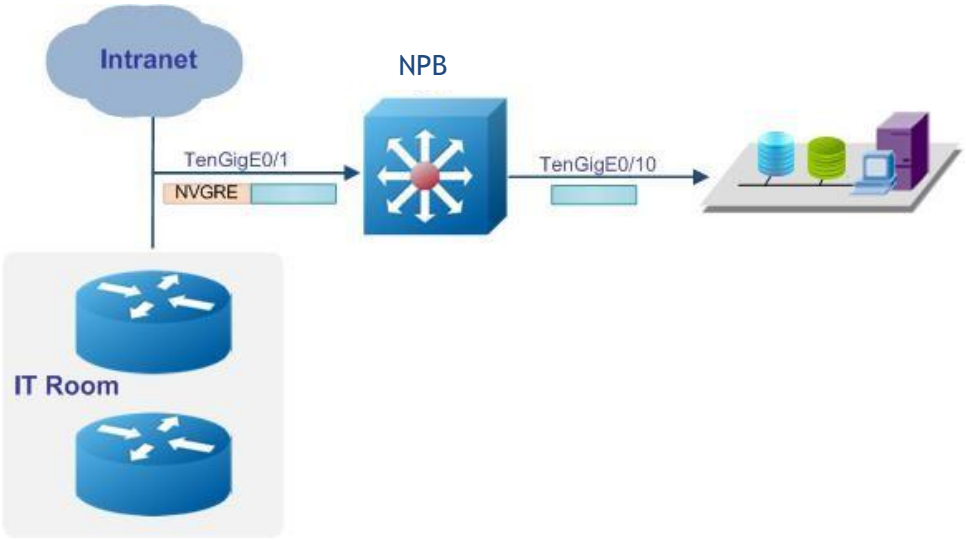

Figure 22-2 Topology of stripping NVGRE header

22.2.2 Configuration Ideas

In some cases, server or analyzer cannot parse the packet with VXLAN/NVGRE/GRE header. The packet header stripping application can resolve the problem.

Reference to the Figure the packet enter eth-0-1, the NVGRE header should be stripped

22.2.3 Configuration

The following example shows how to create a flow rule the match the NVGRE packets and strip the header:

```
Network Packet Broker(config)# flow flow1
Network Packet Broker(config-flow-flow1)# permit nvgre src-ip any dst-ip any strip-header
Network Packet Broker(config-flow-flow1)# exit
```

The following example shows how to create a Network Packet Broker group with ingress port eth-0-1 and flow1:

```
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow flow1
Network Packet Broker(config-tap-tap1)# egress eth-0-10
Network Packet Broker(config-tap-tap1)# end
```

22.2.4 Validation

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show tap-group

tap-group tap1
ID: 10
 Ingress:
     eth-0-1        flow flow1
 egress:
     eth-0-10
```

22.2.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
flow flow1
sequence-num 10 permit nvgre src-ip any dst-ip any strip-header
!
tap-group tap1 1
 ingress eth-0-1 flow flow1
egress eth-0-10
```

22.3 Configuring strip the GRE header

22.3.1 Networking requirements

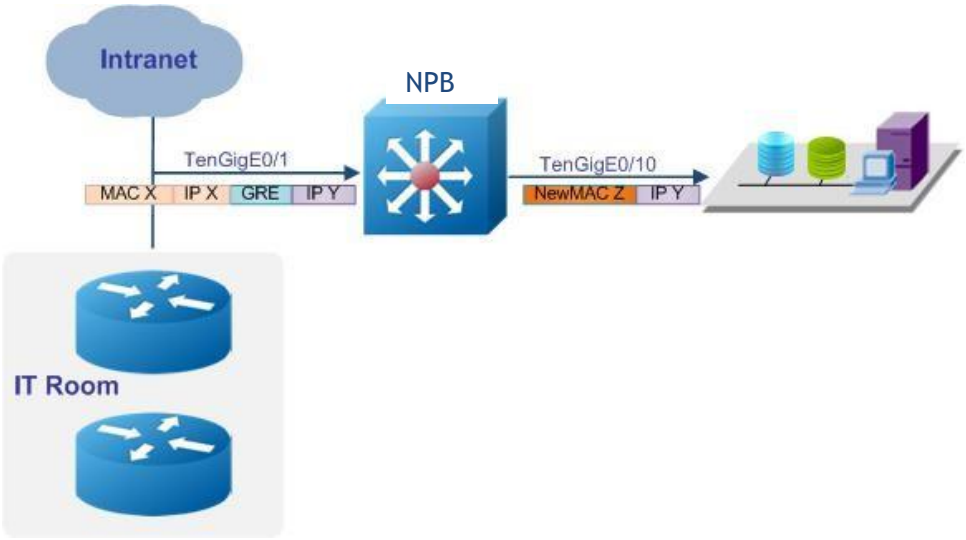

Figure 22-3 Topology of stripping GRE header

22.3.2 Configuration Ideas

In some cases, server or analyzer cannot parse the packet with VXLAN/NVGRE/GRE header. The packet stripping application for GRE packet should strip the outer IP address, MAC address and GRE header, only inner IP address and payload are left. Packet editing application should be configured together with packet header stripping, in order to add outer MAC address.

Reference to the Figure the packet enter eth-0-1, the GRE header should be stripped and a new MAC address should be added.

22.3.3 Configuration

The following example shows how to create a flow rule the match the GRE packets and strip the header:

```
Network Packet Broker(config)# flowflow1
Network Packet Broker(config-flow-flow1)# permit gre src-ip any dst-ip any strip-header edit-macsa a.a.a edit-macda b.b.b
Network Packet Broker(config-flow-flow1)# exit
```

The following example shows how to create a Network Packet Broker group with ingress port eth-0-1 and flow1:

```
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow flow1
Network Packet Broker(config-tap-tap1)# egress eth-0-10
Network Packet Broker(config-tap-tap1)# end
```

22.3.4 Validation

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show tap-group

tap-group tap1
ID: 10
 Ingress:
     eth-0-1        flow flow1
 egress:
     eth-0-10
```

NOTE: GRE header length is flexible. In the example above, the flow only match GRE field, and only strip the standard GRE header which is 4 bytes. If the packets need to strip header include GRE-key, the configuration is as following(Match GRE and GRE-KEY field). It means that, if the flow only matches GRE filed, the stripped length is 4 bytes; if the flow matched GRE and GRE-KEY field, the stripped length is 8 bytes. If the packet with a GRE header which is more than 8 byte, or with variable types of GRE packets(For example, the packets with 4/8/12/16 bytes GRE header exist at same time), please reference to the chapter "Configuring strip the User Defined header".

