# ZXR10 5900E Series
## All Gigabit-Port Intelligent Routing Switch
User Manual (Ethernet Switching Volume)

Version 2.8.23.B

ZTE CORPORATION
ZTE Plaza, Keji Road South,
Hi-Tech Industrial Park,
Nanshan District, Shenzhen,
P. R. China
518057
Tel: (86) 755 26771900
Fax: (86) 755 26770801
URL: http://ensupport.zte.com.cn
E-mail: support@zte.com.cn

**Revision History**

| Revision No. | Revision Date | Revision Reason |
|---|---|---|
| R1.0 | Nov. 9, 2009 | First Release |

Serial Number: sjzl20096393

# Contents

# About This Manual

**Purpose**

This manual is ZXR10 5900E (V2.8.23.B) Series All Gigabit-Port Intelligent Routing Switch User Manual (Ethernet Switching Volume). This manual introduces Ethernet switching functions supported by ZXR10 5900E including VLAN configuration, STP configuration, ZESS configuration and Ethernet OAM configuration.

**Intended Audience**

This manual is intended for the following engineers:

- on-site maintenance engineers
- network monitor engineers
- system maintenance engineer

**What Is in This Manual**

ZXR10 5900E (V2.8.23.B) Series All Gigabit-Port Intelligent Routing Switch User Manual (Ethernet Switching Volume) contains the following chapters:

| Chapter | Summary |
|---|---|
| Chapter 1 VLAN Configuration | This chapter introduces VLAN concept, related configuration command and configuration example. |
| Chapter 2 SVLAN Configuration | This chapter introduces SVLAN concept, related configuration command and configuration example. |
| Chapter 3 SVLAN COS Configuration | This chapter introduces SVLAN COS concept, related configuration command and configuration example. |
| Chapter 4 MAC Table Configuration | This chapter introduces MAC Table concept, related configuration command and configuration example. |
| Chapter 5 STP Configuration | This chapter introduces STP concept, related configuration command and configuration example. |
| Chapter 6 ZESR/ZESR+ Configuration | This chapter introduces ZESR/ZESR+ concept, related configuration command and configuration example. |
| Chapter 7 ZESS Configuration | This chapter introduces ZESS concept, related configuration command and configuration example. |
| Chapter 8 ZESR and SVLAN Linkage Networking Configuration | This chapter introduces ZESR and SVLAN Linkage Networking concept, related configuration command and configuration example. |
| Chapter 9 Link Aggregation Configuration | This chapter introduces Link Aggregation concept, related configuration command and configuration example. |

| Chapter | Summary |
|---|---|
| Chapter 10 IGMP Snooping Configuration | This chapter introduces IGMP Snooping concept, related configuration command and configuration example. |
| Chapter 11 UDLD Configuration | This chapter introduces UDLD concept, related configuration command and configuration example. |
| Chapter 12 LLDP Configuration | This chapter introduces LLDP concept, related configuration command and configuration example. |
| Chapter 13 L2PT Configuration | This chapter introduces L2PT concept, related configuration command and configuration example. |
| Chapter 14 Ethernet OAM Configuration | This chapter introduces Ethernet OAM concept, related configuration command and configuration example. |
| Chapter 15 sflow Configuration | This chapter introduces sflow concept, related configuration command and configuration example. |
| Chapter 16 IPFIX Configuration | This chapter introduces IPFIX concept, related configuration command and configuration example. |

**Related Documentation**

- ZXR10 5900E (V2.8.23.B) Series All Gigabit-Port Intelligent Routing Switch Hardware Manual
- ZXR10 5900E (V2.8.23.B) Series All Gigabit-Port Intelligent Routing Switch User Manual (Basic Configuration Volume)
- ZXR10 5900E (V2.8.23.B) Series All Gigabit-Port Intelligent Routing Switch User Manual (Ethernet Switching Volume)
- ZXR10 5900E (V2.8.23.B) Series All Gigabit-Port Intelligent Routing Switch User Manual (IPv4 Routing Volume)
- ZXR10 5900E (V2.8.23.B) Series All Gigabit-Port Intelligent Routing Switch User Manual (IPv6 Routing Volume)

Chapter 1

# VLAN Configuration

**Table of Contents**



# VLAN Overview

Virtual Local Area Network (VLAN) is a technology that divides a physical network into several logical (virtual) Local Area Networks (LANs). Each VLAN is identified by a VLAN ID (VID).

VLAN technology divides users within a physical LAN into different broadcast domains (VLANs) according to requirements. Users with the same demands are grouped to the same broadcast domain, while those with different demands are separated.

Each VLAN, like a logically independent LAN, shares the same attributes as those physical LANs. All broadcast and unicast traffics within a VLAN are limited to the VLAN but are not forwarded to any other VLAN. Devices in different VLANs must rely on L3 routing switching for communication between them.

VLAN provides the following advantages:

1. Lower broadcast traffic on the network
2. Enhanced network security
3. Streamlined network management

## VLAN Type

LAN type of a device depends on how it will divide a received frame to a VLAN. ZXR10 5900E supports port-based VLAN, the simplest and most effective method of VLAN division. It divides its various ports into different VLANs, so that any traffic received on a port belongs to its corresponding VLAN.

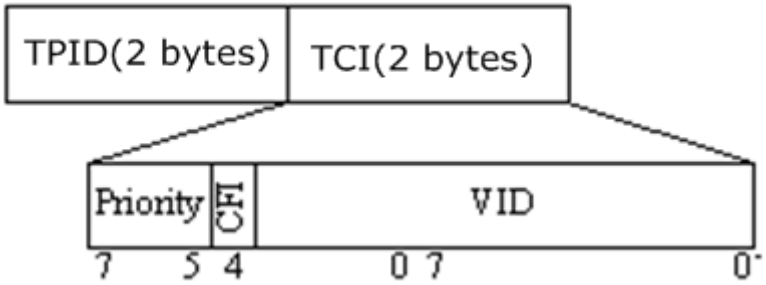
Assume ports 1, 2 and 3 belong to the same VLAN, while the other ports belong to other VLANs, then frames received on port 1 are broadcast to ports 2 and 3 only while they are not passed to any other port. When a user in a VLAN moves to a new location, it no longer belongs to the original VLAN unless the user is assigned to that VLAN again.

# VLAN Tag

It is possible to transmit services of several VLANs over a single link if a frame carries information about its native VLAN while being passed through a network. IEEE 802.1Q implements this function through attaching a VLAN tag to the Ethernet frame.

A VLAN tag is a four-byte long number, and it comes after the source MAC address and before the length/type field in an Ethernet frame. Figure 1 shows the VLAN tag format.

**FIGURE 1 VLAN TAG FORMAT**




VLAN tag is applied to cross-switch VLANs, when the link between switches is usually called a trunk. VLAN tag allows VLANs cross several switches to be created through one or more trunks. When the ports connecting these switches receive a tagged frame, the ports can judge which VLAN the frame belongs to based on the VLAN tag.

Each 802.1Q port is allocated a default VLAN ID, called PVID. Untagged frames received on a port is considered belonging to the default VLAN, and then broadcasted in that VLAN.

ZXR10 5900E supports IEEE 802.1Q tag.

# VLAN Link Type

ZXR10 5900E ports support the following links:

1. Access link

   It connects devices (such as workstation) that cannot identify VLAN tags to the VLAN switch port. It transmits untagged frames only to a single VLAN.

2. Trunk link

It connects two devices that can identify VLAN tags and carries several VLAN's services. It transmits tagged frames only to several VLANs. The most common trunk link is the one between two VLAN switches.

3. Hybrid Link

   It transmits both tagged and untagged frames. For a given VLAN, however, it only transmits frames of the same type.

# Default VLAN

ZXR10 5900E has a default VLAN initially, which has the following features:

- VLAN ID as 1
- VLAN name as VLAN0001
- All ports included
- Untagged by default on all ports

# PVLAN

To improve network security, messages among different users shall be separated. The traditional method is to assign a VLAN to each user. The method has obvious limitation, which can be seen from the following aspects:

1. At present, IEEE 802.1Q standard supports utmost 4094 VLANs, which limits the number of users and network expansion.
2. Each VLAN corresponds to one IP subnet, so vast divided subnets will cause the waste of IP addresses.
3. Planning and management to a mass of VLANs and IP subnets is extremely complicated.

PVLAN (Private VLAN) technology is developed to solve these problems.

PVLAN divides the ports in VLAN into three types: the port connecting to the user is called Isolate Port, the port connecting to a group of users that need interconnection and intercommunication is called Community Port and the port connecting to the upstream router is called Promiscuous Port. The isolated port communicates with the promiscuous port only, but not with any other isolated port or community port. Community port can communicate with promiscuous port and any other community port, but not with isolated port. Thus ports in the same VLAN are separated. The user who connects with isolated port can only communicate with its default gateway, the user who connects community port can interconnect and intercommunicate. Network security is ensured.

ZXR10 5900E supports 20 PVLAN groups, each group having customized isolated ports and at most 256 isolated ports, 16 community ports and 8 promiscuous ports.

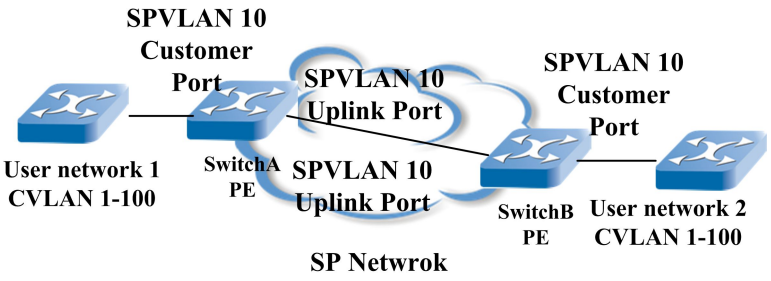
# QinQ

QinQ, also known as VLAN stack, is a graphic name for the IEEE 802.1Q based tunnel protocol. QinQ technology encapsulates the original VLAN tag (inner tag) with another VLAN tag (outer tag) so that the inner tag is masked.

QinQ implements simple Layer 2 VPN (L2VPN) without protocol support, applicable to small-sized LANs with L3 switches as their core.

QinQ typical networking is shown in Figure 2. Port to the customer network is called the customer port. The port to the Service Provider network is called the uplink port, and the SP edge access device is called Povider Edge (PE).

**FIGURE 2 QINQ TYPICAL NETWORKING**




The customer network is usually connected to the PE through trunk VLAN. Uplinks ports in the SP network are connected symmetrically in the trunk VLAN mode.

When a packet (tagged/untagged) from customer network 1 reaches the customer port of switch A. Switch A attaches an outer tag (VLAN ID as 10) to its forcibly. Within the SP network, packet is forwarded to all ports in VLAN 10. This packet finally arrives at switch B. Switch B recognizes that the port to customer network 2 is a customer port, then dispatches the outer tag complying with the traditional 802.1Q to restore the original packet. Switch B sends the packet to customer network 2.

Thus data is transparently transmitted between customer networks 1 and 2 through the SP network. This allows the customer considerable flexibility for Private VLAN ID planning, without any conflict with those of the SP network.

# Subnet VLAN

Subnet-based VLAN applies to L2 VLAN networks for flexible configuration of data frame forwarding. Subnet-based VLAN forward a data frame to a VLAN based on the source IP address. This source IP address based VLAN can forward user data from different subnets cross several VLANs, while remain the original VLAN membership unchanged.

Subnet VLAN isolates data frames from different source IP addresses so that a user has access to data from its own subnet only. The subnet VLAN priority in untagged frame forwarding is higher than that of protocol VLAN or PVID; the priority of tagged frame forwarding in the tagged mode is higher than that of subnet VLAN.

Port is enabled on Subnet VLAN by default and also can be disabled according to actual demands.

ZXR10 5900E supports up to 256 subnet VLANs, that is, supports processing data frames from 256 source IP subnets.

# Protocol VLAN

Protocol-based VLAN applies to L3 networks or those running many protocols. Protocol-based VLAN divides packets based on their network layer encapsulation protocol. Packets with the same tag belong to the same protocol VLAN. This network layer protocol based VLAN can broadcast packets cross several VLAN switches. It allows users to move freely in the network while remain their VLAN membership unchanged.

This solution eliminates the need to reconfigure the VLAN when a user moves to another physical location in the network. In addition, as VLANs are identified based on the protocol type instead of attached frame tag, traffic through the network reduces.

Protocol VLAN is not only enabled on all physical ports by default but also disabled on ports according to demands, which identifies that VLANs are based on the packet tag only. It separates packets with different tags so that users have access to data from other users in the same VLAN only.

ZXR10 5900E supports up to 16 protocol VLANs. This means that protocol VLANs support processing packets with 16 kinds of tags.

# VLAN Translation

VLAN translation permits the switches of different users to have same VLAN ID. With VLAN translation, core switch modifies the same VLAN ID of edge switches into the different VLAN ID. This function isolate user in core switch and simply the configuration of edge switch.

ZXR10 5900E support 768 VLAN translation.

# SuperVLAN

On a traditional ISP network, one IP subnet is allocated to each user, which means that the occupation of three IP addresses by a single user as its subnet address, broadcast address and default gateway address. Even if there are a number of idle IP addresses

in a user's subnet, they cannot be allocated to other users. This causes waste of IP addresses.

SuperVLAN effectively resolves this problem. It merges several VLANs (called sub-VLAN) to a SuperVLAN so that they can use the same IP subnet and default gateway.

With the SuperVLAN technology, the ISP needs only one IP subnet for its SuperVLAN. It creates a sub-VLAN for each of its users. These sub-VLANs can use flexibly the IP addresses in the SuperVLAN subnet and share the default gateway of the SuperVLAN. Each sub-VLAN is an independent broadcast domain, ensuring user isolation, and communicates with other sub-VLANs through SuperVLAN routing.

# Configuring VLAN

## Creating VLAN

| Command | Function |
|---|---|
| `ZXR10(config)#` **vlan** {<*vlan-id*>\|<*vlan-name*>} | This creates specific VLAN and enters into VLAN configuration mode. |

## Creating a VLAN in VLAN Database

To create a VLAN in VLAN database, use the following command.

| Command | Function |
|---|---|
| `ZXR10(config)#`**vlan list** <*vlan-list*>[**name** <*vlan-name*>] | This creates a VLAN in VLAN database. |

## Setting VLAN Name

To set VLAN name, use the following command.

| Command | Function |
|---|---|
| `ZXR10(config-vlanX)#`**name** <*vlan-name*> | This sets VLAN name. |

VLAN name uniquely identifies a VLAN. This can be a group, department and region name. By default, a VLAN name is "VLAN" + VLAN ID. VLAN ID includes four digits (0s are pretended when there are less than four digits). Name of VLAN 4 is VLAN0004.

# Setting VLAN Link Type on Ethernet Interface

To set VLAN link type on Ethernet port, use the following command.

| Command | Function |
|---|---|
| ZXR10(config-gei_1/x)#**switchport mode** {**access**\|**trunk**\|**hybrid**} | This sets VLAN link type on Ethernet port. |

There are three VLAN link types for Ethernet interface of ZXR10 5900E: Access mode, Trunk mode and Hybrid mode. Access mode is used by default.

- Ports of access mode belong to only one VLAN, support untagged frames and are usually connected to computers.
- Ports of trunk mode can belong to several VLANs (receives/sends packets from/to several VLANs), support tagged frames, and are usually used as trunk ports between switches.
- Ports of hybrid mode can belong to several VLANs (receives/sends packets from/to several VLANs), support both tagged and untagged frames (customized), and can be used to connect both switches and computers.

  Ports of hybrid mode are different from trunk ports. They send both tagged and untagged frames (trunk ports send untagged frames only when they are from the default VLAN).

# Adding VLAN Member Port

To add an access, trunk or hybrid port to a specified VLAN, use following commands.

Access port only can be added to one VLAN, Trunk port and Hybrid port can be added into multiple VLANs.

- To add an access port into a specific VLAN, use the following commands.

| Command | Function |
|---|---|
| ZXR10(config-gei_1/x)#**switchport access vlan** {<*vlan-id*>\|<*vlan-name*>} | This command adds an access port to a specified VLAN. |

- To add a trunk port to a specific VLAN, use the following command.

| Command | Function |
|---|---|
| ZXR10(config-gei_1/x)#**switchport trunk vlan** <*vlan-list*> | This command adds a trunk port to a specified VLAN. |

- To add a hybrid port into a specific VLAN, use the following commands.

| Command | Function |
|---|---|
| ZXR10(config-gei_1/x)#**switchport hybrid vlan** <*vlan-list*>[**tag**\|**untag**] | This command adds a hybrid port to a specified VLAN. |

# Adding Ports to a VLAN in Batches

To add ports to a VLAN in batches, use the following command.

| Command | Function |
|---|---|
| ZXR10(config-vlanX)#**switchport** {**pvid**\|**tag**\|**untag**}<*port-list*> | This adds ports to a VLAN in batches. |

# Setting the Native VLAN for a Trunk or Hybrid Port

An access port belongs to only one VLAN, its native VLAN is the VLAN to which it belongs. This requires no additional configuration.

Trunk port and hybrid port belong to multiple vlans and they need to set native vlan. If native vlan is set on port, when one frame with no vlan tag is received on port, it will be forwarded to the port belonging to this native vlan. Native vlan of trunk port and hybrid port is vlan 1 by default.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config-gei_1/x)#**switchport trunk native vlan** {<*vlan-id*>\|<*vlan-name*>} | This command sets native VLAN for a trunk port. |
| 2 | ZXR10(config-gei_1/x)#**switchport hybrid native vlan** {<*vlan-id*>\|<*vlan-name*>} | This command sets native VLAN for a hybrid port. |

# Setting VLAN Filtering on a Port

To set VLAN filtering on a port, use the following command.

| Command | Function |
|---|---|
| ZXR10(config-gei_1/x)# **ingress filtering** {**enable**\|**disable**} | This sets VLAN filtering on a port. |

When ingress filtering is enabled on a port, the port drops a received frame if the VLAN to which the frame belongs does not include the ingress. By default, VLAN ingress filtering is enable.

# Setting Frame Filtering Type of a Port

To set frame filtering type of a port, use the following command.

| Command | Function |
|---|---|
| ZXR10(config-gei_1/x)#**acceptable frame types** {**all**\|**tag**} | This sets frame filtering type of a port. |

This sets the frame type of the port which can receive all types of frames including untagged and tagged frames. By default, all frames are received.

# Creating VLAN Layer 3 Interface

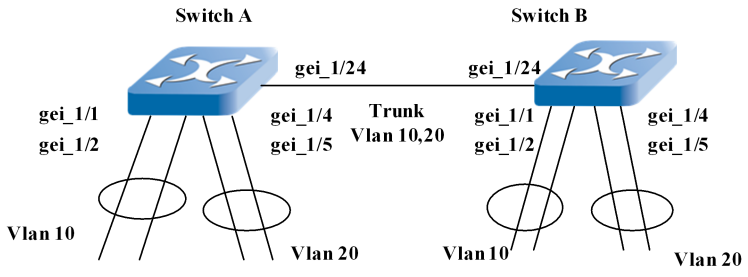
| Command | Function |
|---|---|
| ZXR10(config)#**interface** {**vlan** <*vlan-id*>\|<*vlan-if*>} | This creates VLAN layer 3 interface. |

It is necessary to create this VLAN before creating VLAN layer 3 interface.

# VLAN Configuration Example

The ports gei_1/1 and gei_1/2 on Switch A, and the ports gei_1/1 and gei_1/2 on switch B, belong to VLAN 10; the ports gei_1/4 and gei_1/5 on switch A, and the ports gei_1/4 and gei_1/5 on switch B, belong to VLAN 20. The four ports are all access ports, as shown in Figure 3. Switches A and B are connected through ports gei_1/24 (two trunk ports) over a trunk link.

**FIGURE 3 VLAN NETWORKING**



Configuration of Switch A:

```
ZXR10_A(config)#vlan 10
ZXR10_A(config-vlan10)#switchport pvid gei_1/1-2
ZXR10_A(config-vlan10)#exit
ZXR10_A(config)#vlan 20
ZXR10_A(config-vlan20)#switchport pvid gei_1/4-5
ZXR10_A(config-vlan20)#exit
ZXR10_A(config)#interface gei_1/24
ZXR10_A(config--gei_1/24)#switchport mode trunk
ZXR10_A(config-gei_1/24)#switchport trunk vlan 10
ZXR10_A(config-gei_1/24)#switchport trunk vlan 20
ZXR10_A(config-gei_1/24)#exit
```

Configuration of Switch B:

```
ZXR10_B(config)#vlan 10
ZXR10_B(config-vlan10)#switchport pvid gei_1/1-2
ZXR10_B(config-vlan10)#exit
ZXR10_B(config)#vlan 20
ZXR10_B(config-vlan20)#switchport pvid gei_1/4-5
ZXR10_B(config-vlan20)#exit
ZXR10_B(config)#interface gei_1/24
ZXR10_B(config-gei_1/24)#switchport mode trunk
ZXR10_B(config-gei_1/24)#switchport trunk vlan 10
ZXR10_B(config-gei_1/24)#switchport trunk vlan 20
ZXR10_B(config-gei_1/24)#exit
```

# Configuring PVLAN

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**vlan private-map session-id** <*id*>[**isolate** <*port-list*>][**promis** <*port-list*>][**community** <*port-list*>] | This configures isolated ports, promiscuous ports and community port of Private VLAN. |
| 2 | ZXR10(config)#**show vlan private-map** | This displays the configuration information of PVLAN. |

**Example** The configuration of two isolated groups is shown below.

Isolated group 1: gei_1/1,gei_1/2,xgei_2/1 and xgei_3/1 are isolated ports, port gei_1/10 is a promiscuous port.

Isolated group 2: gei_1/3,gei_1/4 and gei_4/1 are isolated ports, gei_1/5,gei_1/6 and gei_5/1 are community ports, gei_1/11 and gei_1/12 are promiscuous ports.

The detailed configuration is as follows:

```
ZXR10(config)#vlan private-map session-id 1 isolate
 gei_1/1-2,gei_2/1,gei_3/1 promis gei_1/10
ZXR10(config)#vlan private-map session-id 2 isolate
gei_1/3-4,gei_4/1 promis gei_1/11-12 community gei_1/5-6,gei_5/1
ZXR10(config)#show vlan private-map
Session Isolate_Ports Promis_Ports Community_ports   Vlan_Cfg Vlan
-------------------------------------------------------------------
1       gei_1/1-2,gei_1/10
        _2/1,gei_3/1
2       gei_1/3-4,gei_1/11-12  gei_1/5-6,gei_5/1
        _4/1
ZXR10(config)#
```

# Configuring QinQ

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config-if)#**switchport** <*port-list*> **qinq** {**normal**\|**uplink**\|**customer**\|**tpid** <*tpid*>} | When configuring QinQ, it needs to set customer port of SPVLAN to untagged port and uplink port to tagged port. |
| 2 | ZXR10(config)# **show qinq** | This views configuration information of QinQ. |

**Example** In figure QINQ TYPICAL NETWORKING, assume switch A's customer port is gei_1/1, its uplink port is gei_1/24, switch B's customer port is gei_1/1 and its uplink port is gei_1/24.

Configuration of Switch A:

```
ZXR10_A(config)#vlan 10
ZXR10_A(config-vlan)#exit
ZXR10_A(config)#interface gei_1/1
ZXR10_A(config-if)#switchport qinq customer
ZXR10_A(config-if)#switchport access vlan 10
ZXR10_A(config-if)#exit
ZXR10_A(config)#interface gei_1/24
ZXR10_A(config-if)#switchport qinq uplink
ZXR10_A(config-if)#switchport mode trunk
ZXR10_A(config-if)#switchport trunk vlan 10
ZXR10_A(config-if)#exit
```

Configuration of Switch B:

```
ZXR10_B(config)#vlan 10
ZXR10_B(config-vlan)#exit
ZXR10_B(config)#interface gei_1/1
ZXR10_B(config-if)#switchport qinq customer
ZXR10_B(config-if)#switchport access vlan 10
ZXR10_B(config-if)#exit
ZXR10_B(config)#interface gei_1/24
ZXR10_B(config-if)#switchport qinq uplink
ZXR10_B(config-if)#switchport mode trunk
ZXR10_B(config-if)#switchport trunk vlan 10
ZXR10_B(config-if)#exit
```

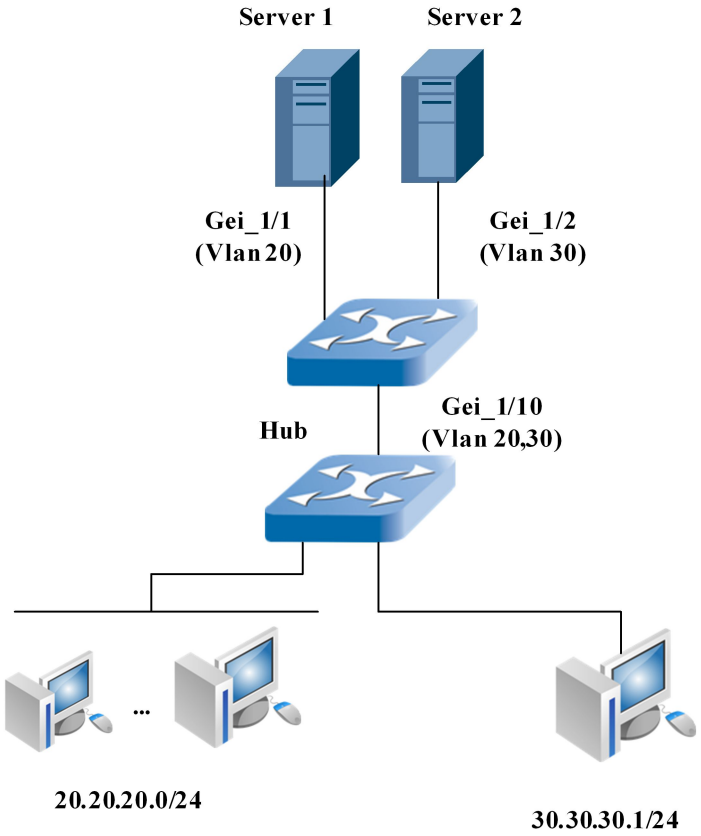
# Configuring Subnet VLAN

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**vlan subnet-map session-no**<*session-no*><*ipaddr*><*mask*> **vlan** {<*vlanid*><*name*>} | This creates a subnet VLAN. |

| Step | Command | Function |
|---|---|---|
| 2 | ZXR10(config)#**show vlan subnet-map** | This displays configuration of subnet VLAN. |

**Example** As shown in Figure 4, configure VLAN data on the switch. Configure VLAN20 and VLAN30. Port gei_1/1 belongs to VLAN20, port gei_1/2 belongs to VLAN30, port gei_1/10 belongs to both VLAN20 and VLAN30. Configure different PVIDs for gei_1/1, gei_1/2 and gei_1/10. PCs in 20.20.20.0/24 have access to server 1, and the PC whose IP address is 30.30.30.1 has access to server 2.

**FIGURE 4 SUBNET VLAN CONFIGURATION EXAMPLE**




Configuration of switch:

```
/*Create a VLAN and assign ports to it*/
ZXR10(config)#interface gei_1/1
ZXR10(config-gei_1/1)#switchport mode hybrid
ZXR10(config-gei_1/1)#switchport hybrid native vlan 20
ZXR10(config-gei_1/1)#switchport hybrid vlan 20 untag
ZXR10(config-gei_1/1)#exit
ZXR10(config)#interface gei_1/2
ZXR10(config-gei_1/2)#switchport mode hybrid
ZXR10(config-gei_1/2)#switchport hybrid native vlan 30
ZXR10(config-gei_1/2)#switchport hybrid vlan 30 untag
ZXR10(config-gei_1/2)#exit
```

```
ZXR10(config)#interface gei_1/10
ZXR10(config-gei_1/10)#switchport mode hybrid
ZXR10(config-gei_1/10)#switchport hybrid vlan 20,30 untag
ZXR10(config-gei_1/10)#exit
/*Create subnet VLAN data*/
ZXR10(config)#vlan subnet-map session-no
1 20.20.20.0 255.255.255.0 vlan 20
ZXR10(config)#vlan subnet-map session-no
2 30.30.30.1 255.255.255.255
vlan 30
/*disable subnet VLAN in some ports which don’ t need subnet VLAN*/
ZXR10(config)#interface gei_1/5
ZXR10(config-gei_1/5)#vlan subnet-map disable
ZXR10(config-gei_1/5)#exit
```

# Configuring Protocol VLAN

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**vlan protocol-map session-no** *<session-no>*{**ethernet2**\| **llc**\| **snap**}*<0xHHHH>* **vlan** {*<vlanId>*\|*<name>*} | This configures protocol VLAN. |
| 2 | ZXR10(config)#**show vlan protocol-map** | This views the configuration of protocol VLAN. |

**Example** Customer port gei_1/1 of a switch receives 0X1000 packets and 0X1001 packets. These packets with different tags can be observed on the other two ports, gei_1/2 and gei_1/3 respectively.

The detailed configuration is as follows:

Configuration of switch:

```
/*create protocol vlan data*/
ZXR10(config)#vlan protocol-map session-no 1 ethernet2
 0x1000 vlan 10
ZXR10(config)#vlan protocol-map session-no 2 ethernet2
0x1001 vlan 20
/*put port into corresponding VLAN*/
ZXR10(config)#interface gei_1/1
ZXR10(config-if)#switchport mode trunk
ZXR10(config-if)#switchport trunk vlan 10,20
ZXR10(config-if)#exit
ZXR10(config)#interface gei_1/2
ZXR10(config-if)#switchport mode trunk
ZXR10(config-if)#switchport trunk vlan 10
ZXR10(config-if)#exit
ZXR10(config)#int gei_1/3
ZXR10(config-if)#switchport mode trunk
ZXR10(config-if)#switchport trunk vlan 20
ZXR10(config-if)#exit
/*disable protocol vlan in some ports which don’ t
 need protocol vlan*/
ZXR10(config)#interface gei_1/5
ZXR10(config-gei_1/5)#vlan protocol-map disable
ZXR10(config-gei_1/5)#exit
```

# Configuring VLAN Translation

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**vlan translate session-no** <*session_id*> **ingress-port** <*interface-name*> **ingress-vlan** <*vlan-list*> **egress-vlan** <*vlanId*> | This configures VLAN Translation. |
| 2 | ZXR10(config)#**show vlan translate** | This views the configuration of VLAN Translation. |
| 3 | ZXR10(config)#**vlan egr-translate session-no** <*session_id*> **egress-port** <*interface-name*> **egress-vlan** <*vlan-list*> **ingress-vlan** <*vlanId*> | This configures VLAN egr-translation. |

**Example** Port gei_1/1 receives a packet which belongs to vlan100. This packet is to be sent to xgei_2/1. Port xgei_2/1 belongs to VLAN 200. As for the downlink data, users hope that VLAN 200 forwarded from xgei_2/1 is converted to VLAN 100.

The detailed configuration is as follows:

Configuration of switch:

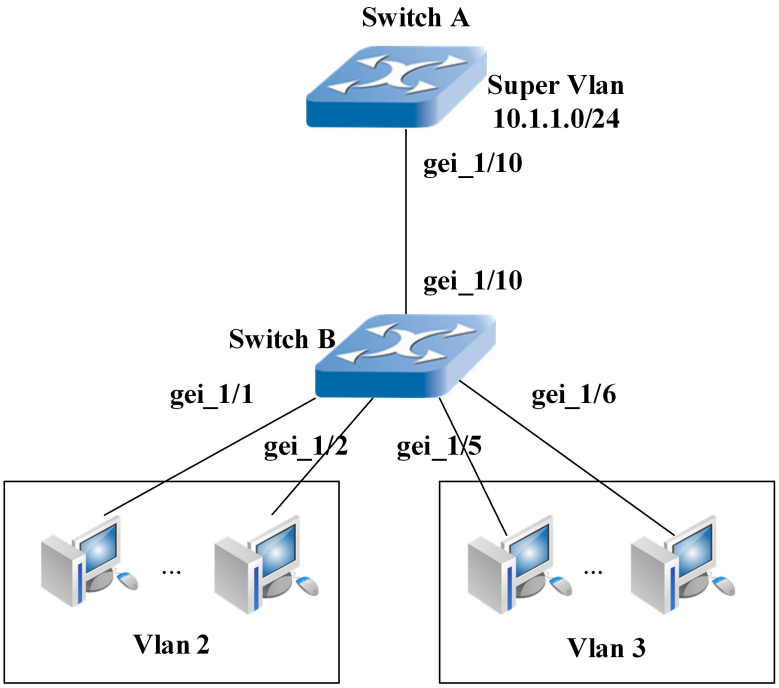
```
ZXR10(config)#vlan translate session-no 1
 ingress-port gei_1/1 ingress-vlan 100 egress-vlan 200
ZXR10(config)#vlan egr-translate session-no 1
egress-port gei_1/1 egress-vlan 200 ingress-vlan 100
ZXR10(config)#interface gei_1/1
ZXR10(config-gei_1/1)#ingress filtering disable
ZXR10(config-gei_1/1)#switchport mode hybrid
ZXR10(config-gei_1/1)#switchport hybrid vlan 100,200
ZXR10(config-gei_1/1)#exit
ZXR10(config)#interface gei_2/1
ZXR10(config-gei_2/1)#switchport mode hybrid
ZXR10(config-gei_2/1)#switchport hybrid vlan 200
ZXR10(config-gei_2/1)#exit
```

# Configuring SuperVLAN

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**interface** {**supervlan** <*supervlan-id*>|<*supervlan-name*>} | This creates a SuperVLAN. |
| 2 | ZXR10(config-vlanX)#**supervlan** <*supervlan-id*> | This adds sub-VLANs onto SuperVLAN. |
| 3 | ZXR10(config)#**supervlan inter-subvlan-routing** {**enable**|**disable**} | This enables/disables routing function among sub-VLANs. |
| 4 | ZXR10#**show supervlan** | This views SuperVLAN configuration information. |

**Example** As shown in Figure 5, configure a SuperVLAN on switch A, with its subnet as 10.1.1.0/24 and gateway as 10.1.1.1. Configure two sub-VLANs on switch B, VLAN 2 and VLAN 3, and configure them to belong to the SuperVLAN. Switch A and switch B are connected through trunk ports.

**FIGURE 5 SUPERVLAN CONFIGURATION EXAMPLE**




Configuration of Switch A:

```
/*Create a SuperVLAN, and assign subnet and gateway for it*/
ZXR10_A(config)#interface supervlan10
ZXR10_A(config-supervlan10)#ip address 10.1.1.1 255.255.255.0
ZXR10_A(config-supervlan10)#exit
/*Add the SubVLAN to the SuperVLAN*/
ZXR10_A(config)#vlan 2
ZXR10_A(config-vlan2)#supervlan 10
ZXR10_A(config-vlan2)#exit
ZXR10_A(config)#vlan 3
ZXR10_A(config-vlan3)#supervlan 10
ZXR10_A(config-vlan3)#exit
 /*Set vlan trunk port*/
ZXR10_A(config)#interface gei_1/10
ZXR10_A(config-gei_1/10)#switchport mode trunk
ZXR10_A(config-gei_1/10)#switchport trunk vlan 2-3
ZXR10_A(config-gei_1/10)#exit
```

Configuration of Switch B

```
ZXR10_B(config)#interface gei_1/1
ZXR10_B(config-gei_1/1)#switchport access vlan 2
ZXR10_B(config-gei_1/1)#exit
ZXR10_B(config)#interface gei_1/2
ZXR10_B(config-gei_1/2)#switchport access vlan 2
ZXR10_B(config-gei_1/2)#exit
ZXR10_B(config)#interface gei_1/5
ZXR10_B(config-gei_1/5)#switchport access vlan 3
ZXR10_B(config-gei_1/5)#exit
ZXR10_B(config)#interface gei_1/6
ZXR10_B(config-gei_1/6)#switch access vlan 3
ZXR10_B(config-gei_1/6)#exit
ZXR10_B(config)#interface gei_1/10
```

```
ZXR10_B(config-gei_1/10)#switch mode trunk
ZXR10_B(config-gei_1/10)#switch trunk vlan 2-3
ZXR10_B(config-gei_1/10)#exit
```

# VLAN Maintenance and Diagnosis

For convenient VLAN maintenance and diagnosis, ZXR10 5900E provides the following commands:

**show vlan** [**brief**|**access**|**trunk**|**hybrid**|**id** <*vlan-id*>[**ifindex**]|**name** <*vlan-name*>[**ifindex**]]

This command can be used to view information about all VLANs, VLAN with specified ID/name, and VLANs with their ports mode as Access/Trunk/Hybrid. The two examples are as follows.

1. This example shows how to view the configuration information of VLANs.

```
ZXR10(config)#show vlan
VLAN  Name  Status  Said  MTU  IfIndex     PvidPorts   UntagPorts
TagPorts
------------------------------------------------------------------
   1 VLAN0001  active  100001  1500   0    gei_1/5-12
  10 VLAN0010  active  100010  1500   0    gei_1/1-3
 100 VLAN0100  active  100100  1500   0
 gei_1/3-4
 130 VLAN0130  active  100130  1500   0    gei_1/4      gei_1/4
 136 VLAN0136  active  100136  1500   0
gei_1/4
 200 VLAN0200  active  100200  1500   0
gei_1/3
ZXR10(config)#
```

2. This example shows information of all the VLANs with their port mode as Trunk.

```
ZXR10(config)#show vlan trunk
VLAN  Name  Status  Said  MTU  IfIndex   PvidPorts   UntagPorts
TagPorts
------------------------------------------------------------------
   1 VLAN0001  active  100001  1500   0
  10 VLAN0010  active  100010  1500   0    gei_1/3
 100 VLAN0100  active  100100  1500   0
gei_1/3
 130 VLAN0130  active  100130  1500   0
 136 VLAN0136  active  100136  1500   0
 200 VLAN0200  active  100200  1500   0
gei_1/3
ZXR10(config)#
```

Chapter 2

# SVLAN Configuration

**Table of Contents**



## SVLAN Overview

The full name of SVLAN is selective VLAN. SVLAN is a kind of VLAN tunnel technology. It provides multi-point to multi-point VLAN transparent transportation service and simple Layer 2 VPN tunnel by means of adding a VLAN tag outside original 802.1Q tag and getting rid of outside VLAN tag when the packet is transported to edge switch.

SVLAN has the function of providing SPVLAN tag according to traffic, which is different from that ordinary QinQ adds SPVLAN tag based on ports. That is, in the same Customer port, according to difference between traffic carried CVLAN tags, provide corresponding SPVLAN tag based on user demands.

SVLAN can modify outer tag value according to inner tag, outer tag, or the combination of the former tages. Also it can control if downlink stream need to be redirected from uplink port to customer port.

With SVLAN function, User can implement map from QOS to SPVLAN of CVLAN tag.

## SVLAN Configuration

1. To configure SVLAN, use the following command.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**vlan qinq session-no** <*session_id*> **customer-port** <*interface-name*> **uplink-port** <*interface-name*> **in-vlan** <*vlan-list*>{**ovlan** {<*vlanId*>\|<*name*>}[ **priority** < *priority-id*>]\| **untag** } | This command configures SVLAN. |
| 2 | ZXR10(config)#**vlan qinq extend-session-no** <*session_id*> **customer-port** < *interface-name* > **uplink-port** < *interface-name* >{**in-vlan** <*vlan-list*>\|**outer-vlan** <*vlanId*>\| **untag** }[**outer-vlan** <*vlanId*>]{**ovlan** <*vlanId*>[ **priority** < *priority-id* >\| **map** ]\| **helpvlan** <*helpvlanId*>}[**unredirect**] | This command configures SVLAN. |

Paramters Description:

| Parameter | Description |
|---|---|
| session-no | <1 - 768> |
| customer-port | CUSTOMER port, which connects user. |
| uplink-port | UPLINK port, which connects service provider. |
| in-vlan | VID of CVLAN |
| ovlan | VID of SPVLAN |
| Priority | designate SPVLAN 802.1p priority <0~7> |
| untag | transparent transportation CVLAN TAG |
| extend-session-no | <1 - 1000> |
| untag | Not carrying CVLAN or VID of CVLAN is 0. |
| outer-vlan | The packet has two layer tages before entering into customer port, it designates outer VID. |
| map | designate 802.1p priority in SPVLAN as 802.1p priority in CVLAN. |
| helpvlan | When transporting CVLAN TAG transparently, the needed auxiliary VLAN VID. When single tag transporting transparently, packet only carries CVLAN TAG when sended from UPLINK port. When double tags transporting transparently, auxiliary VLAN VID should be the same as outer one. The packet still has two layer tags sended from UPLINK port. |
| unredirect | The downlink packet received from uplink port needn't redirect to customer port forcibly. |

2. To delete SVLAN configuration, use the following command.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**no vlan qinq session-no** <*session_id*> | This deletes SVLAN configuration. |
| 2 | ZXR10(config)#**no vlan qinq extend-session-no** {<*session_id*>\|**all**} | This deletes SVLAN configuration. |

Paramters Description:

| Parameter | Description |
|---|---|
| session-no | <1 - 768> |
| extend-session-no | <1 - 1000> |

# SVLAN Configuration Example

## Basic SVLAN Configuration

Example 1: Port 1 is a customer port, and port 2 is an uplink port. When CVLAN is 10, 12 and untag, the packet from port1 SPVLAN is 997,998 and 999 respectively.

```
ZXR10(config)#switchport gei_1/1 qinq customer
ZXR10(config)#interface gei_1/2
ZXR10(config-if)#switchport mode hybrid
ZXR10(config-if)#switchport hybrid vlan 997 tag
ZXR10(config-if)#switchport hybrid vlan 998 tag
ZXR10(config-if)#switchport hybrid vlan 999 tag
ZXR10(config-if)#exit
ZXR10(config)#vlan qinq extend-session-no 1 customer-port gei_1/1
uplink-port gei_1/2 in-vlan 10 ovlan 997
ZXR10(config)#vlan qinq extend-session-no 2 customer-port gei_1/1
uplink-port gei_1/2 in-vlan 12 ovlan 998
ZXR10(config)#vlan qinq extend-session-no 3 customer-port gei_1/1
uplink-port gei_1/2 untag ovlan 999
```

The SVLAN example of viewing configuration:

```
ZXR10(config)#show vlan qinq extend-session
Session Customer Uplink   In_Vlan Outer-vlan  Ovlan Helpvlan
Priority unredirect
-------------------------------------------------------------
1       gei_1/1  gei_1/2  10                  997
2       gei_1/1  gei_1/2  12                  998
3       gei_1/1  gei_1/2  untag               999
ZXR10(config)#
```

Example 2: Port 1 is a customer port, and port2 is an uplink port. For the packet from port1: CVLAN is 10, outer tag is 100, new SPVLAN(modified outer tag) is 200; outer tag VID is 101, new SPVLAN(modified outer tag) is 201, downlink stream needn't redirection.

```
ZXR10(config)# vlan qinq extend-session-no 1 customer-port
  gei_1/1 uplink-port  gei_1/2 in-vlan 10 outer-vlan 100 ovlan 200
ZXR10(config)# vlan qinq extend-session-no 2 customer-port  gei_1/1
uplink-port  gei_1/2 outer-vlan 101 ovlan 201  unredirect
```

The SVLAN example of viewing configuration:

```
ZXR10(config)#show vlan qinq extend-session
Session Customer Uplink   In_Vlan Outer-vlan  Ovlan Helpvlan
 Priority unredirect
-------------------------------------------------------------
1       gei_1/1  gei_1/2    10       100         200
2       gei_1/1  gei_1/2              101         201
     1
```

# Transparent Transmission SVLAN Configuration

Example 1: single tag transparent transmission, port 1 is customer port. Port 2 is uplink port, for the message from port 1: when CVLAN is 10 ,transmitted transparently, helper vlan is 100.

```
ZXR10(config)#switchport gei_1/1 qinq customer
ZXR10(config)#interface gei_1/2
ZXR10(config-if)#switchport mode hybrid
ZXR10(config-if)#switchport hybrid vlan 100 untag
ZXR10(config-if)#exit
Z ZXR10(config)#vlan qinq extend-session-no 1 customer-port gei_1/1
uplink-port gei_1/2 in-vlan 10 helpvlan 100
```

The SVLAN example of viewing configuration is as follows.

```
ZXR10(config)#show vlan qinq extend-session 1
Session Customer Uplink   In_Vlan Outer-vlan  Ovlan Helpvlan
Priority unredirect
-------------------------------------------------------------
1       gei_1/1  gei_1/2  10                            100
```

Example 2: double tags transparent transmission, port 1 is customer port, port 2 is uplink port, for the message from port 1: when CVLAN is 10 and outer tag is 100, transmitted transparently, helper vlan is 100.

```
ZXR10(config)#interface gei_1/2
ZXR10(config-if)#switchport mode hybrid
ZXR10(config-if)# switchport hybrid vlan 100 tag
ZXR10(config-if)#exit
Z ZXR10(config)#vlan qinq extend-session-no 1 customer-port gei_1/1
 uplink-port gei_1/2 in-vlan 10 outer-vlan 100 helpvlan 100
```

The SVLAN example of viewing configuration is as follows.

```
ZXR10(config)#show vlan qinq extend-session 1
Session Customer Uplink   In_Vlan Outer-vlan  Ovlan Helpvlan
Priority unredirect
-------------------------------------------------------------
1       gei_1/1  gei_1/2  10      100                 100
```

## 802.1P Priority Configuration

Example: port 1 is customer port, port 2 is uplink port. For the message from port 1: when CVLAN is 10 and SPVLAN is 100, SPVLAN priority is 5; when CVLAN is 12 and SPVLAN is 200, SPVLAN priority is CVLAN priority.

```
ZXR10(config)#switchport gei_1/1 qinq customer
ZXR10(config)#interface gei_1/2
ZXR10(config-if)#switchport mode hybrid
ZXR10(config-if)#switchport hybrid vlan 100 tag
ZXR10(config-if)#switchport hybrid vlan 200 tag
ZXR10(config-if)#exit
ZXR10(config)#vlan qinq extend-session-no 1 customer-port gei_1/1
uplink-port gei_1/2 in-vlan 10 ovlan 100 priority 5
ZXR10(config)#vlan qinq extend-session-no 2 customer-port gei_1/1
uplink-port gei_1/2 in-vlan 12 ovlan 200 map
```

The SVLAN example of viewing configuration is as follows.

```
ZXR10(config)#show vlan qinq extend-session 1
Session Customer Uplink   In_Vlan Outer-vlan  Ovlan Helpvlan
Priority unredirect
-------------------------------------------------------------
1       gei_1/1  gei_1/2  10                   100
   5
2       gei_1/1  gei_1/2  12                   200
  mapping
ZXR10(config)#
```

# SVLAN Maintenance and Diagnosis

For convenient SVLAN maintenance and diagnosis, ZXR10 5900E provides the following commands:

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**show vlan qinq session-no** <*session_id*> | This views one or all sessions configuration of SVLAN. |
| 2 | ZXR10(config)#**show vlan qinq extend-session-no** <*session_id*> | This views one or all extend-session configuration of SVLAN. |

This page is intentionally blank.

Chapter 3

# SVLAN COS Configuration

**Table of Contents**



## SVLAN COS Overview

In SVLAN QinQ mode, when receiving tagged data packet from user trunk port, uplink port reserves the original data packet tag and attaches service provider tag. This tag includes 2 bytes Ethernet type (0x8100) and 2 bytes priority and VID, in which priority field is 3bits and this field is COS, we call this as service type, service level. Or service priority. The function is to configure COS priority value.

## Configuring SVLAN COS

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**cos-session** < *session_id* >[**cos0** <*0-7*>],[**cos1** <*0-7*>],[**cos2** <*0-7*>],[**cos3** <*0-7*>],[**cos4** <*0-7*>],[**cos5** <*0-7*>],[**cos6** <*0-7*>],[**cos7** <*0-7*>] | This configures SVLAN COS.<br>session-id<1 - 16><br>This configures a cos one time or many coses. |
| 2 | ZXR10(config)#**interface** <*port-name*> | This enters interface configuration mode. |
| 3 | ZXR10(config-gei_1/x)#**cos-mode cos-map-session** <*session_id*> | This applies session corresponding cos to physical interface. |

| Step | Command | Function |
|---|---|---|
| 4 | ZXR10(config)#**no cos-session** <*session_id*> | This deletes SVLAN COS configuration. |
| 5 | ZXR10(config-gei_1/x)#**no cos-mode cos-map-session** <*session_id*> | This deletes the binding of session on physical port. |

**Note:**

Each physical port can only apply one session. The new configuration will replaces the old one. For example, configure the following two commands on gei_1/1 interface configuration mode:

1. cos-mode cos-map-session 1
2. cos-mode cos-map-session 2

Here only 2 takes effect.

# SVLAN COS Configuration Example

For example, assume that on port gei_1/1, configure cos0 priority map is 7, cos1 priority map is 6, cos2 priority map is 5, cos3 priority map is 4, cos4 priority map is 3, cos5 priority map is 2, cos6 priority map is 1, cos7 priority map is 7; on port gei_1/2, configure cos1 priority map is 5.

```
/*configure cos session*/
ZXR10(config)# cos-session 1 cos0 7 cos1 6 cos2 5 cos3 4
 cos4 3 cos5 2 cos6 1 cos7 7
ZXR10(config)#cos-session 2 cos1 5
/*apply cos session on physical port*/
ZXR10(config)#interface gei_1/1
ZXR10(config-gei_1/1)#cos-mode cos-map-session 1
ZXR10(config-gei_1/1)#exit
ZXR10(config)#interface gei_1/2
ZXR10(config-gei_1/2)#cos-mode cos-map-session 2
ZXR10(config-gei_1/2)#exit
```

# SVLAN COS Maintenance and Diagnosis

To perform SVLAN maintenance and diagnosis, ZXR10 5900E provides the following commands to view all SVLAN session configuration information.

1. This views SVLAN COS one or all session configuration.
   **show qos cos-session**<*session_id*>
2. This views if a physical port applies ACL.
   **show running-config interface** <*port-name*>

This page is intentionally blank.

Chapter 4

# MAC Address Table Configuration

**Table of Contents**



# Introduction to MAC Address

MACMedia Access Control) address is the hardware identification of a network device. The switch forwards packets based on MAC address. MAC address is unique, ensuring accurate packet forwarding.

Each switch maintains a MAC address table called forwarding database (FDB). FDB records one-to-one mapping relationship between MAC addresses and switch ports. When receiving a data frame, the switch decides whether to drop it or forward it to the proper port based on this table. The FDB is the basis and prerequisite for fast forwarding.

## Composition and Meaning of MAC Address Table

A MAC address and a VLAN ID pair uniquely identify a MAC address table entry. ZXR10 5900E MAC address table entry includes the following items:

1. MAC address: such as 00D0.D056.95CA.
2. Port No.: MAC address corresponding port such as gei_1/1, smartgroup1.
3. VLAN ID: MAC address corresponding VLAN ID such as 10.
4. Other marks: Indicate MAC address state and operation.

ZXR10 5900E MAC address table entry has the following marks:

- stc: Whether the MAC address is a static one.
- per: Whether the MAC address is permanent.
- toS: Whether the MAC address is solid.

- srF: Whether to drop frames from the source MAC address.
- dsF: Whether to drop frames from the destination MAC address.
- Time: the time of MAC address on the switch. Desiganated by day:hour:minute:second.

During L2 forwarding, the switch checks its MAC address table for the destination MAC address of the received frame, and then forward data to the corresponding port.

# MAC Address Classification

ZXR10 5900E MAC address is divided into the following types:

1. Dynamic MAC address

   Dynamic MAC addresses are learned by the switch from data frames it receives, and are deleted when the aging time is due. When the device connects to another port on the switch, the corresponding mapping relationship between the MAC addtess and port number also changes in the MAC address table. Dynamic MAC addresses are lost when the switch is powered off and must be learned again when the switch is rebooted.

2. Static MAC address

   Static MAC addresses are configured manually and will never age. The mapping relationship between MAC address and port number in the MAC address table remains unchanged despite of changes of the connecting port between switch and device. Static MAC addresses are also lost when the switch is powered off and must be configured again when the switch is rebooted.

3. Permanent MAC address

   Permanent MAC addresses are also configured manually. The mapping relationship between MAC address and port number in the MAC address table remains unchanged despite of changes of the connecting port between switch and device. Permanent MAC addresses will not disappear when the switch is powered off.

# MAC Address Table Establishment and Deletion

The MAC address table of a switch is null initially. It is created for fast forwarding. As the size of the MAC address table is limited and network device changes are frequent, invalid MAC address table entries should be deleted from the switch in time.