```
Network Packet Broker(config)# flow flow1
Network Packet Broker(config-flow-flow1)# permit gre gre-key any src-ip any dst-ip any strip-header edit-macsa a.a.a edit-macda b.b.b
Network Packet Broker(config-flow-flow1)#exit
```

22.3.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
flow flow1
sequence-num 10 permit gre src-ip any dst-ip any strip-header edit-macda 000B.000B.000B edit-macsa 000A.000A.000A
!
tap-group tap1 1
 ingress eth-0-1 flow flow1
egress eth-0-10
```

22.4 Configuring strip the IPIP header

22.4.1 Networking requirements

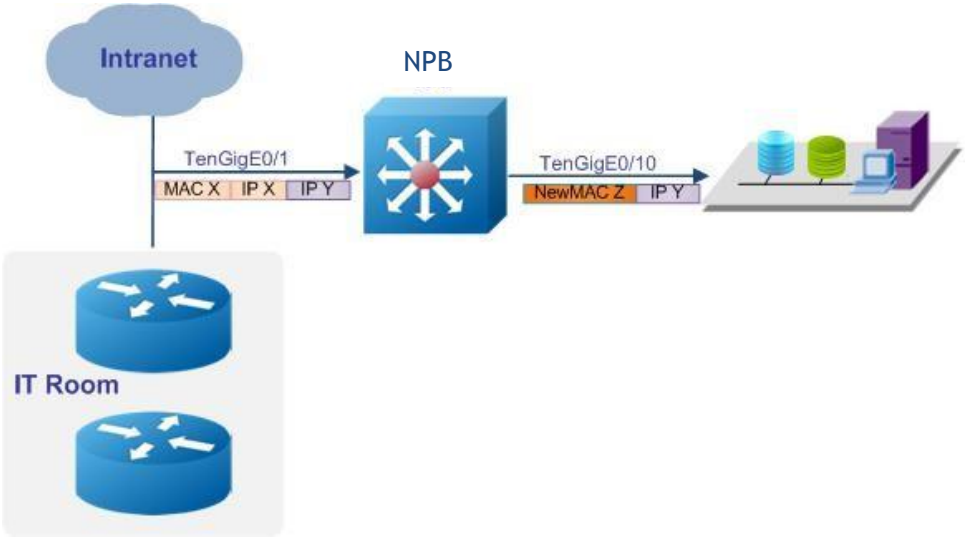

Figure 22-4 Topology of stripping IPIP header

22.4.2 Configuration Ideas

In some cases, server or analyzer cannot parse the packet with IPIP header. The packet stripping application for IPIP packet should strip the outer IP address, MAC header, only inner IP address and payload are left. Packet editing application should be configured together with packet header stripping, in order to add outer MAC address.

Reference to the Figure the packet enter eth-0-1, the IPIP header should be stripped and a new MAC address should be added.

22.4.3 Configuration

The following example shows how to create a flow rule the match the IPIP packets and strip the header:

```
Network Packet Broker(config)# flowflow1
Network Packet Broker(config-flow-flow1)# permit ipip src-ip any dst-ip any strip-header edit-macsa a.a.a edit-macda b.b.b
Network Packet Broker(config-flow-flow1)# exit
```

The following example shows how to create a Network Packet Broker group with ingress port eth-0-1 and flow1:

```
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow flow1
Network Packet Broker(config-tap-tap1)# egress eth-0-10
Network Packet Broker(config-tap-tap1)# end
```

22.4.4 Validation

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# showtap-group
```

```
tap-group tap1
ID: 10
 Ingress:
     eth-0-1          flow flow1
 egress:
     eth-0-10
```

22.4.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
flow flow1
sequence-num 10 permit ipip src-ip any dst-ip any strip-header edit-macda 000B.000B.000B edit-macsa 000A.000A.000A
!
tap-group tap1 1
 ingress eth-0-1 flow flow1
egress eth-0-10
```

22.5 Configuring strip the User Defined header

22.5.1 Networking requirements

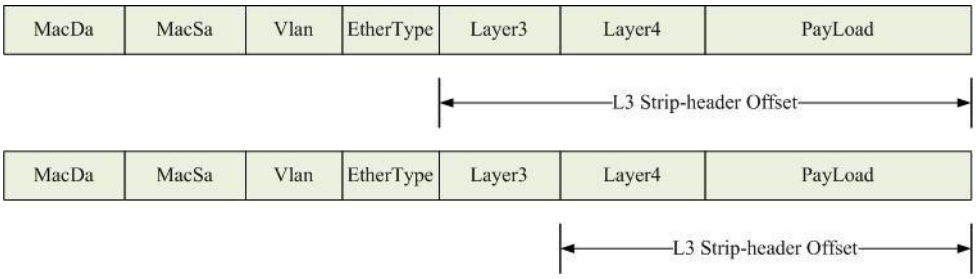

Figure 22-5 Packet structure

22.5.2 Configuration Ideas

Normal packet header stripping application can strip the standard VXLAN/GRE/NVGRE header, which cannot match all cases. e.g. GRE header may have variable length because GRE-KEY/Checksum/Sequence Num inserted. By default, packet header stripping can strip GRE header and one option field of 4 bytes. When the GRE packet has more than one option fields, the packet header stripping cannot strip them correctly.

The user defined header stripping application can resolve the problem. A starting position (L2, L3 or L4) and offset (up to 30 bytes) should be specified before using user defined header stripping.

The following example shows how to strip the GRE packets with GRE-KEY/Checksum/Sequence Number

22.5.3 Configuration

Create a flow rule to match GRE packets and enable user defined stripping:

```
Network Packet Broker(config)# flow flow1
Network Packet Broker(config-flow-flow1)# permit gre src-ip any dst-ip any strip-header strip-position l4 strip-offset 16 edit-macsa a.a.a edit-macda b.b.b
Network Packet Broker(config-flow-flow1)# exit
```

NOTE:Strip-position is L4 and offset is 16 means remove 16 bytes after L4 header and remove all fields before L4 header.

Create a Network Packet Broker group with ingress port eth-0-1 and flow1:

```
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow flow1
Network Packet Broker(config-tap-tap1)# egress eth-0-10
Network Packet Broker(config-tap-tap1)# end
```

22.5.4 Validation

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show tap-group

tap-group tap1
ID: 10
 Ingress:
```

```
    eth-0-1         flow flow1
 egress:
    eth-0-10
```

22.5.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
flow flow1
sequence-num 10 permit gre src-ip any dst-ip any strip-header strip-position l4 strip-offset 16 edit-macda 000B.000B.000B edit-macsa 000A.000A.000A
!
tap-group tap1 1
 ingress eth-0-1 flow flow1
egress eth-0-10
```

22.6 Configuring strip the ERSPAN header

22.6.1 Networking requirements

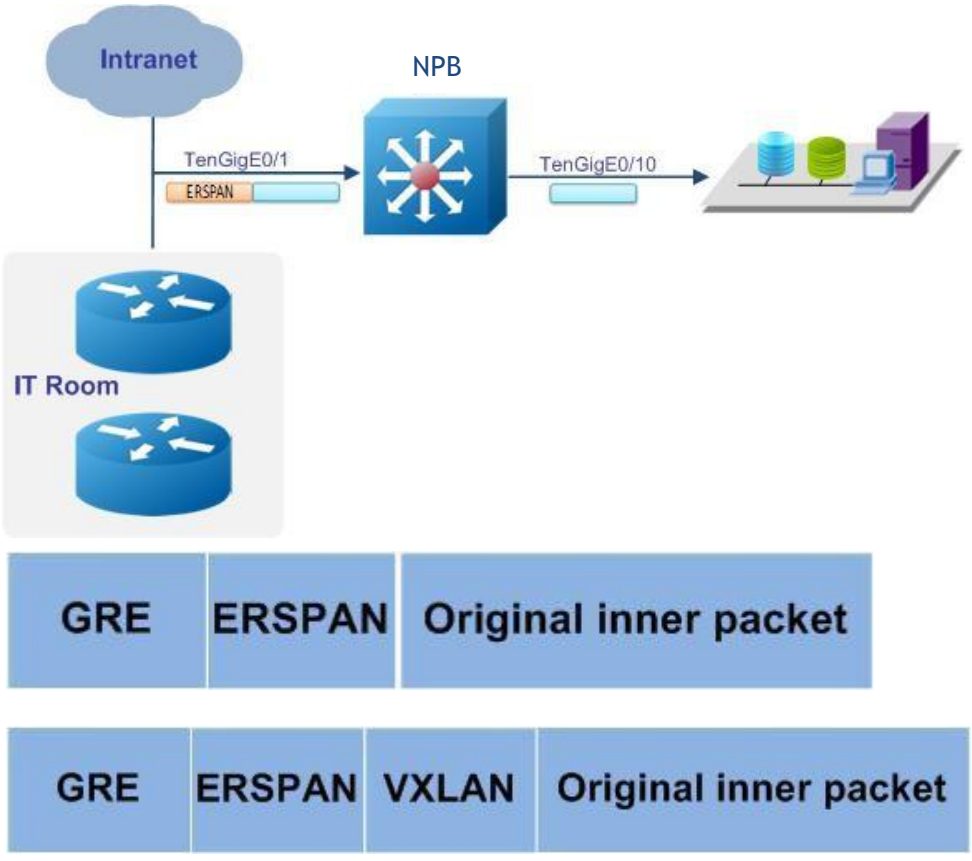

Figure 22-6 Packet structure

22.6.2 Configuration Ideas

In some cases, user needs to match the packets with ERSPAN ID among those packets with GRE+ERSPAN+VXLAN header, and need to strip the ERSPAN header. VXLAN header may also need to be stripped. Matching ERSPAN ID can meet the requirement.