1. Dynamic Learning

   Dynamic MAC addresses in the MAC address table are learned by the switch. The procedure of switch learning MAC address is as follows:

Switch analyzes the source MAC address and VLANID (for example, MAC1+VID1) once it receives a data frame on a port.

If judging the MAC address to be valid and learnable, the switch checks MAC1+VID1 in its MAC address table. If the entry is not found, the address is added to the MAC address table, otherwise the entry is updated.

**Note:**

i. MAC address learning is to learn the source MAC address of received data frames, not the destination MAC address.

ii. MAC address learning applies to unicast addresses only, not to broadcast or multicast addresses.

2. MAC Address Aging

   The size of the MAC address table is limited, so a MAC address aging mechanism is provided for effective resource utility of the MAC address table.

   A switch considers a device that has got offline or is not in communication when it fails to receive any data frame from that device for a certain time period (set aging time), that is, it does not receive any data frame from its source MAC address as that of device's MAC address.

   Then the switch deletes that MAC address of the device from its MAC address table and updates the MAC address table.

   MAC address aging applies to dynamic MAC addresses only.

3. Manual Addition and Deletion

   An entry can be added to the MAC address table of a switch with a configuration command when the network is relatively stable and device is connected to a fixed switch port. Configuration can take place for dynamic, static or permanent MAC address. Configuring static or permanent MAC addresses can prevent MAC spoofing attacks.

   MAC address can be deleted with the MAC address deletion command. Deleting command of ZXR10 5900E can forcibly delete a dynamically learned MAC address.

# Configuring MAC Address Table

MAC address table of switch can run normally with the default configuration. But some appropriate configuration on MAC address table can improve the network stability.

The configuration of MAC address table includes the following contents.

# Setting MAC Address Aging Time

MAC address aging time influences the switch's performance.

A shorter aging time may make the switch delete useful MAC address table entries. As a result, it broadcasts many packets it loses track to their destination MAC addresses, which consumes the bandwidth.

A longer aging time may lead to too many useless entries in the MAC address table, which use up the MAC address table resources. New MAC addresses cannot be added to the MAC address table, so forwarding performance also reduces.

To set MAC Address Aging Time, use the following command.

| Command | Function |
|---|---|
| `ZXR10(config)#` **mac aging-time** <*time*> | This sets MAC Address Aging Time. |

The address aging time on ZXR10 5900E by default is up to 360s, configurable range is from 60s to 630s.

# Burning MAC Addresses

Learned MAC addresses can be burn in the switch after a period of stable running if there is not any change in the connecting ports between switch and device (or in the mapping relationship between device MAC address and port number. in the MAC address table).

MAC address burning is to convert all dynamic MAC addresses in the MAC address table to static MAC addresses that will not age. After address burning, data frames from these MAC addresses are not learned when they are received on other ports.

To burn MAC Address, use the following command.

| Command | Function |
|---|---|
| `ZXR10(config)#`**mac to-static** {**enable**\|**disable**\|**interface** <*port-name*>{**enable**\|**disable**}} | Burnt MAC addresses are not stored permanently but are lost after power-off. |

# Configuring MAC Address Permanent

This is a security policy for MAC virus flood. When network runs for a while, if it is stable user and the location of devices that each port of switch connects are fixed, that is, the interface that each device MAC address in switch MAC address table corresponds is fixed, then make the mac address that learns from port and will learn from port permanent to prevent MAC deceiving format network

attack. Even if after device is rebooted, these MAC addresses still can be used. Of course, the premise is that automatic writing function is opened or write operation is implemented.

MAC address permanent means making dynamic MAC address of corresponding port in MAC address table permanent. After this configuration, the MAC address learns from the corresponding port can convert to permanent automatically. These MAC addresses don't join aging after converting and can be saved into disk. Meanwhile if data frame with this source address occurs on other ports switch won't learn again. The MAC permanent entry number is limited. Therefore when the newly learned MAC entry on port can't continue converting, alarm will occur and subsequent MAC will still be handled as dynamic entry. When this function is disabled, all MAC entries converted by this way on this port will be deleted.

To configure MAC address permanent, perform the following commands.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**mac to-permanent interface** <*port-name*> **enable** | After MAC address is made permanent, these MAC address will be saved permanently and won't lost when switch is rebooted. |
| 2 | ZXR10(config)#**mac to-permanent interface** <*port-name*> **disable** | After disabling MAC address permanent, these permanent MAC addresses will be deletes. |
| 3 | ZXR10(config)#**mac auto-write** { **disable** \| **enable** **interval** <*time*>} | This configures writing permanent MAC address and save period. |

# Binding MAC Address to a Port

A MAC address can be bound to a port of ZXR10 5900E by adding a static/permanent MAC address to the FD. The mapping relationship between MAC address and port is fixed for static or permanent MAC address. The relationship will not be released until it is manually deleted.

For binding a MAC address, perform the following steps.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**mac add** {**dynamic**\|**static**\|**permanent**}<*mac-address*> **interface** <*port-name*>[**vlan** <*vlan-id*>] | When adding MAC address, if the VLAN ID is not designated , add according to port PVID. |
| 2 | ZXR10(config)#**mac delete** {<*mac-address*>\| **interface** <*port-name*>\| **vlan** <*vlan-id*>} | If a VLAN ID is not specified while deleting a MAC address, all entries matching **<***mac-address*>will be deleted. |

# Enabling MAC Address Learning

MAC address learning is enabled on all switch ports by default. Ports can learn MAC addresses freely. The device can bind MAC addresses to a port (configure these addresses manually) when these devices will always be connected to that port. Disable MAC address learning so that the port will no longer learn MAC addresses.

To enable MAC Address learning on a port, use the following command.

| Command | Function |
|---|---|
| ZXR10(config)#**mac learning** {**enable**\|**disable**\|**interface** <*port-name*>{**enable**\|**disable**}} | This enables MAC address learning on a port. |

# Limiting MAC Address Count

When there are too may users online and MAC address table size is near its upper limit, the count of MAC addresses learned by low priority user ports can be limited.

Limiting the MAC address count of a port is a good countermeasure against MAC address flood attacks.

For limiting MAC Address count on a port, perform the following steps.

| Command | Function |
|---|---|
| ZXR10(config)# **mac limit-num** [**interface** <*port-name*>]<*max-number*> | This limits MAC address count on a port. |

MAC address count is unlimited by default. To cancel MAC address count limit, set the limit to 0.

# Setting MAC Address Learning Protection

ZXR10 5900E supports MAC address learning protection function for ports. When a port is detected to be learning MAC addresses abnormally, its address learning will be suspended for some time. A port in protection state cannot learn new addresses. It recovers learning when the protect time is due.

For setting MAC Address learning protection on a port, perform the following steps.

1. Set port MAC address learning count limit.
2. Open port MAC address learning protection enabled switch.
3. Set the protected port protection time.

The detailed configuration is as follows:

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**mac protect** [**interface** <*port-name*>]{**disable**\|**enable**} | This sets port MAC address learning protection. |
| 2 | ZXR10(config)#**mac protect time** <*time*> | This sets port MAC address learning protection time. |

MAC address learning protection is disabled on all switch ports by default. To enable this function, it is recommended to set a smaller MAC address count limit.

# Setting Port Unkown Source MAC Address Filtering

By default, the port unkown source MAC address filtering function is closed. A port does not filter unkown source MAC address. If one port enables unkown source MAC address filtering function, the corresponding port will discard the unkown source MAC address packets from this port.

To set port unkown source MAC address filtering on a port, use the following command.

| Command | Function |
|---|---|
| ZXR10(config)#**mac unknowsource-filter interface** <*port-name*>{**disable**\|**enable**} | This sets port unkown source MAC address filtering on a port. |

# Setting MAC Address Filtering

To prevent unauthorized access, ZXR10 5900E support filtering data frames according to MAC addresses. There are three filtering modes:

- Source MAC address matching only, that is, a data frame is dropped when its source MAC address matches the setting.
- Destination MAC address matching only, that is, a data frame is dropped when its destination MAC address matches the setting.
- Source/Destination MAC address matching, that is, a data frame is dropped when its source or destination MAC address matches the setting.

To set MAC address filtering, use the following command.

| Command | Function |
|---|---|
| ZXR10(config)#**mac filter** {**source**\|**both**\|**destination**}<*mac-address*> **vlan** <*vlan-id*> | This sets MAC address filtering. |

There is no need to specify a port name while setting MAC address filtering. This function filters data frames from all ports of the switch. To cancel MAC address filtering, simply delete the MAC address.

## Viewing MAC Address Table

View MAC address entry by using the following command. The viewed MAC address are dynamic learning and added manually.

| Command | Function |
|---|---|
| ZXR10(config)#**show mac** [**dynamic**\|**static**\|**permanent**\| **src-filter**\|**dst-filter**\|<*mac-address*>\| **interface** <*port-name*>\|**vlan** <*vlan-id*>] | This views MAC address table. |

**Example** Example: display MAC address entry.

```
ZXR10(config)#show mac
 Total MAC address : 2

 Flags: vid --VLAN id,stc --static, per --permanent, toS
--to-static,wtd--is wri
ted,
        srF --source filter,dsF --destination filter, time
--day:hour:min:sec
        Frm --mac from where:0,drv; 1,config; 2,VPN; 3,802.1X;
 4,micro;5,dhcp

  MAC_Address    vid  vpn        port  per stc toS wtd srF dsF
Frm     Time
-------------------------------------------------------------------
0000.2222.2222     2    0     fei_2/1/2 1   0   0   0   0   0
 1     0:00:14:19
0000.1111.1111     2    0     fei_2/1/2 1   0   0   0   0   0
 1     0:00:14:28
```

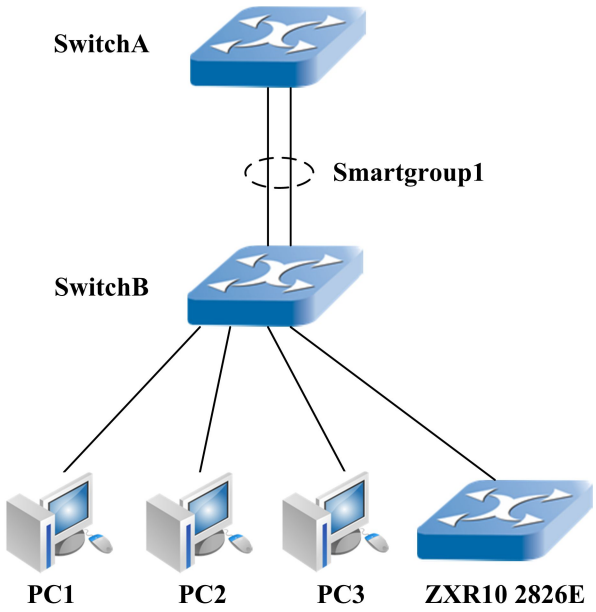
# MAC Address Table Configuration Example

As shown in Figure 6, switches A and B are connected over trunk link smartgroup1, three PCs and a ZXR10 2826E are connected to switch B. The details is shown as .

| Equipment | MAC Address | Switch Port | VLAN |
|---|---|---|---|
| PC1 | 0X00D0.8765.95CA | gei_1/1 | 1 |
| PC2 | 0X00D0.8765.95CB | gei_1/3 | 2 |
| PC3 | 0X00D0.8765.95CC | gei_1/5 | 3 |
| ZXR10 2826E | ---------- | gei_1/7 | 4 |

PC1, PC2 and PC3 work as servers, and their MAC addresses are bound to the ports of switch B. There are a number of individual users connecting to the ZXR10 2826E, so MAC address learning protection (with the MAC address count as 1000 and protect time as 120s) is need to be enabled on the proper switch B port. In addition, it is need to set the MAC address aging time to 180s on switch B.

**FIGURE 6 MAC ADDRESS TABLE CONFIGURATION EXAMPLE**




Configuration of switch B

```
/*Configure MAC address binding on the port*/
ZXR10_B(config)#mac add permanent  00D0.8765.95CA
 interface  gei_1/1  vlan 1
ZXR10_B(config)#mac add permanent  00D0.8765.95CB
 interface  gei_1/3  vlan 2
ZXR10_B(config)#mac add permanent  00D0.8765.95CC
 interface  gei_1/5  vlan 3
/*Configure MAC address learning protection on the port*/
ZXR10_B(config)#mac limit-num interface  gei_1/7 1000
ZXR10_B(config)#mac protect interface  gei_1/7 enable
ZXR10_B(config)#mac protect time 120

/*Configure MAC address aging time*/
ZXR10_B(config)#mac aging-time 180
```

This page is intentionally blank.

Chapter 5

# STP Configuration

**Table of Contents**



# STP Overview

Spanning Tree Protocol (STP) is applied to a loop network. It blocks some redundant paths with certain algorithms so that the loop network is pruned into a tree network without any loop, thus avoiding the infinite loop of packets in the loop network.

STP is implemented by exchanging Bridge Protocol Data Unit (BPDU) messages among involved switches in an extended LAN. The following operations can be performed by exchanging BPDU messages:

1. Selecting a root switch from the stable spanning tree topology.
2. Selecting a designated switch from the network.
3. Setting redundant switch ports to Discarding, to avoid loops in the topology.

STP module of ZXR10 5900E supports three modes: SSTP, RSTP and MSTP, which respectively observes IEEE802.1d, IEEE802.1w and IEEE802.1s standards.

## SSTP Mode

Single Spanning Tree Protocol (SSTP) fully observes IEEE802.1d standards in terms of function. The bridge running SSTP can fully inter work with those running RSTP and MSTP.

# RSTP Mode

Rapid Spanning Tree Protocol (RSTP) provides a faster aggregation speed than STP (that is, SSTP mode). When the network topology changes, the state of the redundant switch port can make a fast shift (Discard > Forword) in the case of point-to-point connection.

# MSTP Mode

Two concepts are added to Multiple Spanning Tree Protocol (MSTP): instance and VLAN mapping. SSTP/RSTP mode can be regarded as a special case of the MSTP mode. There only exists one instance, that is, instance 0. The MSTP mode also provides fast aggregation and load balance under the VLAN environment.

In SSTP and RSTP modes, there is no concept of VLAN. There exists one port state, that is, forwarding state. The state of a port in different VLANs is the same. In MSTP mode there can exist multiple spanning-tree instances. Forwarding states of a port under different VLANs can be different. Many independent sub-tree instances can be formed inside the MST area to implement load balance.

The following are the basic concepts of MSTP:

1. MST Config ID

   It is a forwarding scheme for frames with different VIDs. That is in an MST area all the bridges are forwarded to specific spanning trees (CIST or an MST instance) according to the VID in the frame.

   MST Config ID is composed of the following parts:

   - Configuration name: a character string of 32 bytes.
   - Version: non-negative integer of two bytes.
   - Configuration summary: signature based on the MST Config Table and after MD5 processing, with a length of 16 bytes.

   MST Config Table is composed of 4,096 continuous dual-types. The First and last dual-bytes are 0. Other dual-byte represents a binary number. Second dual-byte stands for the MSTID to which VID 1 corresponds. Third dual-byte stands for the MSTID to which VID 2 corresponds and the second last dual-byte represents the MSTID to which VID 4094 corresponds. Configuration summary is obtained by calculating the MST Config Table and a fixed KEY value through the HMAC-MD5 algorithm. By resolution, it can know a certain VID belongs to which MST instance or CIST.

2. MST Area

   Each MST area is composed of one or several connected bridges with the same MST Config ID. These bridges use the same instances. This area also includes the LAN designating the bridge in the CIST instance.

**Note:**

Bridges in an MST area mush have same MST config ID. Two bridges with same MST config ID are probably in different MST area. For example, if two bridges with the same MST config ID are connected through the LAN of another MST area, they should belong to different MST area.

MST area may have different spanning tree structures: Internal Spanning Tree (IST), MST1, MST2 and MSTn. Each MSTi can be regarded as an MSTI. The bridge forwards the frame with the designated VID based on the path to which the VID corresponds. The mapping between VID and MSTI is shown by the MST Config ID. The spanning tree structure of MSTI is decided by priority parameters configured by the system.

3. MST Instance

   MST bridge must support two kinds of instances that is IST instance and multiple MST instances. By default, IST runs in an area. All the VLANs are configured to the IST by default. IST connects all the switches in the area. IST is responsible for communication with other MST areas and SST areas outside the area. MST instance does not send the BPDU packet independently. Spanning tree information is included in the M-record, and transmitted as part of the IST BPDU inside the area.

4. CIST

   internal IST and external CST of each MST area jointly constitute Common and Internal Spanning Tree (CIST). It means that CIST is same as IST inside the MST area, and CST outside the MST area.

5. IST Region Root

   Each MST area has an IST Region Root switch which has minimum overhead from the CST Root path. When CIST Root is in a certain MST area, the CIST Root is the IST Region Root of this MST area. After the IST Region Root is selected, other ports towards the CIST Root in the area will be blocked.

6. MST BPDU

   MSTI inside the MST area does not communicate with the outside, while the IST exchanges BPDU packets with the outside. Inside the area MSTI does not send BPDU packets independently. MST BPDU packets are sent by the IST includes MSTI information. MSTI uses a flag to show there is need to send the MST BPDU packet. IST is responsible for sending packets. All MSTIs that need to send BPDU packets place their information in the M-record structure which is sent as part of the IST BPDU.

# BPDU Protection

**BPDU Overview**

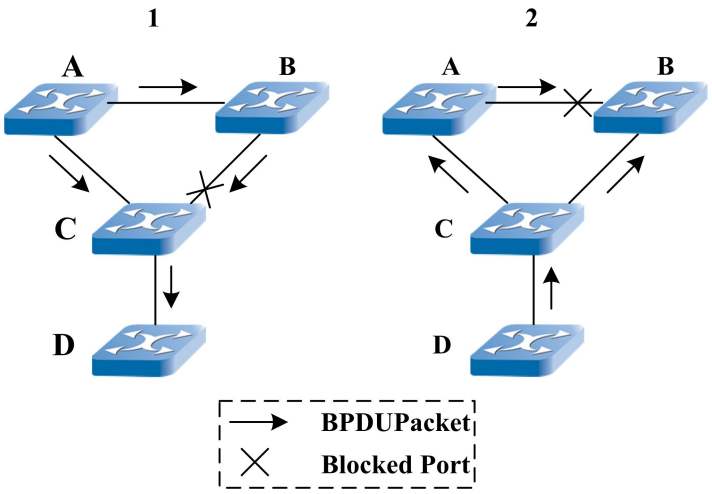
Switches calculating spanning tree according to the content of BPDU packet. Network topology once changes spanning tree will be calculated again. If the network is large, the calculation will be very frequent. This influence switches to transmit packet. At the same time, the change of Root Bridge also brings some problem. BPDU protection overcomes these problems.

**BPDU Protection of Edge Port**

If a port is set as an edge port, BPDU protection function will shut down this port when the edge port receives a BPDU packet. It outputs alarm information to monitor terminal.

BPDU protection of edge port maintains the stable of network topology. Device which connects to edge port can not influence the spanning-tree. It can implement by setting port DOWN and outputing alarm information at terminal when receiving BPDU packet at the edge port. The example is shown as Figure 7.

**FIGURE 7 BPDU PROTECTION OF EDGE PORT**




See the first status in the figure, priority of switch A is 8192. Switch A is a root switch. Priority of switch B is 16384. Link between Switch A and Switch B is 1000M and other link is 100M. Switch A and Switch B are both core switches. Switch C is a edge switch. Port of switch C connects to switch D edge port. spanning tree is enabled in switches except switch D. After switches calculate spanning tree the port of switch C which connects to switch B is blocked. Switch D doesn't take part in calculating spanning-tree. Direction of arrow represents the direction of BPDU.

See the second status in the figure, spanning tree is enabled in switch D. Its priority is smaller than switch A which is root switch. Switch D will become a root switch. After switches calculate spanning tree, port of switch B which connects to switch A will be blocked. Its priority is smaller than switch A which is root switch. Switch D will become a root switch. After switches calculate spanning tree, port of switch B which connects to switch A will be blocked.

If configuration of BPDU protection in edge port of switch C takes place. Switch D sends BPDU packet, switch C will receive this

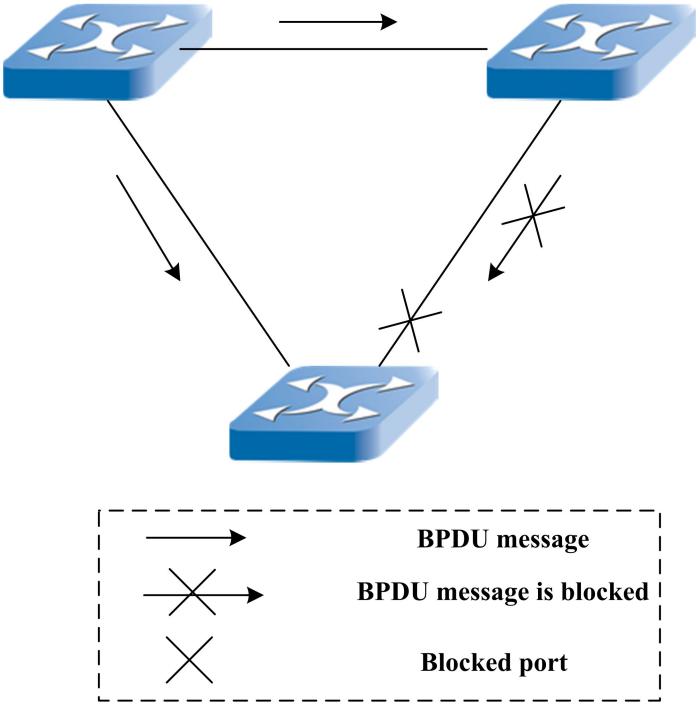
packet and shut down the port which connects to switch D. This solves the problem of network performance.

**Port Loopback Protection Function**

Loopback protection provides addtional layer 2 protection function. One reason for STP loop takes places in a network with redundant link is that one port which in blocking state becomes a designated port and enters into FORWARDING state. A blocking port when doesn't receive a BPDU packet. STP thinks that there isn't a loop. Port will transmit from BLOCKING state to FORWARDING state, this will create a loop.

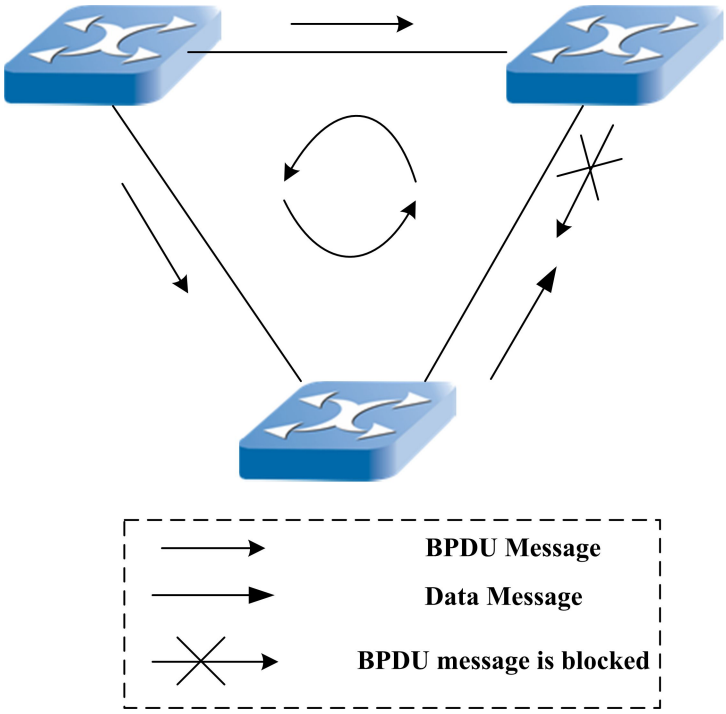
When port loopback protection is configured and blocking port don't receive BPDU packet. Port will transmit into LOOP_INCONSISTENT state. This state is blocking state and it doesn't transmit any data.

**FIGURE 8 STP BEFORE MAX_AGE TIMER EXPIRED**



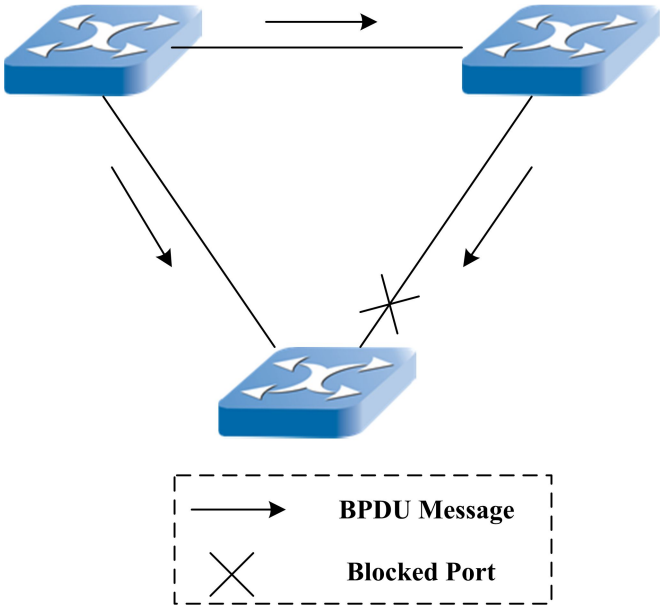

In Figure 8, switch A is a root switch. There is link failure between switch B and switch C which doesn't receive any BPDU packet from switch B. Before MAX_AGE timer expired, port in switch C is still in blocking state.

**FIGURE 9 NETWORK LOOP DIAGRAM**




In Figure 9, if port loopback protection is not configured. After MAX_AGE timer expires, blocking port in switch C will be transmitted into LISTENING state and then FORWARD_DELAY time is transmitted into LEARNING state. This cycle is repeated again then this will lead to loop.