22.6.3 Configuration

The following example shows how to create a flow rule to match the ERSPAN packets with span id 1, and strip the GRE+ERSPAN header and add vlan 1:

```
Network Packet Broker(config)# flow erspan
Network Packet Broker(config-flow- erspan)# permit gre erspan 1 0x0 src-ip any dst-ip any strip-header add-vlan 1
```

The following example shows how to create a flow rule to match the ERSPAN packets with span id 2, and strip the GRE+ERSPAN+vxlan header and add vlan 3:

```
Network Packet Broker(config)# flow erspan
```

```
Network Packet Broker(config-flow- erspan)# permit gre erspan 2 0x0 src-ip any dst-ip any strip-header strip-inner-vxlan-header add-vlan 3
```

The following example shows how to create a Network Packet Broker group with ingress port eth-0-1 and flow erspan:

```
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow erspan
Network Packet Broker(config-tap-tap1)# egress eth-0-10
```

22.6.4 Validation

The following example shows how to display the information of the flow rule:

```
Network Packet Broker# show flow
flow erspan
 sequence-num 10 permit gre erspan 1 0x0 src-ip any dst-ip any strip-header edit
 -vlan 1
 sequence-num 20 permit gre erspan 2 0x0 src-ip any dst-ip any strip-header edit
 -vlan 3
```

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# showtap-group

tap-group tap1
ID: 10
 Ingress:
      eth-0-1         flow erspan
 egress:
      eth-0-10
```

22.6.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
flow erspan
 sequence-num 10 permit gre erspan 1 0x0 src-ip any dst-ip any strip-header add
-vlan 1
 sequence-num 20 permit gre erspan 2 0x0 src-ip any dst-ip any strip-header edit
 -vlan 3
!
tap-group tap1 1
ingress eth-0-1 flow erspan
egress eth-0-10
```

22.7 Configuring strip the MPLS header

22.7.1 Networking requirements

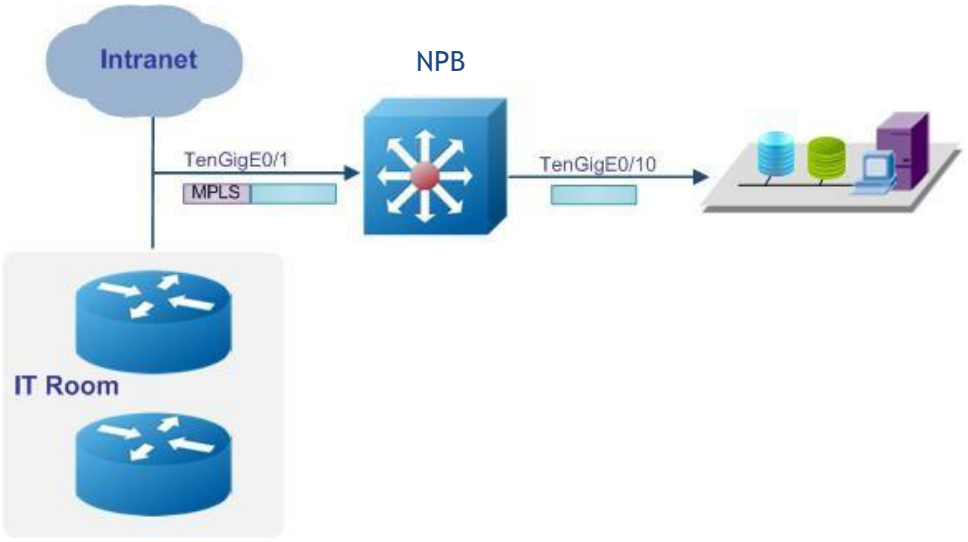

Figure 22-7 Topology of stripping MPLS header

22.7.2 Configuration Ideas

In some cases, server or analyzer cannot parse the packet with MPLS LABEL header. The packet header stripping application can resolve the problem. Network Packet Broker supports to match the number of mpls labers(up to 9) and the value of mpls labers(upp to 3). If the stripped message is a IPv4 message, the operation of adding a mac-header is supported.

Reference to the Figure the packet enter eth-0-1, the MPLS header should be stripped

22.7.3 Configuration

The following example shows how to create a flow rule the match the MPLS packets and strip the header:

```
Network Packet Broker(config)# flowflow1
Network Packet Broker(config-flow-flow1)# permit mpls label-num 2 mpls-label1 any mpls-label2 100 strip-header
Network Packet Broker(config-flow-flow1)# exit
```

The following example shows how to create a flow rule the match the MPLS packets, strip the header and add mac-header:

```
Network Packet Broker(config)# flow flow2
Network Packet Broker(config-flow-flow1)# permit mpls label-num 3 mpls-label1 any mpls-label2 100 mpls-label3 200 strip-header add-l2macda 1.1.1 add-l2macsa 2.2.2
Network Packet Broker(config-flow-flow1)# exit
```

The following example shows how to create a Network Packet Broker group with ingress port eth-0-1 and flow1:

```
Network Packet Broker(config)# tap-grouptap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow flow1
Network Packet Broker(config-tap-tap1)# egress eth-0-10
Network Packet Broker(config-tap-tap1)# exit

Network Packet Broker(config)# tap-group tap2
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow flow2
Network Packet Broker(config-tap-tap1)# egress eth-0-10
Network Packet Broker(config-tap-tap1)# end
```

22.7.4 Validation

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show tap-group

tap-group tap1
ID: 10
 Ingress:
      eth-0-1         flow flow1
 egress:
      eth-0-10
Network Packet Broker-group tap2
ID: 20
 Ingress:
      eth-0-1         flow flow2
 Egress:
      eth-0-10
```

22.7.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
flow flow1
 sequence-num 10 permit mpls label-num 2 mpls-label2 100 strip-header
 exit
!
flow flow2
 sequence-num 10 permit mpls label-num 3 mpls-label2 100 mpls-label3 100 strip-header add-l2macda 0001.0001.0001 add-l2macsa
0002.0002.0002
 exit
!
tap-group tap1 1
 ingress eth-0-1 flow flow1
 egress eth-0-10
!
```

```
tap-group tap2 2
 ingress eth-0-1 flow flow2
 egress eth-0-10
```