**FIGURE 10 PORT LOOPBACK PROTECTION**



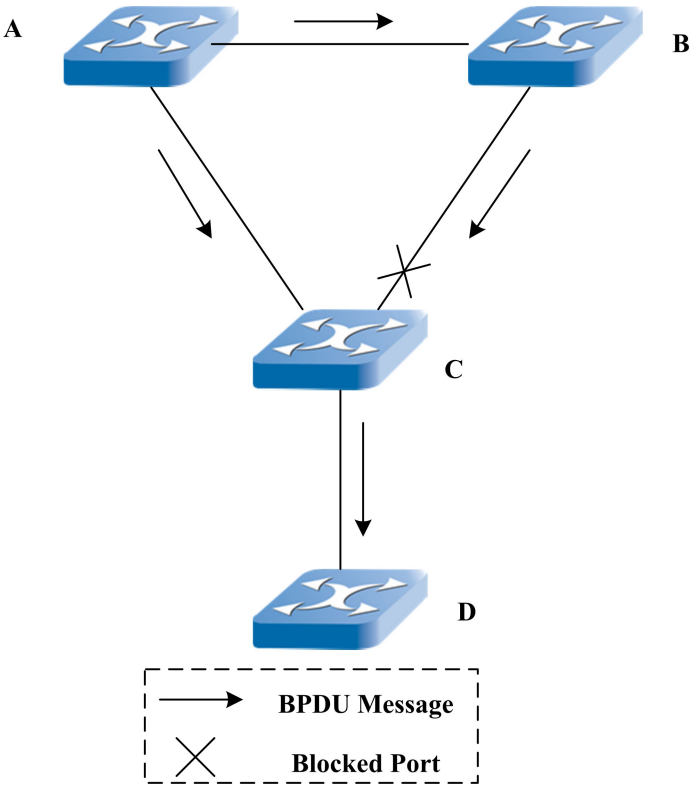

In Figure 10, if port loopback protection is configured. Blocking port in switch C will be transmitted into LOOP_INCONSISTENT

state after MAX_AGE time. Port in LOOP_INCONSISTENT state won't transmit data. This will avoid looping.

**Port Root Protection Function**

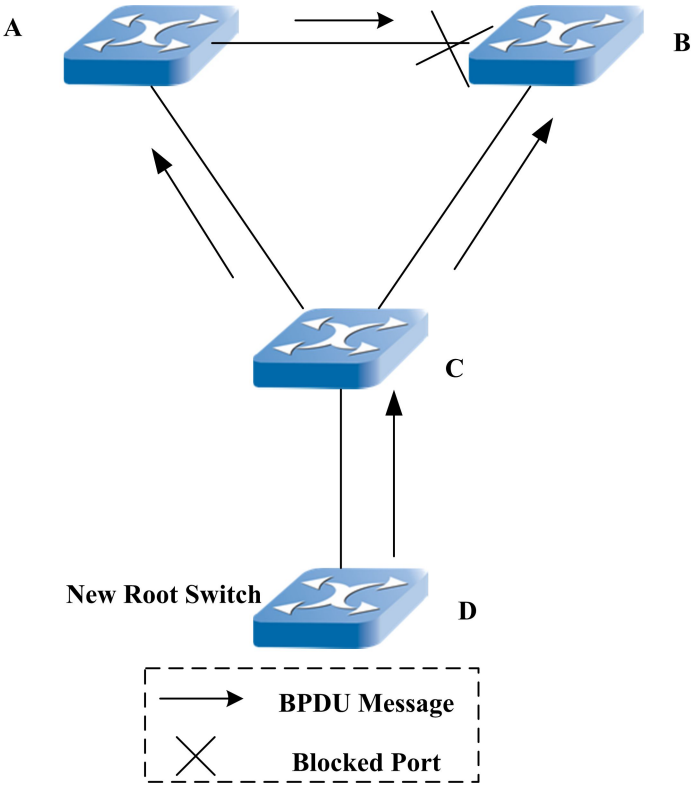
Port root protection function provides a way to protect root switch. In switch environment, all switches which enable spanning-tree will take part in the election of root switch. Switch which has the lowest priority will be root switch. After election, if one new switch which has smaller priority than the root switch connects to network, this new switch will replace the original root switch to become new root switch. This results in calculating spanning tree again and interrupting network for a while. The new spanning tree may contain sub-optimum path and lower network performance. Here, let's see how port root protection function solves this problem.

**FIGURE 11 ROOT BRIDGE**




In Figure 11, switch A and switch B are both core switches and switch A is a root switch. Switch C is a edge switch. Port of switch C which connects to switch B is blocked. The flow of BPDU is as direction of arrow.

**FIGURE 12 NEW ROOT BRIDGE**




In Figure 12, switch D takes part in calculating spanning tree. If priority of switch D is lower than that of switch A. Switch D is elected to be a new root switch. After calculating spanning tree, port of switch B which connects to switch A is blocked.

Port root protection command is configured in interface mode. It is only permitted in designated port and is not permitted in root port. A port which enables root protection receives BPDU packets from a neighbor switch and knows the neighbor switch has smaller priority. Port will enter ROOT_INCONSISTENT state.

In Figure 12, port of switch C that is connected to switch D should be configured as protection. Once this port receives a BPDU packet which contains smaller priority than root switch, this port will enter ROOT_INCONSISTENT state and stops transmitting any data packet. Once switch D stops sending the BPDU packet which contains smaller priority. Port of switch C will resume transmitting data packets automatically.

# Configuring STP

## Enabling/Disabling STP

| Command | Function |
|---|---|
| ZXR10(config)#**spanning-tree** {**enable**\|**disable**} | This enables or disables STP. |

**Note:**

Disable STP on ZXR10 5900E, all ports that physical status is up are set Forwarding status. In default, STP is disabled.

## Configuring STP Mode

| Command | Function |
|---|---|
| ZXR10(config)#**spanning-tree mode** {**sstp**\|**rstp**\|**mstp**} | This configures STP mode. |

By default, STP mode of ZXR10 5900E is MSTP. Whichever mode is selected, it can be fully compatible and interoperable with other two modes.

## Configuring STP Parameters

The parameters of STP protocol is as follow:

- max-age

  In the CST structure, latest BPDU packets are transferred from the Root switch to the leave switch along the CST structure. The message-age value of the BPDU packet sent by the Root switch is 0 and increases by 1 after each middle switch. The max-age value does not change at all. When the message-age value of a BPDU packet is greater than the max-age value, this BPDU packet is invalid.

- hello-time

  It is to control the interval for sending BPDU packet.

- forward-delay

  In case of non-state fast transition, it decides the delay (2 x forward-delay) of the port from Blocking to Forwarding.

- max-hops

It is decided by the area root node of an instance in the MST area and decreases by 1 after each switch. When it decreases to 0 the BPDU packet is invalid. Message-age and max-age values of the BPDU packet in the MST area do not change during area transmission.

To configure STP protocol parameters , use the following commands.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**spanning-tree hello-time**<*time*> | This set Hello time of STP. |
| 2 | ZXR10(config)#**spanning-tree forward-delay**<*time*> | This sets the forward delay time of STP. |
| 3 | ZXR10(config)#**spanning-tree max-age**<*time*> | This sets the maximum time of BPDU packet. |
| 4 | ZXR10(config)#**spanning-tree mst max-hops**<*hop*> | This sets the maximum hops of BPDU packet. |

**Note:**

In CST structure, hello-time of every switch is determined by the root of switch. Max-hops takes effect only when this node serves as the area root node of an instance in the MST area.

# Creating Instances

In MSTP mode, users can turn connected switches into an MST area by creating or deleting instances. Implementing fast aggregation and load balance of the whole network.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**spanning-tree mst configuration** | This enters MST configuration mode. |
| 2 | ZXR10(config-mstp)#**instance** <*instance*> **vlans** <*vlan-id*> | This creates an instance. |

**Note:**

ZXR10 5900E has on instance 0 only in SSTP/RSTP mode. In MSTP mode, the instance 0 exists by default and cannot be deleted at all.

# Configuring MSTP Name and Version

| Step | Command | Function |
|---|---|---|
| 1 | `ZXR10(config-mstp)#`**name** <*string*> | This configures MST configuration name. |
| 2 | `ZXR10(config-mstp)#`**revision**<*version*> | This configures version number. |

**Note:**

Four conditions decide whether switches belong to same MST area: same MST configuration, same MST configuration version No., same INS-VLAN mapping table, and switch interconnection or not.

# Configuring Switch and Port Priority

In the entire spanning tree structure, bridge priority of an instance can decide the position of this switch in the whole CST structure (whether this switch can be selected as the root of the entire spanning tree). It can also be in a certain instance spanning tree structure in the MST area (whether this switch can be selected as the area root of the instance).

Bridge can be designated as the spanning tree root by setting a lower priority for the bridge.

A specific port is included in the spanning tree by setting its priority. Smaller the set value of a port is the higher port priority is. It is most likely for such a port to be included in the spanning tree. If all ports on the bridge have the same priority value, the port priority depends on the port index number.

| Function | Example |
|---|---|
| `ZXR10(config)#`**spanning-tree mst instance** <*instance*> **priority** <*priority*> | This configures switch priority. |

**Note:**

Bridge and port priority of ZXR10 5900E must be configured after the instance has been created.

## Excluding a Port from Spanning Tree Calculation

In some cases, it is necessary to exclude a port from spanning tree calculation such as uplink port of switch or the port connecting to PC.

| Command | Function |
|---|---|
| ZXR10(config-gei_1/x)#**spanning-tree** {**enable**\|**disable**} | This excludes a port from spanning tree calculation |

# BPDU Protection Configuration

## Configuring BPDU Protection on Edge Port

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config-gei_1/x)#**spanning-tree edged-port enable** | This changes a port into edge port. |
| 2 | ZXR10(config-gei_1/x)#**spanning-tree edged-port disable** | This closes edge port. |
| 3 | ZXR10(config-gei_1/x)#**spanning-tree bpduguard action discard** | This configures port is in discard state after enable BPDU protection function. |
| 4 | ZXR10(config-gei_1/x)#**no spanning-tree bpduguard action** | This deletes discard state. |
| 5 | ZXR10(config-gei_1/x)#**spanning-tree bpduguard action shutdown** | This configures port is in shutdown state after enable BPDU protection function. |
| 6 | ZXR10(config-gei_1/x)#**no spanning-tree bpduguard action** | This deletes shutdown state. |

## Configuring Port Loopback Function

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config-gei_1/x)#**spanning-tree guard loop instance 1** | This enables port loopback protection function in instance 1. |
| 2 | ZXR10(config-gei_1/x)#**no spanning-tree guard loop instance 1** | This deletes the port that has a loopback protection function in instance 1. |

## Configuring Port Root Protection Function

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config-gei_1/x)#**spanning-tree guard root instance 1** | This enables port root protection function in instance 1. |
| 2 | ZXR10(config-gei_1/x)#**no spanning-tree guard root instance 1** | This deletes port root protection function in instance 1. |

# STP Configuration Examples

MSTP supports multiple MST areas. However, it is suggested to configure one MST area running on the backbone network and serving as the root of the entire CST, for better fast aggregation and load balance of the entire network.

1. As shown in Figure 13, MSTP is run on the backbone network. MST area serves as the CST root. That is, the CIST root bridge is inside the MST area. Three switches A, B and C are configured in the same area, with an initial priority of 32768. The CIST root and IST root are decided according to their MAC addresses. MAC addresses of Switches A, B and C are

   Switch A000d.0df0.0101

   Switch B000d.0df0.0102

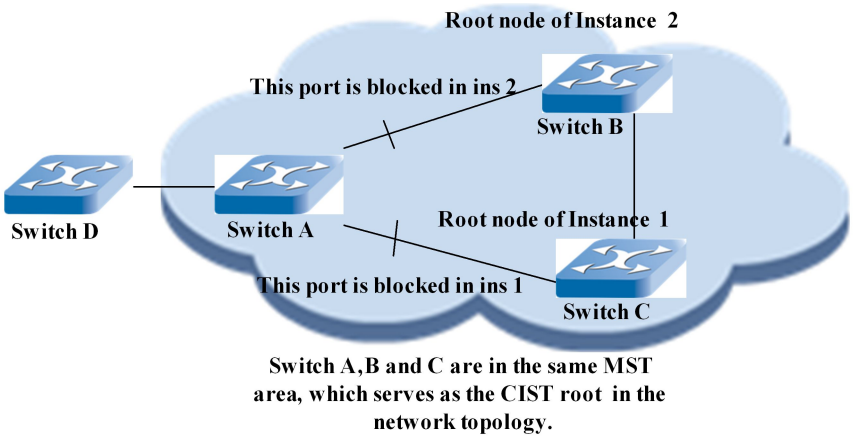
   Switch C000d.0df0.0103

   Create two MST instances and map the VLAN in the area to them.

   Run CST on Switch D whose MAC address is 000d.0df0.0104 and priority is 32768.

Implement fast aggregation of the entire network and load balance of two links on Switch A in the area.

**FIGURE 13 MSTP CONFIGURATION**




### Switch A Configuration

```
/*Configure the MST area*/
ZXR10_A(config)#spanning-tree enable
ZXR10_A(config)#spanning-tree mode mstp
ZXR10_A(config)#spanning-tree mst configuration
ZXR10_A(config-mstp)#name zte
ZXR10_A(config-mstp)#revision 2

/*Map VLANs 1 to 10 to Instance 1 and VLANs 11 to 20 to Instance 2*/
XR10_A(config-mstp)#instance 1 vlan 1-10
ZXR10_A(config-mstp)#instance 2 vlan 11-20
```

### Switch B Configuration

```
/* Configure the MST area*/
ZXR10_B(config)#spanning-tree mode mstp
ZXR10_B(config)#spanning-tree mst configuration
ZXR10_B(config-mstp)#name zte
ZXR10_B(config-mstp)#revision 2

/*Map VLANs 1 to 10 to Instance 1 and VLANs 11 to 20 to Instance 2*/
ZXR10_B(config-mstp)#instance 1 vlan 1-10
ZXR10_B(config-mstp)#instance 2 vlan 11-20

/*Change the priority of Switch B in Instance 2 so that
 it becomes the Root of Instance 2*/
ZXR10_B(config-mstp)#spanning-tree mst instance 2 priority 4096
```

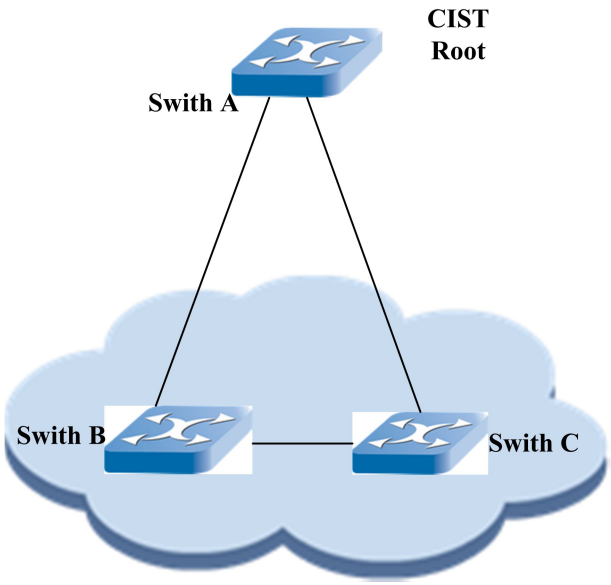
### Switch C Configuration

```
/*Configure the MST area*/
ZXR10_C(config)#spanning-tree mode mstp
ZXR10_C(config)#spanning-tree mst configuration
ZXR10_C(config-mstp)#name zte
ZXR10_C(config-mstp)#revision 2

/*Map VLANs 1 to 10 to Instance 1 and VLANs 11 to 20 to Instance 2 */
ZXR10_C(config-mstp)#instance 1 vlan 1-10
ZXR10_C(config-mstp)#instance 2 vlan 11-20

/*Change the priority of Switch C in Instance 1 so that
 it becomes the Root of Instance 1*/
ZXR10_C(config-mstp)#spanning-tree mst instance 1 priority 4096
```

Keep the default configuration of Switch D.

2. As shown in Figure 14 Switches B and C run in an area. CIST Root Bridge is outside the area. One edge port of Switch B/C is blocked.

**FIGURE 14 CONFIGURATION OF MSTP**



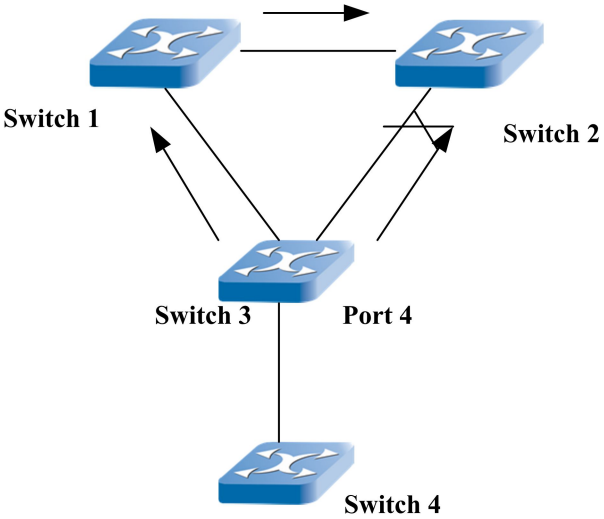

The difference between this example and the previous example is:

One area can have only one instance to communicate with the external network Edge port is in block or forward state for all the VLANs. Therefore, there is no possibility of load balance. Advantages of MSTP are not brought into play. In Figure 14 the link from Switch C to Switch A is in block state for all the VLANs, while the link from Switch B to Switch A is in forward state for all the VLANs.

# BPDU Protection Configuration Example

## Edge Port BPDU Protection Configuration Examples

**FIGURE 15 BPDU CONFIGURATION**

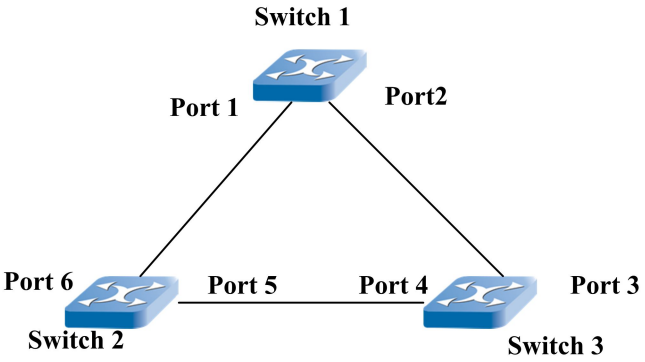



In Figure 15, port 4 of switch 3 is enabled BPDU protection function, the configuration of the port that need to be configured edge port BPDU protection is as follows

```
ZXR10(config-gei_1/4)#spanning-tree bpduguard action shutdown
ZXR10(config-gei_1/4)#spanning-tree edged-port enable
```

# Port Loopback Protection Configuration Example

**FIGURE 16 BPDU CONFIGURATION 2**



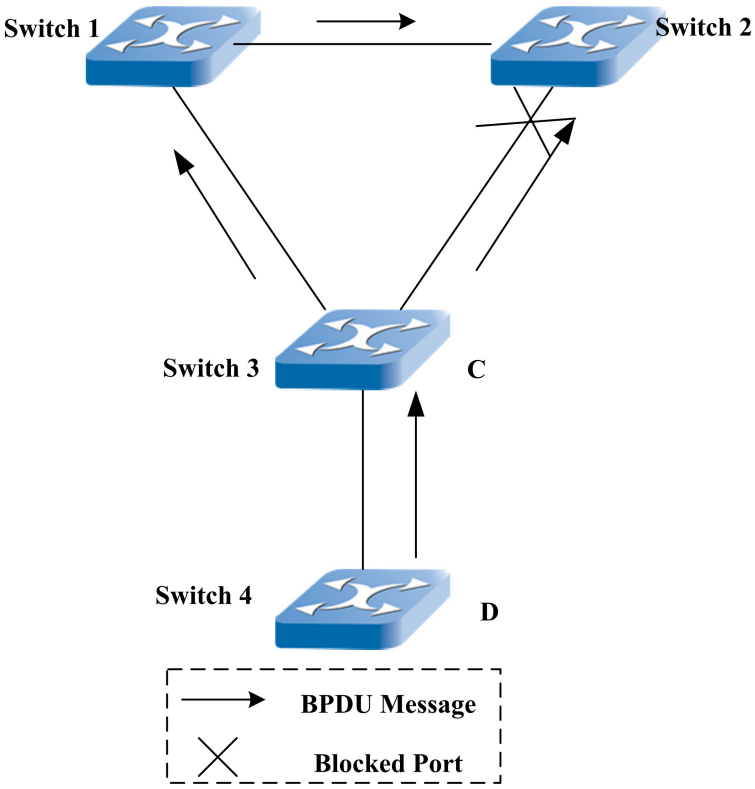

In Figure 16, switch 1 is root switch, port 4 of switch 3 is blocked port, which is enabled loop protection function. Port 5 of switch 2 is disabled.

The configuration of the port that need to be configured port loop-back is as follows:

```
ZXR10(config-gei_1/5)#spanning-tree bpduguard action shutdown
ZXR10(config-gei_1/5)#spanning-tree guard loop instance  <0-16>
ZXR10(config-gei_1/5)#spanning-tree disable
```

# Port Root Protection Configuration Example

**FIGURE 17 BPDU CONFIGURATION 3**




In Figure 17, switch 1 is root switch. Port 4 of switch 3 is enabled port root protection function, The configuration of the port that need to be configured port root protection is as follows:

```
ZXR10(config-gei_1/4)#spanning-tree bpduguard action shutdown
ZXR10(config-gei_1/4)#spanning-tree guard root instance <0-16>
```

# STP Maintenance and Diagnosis

ZXR10 5900E provides show command to view STP related information for fault diagnosis.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10#**show spanning-tree instance** <*instance*> | This shows details of the instance-based spanning tree. |
| 2 | ZXR10#**show spanning-tree interface** <*port-name*> | This shows spanning tree information of a designated interface. |
| 3 | ZXR10#**show spanning-tree statistics** <*port-name*> | This shows statistics on BPDU packets sent and received by a designated interface. |
| 4 | ZXR10#**show spanning-tree inconsistentports** | This shows maintenance information of BPDU protection. |

**Note:**

In the following three cases, loops cannot been avoided even if STP function of switch is enabled.

1. Two switches but multiple parallel links. One converge port configurations and the other does not do so.
2. One switch converge configurations of multiple ports but one port in the aggregation port group is connected to other ports of the local equipment in self-loop mode.
3. Two switches but two parallel links. Due to unknown reasons, both of them cannot receive BPDU packets from the opposite side.

This page is intentionally blank.

Chapter 6

# ZESR/ZESR+ Configuration

**Table of Contents**



## ZESR/ZESR+ Overview

ZESRZTE Ethernet Smart Ringis a solution for solving the layer 2 loop problem (RFC 3619). Compared with STP, the biggest advantage is that the link will switch and recover quickly when one way is disconnected and the shortest time is 50ms.

ZESR is applicable with multi-ring area. Multi-ring is designated that every level is an independent ring and low-level has two entry points to connect with high-level ring. The highest level ring is named as major-level ring and others are named as access rings. Multi-area is named that there are many protection instances on the same ring suitable to different service vlan. Their logic routes are different and independent.

ZESR+ , in double nodes double uplinks networking, improves the current ZESR to meet redundancy protection for uplink and node at the same time in double nodes double uplinks networking.

# Configuring ZESR/ZESR+

## Configuring ZESR Area Protection Instance

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**zesr ctrl-vlan** *<1-4094>* **protect-instance** *<<0-16>* | <1-4094> area control vlan, indicates zesr area, < 0-16>the protected instance ID ,samed as stp instance |
| 2 | ZXR10(config)#**no zesr ctrl-vlan** *<1-4094>* **protect-instance** | <1-4094> area control vlan, indicates zesr area |

ZESR protection instance is same as STP. Service vlan is put into protection instance, so generally enabling STP to cooperate with ZESR. Control vlan should use vlan except service and shouldn't conflict with service and network management. Note that pvid of the port shouldn’t be selected as control vlan. Outside port shouldn’t be put into control vlan.

**Example**

1. This example shows how to configure control vlan as 4000 protection instance as 1.

```
ZXR10(config)# zesr ctrl-vlan 4000 protect-instance 1
```

2. This example shows how to delete control vlan as 4000 protection instance

```
ZXR10(config)# no zesr ctrl-vlan 4000 protect-instance
```

## Configuring Major-level Ring ZESR

To configure ZESR/ZESR+ on major-level ring , use the following commands. Major-level ring is the highest level ring, others are access rings.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**zesr ctrl-vlan** < *1-4094*> **major-level** {**( preforward** <*1-600*>[ **preup** <*0-500*>]**)**| **(role** {**master** |**transit**|**zess-master**|**zess-transit**} | This configures major-level ring ZESR. |
| 2 | ZXR10(config)#**no zesr ctrl-vlan** < *1-4094*> **major-level** | This cancels the configuration of major-level ring ZESR. |

Parameter Description:

| Parameter | Description |
|---|---|
| < 1-4094> | Area control vlan, indicating zesr area |
| <1-600> | Preforward value, the unit is second. After the disconnected port reconnecting, unless ZESR protocol is set or after waiting for preforward time open automatically and the default is 10s. |
| <0-500> | Preup value, the unit is second. After Master detects that loop is up, the status is switched until delaying preup time. The default value is 0. |
| <primary-interface-name> <secondary-interface-name> | major ring two interfaces. To master, secondary interface is blocked to ensure ring is disconnected and no storm. |
| < 1-6> | Hello value, the unit is second. the time of master/zess-transit major interface sending hello protocol message, the default is 1s. |
| < 3-18> | The maximum time daley that master/zess-transit hasn't received hello packet. The unit is second. The default value is 3s. |
| master\|transit\|zess-master\|zess-transit | configuration node role, master\|transit is ZESR master node/transit node, zess-master\|zess-transit is ZESR+ master node/transit node. |

After node role and interface are ensured, preforward and preup can be configured, of which hello, fail and preup only can be used for master or zess-tranist, preup only can be configured as master or zess-master. Interface must be configured in control vlan before it is configured. Interface can use lacp interface but must be dynamic lacp and member interface must close stp.

Besides secondery interface of zess-master node decides blocking location. Therefore the interface must be placed on the uplink which need to be blocked, but secondery interface of zess-transit is suggested to be placed on uplink.

**Example**

1. This example shows how to configure control vlan as 4000, role as master, interface as gei_2/10 and gei_2/20.

```
ZXR10(config)# zesr ctrl-vlan 4000 major-level role master
 gei_2/10 gei_2/20
```

2. This example shows how to configure control vlan as 4000, role as zess-master, interface as gei_2/10 and gei_2/20.

```
ZXR10(config)# zesr ctrl-vlan 4000 major-level role zess-master
gei_2/10 gei_2/20
```

3. This example shows how to configure control vlan as 4000, role as master, preforward as 20s, preup as 20s.

```
ZXR10(config)#zesr ctrl-vlan 4000 major-level preforward 20 preup 20
```

4. This example shows how to configure control vlan as 4000, role as master, hello as 2s, fail as 4s.

```
ZXR10(config)#zesr ctrl-vlan 4000 major-level hello 2 fail 4
```

# Configuring Access Ring ZESR

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config-router)#**zesr ctrl-vlan** < *1-4094*> **level** <*1-2*> **seg** <*1-4*>{**preforward** <*1-600*>[ **preup** <*0-500*>]| **role** {**master** |**transit**}<*primary-interface-name*><*secondary-interface-name*>| { **edge-assistant** | **edge-control**}<*edge-interface-name*>}| **hello** < *1-6*> **fail** < *3-18*>} | This configures access ring ZESR. |
| 2 | ZXR10(config)#**no zesr ctrl-vlan** < *1-4094*> **level** <*1-2*> **seg** <*1-4*> | This cancels the configuration of access ring ZESR. |

Parameter description

< 1-4094> Area control vlan, indicating zesr area

<1-2> Level of access ring

<1-4> access ring SN, at most 4 access rings in each level.

<1-600> Preforward value, the unit is second. After the disconnected port reconnecting, unless ZESR protocol is set or after waiting for preforward time open automatically and the default is 10s.

<0-500> Preup value, the unit is second. After Master or edge-control detects that loop is up, the status is switched until delaying preup time. The default value is 0.

<primary-interface-name> <secondary-interface-name> access ring two interfaces.

< 1-6> Hello value, the unit is second. The default is 1s.

< 3-18> The maximum time dalay that master or edge-control hasn't received hello packet. The unit is second. The default value is 3s.

<edge-interface-name> edge node interface

Switch could be in the entry that major-ring and access ring connect. At that time, it can be in major-ring or access ring . There are two interfaces in major-ring and one interface in access ring . Switch is named as entry node. The entry node could be edge-assistant and edge-control in access ring and edge-control plays a general node master role.

**Example**

1. This example shows how to configure control vlan as 4000, role as master, level as 1 , seg as 1, ports as gei_2/10 gei_2/10

```
ZXR10(config)# zesr ctrl-vlan 4000  level 1 seg 1 role
master gei_2/10 gei_2/20
```

2. This example shows how to configure control vlan as 4000, role as edge-assistant, level as 1 , seg as 1, ports as gei_2/1/10

```
ZXR10(config)# zesr ctrl-vlan 4000  level 1 seg 1 role
edge-assistant gei_2/1/10
```

3. This example shows how to configure control vlan as 4000, level as 1, seg as 1, preforward as 20s, preup as 20s

```
ZXR10(config)#zesr ctrl-vlan 4000 level 1 seg 1
preforward 20 preup 20
```

4. This example shows how to configure control vlan as 4000, level as 1, seg as 1, hello as 2s, fail as 4s

```
ZXR10(config)#zesr ctrl-vlan 4000 level 1 seg 1  hello 2 fail 4
```

## Configuring ZESR Restart-Time

| Command | Function |
|---|---|
| ZXR10(config)#**zesr restart-time** *<30-600>* | <30-600>the specific timethe unit is second, the default is 120s |

**Example** This example shows how to configure ZESR restart-time as 60s.

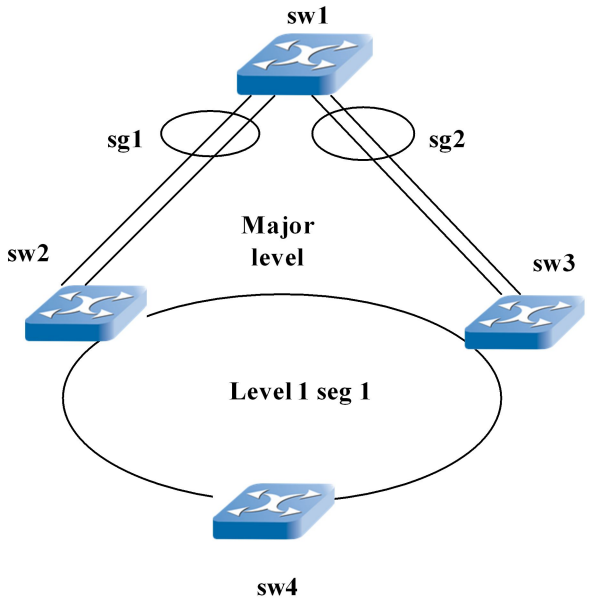
```
ZXR10(config)#zesr restart-time 60
```

# ZESR/ZESR+ Configuration Example

## ZESR Configuration Example

As shown in Figure 18,

**FIGURE 18 ZESR CONFIGURATION EXAMPLE**



SW1-SW4 buildup ring network, transparently transform 100-200, SW1 is core switch and the entire network exit. SW2-SW4 are convergence switch. Demand that service is not be affected if any link is down.

SW1: sg1gei_1/1, gei_1/2connects SW2, sg2gei_1/3, gei_1/4connects SW3

SW2: gei_1/1 connects SW3, gei_1/2 connects SW4, sg2gei_1/3, gei_1/4connects SW1

SW3: gei_1/1 connects SW2, gei_1/2 connects SW4, sg2gei_1/3, gei_1/4connects SW1

SW4: gei_1/1 connects SW2, gei_1/2 connects SW3.

The network formed by SW1, SW2 and SW3 is major level. SW2 is master node. The port that SW2 connect with SW1 is major port (sg1). The network formed by SW2-SW4 is slave ring level 1 seg 1. Take SW4 as master node and select the port connects with SW3 as slave port (gei_1/2), control vlan as 4000.

SW1 configuration:

```
ZXR10_S1(config)#spanning-tree enable
ZXR10_S1(config)#spanning-tree mst configuration
ZXR10(config-mstp)# instance 1 vlan 100-200
ZXR10(config-mstp)#exit

ZXR10_S1(config)#interface smartgroup1
ZXR10_S1(config-smartgroup1)#switchport mode trunk
ZXR10_S1 (config-smartgroup1)#smartgroup mode 802.3ad
ZXR10_S1(config-smartgroup1)switchport trunk vlan 100-200
ZXR10_S1(config-smartgroup1)switchport trunk vlan 4000
ZXR10_S1(config-smartgroup1)exit

ZXR10_S1(config)#interface smartgroup2
ZXR10_S1(config-smartgroup2)#switchport mode trunk
ZXR10_S1 (config-smartgroup2)#smartgroup mode 802.3ad
ZXR10_S1(config-smartgroup2)#switchport trunk vlan 100-200
ZXR10_S1(config-smartgroup2)#switchport trunk vlan 4000
ZXR10_S1(config-smartgroup2)#exit

ZXR10_S1(config)#interface gei_1/1
ZXR10_S1(config-gei_1/1)#negotiation auto
ZXR10_S1(config-gei_1/1)#switchport mode trunk
ZXR10_S1(config-gei_1/1)#switchport trunk vlan 100-200
ZXR10_S1(config-gei_1/1)#switchport trunk vlan 4000
ZXR10_S1(config-gei_1/1)#smartgroup 1 mode active
ZXR10_S1(config-gei_1/1)#spanning-tree disable
ZXR10_S1(config-gei_1/1)#exit

ZXR10_S1(config)#interface gei_1/2
ZXR10_S1(config-gei_1/2)#negotiation auto
ZXR10_S1(config-gei_1/2)#switchport mode trunk
ZXR10_S1(config-gei_1/2)#switchport trunk vlan 100-200
ZXR10_S1(config-gei_1/2)#switchport trunk vlan 4000
ZXR10_S1(config-gei_1/2)#smartgroup 1 mode active
ZXR10_S1(config-gei_1/2)#spanning-tree disable
ZXR10_S1(config-gei_1/2)#exit

ZXR10_S1(config)#interface gei_1/3
ZXR10_S1(config-gei_1/3)#negotiation auto
ZXR10_S1(config-gei_1/3)#switchport mode trunk
ZXR10_S1(config-gei_1/3)#switchport trunk vlan 100-200
ZXR10_S1(config-gei_1/3)#switchport trunk vlan 4000
ZXR10_S1(config-gei_1/3)#smartgroup 2 mode active
ZXR10_S1(config-gei_1/3)#spanning-tree disable
ZXR10_S1(config-gei_1/3)#exit

ZXR10_S1(config)#interface gei_1/4
ZXR10_S1(config-gei_1/4)#negotiation auto
ZXR10_S1(config-gei_1/4)#switchport mode trunk
```

```
ZXR10_S1(config-gei_1/4)#switchport trunk vlan 100-200
ZXR10_S1(config-gei_1/4)#switchport trunk vlan 4000
ZXR10_S1(config-gei_1/4)#smartgroup 2 mode active
ZXR10_S1(config-gei_1/4)#spanning-tree disable
ZXR10_S1(config-gei_1/4)#exit

ZXR10_S1(config)zesr ctrl-vlan 4000 protect-instance 1
ZXR10_S1(config)zesr ctrl-vlan 4000 major level role transit
smartgroup1 smartgroup2
```

### SW2 Configuration

```
ZXR10_S2(config)#spanning-tree enable
ZXR10_S2(config)#spanning-tree mst configuration
ZXR10(config-mstp)# nstance 1 vlan 100-200
ZXR10(config-mstp)#exit

ZXR10_S2(config)#interface smartgroup1
ZXR10_S2(config-smartgroup1)switchport mode trunk
ZXR10_S2 (config-smartgroup1)#smartgroup mode 802.3ad
ZXR10_S2(config-smartgroup1)switchport trunk vlan 100-200
ZXR10_S2(config-smartgroup1)switchport trunk vlan 4000
ZXR10_S2(config-smartgroup1)exit

ZXR10_S2(config)#interface gei_1/1
ZXR10_S2(config-gei_1/1)switchport mode trunk
ZXR10_S2(config-gei_1/1)switchport trunk vlan 100-200
ZXR10_S2(config-gei_1/1)switchport trunk vlan 4000
ZXR10_S2(config-gei_1/1)exit

ZXR10_S2(config)#interface gei_1/2
ZXR10_S2(config-gei_1/2)switchport mode trunk
ZXR10_S2(config-gei_1/2)switchport trunk vlan 100-200
ZXR10_S2(config-gei_1/2)switchport trunk vlan 4000
ZXR10_S2(config-gei_1/2)exit

ZXR10_S2(config)#interface gei_1/3
ZXR10_S2(config-gei_1/3)negotiation auto
ZXR10_S2(config-gei_1/3)switchport mode trunk
ZXR10_S2(config-gei_1/3)switchport trunk vlan 100-200
ZXR10_S2(config-gei_1/3)switchport trunk vlan 4000
ZXR10_S2(config-gei_1/3)smartgroup 1 mode active
ZXR10_S2(config-gei_1/3)spanning-tree disable
ZXR10_S2(config-gei_1/3)exit

ZXR10_S2(config)#interface gei_1/4
ZXR10_S2(config-gei_1/4)negotiation auto
ZXR10_S2(config-gei_1/4)switchport mode trunk
ZXR10_S2(config-gei_1/4)switchport trunk vlan 100-200
ZXR10_S2(config-gei_1/4)switchport trunk vlan 4000
ZXR10_S2(config-gei_1/4)smartgroup 1 mode active
ZXR10_S2(config-gei_1/4)spanning-tree disable
ZXR10_S2(config-gei_1/4)exit

ZXR10_S2(config)#zesr ctrl-vlan 4000 protect-instance 1
ZXR10_S2(config)#zesr ctrl-vlan 4000 major level role transit
 smartgroup1 gei_1/1
ZXR10_S2(config)#zesr ctrl-vlan 4000 level 1 seg 1 role
edge- assistant gei_1/2
```

### SW3 Configuration

### Interface instance configuration is as SW2

```
ZXR10_S3(config)#zesr ctrl-vlan 4000 protect-instance 1
ZXR10_S3(config)#zesr ctrl-vlan 4000 major level role master
 smartgroup1 gei_1/1
ZXR10_S3(config)#zesr ctrl-vlan 4000 level 1 seg 1 role
edge- assistant gei_1/2
```

### SW4 configuration

### Interface instance configuration is as SW2

```
ZXR10_S4(config)#zesr ctrl-vlan 4000 protect-instance 1
ZXR10_S4(config)#zesr ctrl-vlan 4000 level 1 seg 1 role master
```

```
gei_1/1 gei_1/2
```

# ZESR and ZESR+ Hybrid Configuration Example

**FIGURE 19 ZESR+ AND ZESR HYBRID NETWORKING TOPOLOGY FIGURE**

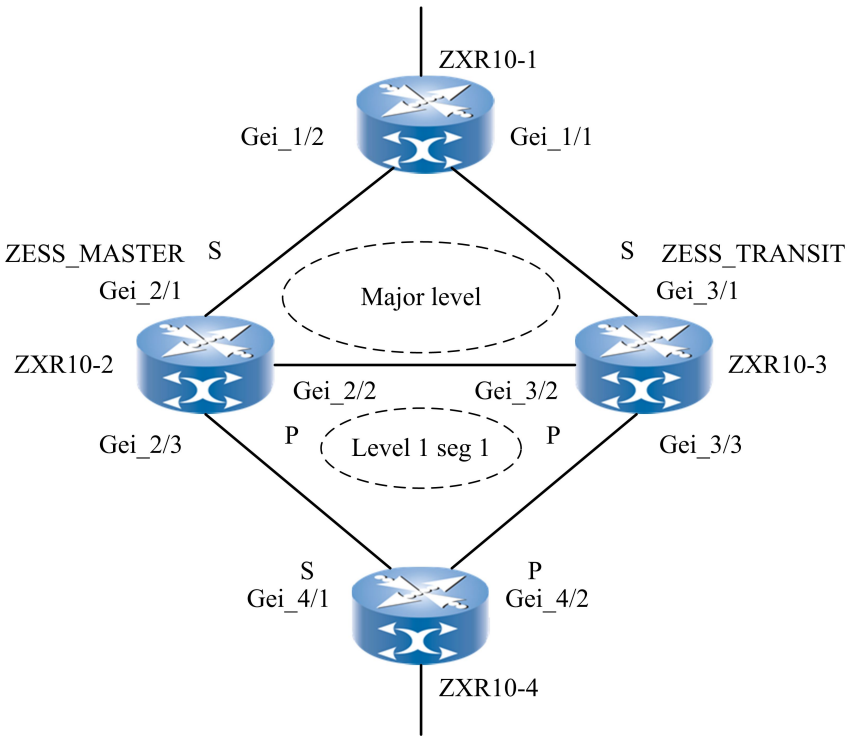



Figure 19 is typical ZESR+ and ZESR hybrid networking topology figure. Node ZXR10-2 , ZXR10-3 and ZXR10-1 form double nodes double uplinks, that is ZESR+. Also three nodes form a main loop virtually. Node ZXR10-2, ZXR10-3 and ZXR10-4 form a level 1seg 1 secondary ring, that is ZESR.

Node 1 configuration:

```
//as a normal switch, the major function is to
transparently transmit data package
//VLAN information need to be configured.
(port with tagged belongs to ctrl-vlan)
//close port broadcast and unknown unicast suppression
//connect ZXR10-3
ZXR10_S1(config)#interface gei_1/1
//configure interface working mode as auto negotiation
ZXR10_S1(config-gei_1/1)#negotiation auto
ZXR10_S1(config-gei_1/1)#switchport mode trunk
ZXR10_S1(config-gei_1/1)#switchport trunk vlan 100-200
ZXR10_S1(config-gei_1/1)#switchport trunk vlan 4000
ZXR10_S1(config-gei_1/1)#exit
 //connect ZXR10-2
ZXR10_S1(config)#interface gei_1/2
//configure interface working mode as auto negotiation
ZXR10_S1(config-gei_1/2)#negotiation auto
ZXR10_S1(config-gei_1/2)#switchport mode trunk
ZXR10_S1(config-gei_1/2)#switchport trunk vlan 100-200
```

```
ZXR10_S1(config-gei_1/2)#switchport trunk vlan 4000
ZXR10_S1(config-gei_1/2)#exit
```

Node 2 configuration:

```
//configure ZESR+ Master node
ZXR10_S2(config)#spanning-tree enable
ZXR10_S2(config)#spanning-tree mst configuration
ZXR10(config-mstp)# instance 1 vlan 100-200
ZXR10(config-mstp)#exit
 //connect ZXR10-1
ZXR10_S2(config)#interface gei_2/1
ZXR10_S2(config-gei_2/1)switchport mode trunk
ZXR10_S2(config-gei_2/1)switchport trunk vlan 100-200
ZXR10_S2(config-gei_2/1)switchport trunk vlan 4000
ZXR10_S2(config-gei_2/1)exit
//connect ZXR10-3
ZXR10_S2(config)#interface gei_2/2
ZXR10_S2(config-gei_2/2)negotiation auto
ZXR10_S2(config-gei_2/2)switchport mode trunk
ZXR10_S2(config-gei_2/2)switchport trunk vlan 100-200
ZXR10_S2(config-gei_2/2)switchport trunk vlan 4000
ZXR10_S2(config-gei_2/2)exit
 //connect ZXR10-4
ZXR10_S2(config)#interface gei_2/3
ZXR10_S2(config-gei_2/3)negotiation auto
ZXR10_S2(config-gei_2/3)switchport mode trunk
ZXR10_S2(config-gei_2/3)switchport trunk vlan 100-200
ZXR10_S2(config-gei_2/3)switchport trunk vlan 4000
ZXR10_S2(config-gei_2/3)exit

ZXR10_S2(config)#zesr ctrl-vlan 4000 protect-instance 1
ZXR10_S2(config)#zesr ctrl-vlan 4000 major level role
zess-master gei_2/2 gei_2/1          //configure zess-master node
/*Note:Secondary interface decides blocking location, therefore ,
therefore Secondary interface can't be configured on corresponding
 interface of link between ZXR10-2 and ZXR10-3 or blocking interface
 faulty will occur.*/
ZXR10_S2(config)#zesr ctrl-vlan 4000 level 1 seg 1 role
edge- assistant gei_2/3  //configure ordinary ZESR border node role
```

Node 3 configuration:

The configuration such as interface instance of node 3 is the same as that of node 2.

```
//Configure ZESR+ Tansit node
ZXR10_S3(config)#zesr ctrl-vlan 4000 protect-instance 1
ZXR10 s3(config)#zesr ctrl-vlan 4000 major-level role zess-transit
 gei_3/2 gei_3/1        //configure zess-transit node
/*When configuring zess-transit role, note that Primary interface
 decides the direction that node sends hello frame, therefor
Primary interface must be configured the corresponding interface
 of link between ZXR10-2 and ZXR10-3, or configuration error will occur.*/
ZXR10 s3(config)#zesr ctrl-vlan 4000 level 1 seg 1 role edge-assistant
 gei_3/3 //configure ordinary ZESR border node role
```

Node 4 configuration:

The configuration such as interface instance of node 4 is the same as that of node 2.

```
//Configure ZESR low-level main node
ZXR10_S4(config)#zesr ctrl-vlan 4000 protect-instance 1
ZXR10 s4(config)#zesr ctrl-vlan 4000 level 1 seg 1 role master
 gei_4/2 gei_4/1  //configure ordinary ZESR master role
```

This page is intentionally blank.

Chapter 7

# ZESS Configuration

**Table of Contents**

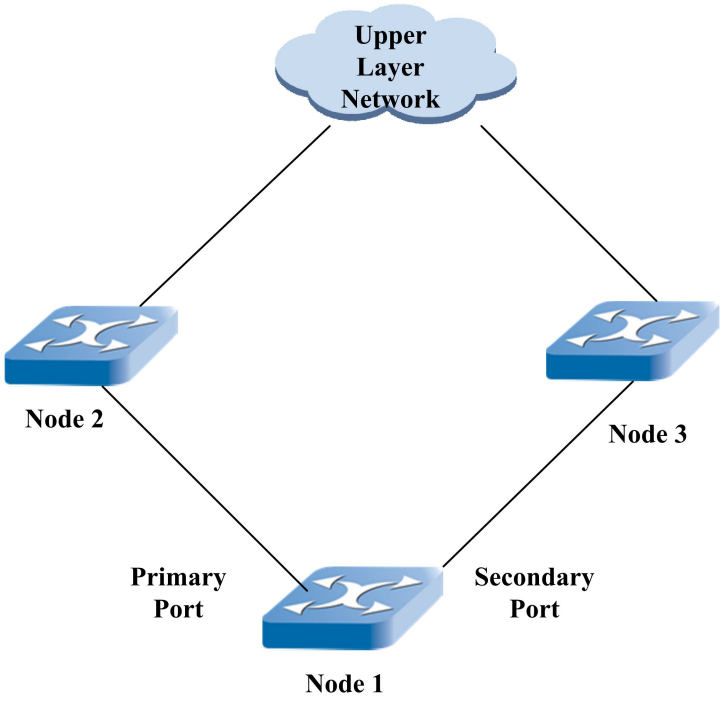


## ZESS Overview

ZESS is ZTE Ethernet Smart Switch technology. It is efficient link switch mechanism. When fault occurs, main link can switch to standby link automatically and quickly to ensure service data normal transmission.

Function description is as follows: As shown in Figure 20, node 1 supports ZESS function, of which port 1 is primary port and port 2 is secondary port. When node 1 detects that both primary port and secondary port are up, blocks protection service VLAN forwarding function of secondary port. When node 1 detects that primary port is DOWN, blocks that of primary port and open that of secondary port. When node 1 detects that primary port recovers as UP, in inversion mode, open primary port and block secondary port again, whereas in non-inversion mode, keep primary port as blocked and secondary port as open. In addition, when ZESS is switching, FDB of block port need to be updated.

FIGURE 20 ZESS NETWORK TOPOLOGY




# Configuring ZESS

## Creating ZESS Domain

Create ZESS domain protect-instance and configure ports. Each domain protects one instance. The maximum number of domain is 4. This is the first step of creating ZESS.

| Step | Command | Function |
|---|---|---|
| 1 | `ZXR10(config)#`**zess domain** *<1-4>* **protect-instance** *<0-16>* **primary** *<primary-interface-name>* **secondary** *<secondary-interface-name>* | This creates ZESS Domain. |
| 2 | `ZXR10(config)#`**no zess domain** *<1-4>* | This deletes ZESS Domain. |

Paramters Description

| Parameter | Description |
|---|---|
| **<** *1-4*> | Domain ID, it indicates ZESS domain. |
| **<** *0-16*> | ZESS protect instance, same as ZESR and STP, put service vlan into protect instance. |

| Parameter | Description |
|---|---|
| <*1-600*> | Preforward value, the unit is second. When the disconnected port reconnects, wait for preforward time and open automatically unless has ZESR protocol configuration. The default value is 10 seconds. |
| <*primary-interface-name*><*secondary-interface-name*> | Two ports of ZESS. In normal cases, secondary port is block, which avoids forming storm. |

**Example**

1. Create zess node that domain is 1, protect-instance is 1 ,ports are gei_1/10 and gei_1/20.

```
ZXR10(config)# zess domain 1 protect-instance 1
primary gei_1/10 secondary gei_1/20
```

2. This deletes zess that domain is 1.

```
ZXR10(config)# no zess domain 1
```

# Configuring Preup Time

To configure preup time of zess node, use the following command.

| Command | Function |
|---|---|
| ZXR10(config)#**zess domain** < *1-4*> **preup** <*1-600*> | The default preup time is 5 seconds. |

Paramters Description

| Parameter | Description |
|---|---|
| <*1-600*> | Preup value, the unit is second. After ZESS detects that link recovers, it doesn't switch state quickly until delaying preup time. The default value is 5 seconds. |

**Example** This example shows how to configure domain 1 preup time as 10 seconds.

```
ZXR10(config)# zess domain 1 preup 10
```

# Configuring ZESS Mode

To configure ZESS mode as inversion or non-inversion mode, use the following command.

| Command | Function |
|---|---|
| ZXR10(config)#**zess domain** <*1-4*> **mode** {**revertive** \| **non_revertive**} | This configures ZESS mode as inversion or non-inversion mode. |

Paramters Description

| Parameter | Description |
|---|---|
| **<** *1-4*> | Domain ID, it indicates ZESS domain. |
| **revertive** \| **non_revertive** | inversion or non-inversion modes, In inversion mode, when main link recovers, service will be cut again, whereas in non-inversion mode, service will still be transmitted on standby link. |

**Example** Configure ZESS node that domain is 1 and in non-inversion mode.

```
ZXR10(config)# zess domain 1 mode non_revertive
```

# Configuring ZESS Control VLAN

ZESS node can send flush packet to upper node for cleaning FDB.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**zess domain** <*1-4*> **ctl-vlan** <*1-4094*> | This configures ZESS ctl-vlan. |
| 2 | ZXR10(config)#**no zess domain** <*1-4*> **ctl-vlan** | This deletes ZESS control VLAN. |

Parameter Description:

| Parameter | Description |
|---|---|
| **<** *1-4*> | Domain ID, it indicates ZESS domain. |
| **<** *1-4094*> | Control vlan Id, it is used to send flush packet which is multicast in this vlan. |

**Example**

1. Configure domain 1 control vlan as 2000.

   ```
   ZXR10(config)#zess domain 1 ctl-vlan 2000
   ```

2. Deletes domain 1 control vlan.

   ```
   ZXR10(config)#no zesr domain 1 ctl-vlan
   ```

# Configuring ZESS Port

To configure the designated port as ZESS packet receiving port, use the following command. This command must be carried out in the corresponding port mode.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config-gei_1/x)#**zess receive-vlan** *<1-4094>* | This adds port into ZESS VLAN receiving table. |
| 2 | ZXR10(config-gei_1/x)#**no zess receive-vlan** *<1-4094>* | This deletes port from ZESS VLAN receiving table. |

Paramters Description

| Parameter | Description |
|---|---|
| **<** *1-4094*> | control vlan identifier, after configuring, this port will receive zess flush packet with same control vlan and forward it. It shall be noted that zess domain used port can't be added into the same vlan receiving table. |

**Example**

1. Configure port gei_1/4 receive control vlan 2000 flush packet.

```
ZXR10(config-gei_1/4)# zess receive-vlan 2000
```

2. Delete port gei_1/4 from the table that control vlan is 2000.

```
ZXR10(config-gei_1/4)# no zess receive-vlan 2000
```

## Clearing ZESS receive-vlan Ports

To clear ports in receive-vlan table on configuration mode, use the following commands.

| Command | Function |
|---|---|
| ZXR10(config)#**zess clear receive-vlan** {*<1-4094>*| **all**} | This clears all ports with zess receive-vlan in designated vlan. |

Paramters Description

| Parameter | Description |
|---|---|
| **<** *1-4094*> | ZESS control vlan Id. |

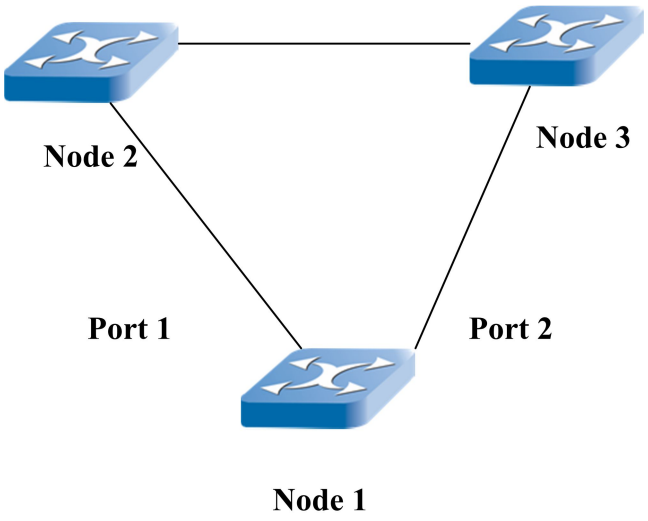
**Example** This example shows how to clear all ports with zess receive-vlan.

```
ZXR10(config)# zess clear receive-vlan all
```

# ZESS Configuration Example

The networking figure is as shown in Figure 21.