22.8 Configuring strip the PPPOE header

22.8.1 Networking requirements

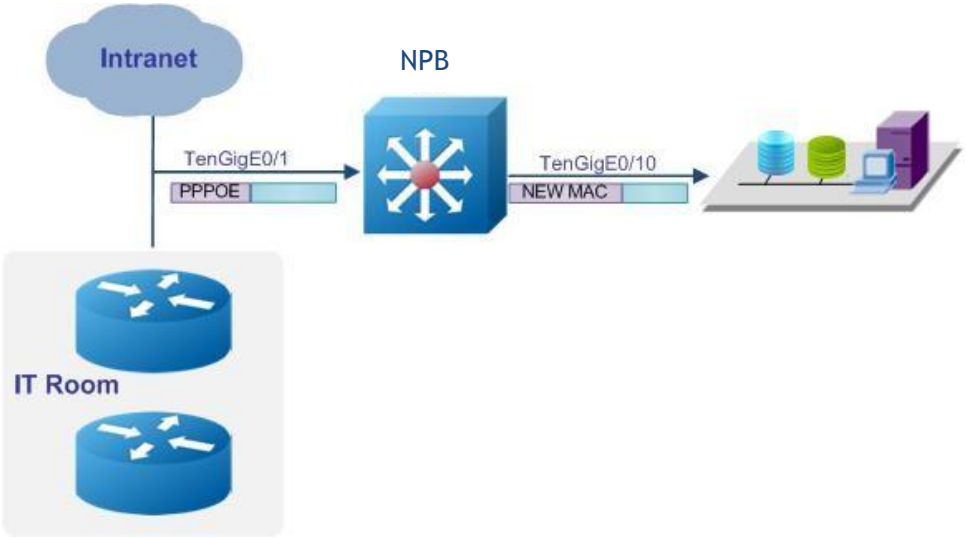

Figure 22-8 Topology of stripping PPPOE header

22.8.2 Configuration Ideas

In some cases, server or analyzer cannot parse the packet with PPPOE LABEL header. The packet header stripping application can resolve the problem. Network Packet Broker supports to match point-to-point protocol type of ipv4 or ipv6. Mac-header needs to be added after stripping.

Reference to the Figure the packet enter eth-0-1, the PPPOE header should be stripped and a new MAC address should be added.

22.8.3 Configuration

The following example shows how to create a flow rule the match the PPPOE packets:

```
Network Packet Broker(config)# flow flow1
Network Packet Broker(config-flow-flow1)# permit pppoe ppp-type ipv6
Network Packet Broker(config-flow-flow1)# exit
```

The following example shows how to create a flow rule the match the PPPOE packets and strip the header:

```
Network Packet Broker(config)# flow flow2
Network Packet Broker(config-flow-flow1)# permit pppoe ppp-type ipv4 strip-header add-l2macda 1.1.1 add-l2macsa 2.2.2 add-l2vlan 10
Network Packet Broker(config-flow-flow1)# exit
```

The following example shows how to create a Network Packet Broker group with ingress port eth-0-1 and flow1:

```
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow flow1
Network Packet Broker(config-tap-tap1)# egress eth-0-10
Network Packet Broker(config-tap-tap1)# exit
Network Packet Broker(config)# tap-group tap2
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow flow2
Network Packet Broker(config-tap-tap1)# egress eth-0-10
Network Packet Broker(config-tap-tap1)# end
```

22.8.4 Validation

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show tap-group

tap-group tap1
ID: 10
 Ingress:
     eth-0-1         flow flow1
 egress:
     eth-0-10
```

```
Network Packet Broker-group tap2
ID: 20
 Ingress:
      eth-0-1          flow flow2
 egress:
      eth-0-10
```

22.8.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
flow flow1
 sequence-num 10 permit pppoe ppp-type ipv6
 exit
!
flow flow2
 sequence-num 10 permit pppoe ppp-type ipv4 strip-header add-l2macda 0001.0001.0001 add-l2macsa 0002.0002.0002 add-l2vlan 10
 exit
!
tap-group tap1 1
 ingress eth-0-1 flow flow1
 egress eth-0-10
!
tap-group tap2 2
 ingress eth-0-1 flow flow2
 egress eth-0-10
```

23 AAA Configuration

AAA(Authentication/Authorization/Accounting)is an security mechanism for network management, which support 3 applications: Authentication, Authorization and Accounting. The Network Packet Broker series switches support to certify the users access the network.

23.1 Configuring Radius Authentication

23.1.1 Networking requirements

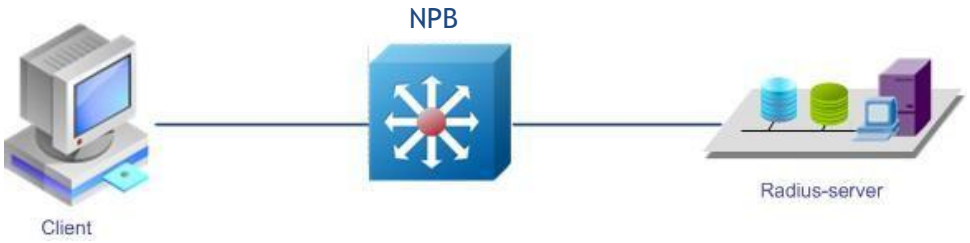

Figure 23-1 Topology of Radius Authentication

23.1.2 Configuration Ideas

Radius is a distributed server/client system to prevent unauthorized access and to guarantee the security of the network. Radius server keeps all information of users' authentication and network service accessing. Radius server should do Authentication/Authorization/Accounting according the user information in local database, after it received request from a client.

23.1.3 Configuration

The following example shows how to enable AAA and set the mode of Authentication/Authorization/Accounting:

```
Network Packet Broker(config)# aaa new-model
Network Packet Broker(config)# aaa authentication login radius-authen radius
Network Packet Broker(config)# aaa authorization exec radius-author radius
Network Packet Broker(config)# aaa accounting exec radius-acct start-stop radius
```

The following example shows how to set the parameter of the radius server:

```
Network Packet Broker(config)# radius-server host mgmt-if 10.10.1.1 key test auth-port 1819
```

The following example shows how to set the login mode to radius:

```
Network Packet Broker(config)# line vty 0 7
Network Packet Broker(config-line)# login authentication radius-authen
Network Packet Broker(config-line)# privilege level 4
Network Packet Broker(config-line)# no line-password
```

23.1.4 Validation

Use the username and password on radius server to login the device.

23.1.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
aaa new-model
!
aaa authentication login radius-authen radius
!
aaa authorization exec radius-author radius
!
aaa accounting exec radius-acct start-stop radius
!
line vty 0 7
 exec-timeout 35791 0
 privilege level 4
 no line-password
 login authentication radius-authen
```

24 Sflow Configuration

Sflow (Sampled Flow) is a traffic monitoring technology based on packet sampling.
Sflow is used to analyze the network traffic.
Sflow has 2 types of message: statistics information for ports and sampled packets information.

24.1.1 Networking requirements

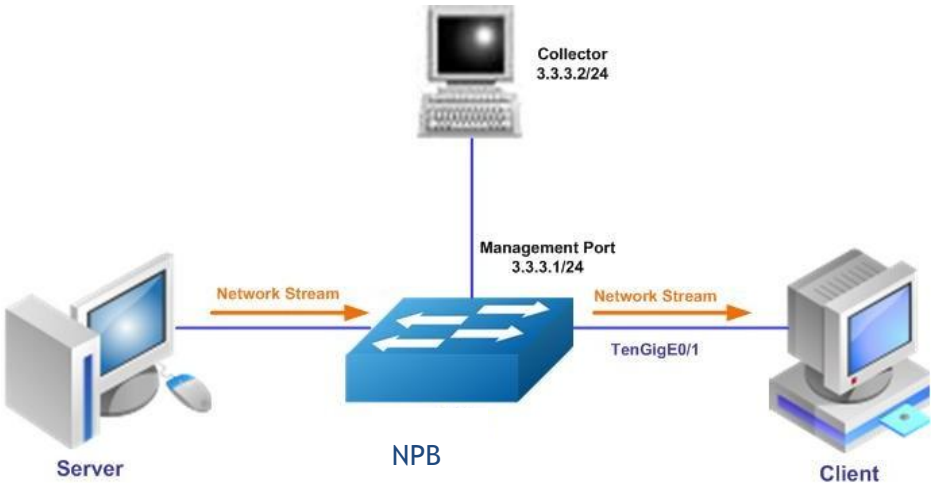

Figure 24-1 Topology of Sflow

24.1.2 Configuration Ideas

Traffic monitoring is a basic requirement of network management.
User need to find the source abnormal traffic and attacking traffic in time. Sflow, which is a traffic monitoring technology based on packet sampling can meet the requirement.

24.1.3 Configuration

The following example shows how to enable sflow and set the sampling interval, IP address of the agent and IP address of the collector:

```
Network Packet Broker(config)# sflowenable
Network Packet Broker(config)# sflow counter interval 20
Network Packet Broker(config)# sflow agent ip 3.3.3.1
Network Packet Broker(config)# sflow collector mgmt-if3.3.3.2
```

The follow example shows how to enable sflow on a port and set the sampling rate:

```
Network Packet Broker(config)# interface eth-0-1
Network Packet Broker(config-if-eth-0-1)# sflow flow-sampling rate 32768
Network Packet Broker(config-if-eth-0-1)# sflow flow-sampling enable input
Network Packet Broker(config-if-eth-0-1)# sflow counter-sampling enable
```

24.1.4 Validation

The following example shows how to display the information of sflow:

```
Network Packet Broker# show sflow
sFlow Version: 4
sFlow Global Information:
 Agent IPv4 address          : 3.3.3.1
 Counter Sampling Interval    : 20 seconds
 Collector 1:
  IPv4 Address: 3.3.3.2
  Port: 6343

sFlow Port Information:
                          Flow-Sample Flow-Sample
 Port       Counter  Flow     Direction    Rate
-------------------------------------------------------------------
 XGe0-1     enable   enable   Input        32768
```

24.1.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
```

```
!
sflow enable
sflow agent ip 3.3.3.1
sflow counter interval 20
!
sflow collector mgmt-if 3.3.3.2
!
interface eth-0-1
 speed 1000
 duplex full
 sflow counter-sampling enable
 sflow flow-sampling enable input
!
```

25 RPC API Configuration

RPC API service allows user to configure and monitor the switch system through Remote Procedure Calls (RPC) from your program. RPC API service uses JSON over HTTP protocol to communicate the switch from your program. User may issue switch CLI commands through RPC method. By default, the CLI mode is in EXEC mode.

User could send RPC request via an HTTP POST request to URL: http://switch_management_ip_address:switch_tcp_port_number/api/cmd_api/ .

The detailed RPC request and response are show below by JSON format.
RPC server and HTTP server listen same port by default. The HTTP server should be disabled first when we use same port.

25.1.1 Configuration

25.1.2 RPC API Service configuration

RPC API service via http(tcp port 80) is disabled by default. The following example shows how to enable it:

```
Network Packet Broker# configure terminal
Network Packet Broker(config)# service rpc-api enable
Network Packet Broker(config)# exit
```

RPC API service via https (tcp port 443) is enabled by default. The following example shows how to enable it:

```
Switch# configure terminal
Switch(config)# service rpc-api enable ssl
Switch(config)# exit
```

The following example shows how to disable RPC API:

```
Switch# configure terminal
Switch(config)# service rpc-api disable
Switch(config)# exit
```

25.2 JSON-RPC Request

25.2.1 Request

```
{
   "params":
      {
         "format":"json",
         "version":1,
         "cmds":["show services"]
      }
}
```

25.2.2 Response

```
0:
cmd: 'show version'
sequence: 0
sourceDetails: #cli output result
error: False
err_code: 0
err_reason:
1:
cmd: 'config terminal'
sequence: 1
sourceDetails: #cli output result
error: False
err_code: 0
err_reason:
2:
cmd: 'vlan 2'
sequence: 2
sourceDetails: #cli output result
error: False
err_code: 0
err_reason:
3:
```

```
cmd: 'end'
sequence: 3
sourceDetails: #cli output result
error: False
err_code: 0
err_reason:
4:
cmd: 'show running-config'
sequence: 4
sourceDetails: #cli output result
error: False
err_code: 0
err_reason:
```

25.2.3 RPC Error Code

Error code	Description
RPC_ERROR_CLI_TIMEOUT = -1000	RPC TIMEOUT, Don't load too much CLI to system in one message.
RPC_ERROR_CLI_FAIL = -1001	CLI Fail, User should Note the source Details information for detail
RPC_ERROR_CLI_AUTH_FAIL = -1002	Username or password error
RPC_ERROR_CLI_AUTH_LOW = -1003	User privilege is to low
RPC_ERROR_CLI_NOT_SUPPORT = -1004	Unsupported CLI by RPC
RPC_ERROR_CHAR_NOT_SUPPORT = -1005	RPC message format or version can't be supported.
RPC_ERROR_STRING_NOT_SUPPORT = -1006	Unsupported string by RPC, e.g. "service rpc-api disable","ssh","telnet","source","ovs-ofctl snoop","start sh","reboot","reload","format"
RPC_ERROR_MESSAGE_NOT_SUPPORT = -1007	RPC packet format error or version error

25.2.4 Validation

The following example shows how to display the information of system service:

```
DUT1# show services
Networking services configuration:
Service Name Status Port Protocol
----------------+-------------+----------+-----------
dhcp             disable      67/68      UDP
http             disable      80         TCP
https            disable      443        TCP
rpc-api          enable       80         TCP
telnet           enable       23         TCP
ssh              enable       22         TCP
snmp             disable      161        UDP
```

The following example shows how to display the information of rpc-api service:

```
Network Packet Broker # show services rpc-api
RPC-API service configuration:
 Server State       : enable
 Port               80
 Authentication Mode : none
```

```
SSL State              : disable
Message Execute        0
Message Deny           0
```

25.2.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
service rpc-api enable
!
```

26 Packet header add Configuration

26.1 Configuring add the L2-GRE header

26.1.1 Networking requirements

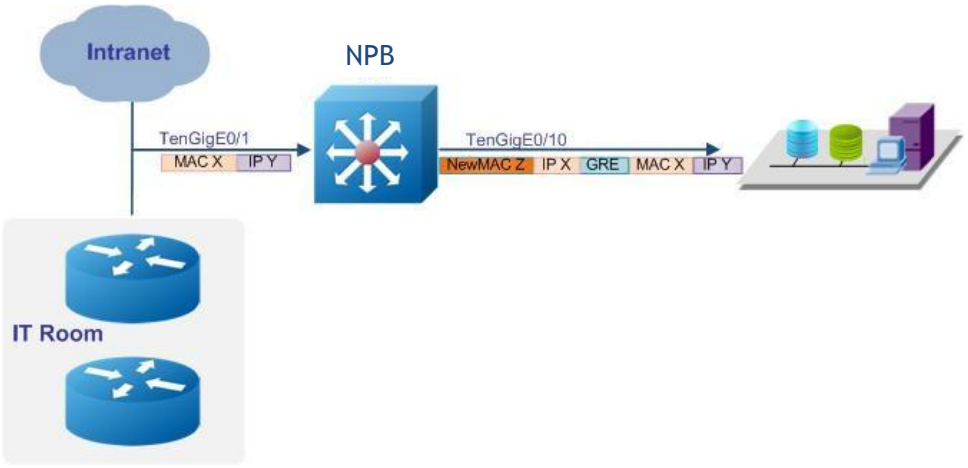

Figure 26-1 Topology of add L2-GRE header

26.1.2 Configuration Ideas

In some cases, server site don not in local place,so traffic with remote sites via L2-GRE. And hold original frame, client need that device have function adding L2-gre packet Header

26.1.3 Configuration

The following example shows how to create a flow rule the match the packets and add L2-GRE header:

```
Network Packet Broker(config)# flowflow1
Network Packet Broker(config-flow-flow1)# permit any src-ip any dst-ip 1.1.0.1 0.0.0.0 add-l2gre l2gre-sip 10.0.0.1 l2gre-dip 10.2.1.1 l2gre-dmac a.a.a l2gre-key 1 l2gre-key-length 24
Network Packet Broker(config-flow-flow1)# exit
```

The following example shows how to create a Network Packet Broker group with ingress port eth-0-1 and flow1:

```
Network Packet Broker(config)# tap-grouptap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow flow1
Network Packet Broker(config-tap-tap1)# egress eth-0-10
Network Packet Broker(config-tap-tap1)# end
```

NOTE: The gre-key-length can config 16,20,24,32 about add-L2-GRE . gre-key-length 16 have gre-key range 1-65535,gre-key-length 20 have gre-key range 1-1048575,gre-key-length 24 have gre-key range 1-16777215,gre-key-length 32 have gre-key range 1-4294967295.

26.1.4 Validation

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show tap-group

tap-group tap1
ID: 10
 Ingress:
     eth-0-1         flow flow1
 egress:
     eth-0-10
```

26.1.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
flow flow1
sequence-num 10 permit any src-ip any dst-ip host 1.1.0.1 add-l2gre l2gre-sip 10.0.0.1 l2gre-dip 10.2.1.1 l2gre-dmac 000a.000a.000a l2gre-key 1 l2gre-key-length 24!
!
tap-group tap1 1
 ingress eth-0-1 flow flow1
```

```
egress eth-0-10
```

26.2 Configuring add the L3-GRE header

26.2.1 Networking requirements

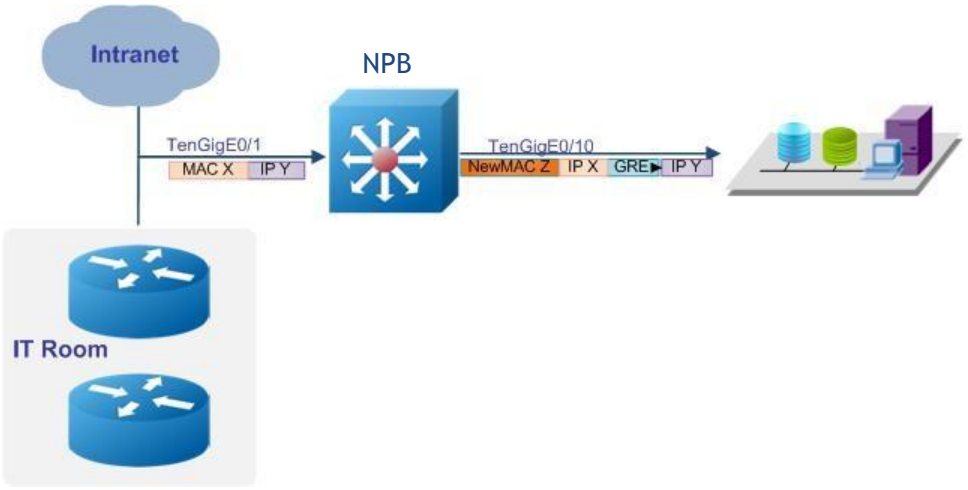

Figure 26-2 Topology of add L3-GRE header

26.2.2 Configuration Ideas

In some cases, server site don not in local place, so traffic with remote sites via L3-GRE.client need that device have function adding L3-gre packet Header

26.2.3 Configuration

The following example shows how to create a flow rule the match the packets and add L3-GRE header:

```
Network Packet Broker(config)# flowflow1
Network Packet Broker(config-flow-flow1)# permit any src-ip host 1.1.0.2 dst-ip any add-l3gre l3gre-sip 3.3.3.3 l3gre-dip 4.4.4.3 l3gre-dmac b.b.b
Network Packet Broker(config-flow-flow1)# exit
```

The following example shows how to create a Network Packet Broker group with ingress port eth-0-1 and flow1:

```
Network Packet Broker(config)# tap-grouptap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow flow1
Network Packet Broker(config-tap-tap1)# egress eth-0-10
Network Packet Broker(config-tap-tap1)# end
```

26.2.4 Validation

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show tap-group

tap-group tap1
ID: 10
 Ingress:
     eth-0-1         flow flow1
 egress:
     eth-0-10
```

26.2.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
flow flow1
sequence-num 10permit any src-ip host 1.1.0.2 dst-ip any add-l3gre l3gre-sip 3.3.3.3 l3gre-dip 4.4.4.3 l3gre-dmac b.b.b
!
tap-group tap1 1
 ingress eth-0-1 flow flow1
egress eth-0-10
```

26.3 Configuring add the VXLAN header

26.3.1 Networking requirements

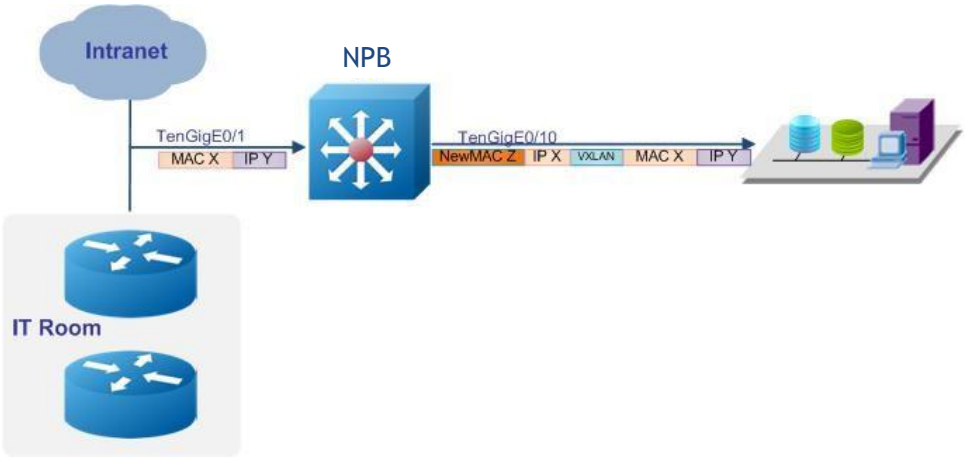

Figure 26-3 Topology of add VXLAN header

26.3.2 Configuration Ideas

In some cases, server site does not in local place, so traffic with remote sites via VXLAN. Client need that device have function adding VXLAN packet Header

26.3.3 Configuration

The following example shows how to create a flow rule the match the packets and add VXLAN header:

```
Network Packet Broker(config)# flowflow1
Network Packet Broker(config-flow-flow1)# permit any src-ip host 1.1.0.2 dst-ip any add-vxlan vxlan-sip 1.1.1.1 vxlan-dip 2.2.2.2 vxlan-dmac a.a.a vxlan-set-vni 100
Network Packet Broker(config-flow-flow1)# exit
```

The following example shows how to create a Network Packet Broker group with ingress port eth-0-1 and flow1:

```
Network Packet Broker(config)# tap-grouptap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow flow1
Network Packet Broker(config-tap-tap1)# egress eth-0-10
Network Packet Broker(config-tap-tap1)# end
```

26.3.4 Validation

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show tap-group

tap-group tap1
ID: 10
 Ingress:
     eth-0-1         flow flow1
 egress:
     eth-0-10
```