FIGURE 21 ZESS NETWORKING CONFIGURATION FIGURE




Node 1-3 comprise ZESS uplink network, node 2-3 connect upper layer network. By default, the upper layer network is connected and simplified as that connection node 2 and node 3 connect directly, node 1 configures ZESS.

Node 1: sg1(gei_1/1,gei_1/2) connect node 2, sg2(gei_1/3, gei_1/4) connect node 3.

Node 2: gei_1/1 connect node 3, sg1 (gei_1/3, gei_1/4) connect node 1.

Node 3: gei_1/1 connects node 2, sg2(gei_1/3, gei_1/4) connect node 1.

Node 1 is configured as ZESS node that domain is 1, protect instance is 1 and control vlan is 4000.

Put node 2-3 port, node 1 port, and the connection port between 2 and 3 into ZESS receive-vlan 4000 table.

Node 1 configuration:

```
ZXR10_S1(config)#spanning-tree enable
ZXR10_S1(config)#spanning-tree mst configuration
ZXR10(config-mstp)# instance 1 vlan 100-200
ZXR10(config-mstp)#exit

ZXR10_S1(config)#interface smartgroup1
ZXR10_S1(config-smartgroup1)#switchport mode trunk
ZXR10_S1 (config-smartgroup1)#smartgroup mode 802.3ad
ZXR10_S1(config-smartgroup1)switchport trunk vlan 100-200
ZXR10_S1(config-smartgroup1)switchport trunk vlan 4000
ZXR10_S1(config-smartgroup1)exit

ZXR10_S1(config)#interface smartgroup2
ZXR10_S1(config-smartgroup2)#switchport mode trunk
ZXR10_S1 (config-smartgroup2)#smartgroup mode 802.3ad
ZXR10_S1(config-smartgroup2)#switchport trunk vlan 100-200
ZXR10_S1(config-smartgroup2)#switchport trunk vlan 4000
ZXR10_S1(config-smartgroup2)#exit

ZXR10_S1(config)#interface gei_1/1
ZXR10_S1(config-gei_1/1)#negotiation auto
ZXR10_S1(config-gei_1/1)#switchport mode trunk
ZXR10_S1(config-gei_1/1)#switchport trunk vlan 100-200
ZXR10_S1(config-gei_1/1)#switchport trunk vlan 4000
ZXR10_S1(config-gei_1/1)#smartgroup 1 mode active
ZXR10_S1(config-gei_1/1)#spanning-tree disable
```

```
ZXR10_S1(config-gei_1/1)#exit

ZXR10_S1(config)#interface gei_1/2
ZXR10_S1(config-gei_1/2)#negotiation auto
ZXR10_S1(config-gei_1/2)#switchport mode trunk
ZXR10_S1(config-gei_1/2)#switchport trunk vlan 100-200
ZXR10_S1(config-gei_1/2)#switchport trunk vlan 4000
ZXR10_S1(config-gei_1/2)#smartgroup 1 mode active
ZXR10_S1(config-gei_1/2)#spanning-tree disable
ZXR10_S1(config-gei_1/2)#exit

ZXR10_S1(config)#interface gei_1/3
ZXR10_S1(config-gei_1/3)#negotiation auto
ZXR10_S1(config-gei_1/3)#switchport mode trunk
ZXR10_S1(config-gei_1/3)#switchport trunk vlan 100-200
ZXR10_S1(config-gei_1/3)#switchport trunk vlan 4000
ZXR10_S1(config-gei_1/3)#smartgroup 2 mode activ
ZXR10_S1(config-gei_1/3)#spanning-tree disable
ZXR10_S1(config-gei_1/3)#exit

ZXR10_S1(config)#interface gei_1/4
ZXR10_S1(config-gei_1/4)#negotiation auto
ZXR10_S1(config-gei_1/4)#switchport mode trunk
ZXR10_S1(config-gei_1/4)#switchport trunk vlan 100-200
ZXR10_S1(config-gei_1/4)#switchport trunk vlan 4000
ZXR10_S1(config-gei_1/4)#smartgroup 2 mode active
ZXR10_S1(config-gei_1/4)#spanning-tree disable
ZXR10_S1(config-gei_1/4)#exit

ZXR10_S1(config)# zess doamin 1 protect-instance 1
primary smartgroup1 secondary smartgroup2
ZXR10_S1(config)# zess domain 1 ctl-vlan 4000
```

## Node 2 configuration:

```
ZXR10_S2(config)#spanning-tree enable
ZXR10_S2(config)#spanning-tree mst configuration
ZXR10(config-mstp)# instance 1 vlan 100-200
ZXR10(config-mstp)#exit

ZXR10_S2(config)#interface smartgroup1
ZXR10_S2(config-smartgroup1)switchport mode trunk
ZXR10_S2 (config-smartgroup1)#smartgroup mode 802.3ad
ZXR10_S2(config-smartgroup1)switchport trunk vlan 100-200
ZXR10_S2(config-smartgroup1)switchport trunk vlan 4000
ZXR10_S2(config-smartgroup1)exit

ZXR10_S2(config)#interface gei_1/1
ZXR10_S2(config-gei_1/1)switchport mode trunk
ZXR10_S2(config-gei_1/1)switchport trunk vlan 100-200
ZXR10_S2(config-gei_1/1)switchport trunk vlan 4000
ZXR10_S2(config-gei_1/1)exit

ZXR10_S2(config)#interface gei_1/3
ZXR10_S2(config-gei_1/3)negotiation auto
ZXR10_S2(config-gei_1/3)switchport mode trunk
ZXR10_S2(config-gei_1/3)switchport trunk vlan 100-200
ZXR10_S2(config-gei_1/3)switchport trunk vlan 4000
ZXR10_S2(config-gei_1/3)smartgroup 1 mode active
ZXR10_S2(config-gei_1/3)spanning-tree disable
ZXR10_S2(config-gei_1/3)exit

ZXR10_S2(config)#interface gei_1/4
ZXR10_S2(config-gei_1/4)negotiation auto
ZXR10_S2(config-gei_1/4)switchport mode trunk
ZXR10_S2(config-gei_1/4)switchport trunk vlan 100-200
ZXR10_S2(config-gei_1/4)switchport trunk vlan 4000
ZXR10_S2(config-gei_1/4)smartgroup 1 mode active
ZXR10_S2(config-gei_1/4)spanning-tree disable
ZXR10_S2(config-gei_1/4)exit

ZXR10_S2(config)#interface smartgroup1
ZXR10_S2(config-smartgroup1)#zess receive-vlan 4000
ZXR10_S2(config)#interface gei_1/1
```

```
ZXR10_S2(config-gei_1/1)#zess receive-vlan 4000
```

Node 3 configuration:

The configuration such as port instance is as node 2.

```
ZXR10_S3(config)#interface smartgroup1
ZXR10_S3(config-smartgroup1)#zess receive-vlan 4000
ZXR10_S3(config)#interface gei_1/1
ZXR10_S3(config-gei_1/1)#zess receive-vlan 4000
```

# ZESS Maintenance

To show ZESS configuration, use the following command.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**show zess brief** | This views all ZESS domain status simply. |
| 2 | ZXR10(config)#**show zess domain** <*1-4*> | This views designated domain status in detail. |
| 3 | ZXR10(config)#**show zess receiver** [ **vlan** <*1-4094*>] | This views designated receive-vlan status. |

Paramters Description

| Parameter | Description |
|---|---|
| **<** *1-4*> | Domain ID, it indicates ZESS domain. |
| **<** *1-4094*> | ZESS control vlan Id. |

Chapter 8

# ZESR and SVLAN Linkage Networking Configuration

**Table of Contents**



## ZESR and SVLAN Linkage Networking Overview

ZESR and SVLAN linkage networking is applicable for multi-ring multi-domain network. SVLAN can switch ports quickly according to ring's connectivity status when fault occurs on the node of ring.

When configuring SVLAN, each configuration data only can designate a customer port as an uplink port. Meanwhile, a group of in-vlan and customer port only can configure a SVLAN data. Therefore it is necessary to configure vlan attribute of another port same as that of uplink port for designating multiple uplink ports for SVLAN.

Configure SVLAN and VLAN two different uplink ports, one is active and another is standby. But only one port can be active at one time, maintain only one logically connective route between any two nodes controlled by ZESR configuration.

# Configuring ZESR and SVLAN Linkage Networking

## Configuring SVLAN

1. To configure SVLAN,
   refer to SVLAN configuration.
2. To configure other uplink ports,
   refer to VLAN configuration.

**Note:**

1. When configuring SVLAN based on ACL, configure downlink data flow is not redirection.
2. The designated uplink port when configuring SVLAN is totally equivalent to the ordinary port that has the same VLAN attribute. The packet with double-layer tag will be broadcast in VLAN that outer tag designates.

**Example** The example shows how to configure SVLAN.

```
ZXR10(config)#vlan qinq extend-session-no 1 customer-port
gei_1/1 uplink-port gei_1/2 in-vlan 10 ovlan 100 unredirect
ZXR10(config)#interface gei_1/1
ZXR10(config-gei_1/1)#switchport qinq customer
ZXR10(config-gei_1/1)#switchport mode hybrid
ZXR10(config-gei_1/1)#switchport hybrid native vlan 10
ZXR10(config-gei_1/1)#switchport hybrid vlan 10,100 untag
ZXR10(config-gei_1/1)#exit
ZXR10(config)#interface gei_1/2
ZXR10(config-gei_1/2)#switchport mode trunk
ZXR10(config-gei_1/2)#switchport trunk native vlan 100
ZXR10(config-gei_1/2)#switchport trunk vlan 100
ZXR10(config-gei_1/2)#exit
ZXR10(config)#interface gei_1/3
ZXR10(config-gei_1/3)#switchport mode trunk
ZXR10(config-gei_1/3)#switchport trunk native vlan 100
ZXR10(config-gei_1/3)#switchport trunk vlan 100
ZXR10(config-gei_1/3)#exit
ZXR10(config)#interface gei_1/4
ZXR10(config-gei_1/4)#switchport mode trunk
ZXR10(config-gei_1/4)#switchport trunk native vlan 100
ZXR10(config-gei_1/4)#switchport trunk vlan 100
ZXR10(config-gei_1/4)#exit
```

# Configuring Port MAC Duplication

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config-gei_1/x )#**mac-duplicate** <*0-4*> **src-vlan** <*vlan-list*> **dest-vlan** <*vlan-list*> | This configures port MAC duplication. |
| 2 | ZXR10(config-gei_1/x )#**no mac-duplicate** <*0-4*> | This cancels port MAC duplication. |

**Note:**

Port is configured as customer port of SVLAN based on ACL. The learning L2 entry on port is vlan id of inner tag , need to enable MAC duplication for customer port. CPU duplicates a L2 entry of outer tag vlan id, downlink packet can get customer port information according to outer vlan id L2 entry.

# Configuring Port LOOPBACK

Configure uplink port of SVLAN based on ACL to implement loopback. At this time ,the port doesn't send packet. All packets are loopback on this port and forwarded to other ports in the same vlan. In one vlan, at least two ports should exist to send packets as SVLAN uplink port.

| Command | Function |
|---|---|
| ZXR10(config-gei_1/x)#**loopback** {**enable**\|**disable**} | This configures port to implement loopback on the interface mode. |

**Note:**

When loopback enable is configured, port learning function will be closed automatically, whereas port learning function is opened automatically when loopback disable is configured. Loopback port as uplink port of SVLAN only receives packet that SVLAN customer port redirects after adding tags and loopbacks to uplink port, which doesn't receive packet forwarded by uplink port. Therefore it needn't port learning function. If port learning function is not disabled, port learns L2 entry when loopbacking message, which causes L2 entry, set by MAC duplication function on customer port, to be coverd.

# Configuring One-Way PVLAN

To configure to forbid uplink port of SVLAN based on ACL to forward data to loopback port, use the following commands.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**vlan private-map-unidirectional session-id** <*id*> **source** <*port-list*> **destination** <*port-list*> | This configures one-way PVLAN source port and destination port. |
| 2 | ZXR10(config)#**no vlan private-map-unidirectional session-id** <*id*> | This cancels one-way PVLAN configuration. |
| 3 | ZXR10(config)#**show vlan private-map-unidirectional** | This displays the configuration information of one-way PVLAN. |

**Note:**

1. Uplink data packet is forwarded by customer port and broadcast in SPVLAN after looped by loopback port. To prevent customer port from receiving data message looped by loopback port, generally, configure a one-way PVLAN data whose source port is loopback port and destination port is customer port.
2. Downlink data packet is forwarded directly to customer port information by uplink port, needn't be forwarded to loopback port. To avoid that the data packet that uplink port forwards to loopback port loops and is forwarded to uplink port again, must configure a one-way PVLAN data whose source port is uplink port and destination port is loopback port.

**Example** This example shows how to configure one-way PVLAN.

```
ZXR10(config)#vlan private-map-unidirectional session-id 1
 source gei_1/3-4 destination gei_1/2
ZXR10(config)#vlan private-map-unidirectional session-id 2
 source gei_1/2 destination gei_1/1
```

# Configuring ZESR

Refer to ZESR configuration chapter.

# Configuration Example

1. Configure gei_1/1 as customer port on the switch, the CVID which receives data packet is VLAN 10, configure gei_1/3/ and gei_1/4 as uplink port, the SPVID which forwards data packet is VLAN 100, configure SVLAN based on ACL and configure

gei_1/2 as auxiliary loopback port. The detailed configuration is as follows:

```
ZXR10(config)#vlan qinq extend-session-no 1 customer-port
gei_1/1 uplink-port gei_1/2 in-vlan 10 ovlan 100 unredirect
ZXR10(config)#interface gei_1/1
ZXR10(config-gei_1/1)#switchport qinq customer
ZXR10(config-gei_1/1)#switchport mode hybrid
ZXR10(config-gei_1/1)#switchport hybrid native vlan 10
ZXR10(config-gei_1/1)#switchport hybrid vlan 10,100 untag
ZXR10(config-gei_1/1)#exit
ZXR10(config)#interface gei_1/2
ZXR10(config-gei_1/2)#switchport mode trunk
ZXR10(config-gei_1/2)#switchport trunk native vlan 100
ZXR10(config-gei_1/2)#switchport trunk vlan 100
ZXR10(config-gei_1/2)#exit
ZXR10(config)#interface gei_1/3
ZXR10(config-gei_1/3)#switchport mode trunk
ZXR10(config-gei_1/3)#switchport trunk native vlan 100
ZXR10(config-gei_1/3)#switchport trunk vlan 100
ZXR10(config-gei_1/3)#exit
ZXR10(config)#interface gei_1/4
ZXR10(config-gei_1/4)#switchport mode trunk
ZXR10(config-gei_1/4)#switchport trunk native vlan 100
ZXR10(config-gei_1/4)#switchport trunk vlan 100
ZXR10(config-gei_1/4)#exit
ZXR10(config)#interface gei_1/1
ZXR10(config-gei_1/1)#mac-duplicate 0 src-vlan 10 dest-vlan 100
ZXR10(config-gei_1/1)#exit
ZXR10(config)#interface gei_1/2
ZXR10(config-gei_1/2)#loopback enable
ZXR10(config-gei_1/2)#exit
ZXR10(config)#vlan private-map-unidirectional session-id 1
source gei_1/3-4 destination gei_1/2
ZXR10(config)#vlan private-map-unidirectional session-id 2
source gei_1/2 destination gei_1/1
```

2. Configure gei_1/1 as customer port on the switch, the CVID which receives data packet is VLAN 10, configure gei_1/3and gei_1/4 as uplink port, the SPVID which forwards data packet is VLAN 100, configure SVLAN based on VLAN translation. The detailed configuration is as follows:

```
ZXR10(config)#vlan qinq session-no 1 customer-port
gei_1/1 uplink-port gei_1/3 in-vlan 10 ovlan 100
ZXR10(config)#interface gei_1/1
ZXR10(config-gei_1/1)#switchport qinq customer
ZXR10(config-gei_1/1)#switchport mode hybrid
ZXR10(config-gei_1/1)#switchport hybrid native vlan 10
ZXR10(config-gei_1/1)#switchport hybrid vlan 10,100 untag
ZXR10(config-gei_1/1)#exit
ZXR10(config)#interface gei_1/3
ZXR10(config-gei_1/3)#switchport mode trunk
ZXR10(config-gei_1/3)#switchport trunk native vlan 100
ZXR10(config-gei_1/3)#switchport trunk vlan 100
ZXR10(config-gei_1/3)#exit
ZXR10(config)#interface gei_1/4
ZXR10(config-gei_1/4)#switchport mode trunk
ZXR10(config-gei_1/4)#switchport trunk native vlan 100
ZXR10(config-gei_1/4)#switchport trunk vlan 100
ZXR10(config-gei_1/4)#exit
```

This page is intentionally blank.

Chapter 9

# Link Aggregation Configuration

**Table of Contents**



## Link Aggregation Overview

Link aggregation (also called trunk) binds multiple physical ports into a logic port. Link aggregation is to implement load balance of outgoing/incoming traffic among these ports. Switch decides from which port packets are sent to the opposite switch depending on the port load balance policy configured by users. When detecting the link of one port is broken, switch stops sending packets from it until it is restored to normal. Link aggregation is quite important in increasing link bandwidth and implementing transmission elasticity and redundancy.

ZXR10 5900E supports two link aggregation modes:

Static trunk directly adds multiple physical ports to the trunk group, thus forming a logic port. This mode is not suitable for observing the state of the link aggregation port.

Link Aggregation Control Protocol (LACP) follows IEEE 802.3ad standards. It converge multiple physical ports into the trunk group to form a logic port. It automatically generates aggregation to obtain the maximum bandwidth.

It is necessary to observe the following rules to configure the link aggregation function for ZXR10 5900E.

- Configuration is done for at most 32 trunk groups, with at most eight ports in each trunk group.
- Cross-interface board aggregation is supported. Member port can be located on any interface board. Selected member ports must operate in full duplex mode and have the same working rate.
- Mode of the member port must be consistent, and can be access, trunk or hybrid.

The logic port formed by link aggregation on the ZXR10 5900E is called smartgroup, which can be used as a common port.

# Configuring Link Aggregation

1. To create a trunk group, use the following command.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**interface smartgroup**<*1-32*> | This creates a trunk group. |
| 2 | ZXR10(config--smartgroupX)#**smartgroup mode**{ **802.3ad**\|**on**} | This creates smartgroup mode. |

2. To add a member port to the trunk group and set the port aggregation mode, use the following command.

| | |
|---|---|
| ZXR10(config-gei_1/x)#**smartgroup** <*smartgroup-id*> **mode** {**passive**\|**active**\|**on**} | This adds a member port to the trunk group and sets the port aggregation mode. |

When aggregation mode is set to on, port runs static trunk and both ends taking part in aggregation should be set to on.

When aggregation mode is set to active or passive, port runs LACP. Active indicates a port is in active negotiation mode. Passive indicates a port is in passive negotiation mode. When configuring dynamic link aggregation, set the aggregation mode of one end to active and that of other to passive or both ends to active.

**Note:**

VLAN link type of member port must be consistent with that of smartgroup. Otherwise, port is not allowed to join trunk group.

3. To set load balance mode for port link aggregation, use the following command.

| | |
|---|---|
| ZXR10(config--smartgroupX)#**smartgroup load-balance** <*mode*> | This sets load balance mode for port link aggregation. |

Port link aggregation of ZXR10 5900E supports six load balance modes. They are based on source and destination IP addresses, source and destination MAC addresses, as well as source and destination ports. By default, load balance mode is based on source and destination MAC addresses.

4. To delete sm, use the following command.

| | |
|---|---|
| ZXR10(config)#**no interface** <*smartgroup-id*> | This deletes sm. |

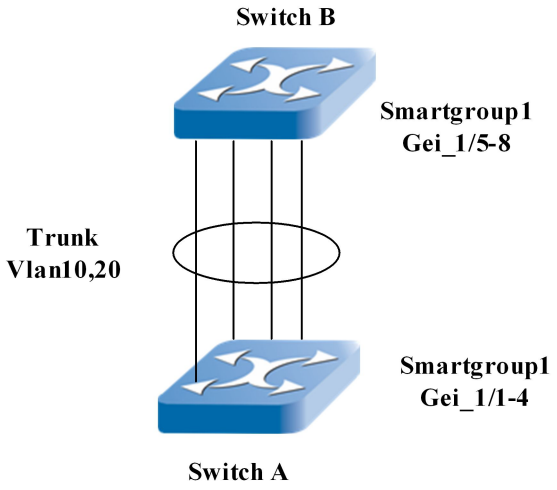
5. To delete port from sm, use the following command.

| | |
|---|---|
| ZXR10(config-gei_1/x)#**no smartgroup** | This deletes port from sm. |

# Link Aggregation Configuration Example

Switches A and B are connected through the smartgroup port, which is converged by four physical ports. The smartgroup operates in trunk mode with VLANs 10 and 20 borne. This is shown in Figure 22.

**FIGURE 22 LINK AGGREGATION CONFIGURATION**




Switch A configuration:

```
/*Create a trunk group*/
ZXR10_A(config)#interface smartgroup1
ZXR10_A(config-smartgroup1)#exit
/* bind ports to Trunk group */
ZXR10_A(config)#interface gei_1/1
ZXR10_A(config-gei_1/1)#smartgroup 1 mode active
ZXR10_A(config-gei_1/1)#exit
ZXR10_A(config)#interface gei_1/2
ZXR10_A(config-gei_1/2)#smartgroup 1 mode active
ZXR10_A(config-gei_1/2)#exit
ZXR10_A(config)#interface gei_1/3
ZXR10_A(config-gei_1/3)#smartgroup 1 mode active
ZXR10_A(config-gei_1/3)#exit
ZXR10_A(config)#interface gei_1/4
ZXR10_A(config-gei_1/4)#smartgroup 1 mode active
ZXR10_A(config-gei_1/4)#exit
/*Modify VLAN link type of smartgroup port*/
ZXR10_A(config)#interface smartgroup1
ZXR10_A(config-smartgroup1)#switchport mode trunk
```

```
ZXR10_A(config-smartgroup1)#switchport trunk vlan 10
ZXR10_A(config-smartgroup1)#switchport trunk vlan 20
ZXR10_A(config-smartgroup1)#switchport trunk native vlan 10
```

Switch B configuration

```
ZXR10_B(config)#interface smartgroup1
ZXR10_A(config-smartgroup1)#smartgroup mode 802.3ad
ZXR10_B(config-smartgroup1)#exit
ZXR10_B(config)#interface gei_1/5
ZXR10_B(config-gei_1/5)#smartgroup 1 mode passive
ZXR10_B(config-gei_1/5)#exit
ZXR10_B(config)#interface gei_1/6
ZXR10_B(config-gei_1/6)#smartgroup 1 mode passive
ZXR10_B(config-gei_1/6)#exit
ZXR10_B(config)#interface gei_1/7
ZXR10_B(config-gei_1/7)#smartgroup 1 mode passive
ZXR10_B(config-gei_1/7)#exit
ZXR10_B(config)#interface gei_1/8
ZXR10_B(config-gei_1/8)#smartgroup 1 mode passive
ZXR10_B(config-gei_1/8)#exit
ZXR10_B(config)#interface smartgroup1
ZXR10_B(config-smartgroup1)#switchport mode trunk
ZXR10_B(config-smartgroup1)#switchport trunk vlan 10
ZXR10_B(config-smartgroup1)#switchport trunk vlan 20
ZXR10_B(config-smartgroup1)#switchport trunk native vlan 10
```

# Link Aggregation Maintenance and Diagnosis

To facilitate link aggregation maintenance and diagnosis, use the following command.

| | |
|---|---|
| ZXR10(config)#**show lacp** {[<*smartgroup-id*>]{**counters**\|**internal**\|**neighbors**} | This facilitates link aggregation maintenance and diagnosis. |

1. This example shows how to view the aggregation state of member port in trunk group 2.

```
ZXR10(config)#show lacp 2 internal
Smartgroup:2
Flag *--LOOP is TRUE
Actor  Agg   LACPDUs    Port      Oper  Port   RX     Mux
Port  State  Interval  Priority  Key  State Machine Machine
--------------------------------------------------------------------
gei_1/7 active   30        32768   0x202  0x3d  current
distributing
gei_1/8 active   30       32768   0x202  0x3d  current
distributing
ZXR10(config)#
```

   When Agg State is selected and Port State is 0x3d, port aggregation succeeds. If the aggregation fails, Agg State is unselected.

2. This example shows how to view the count of the received and transmitted packets of the member port.

```
ZXR10(config)#show lacp 2 counter
Smartgroup:2
Actor     LACPDUs              Marker    LACPDUs    Marker
Port      Tx         Rx        Tx  Rx    Err          Err
```

```
-------------------------------------------------------------------
gei_1/7   11          5          0   0     0            0
gei_1/8   10          6          0   0     0            0
ZXR10(config)#
```

When both protocol transmitted packets Tx and protocol received packets Rx of each member port are not zero, aggregation succeeds. Otherwise, the aggregation fails.

3. This example shows how to view the member port of the opposite side of trunk group 2.

```
ZXR10(config)#show lacp 2 neighbors
Smartgroup 2 neighbors
Actor    Partner              Partner   Port      Oper     Port
Port     System ID            Port No.  Priority  Key      State
-------------------------------------------------------------------
gei_1/7  0x8000,00d0.d0c0.0f60  323     0x8000    0x202   0x3d
gei_1/8  0x8000,00d0.d0c0.0f60  324     0x8000    0x202   0x3d
ZXR10(config)#
```

Partner Port number represents the port number of the partner. When Port State is 0x3d, it indicates that the aggregation succeeds.

This page is intentionally blank.

Chapter 10

# IGMP Snooping Configuration

**Table of Contents**

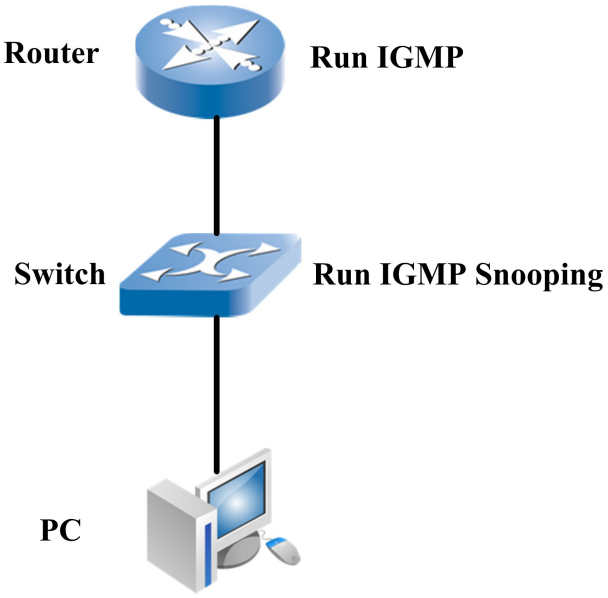


## IGMP Snooping Overview

IGMP snooping is one of layer 2 functions of the switch, which can limit the forwarding of IP multicast traffic.

As shown in Figure 23, IGMP runs between host and multicast router. IGMP communications between host and router so that the switch can learn which ports are multicast members and get multicast forward table before forwarding multicast packets. Multicast packet is sent to the port in the multicast forward table only. IGMP snooping avoid unnecessary network bandwidth waste and improve the switch utilization.

FIGURE 23 IGMP SNOOPING APPLICATION




# Multicast Group Join

The host joins corresponding multicast group by sending an IGMP joining message. When the switch receives the IGMP host report from a host for a particular multicast group, the switch adds the port number of the host to the associated multicast table entry. When other hosts in the same VLAN are interested in the multicast traffic and send a membership report, the switch adds them to the existed forwarding entries.

Switch creates only one forwarding entry for each multicast group in the same VLAN, forwards the multicast traffic of the multicast group in all ports receiving the membership report.

# Multicast Group Leave

Hosts that joined multicast group must respond to IGMP query message transmitted by router periodically. As long as one host responds to IGMP query in a VLAN, the router must continue forwarding traffic of the multicast group that the host joined to the VLAN.

When a host wants to leave a multicast group, it could ignore the IGMP query message transmitted by router periodically (called "leave quietly"), or send IGMPv2 leave message of specific group.

When IGMP Snooping hears IGMPv2 leave message of specific group, the switch sends specific group query message to the port receiving the message to query whether other hosts belonging to the multicast group are available in this port. If IGMP Snooping cannot receive any response message after several queries, it indicates that there are no hosts belonging to the multicast group

in this port, and IGMP Snooping will delete corresponding ports in the Layer 2 forwarding entries; if receiving response message, it is not necessary to modify forwarding table.

## Fast Leave

When switch monitors the IGMPv2 leave message of designated group, it does not send the query message. Instead, the switch directly deletes the corresponding port in the layer 2 forward entry.

Take care when enabling fast leave function in a VLAN, if one of the multiple hosts in a port leaves multicast group, other hosts of the same multicast group in the port cannot receive multicast traffic of the multicast group.

# Configuring IGMP Snooping

## Enabling IGMP snooping

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**ip igmp snooping** | This enables IGMP snooping. |
| 2 | ZXR10(config-vlanX)#**igmp snooping** | This enables IGMP snooping in VLAN. |
| 3 | ZXR10(config-vlanX)#**igmp snooping fast-leave** | This configures group fast leave function in VLAN. |

## Configuring ssm-mapping

To configure ssm-mappingtake received igmp v2 client as v3 client to handle, use the following commands.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**ip igmp snooping ssm-mapping** | This globally enables ssm-mapping. |
| 2 | ZXR10(config)#**ip igmp snooping ssm-mapping-rule** *<group address><source address>* | This configures ssm-mapping rule. |
| 3 | ZXR10(config)#**ip igmp snooping clear-ssm-mapping** | This clears all configured ssm-mapping rules. |

# Configuring Topology Discovery Convergence

After igmp snooping receiving topology changing notification, the action is triggered to speed multicast route convergence.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**ip igmp snooping send-special-leave** | This enables sending special leave message function at the global configuration mode. |
| 2 | ZXR10(config)#**ip igmp snooping send-general-query** | This enables sending general query message function at the global configuration mode. |

# Configuring an Agent Querier

Generally multicast network has at least one multicast router which regularly sends the IGMP query packet. If the multicast network has no multicast router, configure an agent querier to send the IGMP query packet.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**ip igmp snooping querier** | This enables IGMP Snooping querier function.<br>The command with parameter is**ip igmp snooping querier vlan** <*vid*>[**version** <*1-3*>] |
| 2 | ZXR10(config)#**ip igmp snooping query-interval**<*interval*> | This configures query interval of the agent querier. |
| 3 | ZXR10(config)#**ip igmp snooping query-response-interval**<*interval*> | This configures maximum response interval of the agent querier. |
| 4 | ZXR10(config-vlanX)#**igmp snooping querier** <*1-3*> | This configures IGMP Snooping agent querier version in VLAN. |

# Configuring IGMP Agent

When ZXR10 5900E is configured IGMP-SNOOPING to connect multicast router, generally the IGMP agent function need to be opened. There are two functions of agent: one is that when switch receiving query from multicast router, send information got by IGMP-SNOOPING listening to indicate that which group has the client. Another is that when switch listens the first user joining

or last user leaving of a group, send this message to multicast router.

| Command | Function |
|---|---|
| ZXR10(config)#**ip igmp snooping mode proxy vlan** <*vlan id*> | This enables IGMP proxy function. |

# Restricting a Multicast Group

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config-vlanX)#**igmp snooping acl**<*acl-number*> | This makes a ACL filtering for multicast group. |
| 2 | ZXR10(config-vlanX)#**igmp snooping max-group-num** <*number*> | This configures maximum group number allowed by the VLAN. |
| 3 | ZXR10(config-vlanX)#**igmp snooping max-host-in-group**<*ip-address*>[**limit-num**<*num*>] | This configures maximum host group allowed by the VLAN. |

# Limiting Quantity of Users

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**ip igmp snooping max-host-limit interface** <*port-name*> **limit-num** <*1-4096*> | This configures max number of users that port allows to access. |
| 2 | ZXR10(config)#**ip igmp snooping max-host-limit vlan** <*vlan-id*> **limit-num** <*1-4096*> | This configures max number of users that VLAN allows to access. |
| 3 | ZXR10(config)#**ip igmp snooping max-host-limit group** <*A.B.C.D*> **limit-num** <*1-4096*> | This configures max number of users that multicast group allows to access. |
| 4 | ZXR10(config)#**no ip igmp snooping max-host-limit interface** <*port-name*> | This deletes user number limit of port. |
| 5 | ZXR10(config)#**no ip igmp snooping max-host-limit vlan** <*vlan-id*> | This deletes user number limit of VLAN. |
| 6 | ZXR10(config)#**no ip igmp snooping max-host-limit group** <*A.B.C.D*> | This deletes user number limit of multicast group. |

## Configuring Static IGMP SNOOPING

The static configuration is not aging but deleted statically.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config-vlanX)#**igmp snooping static** <*ip-address*> **interface** <*port-name*> | This configures static user in VLAN. If a user needs to join a multicast group without IGMP. IGMP snooping cannot monitor the request. In this case, static configuration can be made. |
| 2 | ZXR10(config-vlanX)#**igmp snooping mrouter interface** <*port-name*> | This configures multicast route port in the VLAN. This command is used when PIM-Snooping is not configured or connecting multicast router that does not send query packets. |

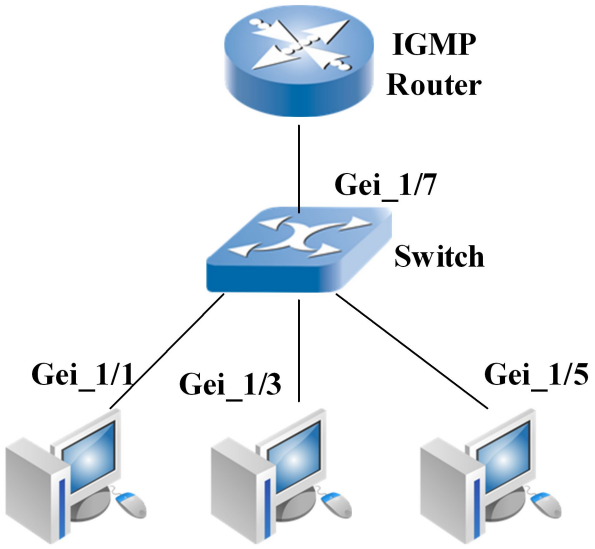
## Modifying Default Time

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config-vlan)#**igmp snooping host-time-out** <*time*> | This modifies aging time of the user. |
| 2 | ZXR10(config-vlan)#**igmp snooping last-member-query-interval**<*interval*> | This modifies the last member query interval. |
| 3 | ZXR10(config-vlan)#**igmp snooping mrouter-time-out**<*time*> | This modifies the route port aging time. |

# IGMP Snooping Configuration Example

Ports gei_1/1, gei_1/3 and gei_1/5 are connected to host. Port gei_1/7 is connected to multicast router. These ports belong to VLAN 10. Enable IGMP Snooping on switch. This is shown in Figure 24.

**FIGURE 24 CONFIGURATION OF IGMP SNOOPING**




Configuration on the switch:

```
ZXR10(config)#ip igmp snooping
ZXR10(config)#ip igmp snooping mode proxy vlan 10
ZXR10(config)#vlan 10
ZXR10(config-vlan10)#igmp snooping
```

# IGMP Snooping Maintenance and Diagnosis

ZXR10 5900E provides **show** command to view information related to IGMP Snooping, to help maintenance and diagnosis.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10#**show ip igmp snooping** | This shows IGMP snooping configuration information. |
| 2 | ZXR10#**show ip igmp snooping vlan**<*vlan-id*> | This shows IGMP snooping configuration information of designated VLAN. |
| 3 | ZXR10#**show ip igmp snooping port-info vlan**<*vlan-id*> | This shows port information related to IGMP snooping. |
| 4 | ZXR10#**show ip igmp snooping statistic**[**clear**] | This shows statistics on IGMP packets. |

ZXR10 5900E also provides **debug ip igmp-snooping** to opens IGMP snooping debugging to debug IGMP snooping.

This example shows IGMP snooping packet receiving and transmitting.

```
ZXR10#debug ip igmp-snooping
ZXR10#
IGMP SNOOPING Rcv 224.1.1.1 Group Report Msg:
From Vlan 1, Port gei_4/7
IGMP SNOOPING Rcv 224.1.1.1 Group Report Msg:
From Vlan 1, Port gei_4/8
...
```

Chapter 11

# UDLD Configuration

**Table of Contents**



## UDLD Overview

UDLD is a layer 2 logic link detection protocol. It can detect Ethernet link logic connectivity and verify physical connectivity. Different from physical connection detection, UDLD detects based on neighbor. The layer 1 device is transparent to UDLD.

UDLD detection builds up neighbor relationship with adjacent layer 2 devices firstly. When the Ethernet port whose status is UP opens UDLD function, this port sends Hello packet that neighbor has entered to inform other adjacent related devices. The adjacent related devices port that opens UDLD function receives this Hello packet and sends an Echo packet back. From the point of view of this device, that receiving this Echo packet means that the two devices are interconnection and the neighbor relationship with peer device has established on this device and sends Echo packet back. After the peer receiving Echo packet, the relationship on the two devices is established.

After both sides establish the neighbor relationship, send detection Hello packet at regular time to detect if the detection link works normally. When receiving the Hello detection packet sent from neighbor, update local storage neighbor buffer information and reset neighbor outtime. If exceed neighbor aging time and still not receiving Hello detection packet, consider that link is on abnormal working condition and need different working modes to handle.

UDLD has two working modes: ordinary mode and aggressive mode. In ordinary mode, only when packets are received and link is verified to be through in uni-direction, this interface can be down; in case corresponding packets are not received or link fails to be verified through in uni-direction, this operation will not be conducted on the interface; In aggressive mode, as long that link fails to be verified to be through bidirectionally, the interface will be down. The common point of the two modes is that in any circumstance, as long as the link is not verified to work normally , print alarm.

Generally speaking, there are several conditions that UDLD sets interface down.

1. In ordinary mode, after sending Hello neighbor joining packet, the received Echo packet displays that the peer port's neighbor is not itself.
2. In aggressive mode, after sending Hello neighbor joining packet, the received Echo packet displays that the peer port's neighbor is not itself.
3. In aggressive mode, after receiving Hello neighbor joining packet, send Echo packet and the peer Echo packet is not received.
4. In aggressive mode, all neighbors in the interface exceed the aging time and don't receive Hello detection packet. When interface is down or the interface can't be used, this device need to send a flush packet to inform the adjacent layer 2 device to delete the information of this device.

Enable UDLD protocol, the receiving echo packet displays the peer port's neighbor is not itself, which indicates fault connection. No matter what mode UDLD applies, the port will be shutdown.

Aging time means protocol packet sending interval (the default is 15 seconds)*3. If aggressive mode is configured, packet is not recieverd when exceed aging time, port will be shutdown.

# Configuring UDLD

## UDLD Global Configuration

| Step | Command | Function |
|---|---|---|
| 1 | `ZXR10(config)#`**Udld enable/disable** <*port-list*> | This globally enables UDLD or enables UDLD in batch. |
| 2 | `ZXR10(config)#`**udld message time** < *7-90*><*port-list*> | This sets protocol packet sending interval. |
| 3 | `ZXR10(config)#`**no udld message time** <*port-list*> | This recovers default protocol packet sending time. |
| 4 | `ZXR10(config)#`**udld recovery enable/disenable** <*port-list*> | This recovers interface up automatically for the reason that UDLD causes interface down (the default is not recovery). |
| 5 | `ZXR10(config)#`**udld recovery timer** <*port-list*> | This sets the time when the interface is recoverd as up automatically for the reason that UDLD causes interface down (the default is 30s). |
| 6 | `ZXR10(config)#`**no udld recovery timer** <*port-list*> | This sets 30s to recover interface up automatically for the reason that UDLD causes interface down. |

| Step | Command | Function |
|---|---|---|
| 7 | ZXR10(config)#**udld reset** <*port-list*> | This resets interface manually for the reason that UDLD causes interface down. |
| 8 | ZXR10(config)#**Debug udld event** | This prints UDLD related information. |
| 9 | ZXR10(config)#**Debug udld packet** | This prints UDLD related information. |

# UDLD Interface Configuration

The configuration in the interface mode is the same as that in global configuration mode.

# UDLD Configuration Notification Items

1. UDLD doesn't support optical-electrical mixed port.
2. The interface configuration can cover global configuration. The global configuration also can cover interface configuration (only suitable for optical interface), for example, optical interface is enabled UDLD in the interface mode. In global configuration mode, no udld mode takes effect.
3. The maximum number of UDLD neighbor is 16.

This page is intentionally blank.

Chapter 12

# LLDP

**Table of Contents**



# LLDP Overview

LLDP Link Layer Discovery Protocol is a new protocol defined in 802.1ab. It makes the adjacent devices send information to each other to update physical topology information and establish device management information base. The workflow of LLDP is as follows:

1. send local device link and management information to adjacent device;
2. Local device receives adjacent device network management information;
3. Store adjacent device network management information in local device MIB database. Network management software can query device layer 2 connection status in MIB database.

LLDP is not configuration protocol of remote system or signal control protocol between ports. LLDP can discover the adjacent devices layer 2 configuration is not same, but it doesn't provide mechanism to solve the problem, it only reports this problem to upper layer management device.

In a word, LLDP is a kind of neighbor finding protocol. It defines a standard for the network devices in the Ethernet such as switch, router and wireless lan access point. It can announce its existence to other nodes in the network and save discovery information of every neighbor device. For example, device configuration, device ID and other information can be announced by this protocol

LLDP defines a common announcement information set, a transmission announcement protocol and a kind of way to save the receiving announcement information. The device that need to announce its information can put multiple pieces of announcement information into one LLDPDU Link Layer Discovery Protocol Data Unit to transmit. This LLDPDU contains a sting of variable length short message units, which is called Type Length Value(TLV). The description is as follows:

- Type means the information type that need to send;
- Length means information byte number;

- Value means the actual information that need to send.

Each LLDPDU contains four forced TLVs and an optional TLV:

1. Device ID TLV
2. Port ID TLV
3. TTL TLV
4. Optional TLV
5. LLDPDU end TLV.

Device ID and port ID are used for identifying transmitter.

TTL TLV tells receiver the reserving period of all information. If the update information from transmitter is not received in the period, the receiver will discard all related information. IEEE has defined a suggesting update frequency, that is, transmit one time every 30s.

Optional TLV includes basic management TLV set( such as port description TLV), special TLV set organized IEEE 802.1, and special TLV set organized IEEE 802.3

The occurrence of LLDPDU end TLV indicates LLDPDU is over.

# Configuring LLDP

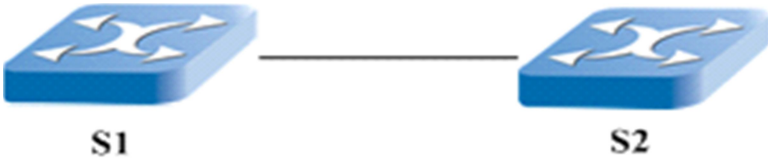
LLDP configuration includes global configuration and interface configuration. Only finishes these two parts of configuration, can this protocol take effect.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**lldp enable** | This enables LLDP. |
| 2 | ZXR10(config)#**lldp hellotime** <*seconds*> | This sets LLDP packet sending interval.<br>The range of lldp hellotime is <5-32768>, the default is 30. |
| 3 | ZXR10(config)#**lldp holdtime** <*multiple*> | This sets LLDP packet aging time, the product of hellotime and parameter is aging time.<br>The range of lldp holdtime is <2-10>, the default is 4. |
| 4 | ZXR10(config)#**lldp** {**enable** \| **rxdisable** \| **txdisable** \| **rxenable** \| **txenable** \| **disabled**}[**interface gei_1/1**] | This configures LLDP global or interface management status. |

# LLDP Configuration Example

Connect the two devices to implement LLDP protocol discovery, as shown in Figure 25.

**FIGURE 25 LLDP CONFIGURATION EXAMPLE**




Configuration of S1:

```
Zxr10#conf t
Zxr10(config)#lldp enable
Zxr10(config)#lldp enable interface gei_1/1
```

Configuration of S2:

```
Zxr10#conf t
Zxr10(config)#lldp enable
Zxr10(config)#lldp enable interface gei_1/1
```

View the configuration result:

1. View LLDP global configuration information

```
Zxr10#show lldp config
---------------------------------------
Lldp init:          1
Lldp enable:        enabledRxTx
Lldp hellotime:     30s
Lldp holdtime:      120s
Lldp maxneighbor:   128
Lldp curneighbor:   2
---------------------------------------
```

2. View LLDP status

```
Zxr10#show lldp statistic
LLDP counters:
Total packets output: 23352, Input: 23266
Total packets error: 0, discard: 0
Total tlvs discard: 0, unrecognized: 0
Total neighbors add: 6, del: 0,
Total neighbors age: 0, drop: 0,
```

3. view LLDP interface configuration information

```
Zxr10#show lldp config int gei_1/1
---------------------------------------
Lldp port enable:        enabledRxTx
Lldp maxneighbor:        8
Lldp curneighbor:        1
Lldp rxstat:             3
Lldp rxPortstus:         1
Lldp rxenable:           1
Lldp rcvChanges:         1
Lldp rcvFrame:           0
Lldp badFrame:           0
Lldp rxTTL:              300
Lldp txstat:             2
Lldp txPortstus:         1
Lldp txenable:           1
Lldp txsLocalchange:     0
Lldp txdelay:            1
Lldp txshutwhile:        0
Lldp txTTR:              10
---------------------------------------
```

4. View LLDP neighbor information

```
Zxr10#show lldp neighbor
Capability Codes: R - Router, T - Trans Bridge, B - Source Route
 Bridge

                  S - Switch, H - Host, I - IGMP, r - Repeater,
P - Phone  W - W
LAN Access Point

Local Intrfce    Device ID    Holdtime  Capability    Platform
        Port ID
------------------------------------------------------------
-
-
gei_1/48       0000d0000100      91    B S            ZXR10 ROS
Version gei_2/1
                                                       V4.08.23
ZX..
gei_1/47       000011110000      296   B S            ZXR10 ROS
Version gei_1/24
                                                    V4.08.23 ZX..
```

5. View LLDP interface neighbor information

```
Zxr10#show lldp neighbor interface gei_1/1
Capability Codes: R - Router, T - Trans Bridge, B - Source
 Route Bridge

        S - Switch, H - Host, I - IGMP, r - Repeater, P - Phone
 W - W
LAN Access Point

Local Intrfce  Device ID  Holdtime Capability Platform Port ID
------------------------------------------------------------------
-
gei_1/48  0000d0000100   92  B S     ZXR10 ROS Version gei_2/1
                                                    V4.08.23 ZX..
```

Chapter 13

# L2PT Configuration

**Table of Contents**



## L2PT Overview

In the VPN mode of QinQ, if the VPN users in different locations want to run their layer 2 protocol, core network need to transparently transmit these layer 2 protocol packets. These packets can't be transparently transmitted, L2PT is used to transparently transmit client network layer 2 protocol packet in QinQ VPN network environment.

The fullname of L2PT is layer 2 protocol tunnel, which is a layer 2 protocol tunnel technology. The principle is that the receiving layer 2 protocol packet is encapsulated by a multicast address on tunnled port of edge switch, and then the encapsulated packet is broadcast in vlan, these packets are de-encapsulated on the port of remote switch that enables tunneled. In the end, transparent transmission is implemented. Layer 2 protocol packet (STP), on the port that doesn't enable L2PT, will not be transmitted in the provider network, which will cause that the client network forms several unconnected stp domain based on area border and the client VPN network can't run a uniform STP topology. L2PT can help user to meet the requirement by transparently transmitting STP BPDU packet in VPN.

## Command Configuration

To enable port or close L2PT tunnel, use the following command.

| Command | Function |
|---|---|
| `ZXR10(config-gei_1/x)#` **l2protocol-tunneled stp** {**enable** \| **disable**} | This enables/disables port tunneled. |

This command is configured in the interface mode. It is used to enable or disable a certain port tunneled. STP field indicates layer

2 protocol mode that need tunneled, which only supports stp protocol. The default is disabled.

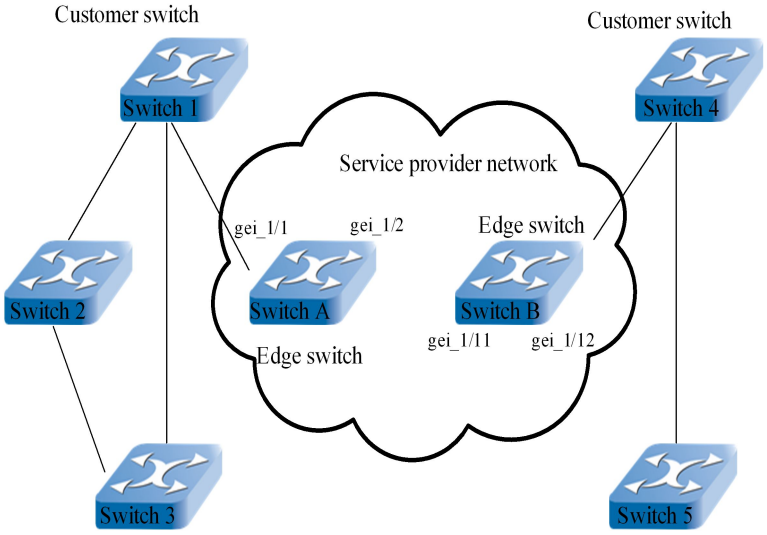
**Note:**

QinQ or SVLAN need to be configured on tunnel access port to take an effect on port tunneled configuration function, which implements L2PT packet transparent transmission function.

# L2PT Configuration Example

Network figure is shown as Figure 26:

**FIGURE 26 L2PT NETWORKING DIAGRAM**




Switch A and switch B are edge switches, which are at the edge of provider network. They are used to connect network device of user. Tunnel is configured on a port of edge switch to implement packet encapsulation or de-encapsulation. Switch 1, switch 2, switch 3, switch 4 and switch 5 are client network switches, which belong to the same VPN.

The client network device that switch A connects transparently transmits STP protocol packet to the client network device that switch B connects through provider network devices, which finishes one-way transparent transmission. The configuration is as follows;

1. On tunnel access edge switch A, the port that connects the client network device is configured L2PT tunnel, QinQ customer port and enabled STP protocol; The port that connects provider network is configured trunk port.

```
Zxr10#conf t
Zxr10(config)#vlan 10
Zxr10(config-vlan10)#exit
```

```
Zxr10(config)#inter gei_1/1
Zxr10(config-gei_1/1)#l2protocol-tunneled stp enable
Zxr10(config-gei_1/1)#switchport qinq customer
Zxr10(config-gei_1/1)#switchport access vlan 10
Zxr10(config-gei_1/1)#exit
Zxr10(config)#interface gei_1/2
Zxr10(config-gei_1/2)#switchport mode trunk
Zxr10(config-gei_1/2)#switchport trunk vlan 10
Zxr10(config-gei_1/2)#exit
Zxr10(config)#spanning-tree enable
```

2. On tunnel exit edge switch B, the port that connects the provider network is configured L2PT tunnel port, the port that connects client network device is configured access port.

```
Zxr10#conf t
Zxr10(config)#vlan 10
Zxr10(config-vlan10)#exit
Zxr10(config)#inter gei_1/11
Zxr10(config-gei_1/11)#l2protocol-tunneled stp enable
Zxr10(config-gei_1/11)#exit
Zxr10(config)#interface gei_1/12
Zxr10(config-gei_1/12)#switchport access vlan 10
```

This page is intentionally blank.