26.3.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
flow flow1
sequence-num 10 permit any src-ip host 1.1.0.2 dst-ip any add-vxlan vxlan-sip 1.1.1.1 vxlan-dip 2.2.2.2 vxlan-dmac a.a.a vxlan-set-vni 100
!
tap-group tap1 1
 ingress eth-0-1 flow flow1
egress eth-0-10
```

26.4 Configuring add the ERSPAN header

26.4.1 Networking requirements

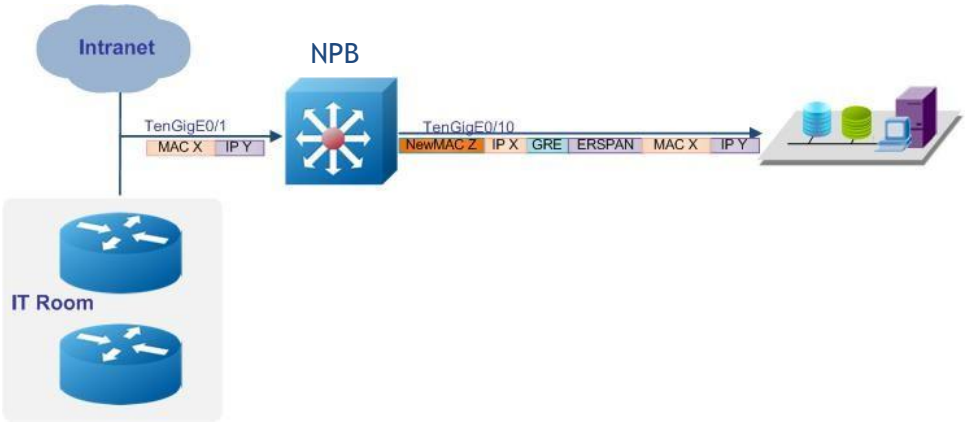

Figure 26-4 Topology of add erspan header

26.4.2 Configuration Ideas

In some cases, server site does not in local place,so traffic with remote sites via erspan.client need that device have function adding erspan packet Header. There are two types of erspan, type1 and type2.

26.4.3 Configuration

The following example shows how to create a flow rule the match the packets and add erspan type1 header:

```
Network Packet Broker(config)# flowflow1
Network Packet Broker(config-flow-flow1)# permit any src-ip host 1.1.0.2 dst-ip any add-erspan erspan-type1 erspan-sip 1.1.1.1 erspan-dip 2.2.2.2 erspan-dmac a.a.a
Network Packet Broker(config-flow-flow1)# exit
```

The following example shows how to create a flow rule the match the packets and add erspan type2 header:

```
Network Packet Broker(config)# flowflow1
Network Packet Broker(config-flow-flow1)# permit any src-ip host 1.1.0.3 dst-ip any add-erspan erspan-type2 erspan-sip 1.1.1.1 erspan-dip 2.2.2.2 erspan-dmac a.a.a erspan-spanid 100
Network Packet Broker(config-flow-flow1)# exit
```

The following example shows how to create a Network Packet Broker group with ingress port eth-0-1 and flow1:

```
Network Packet Broker(config)# tap-grouptap1
Network Packet Broker(config-tap-tap1)# ingress eth-0-1 flow flow1
Network Packet Broker(config-tap-tap1)# egress eth-0-10
Network Packet Broker(config-tap-tap1)# end
```

26.4.4 Validation

The following example shows how to display the information of the Network Packet Broker group:

```
Network Packet Broker# show tap-group

tap-group tap1
ID: 10
 Ingress:
     eth-0-1         flow flow1
 egress:
     eth-0-10
```

26.4.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
flow flow1
sequence-num 10 permit any src-ip host 1.1.0.2 dst-ip any add-erspan erspan-type1 erspan-sip 1.1.1.1 erspan-dip 2.2.2.2 erspan-dmac a.a.a
```

```
sequence-num 20 permit any src-ip host 1.1.0.3 dst-ip any add-erspan erspan-type2 erspan-sip 1.1.1.1 erspan-dip 2.2.2.2 erspan-dmac
000a.000a.000a erspan-spanid 100
!
tap-group tap1 1
 ingress eth-0-1 flow flow1
egress eth-0-10
```

27 Port-group Configuration

27.1 Configuring add the port-group

27.1.1 Networking requirements

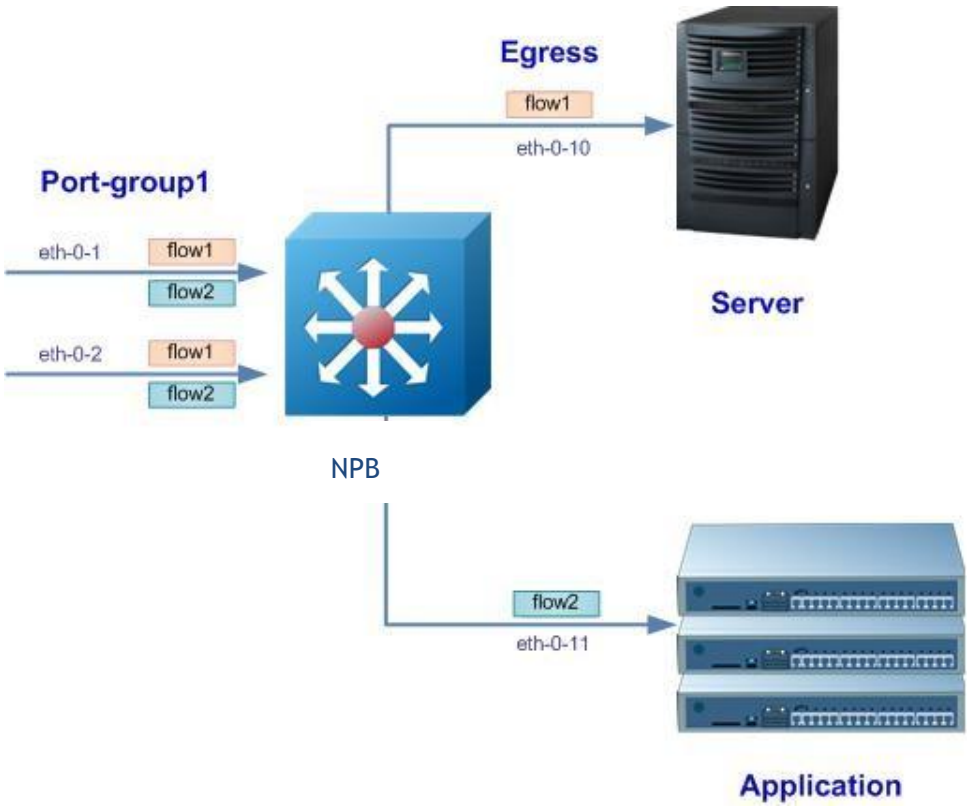

Figure 27-1 Topology of Port-group

27.1.2 Configuration Ideas

In some cases, multiple ports join in a port-group to use an ACL flow resource together.

27.1.3 Configuration

The following example shows how to create a flow rule the match the packets :

```
Network Packet Broker(config)# flowflow1
Network Packet Broker(config-flow-flow1)# permit mpls any
Network Packet Broker(config-flow-flow1)# permit gre src-ip any dst-ip any
Network Packet Broker(config-flow-flow1)# exit
```

The following example shows how to create a port-group and add member interfaces :

```
Network Packet Broker(config)# port-group portgroup1
Network Packet Broker(config-port-portgroup1)# member interface eth-0-1
Network Packet Broker(config-port-portgroup1)# member interface eth-0-2
Network Packet Broker(config-port-portgroup1)# exit
```

The following example shows how to create a Network Packet Broker group with ingress portgroup1 and flow1:

```
Network Packet Broker(config)# tap-group tap1
Network Packet Broker(config-tap-tap1)# ingress portgroup1 flow flow1
Network Packet Broker(config-tap-tap1)# egress eth-0-9
Network Packet Broker(config-tap-tap1)# end
```

The following example shows how to show port-group flow statistics:

```
Network Packet Broker# show port-group flow statistics portgroup1
Network Packet Broker group name: tap1
 flow name: flow1
 sequence-num 10 permit mpls any ( bytes 0 packets 0 )
 sequence-num 20 permit gre src-ip any dst-ip any ( bytes 0 packets 0 )
(total bytes 0 total packets 0 )
```

27.1.4 Validation

The following example shows how to display the information of the flow:

```
Network Packet Broker# show flow
flow flow1
 sequence-num 10 permit mpls any
 sequence-num 20 permit gre src-ip any dst-ip any
```

The following example shows how to display the information of the port-group:

```
Network Packet Broker# show port-group
port-group portgroup1 1
 member interface eth-0-1
 member interface eth-0-2
```

The following example shows how to display the information of the tap-group:

```
Network Packet Broker# show tap-group
truncation            144
timestamp-over-ether : 0000.0000.0000 0000 0000.0000 0x0000

tap-group tap1
 ID: 10
 Ingress:
       portgroup1      flow flow1
 Egress:
       eth-0-9
```

27.1.5 Configuration file

User can display the configuration files as below:

```
Network Packet Broker# show running-config
!
flow flow1
 sequence-num 10 permit mpls any
 sequence-num 20 permit gre src-ip any dst-ip any
exit
!
tap-group tap1 1
 ingress portgroup1 flow flow1
 egress eth-0-9
!
port-group portgroup1 1
 member interface eth-0-1
 member interface eth-0-2
!
```

28 Configuring IPFIX

28.1 Overview

28.1.1 Function Introduction

Traffic on a data network can be seen as consisting of flows passing through network elements. For administrative or other purposes, it is often interesting, useful, or even necessary to have access to information about these flows that pass through the network elements. This requires uniformity in the method of representing the flow information and the means of communicating the flows from the network elements to the collection point. This is what IPFIX can do.

Before IPFIX was introduced, there is a Cisco private method NetFlow. IPFIX is similar to NetFlow and is based on NetFlow version 9.

28.1.2 Principle Description

N/A

28.2 Configuration

step 1 Enter the configure mode

```
Switch# configure terminal
```

step 2 Set the aging time(optional)

Set the aging time as 300 seconds. The aging time is 1800 seconds by default.

```
Switch(config)# ipfix global
Switch(Config-ipfix-global)# flow aging 300
Switch(Config-ipfix-global)# exit
```

step 3 Configuring recorder

```
Switch(config)# ipfix recorder recorder1
Switch(Config-ipfix-reocrder)# match mac source address
Switch(Config-ipfix-reocrder)# match mac destination address
Switch(Config-ipfix-reocrder)# match ipv4 source address mask 32
Switch(Config-ipfix-reocrder)# match ipv4 destination address mask 32
Switch(Config-ipfix-reocrder)# collect counter bytes
Switch(Config-ipfix-reocrder)# collect counter packets
Switch(Config-ipfix-reocrder)# exit
```

step 4 Configuring sampler(optional)

```
Switch(config)# ipfix sampler sampler1
Switch(Config-ipfix-sampler)# 1 out-of 100
Switch(Config-ipfix-sampler)# exit
```

step 5 Configuring exporter

```
Switch(config)# ipfix exporter exporter1
Switch(Config-ipfix-exporter)# destination mgm-if ipv4 9.0.0.1
Switch(Config-ipfix-exporter)# flow data timeout 200
Switch(Config-ipfix-exporter)# exit
```

step 6 Configuring the other condition of exporter send ipfix cache(optional)

```
Switch(config)# ipfix exporter exporter1
Switch(Config-ipfix-exporter)# event flow end timeout
Switch(Config-ipfix-exporter)# flow data flush threshold count 20
Switch(Config-ipfix-exporter)# exit
```

step 7 Configuring monitor

```
Switch(config)# ipfix monitor monitor1
Switch(Config-ipfix-monitor)# recorder recorder1
Switch(Config-ipfix-monitor)# exporter exporter1
Switch(Config-ipfix-monitor)# exit
```

step 8 Enter the interface configure mode and apply ipfix

```
Switch(config)# interface eth-0-1
Switch(config-if)# ipfix monitor input monitor1 sampler sampler1
Switch(config-if)# no shutdown
Switch(config-if)# exit
```

step 9 Exit the configure mode

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

step 10 Send 100 ip packets to eth-0-1

step 11 Validation

Use the following commands to validate the configuration:

```
Switch# show ipfix global
```

```
IPFIX global informaition:
    Current flow cache number              : 1(ingress: 1, egress: 0)
    Flow cache aging interval              : 300 seconds
    Flow cache export interval             : 5 seconds
    Flow cache sampler mode                : all

Switch# show ipfix recorder recorder1
IPFIX recorder information:
   Name          : recorder1
   Description :
   Match info    :
      match Source Mac Address
      match Destination MAC Address
      match IPv4 Source Address
      match IPv4 Destination Address
   Collect info :
      collect Flow Byte Number
      collect Flow Packet Number

Switch# show ipfix exporter exporter1
IPFIX exporter information:
   Name                        : exporter1
   Description                 :
   Domain ID                   0
   Collector Name              : 9.0.0.1
   IPFIX message protocol       : UDP
   IPFIX message destination Port    2055
   IPFIX message TTL value         255
   IPFIX message DSCP value        63
   IPFIX data interval             200
   IPFIX template interval         1800
   IPFIX exporter events           :
      Flow aging event

Switch# show ipfix sampler sampler1
IPFIX sampler information:
   Name                        : sampler1
   Description                 :
   Rate                        100

Switch# show ipfix monitor monitor1
IPFIX monitor information:
   Name                        : monitor1
   Description                 :
   Recorder                    : recorder1
   exporter                    : exporter1

Switch# show ipfix cache observe-point interface eth-0-1 input
 Cache dir           : input
 Cache flow profile : 0
 Cache key profile : 0
 Cache key info :
    Source mac        : 0000.0002.0001
    Destination mac    : 0000.0002.0002
    ipsa              : 10.10.10.3/32
    ipda              : 10.10.10.1/32
 Cache collect info:
    Byte number of ingress              64
```

Packet number of ingress	1

28.3 Application cases

N/A

29 Configuring Import Certificate

29.1 Overview

29.1.1 Function Introduction

The current device is using our own certificate and is not authorized by the relevant authority, so the browser considers the address insecure every time HTTPS access occurs. This command can import the customer's own certificate to improve security.

29.1.2 Principle Description

N/A

29.2 Configuration

step 1 Upload the new certificate file

```
Switch# copy mgmt-if tftp://10.10.38.160/cert.pem flash:/boot/
```

step 2 Enter the configure mode

```
Switch# configure terminal
```

step 3 Turn on HTTPS service

```
Switch(config)# service http disable
Switch(config)# service https enable
Switch(config)#
```

step 4 Load new certificate file

```
Switch(config)# certificate load pem-cert flash:/boot/cert.pem
Switch(config)#
```

step 5 Restart HTTPS service

```
Switch(config)# service https disable
Switch(config)# service https enable
Switch(config)#
```

29.3 Application cases

N/A

Tips

To full fill the keyword of any command line in any command mode, use TAB on the keyboard. It is unnecessary to type every letter of the keywords.

To get the help information of the command line, use the "?" symbol.

To quit to the up level of the command mode, use "quit" or "exit". To return to Privileged EXEC mode, use "end".

To save the current configuration, use "write memory". User should use the "write memory" command on time in order to prevent loss the configuration after device reboot.

To get more description of the command line, please reference to the CLI guide.

To get detailed information about the feature, please reference to the User guide.

The "no" form of the command line is usually used to delete the configuration or restore the default value. E.g.: configuration "speed 1000"should be removed by "no speed".