Chapter 14

# Ethernet OAM Configuration

**Table of Contents**



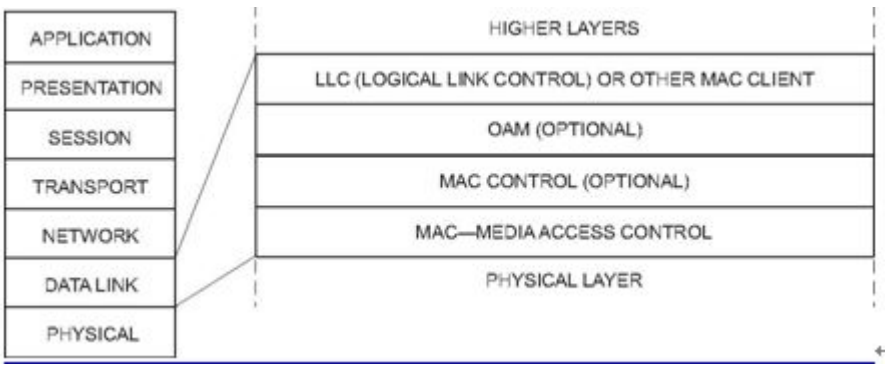
# 802.3ah Overview

IEEE 802.3ah is management of "link" level. It monitors and fault process Point to Point ethernet link. Sometimes "Detection of the last mile" means that. Link layer OAM is mainly used in Point to Point direct-connect link detection.

## Overview

Figure 27 views the location of OAM in ISO/IEC OSI reference module. LLC( logical link control ) or other MAC client layer are above OAM, MAC layer or optional MAC control sub-layer are below OAM. OAM layer is optional. OAM function mainly includes the following three functions:

**FIGURE 27 OAM SUB-LAYER IN ISO/IEC OSI REFERENCE MODULE RELATIONSHIP**




- Remote discovery

- Remote loopback
- Link monitor

DTE joining OAM sub-layer supports active/passive mode. When enabling OAM, DTE that supports the two modes should select active or passive.

# Remote Discovery

OAM provides mechanism for detecting if remote DTE has OAM sub-layer, if find it isn't satisfied, OAM client will know the result and generate fail alarm. There are two cases for fail. One is that peer end doesn't open OAM function, another is link connection fault. During the remote discovery process, the information OAM-PDU tag domain carries current link event (link fault, emergency failure and emergency event). But the specific fault definition , composed of link fault, emergency failure and emergency event, relates to implementation. So there are two ways to know link has fault by remote discovery. One is knew by OAMPDU timeout, another is to define some detailed emergency link events to let client layer know which fault occurs on link from information OAMPDU.

The DTE which is configured active mode lauches discovery process. When discovery process finishes, remote OAM peer entity is in active mode, active DTE is allowed to send any OAMPDU, DTE configured passive mode doesn't launch discovery process, passive DTE feedbacks remote DTE launching discovery process.

# Remote Loopback

OAM provides optional data link layer frame loopback mode. It is controlled by the remote. OAM remote loopback is used for fault location and link performance test. When remote DTE is in OAM remote loop mode, local and remote DTE statistics can be querid and compared at any time. Query can happen before , during and after the process that loop is sent to remote DTE. In addition, analyze OAM sub-layer loop frame to ensure additional information about link health ( namely ensure frame dropping for link fault).

If an OAM client has sent a Loopback Control OAMPDU and is waiting for the peer DTE to respond with an Information OAMPDU that indicates it is in OAM remote loopback mode, and that OAM client receives an OAM remote loopback command from the peer device, the following procedures are recommended: a)If the local DTE has a higher source_address than the peer, it should enter OAM remote loopback mode at the command of its peer. If the local DTE has a lower source_address than the peer, it should ignore the OAM remote loopback command from its peer and continue as if it were never received.

## Link Monitor

Link monitor function is to do statistics for fault symbols or fault frames that physical layer receives during fixed time. The driver has a counter which is always doing the statistics of fault frame, fault symbol, and total receiving frame number. The platform reads these information at specific time, then judge and process according to fault symbol number, fault frame number and total frame number, detect what kind of event happens and generate the corresponding event to inform OAMPDU.

There are four types of link event:

1. Link fault symbol period event, count the fault symbol generated in specific time. Period is defined by symbols number that physical layer receives in some time.
2. Fault frame event, count the fault frame generated in specific time.
3. Fault frame period event, count the fault frame generated in specific time. The period is defined by receiving frame number.
4. Fault frame second accumulated event, count the fault frame second generated in specific time. Period is defined by time interval.

# Configuring 802.3ah

## Function Configuration

| Step | Command | Function |
|---|---|---|
| 1 | `ZXR10(config)#`**set ethernet-oam** {**enable**\|**disable**} | This enables Ethernet-OAM in global configuration mode.<br>Enable: open global link Ethernet-OAM function.<br>Disable: close global link Ethernet-OAM function. |
| 2 | `ZXR10(config)#`**set ethernet-oam** <*oui*> | This sets OUI of Ethernet OAM.<br>**<***oui***>** is character string type, the maximum lengthen is 3. |
| 3 | `ZXR10(config)#`**set ethernet-oam remote-loopback timeout** <*1-10*> | This sets Ethernet OAM remote-loopback timeout interval.<br>The unit of **<***1-10***>** is second. If the function is not set, the default value is 3. |

| Step | Command | Function |
|---|---|---|
| 4 | ZXR10(config-gei_1/x)#**set ethernet-oam** {**enable**\|**disable**} | This enables Ethernet OAM on port.<br>Enable: open port link Ethernet-OAM function.<br>Disable: close port link Ethernet-OAM function. |
| 5 | ZXR10(config-gei_1/x)#**set ethernet-oam period** <*1-10*> **timeout** <*2-10*> **mode** {**active**\|**passive**} | This configures port discovery mode. |
| 6 | ZXR10(config-gei_1/x)#**set ethernet-oam remote-loopback** {**start**\|**stop**} | This enables port remote-loopback.<br>Start: open link ethernet-oam remote-loopback function.<br>Stop: close link ethernet-oam remote-loopback function. |
| 7 | ZXR10(config-gei_1/x) #**set ethernet-oam link-monitor** {{**enable**\|**disable**}\| **(symbol-period (threshold**<*1-65535*> **window** <*1-65535*>**)**\|**(frame (threshold**<*1-65535*>**) window** <*1-60*>**))**\| **(frame-period (threshold**<*1-65535*>**)(window**<*1-600000*>**)**\|**(frame-seconds (threshold** <*1-900*>**)(window** <*10-900*>**))** } | This configures port link monitor event mode. |
| 8 | ZXR10(config)#**show ethernet-oam** [<*port*>{**discovery**\|**link-monitor**\|**satistics**}] | This shows port or global configuration. The show command can be used in other modes. |

# Enhanced Function Configuration

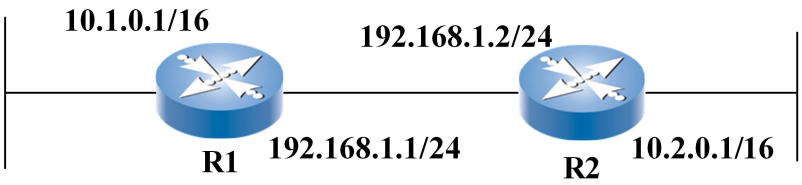
| Step | Command | Function |
|---|---|---|
| 1 | ZXR10#**debug ethernet-oam** { **all** \| **(interface** <*interface-name*>**)**} | This opens OAM Debug function.<br>If select all, all Debug information on all ports will be displayed on foreground.<br>If select interface, only one interface Debug information will be printed on foreground. |
| 2 | ZXR10#**debug ethernet-oam packet interface** <*interface-name*>{**in**\|**out**\|**dual**}**type**{**information**\|**notify**\|**reqst-varb**\|**resps-varb**\|**org-spec**\|**all**} **mode** {**all-time**\|**(number** [**100-1000**]**)**} | This prints Ethernet-oam packet that is in, out or both on one interface, the type of packet can be information PDU, notify PDU, request PDU ,response PDU and so on. |

| Step | Command | Function |
|---|---|---|
| 3 | ZXR10(config)#**clear ethernet-oam** { **all** \|**statistic** } | This clears configuration or statistics data.<br>Select all to clear all ethernet-oam configuration.<br>Select statistic for clearing all statistic information but reserving configuration. |

# Instance Configuration

As shown in Figure 28, run ethernet-oam on R1 and R2. R1 port is gei_1/1, R2 port is gei_1/2.

**FIGURE 28 802.3AH INSTANCE CONFIGURATION**




Configuration of remote discovery

Configuration of R1:

```
ZXR10(config)#set ethernet-oam en
ZXR10(config)#interface gei_1/1
ZXR10(config-gei_1/1)#set ethernet-oam enable
ZXR10(config-gei_1/1)#set ethernet-oam period 10 timeout 3
 mode passive
```

Configuration of R2:

```
ZXR10(config)#set ethernet-oam enable
ZXR10(config)#interface gei_1/2
ZXR10(config-gei_1/2)#set ethernet-oam enable
ZXR10(config-gei_1/2)#set ethernet-oam period 10 timeout 3 mode active
```

When discovering sucess, prompt ETH-OAM gei_1/2 discovery process is successful.

When failing, prompt ETH-OAM gei_1/2 is informed of remote link fault.

ETH-OAM: gei_1/2 is informed of remote unrecoverable failure.

When discovering sucess, the discovery information showed by R2 is as follows:

```
ZXR10(config)#show ethernet-oam gei_1/2 discovery
PortId 2    : ethernet oam enabled
     Local DTE
-----------
     Config:
      Mode          : active
        Period       : 10*100(ms)
        Link TimeOut : 3(s)
        Unidirection : nonsupport
```

```
 PDU max size : 1518
Status:
  Parser       : forward
  Multiplexer  : forward
  Stable       : yes
  Discovery    : done
  Loopback     : off
  PDU Revision : 0

Remote DTE
-----------
Config:
  Mode            : passive
 Link Monitor    : support
  Unidirection    : nonsupport
  Remote Loopback : support
  Mib Retrieval   : nonsupport
  PDU max size    : 1518
Status:
  Parser          : forward
  Multiplexer     : forward
  Stable          : yes
  Mac Address     : 00.19.c6.00.2b.fc
  PDU Revision    : 1
```

# CFM Configuration

## CFM Overview

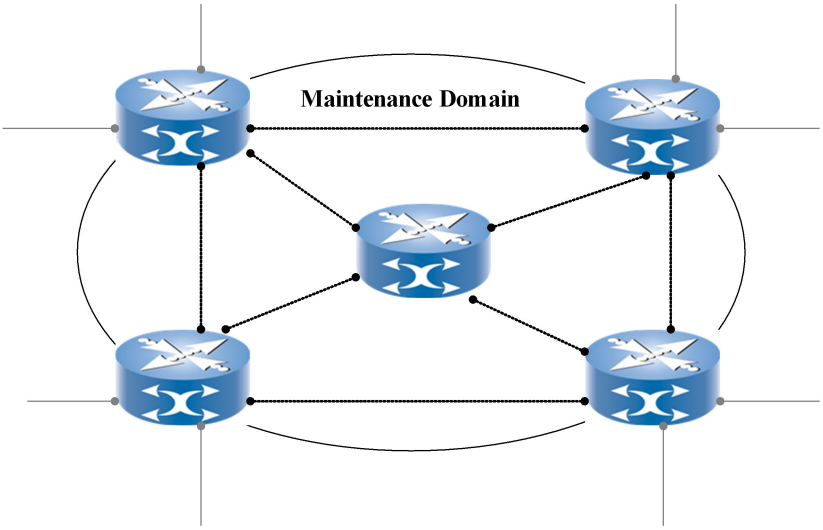
Connectivity Fault Management (CFM) is useful to Virtual Bridged Local Area Networks for detecting, isolating, and reporting connectivity faults. It is aimed primarily at Provider Bridged Networks, but is useful also for C-VLAN networks.

CFM that current switch mainly supports implementation based on IEEE 802.1ag.

The manager of network plans the network service and divides the whole network into multiple MDs for management and diagnosis, single domain is as shown in Figure 29.

The domain in the figure defines a series of ports on edge device and internal device. The gray point on the edge device is service port that connects the devices out of domain, which is defined maintenance edge point (MEP). The black port (includes those devices on the domain intermediate device) is the port that connects devices in the domain, which is defined as maintenance intermediate pointMIP. Implement domain management function by defining MEP and MEP.

**FIGURE 29 MAINTENANCE DOMAIN FIGURE**

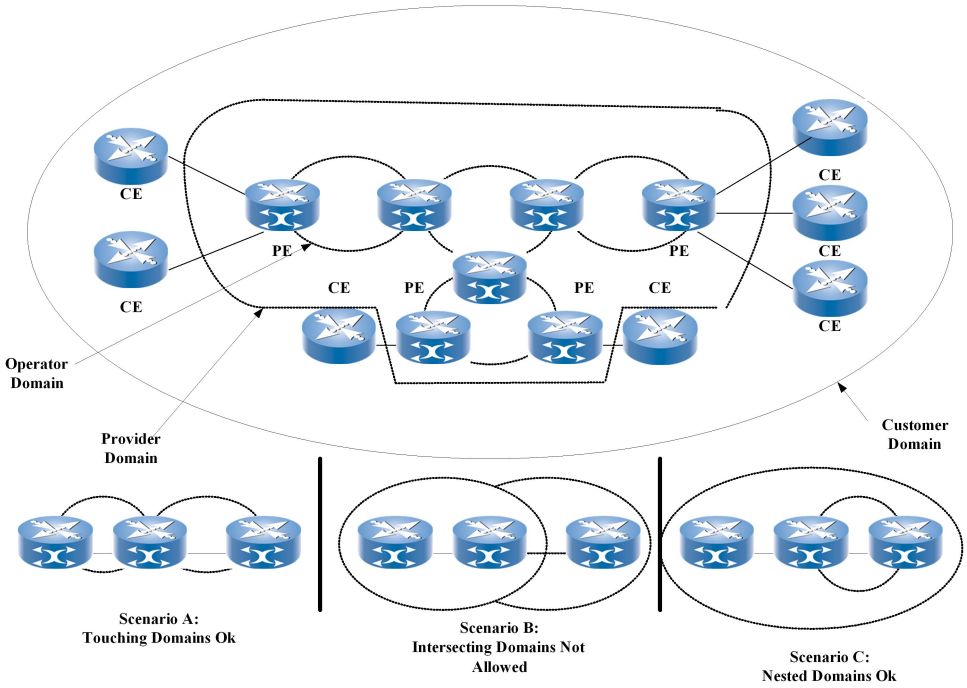



As shown in Figure 30, a network can be divided into user domain, provider domain and operator domain. Each established domain will be designated a level, there are 0-7 levels in total. The domain level will decide domain inclusion relationship. The domain with large level can include the domain with low level, whereas it doesn't work. The domains with same level can't include each other. That is, the domain with the largest range has the highest level. Domain inclusion relationship can not be intersection but tangency(internally-tangent or externally-tangent) and inclusion.

Connectivity Fault Management (CFM) is useful to Virtual Bridged Local Area Networks for detecting, isolating, and reporting connectivity faults. It is aimed primarily at Provider Bridged Networks, but is useful also for C-VLAN networks. IEEE 802.1ag standard defines the following mechanism:

1. Configure multiple embedded MDs by a bridge network. Each domain can be managed by different management organization.
2. Configure MAMaintenance Association ) identified by a lone MD in any designated bridge and a group of VLANs.
3. Protocol, workflow and CFM protocol packet format for detecting, isolating, and reporting connectivity faults.
4. Configure and manage configuration ability of MP (maintenance point) in MA. MP is used for generating CFM packet.
5. Demand MPs to implement specific fault isolating operation and inspect result.

**FIGURE 30 ETHERNET NETWORK MAINTENANCE DOMAIN INCLUSION RELATIONSHIP FIGURE**




Route discovery: MEP use LTM/LTR to track the route from one MEP to another MEP or between MIPs.

Fault detection: MEP use periodically sending and receiving CCM information to detect network connection. It mainly detects connection fault and unwanted connection (fault connection status).

Fault affirmation and isolation: This function belongs to management act, manager affirms fault bill by LBM/LBR, then does the isolation operation.

Fault notification: When MEP has connection fault, the relevant report information will be sent to the designated management system such as NMSTRAP and so on.

Network status detection: estimate network connection status or network delay jitter status through detecting the packet with time stamp between MEPs or packet transceiver with counter value.

MP, includes MEP and MIP, is the smallest entity of management layer realizable function. By comparison, realizable function of MEP is more complicated than that of MIP and management configuration is more complex. In some extent, CFM function is mainly implemented by MEP. MEP can send, receive and handle any one of above information. But MIP only can handle LTM and LBM information and send LTR and LBR information.

# Configuring CFM

## Basic Configuration of CFM

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**cfm** {**enable**\|**disable**} | This enables CFM at global configuration mode.<br>Enable: enable CFM<br>Disable: disable CFM |
| 2 | ZXR10(config)#**cfm create md session** <*1-16*> **name** <*mdname*> **level** <*0-7*> | This establishes and configures MD attribute. When establishing successfully, enter into MD mode automatically. |
| 3 | ZXR10(config)#**cfm delete md** <*1-16*> | This deletes MD. |
| 4 | ZXR10(config)#**cfm md session** <*1-16*> | This enters into MD configuration mode. |
| 5 | ZXR10(config-md)#**ma create session** <*1-32*> **name** <*maname*> | This creates MA.<br>Enter into CFM MA mode successfully. |
| 6 | ZXR10(config-md)#**ma delete** {<*1-32*>\|<*maname*>} | This deletes MA.<br>< 1-32 > : session id of MA to be deleted.<br>< ma-name >: name of MA to be deleted. |
| 7 | ZXR10(config-md)#**ma session** <*1-32*> | This enters into MA configuration mode. |
| 8 | ZXR10(config-md-ma)#**protect** {**vlan**\|**link**} | This sets MA protection mode.<br>The protection link mode can be only used when md's level is 0.<br>The current MA only has protect vlan an protect link modes. |
| 9 | ZXR10(config-md-ma)#**primary vlan** <*1-4094*> | This sets MA main vlan.<br>No matter whether ma's protection mode is vlan or link, main vlan of ma need to be configured. |
| 10 | ZXR10(config-md-ma)#**speed** {**slow**\|**fast**} | This sets MA CCM sending fast and slow mode.<br>The default is slow.<br>**{slow**\|**fast}**set MA allowed CCM sending packet fast or slow. |

| Step | Command | Function |
|---|---|---|
| 11 | ZXR10(config-md-ma)#**ccm timer-interval** < *1-7*> | This sets MA CCM sending time interval, representing CCM sending time interval as 3.33ms,10ms, 100ms, 1s, 10s,1min, 10min respectively.<br>When MA configures as fast, only set the value of 1-3.<br>When MA configures as slow, only set the value of 4-7. |
| 12 | ZXR10(config-md-ma)#**create mep session** <*1-64*><*1-8191*> **direction** {**down**\|**up**} | This creates MEP at MA mode.<br>MEP of down type and up type are supported . The whole device supports 8K MEPs. |
| 13 | ZXR10(config-md-ma)#**create mip session** <*1-64*> **name** <*mipname*> | This creates MIP at MA mode. |
| 14 | ZXR10(config-md-ma)#**create rmep session** <*1-64*><*1-8191*> **remote-mac** <*xxxx.xxxx.xxxx*> | This creates remote-mep.The MAC address of remote MEP, the format is 12 hexadecimal numbers, every four is separated by ".". The MAC address can neither be all 0 nor be multicast address and broadcast address.<br>Creat at most 64 rmeps in one MA, the whole device supports 8K remps. |
| 15 | ZXR10(config-md-ma)#**delete mep** {<*1-8191*>\|**(session** <*1-64*>**)**\|**all**} | This deletes MEP, LMEP or RMEP.<br>Delete mep command can delete a designated MEP by index number and session number. Also can delete the current all MEPs in MA by select all,here MEP includes RMEP. |
| 16 | ZXR10(config-md-ma)#**delete mip** { **(session** <*1-64*>**)**\|**all**} | This deletes MIP.<br>When deleting MIP, delete the designated session number MIP or delete all MIPs in the current MA. |
| 17 | ZXR10(config-md-ma)#**assign mep** <*1-8191*> **to interface** <*interfacename*> | The port can be real port or samartgroup.<br>When assigning port for MEP, the MIP that level is lower than or equal to current MEP level can't exist on the port in the same vlan same level. can't assign port for one MEP many times. |

| Step | Command | Function |
|---|---|---|
| 18 | ZXR10(config-md-ma)#**assign mip** *<1-64>* **interface** *<interfacename>* | This assigns MIP to port.<br>The port can be real port or samartgroup.<br>When assigning port for MIP, the MEP that level is higher than or equal to current MIP level can't exist on the port in the same vlan same level. |
| 19 | ZXR10(config-md-ma)#**no assign mep** *<1-8191>* | This cancels the assignment of MEP to port. |
| 20 | ZXR10(config-md-ma)#**no assign mip** *<1-64>* | This cancels the assignment of MIP to port. |
| 21 | ZXR10(config-md-ma)#**mep***<1-8191>***state**{**enable**\|**disable**} | This sets MEP management enabled. By default, CFM of MEP is disabled. |
| 22 | ZXR10(config-md-ma)#**mep***<1-8191>***alarm-lowest-pri** *<1-5>* | This sets MEP alarm priority.<br>The five priorities from low to high are RDI alarm, MAC alarm, RMEP CCM failure alarm, ERR CCM alarm and XCON alarm. |
| 23 | ZXR10(config-md-ma)#**mep***<1-8191>***priority** *<0-7>* | This sets MEP priority.<br>< 1-8191 >: index number of MEP<br>< 0-7 >: priority of MEP, the range is 0-7. |
| 24 | ZXR10(config-md-ma)#**mep***<1-8191>***ccm-send** {**enable**\|**disable**} | This sets CC packet sending enabled. |
| 25 | ZXR10(config-md-ma)#**mep***<1-8191>***ccm-check**{**enable**\|**disable**} | This checks CC packet.<br>< 1-8191 >: ID number of MEP<br>Enable: ccm-check status of enabled MEP<br>Diable: ccm-check status of disabled MEP |
| 26 | ZXR10(config-md-ma)#**mep** *< 1-8191>* **client-level** *< 0-7>* | This sets level value of client MEP and alarm function use. |

## CFM Function Configuration

1. To enable LB function, use the following command.

   LB (LoopBack) means a MEP sends a designated MP unicast CFM PDU for fault affirmation and isolation. A MP responses LBM and sends unicast packet to LBM initiator MEP.

   When MP loop responser receives a LBM, check its validity first, if invalid then discard. If LBM source address is multi-cast address (not alone MAC address) or destination address

and receiving MP MAC aren't matching, MP discards this LBM packet. If test succeeds, receiving MP sends a LBR to MEP that lauches LBM by taking LBM source address as destination address. When a LBR is received by MHF, ignore this LBR for the reason that MIP hasn't the entity that receiving LBR.

| Command | Function |
|---|---|
| ZXR10#**cfm lbm md** *<1-16>* **ma** *<1-32>* **smep-id** *<1-8191>*{**(dmep-id** *<1-8191>***)** \| **(dmep-mac**<*xxxx.xxxx.xxxx*>**)** \|**(dmip-mac** *<xxxx.xxxx.xxxx>***)**}[{[**repeat** *<1-200>*],[**size**<*1-400*>],[**timeout**<*1-10*>]}] | Rmep must be established first. |

LB can be used only when MD, MA, MP and RMEP establish successfully and global enabled is opened. When using LB function, destination MP parameter can use established RMEP ID , RMEP MAC address or middle MIP MAC address.

LB function supports SG interface.

2. To enable LT function, use the following command.

   LTM (Linktrace Message) : It is originated by MEP. It is used to track the route from MIP to MIP until LTM arrived its destination or MEP can't be forwarded. It is used in fault isolation and route discovery. LTM , multicast packet, whose destination address is selected according to MD level of sending MEP , is forwarded to appropriate MD level MP by bridge network. LTM packet passing middle and MIP of MD and MA all send a LTR to source MEP to ensure the packet arrives here. Destination MP could be MIP.

| Command | Function |
|---|---|
| ZXR10#**cfm ltm md** *<1-16>* **ma** *<1-32>* **smep-id** *<1-8191>*{**(dmep-id** *<1-8191>***)** \| **(dmep-mac**<*xxxx.xxxx.xxxx*>**)** \|**(dmip-mac** *<xxxx.xxxx.xxxx>***)**}[{[**ttl** *<1-64>*],[**timeout** *<5-10>*]]}] | Rmep must be established first. |

   i. LT can be used only when MD, MA, MP and RMEP establish successfully and global enabled is opened.

   ii. When using LT function, destination MP parameter can use established RMEP ID , RMEP MAC address or middle MIP MAC address.

   iii. LT function supports middle and both ends configured SG ports.

   iv. Ttl parameter can't exceed 64 hops. If the middle MIP exceeds 64 hops, even if arrives at the destination, MP will consider that is not arrived.

3. To read one time LTR route information , use the following command.

| Command | Function |
|---|---|
| ZXR10#**cfm ltr-read trans-id** *<1-4294967295>* | This reads one time LTR route information. If read one time arriving a certain MP route, this route must be discovered successfully. |

4. To set ais enabled function , use the following command.

| Command | Function |
|---|---|
| ZXR10(config-md-ma)#**mep**< *1-8191*> **ais** {**enable** \| **disable**} | This sets ais enabled function. If client-level command is configured successfully, enabling ais will send ais message backward. |

5. To set LCK function , use the following command.

| Command | Function |
|---|---|
| ZXR10(config-md-ma)#**mep**< *1-8191*> **lck** {**enable** \| **disable**} | This set LCK function. If cient-level is configured correctly, after LCK is enabled, LCK message will be sent. backward . After LCK message is received, mep will lock this port and stop service flow forwarding. The local or peer end LCK is disabled, port will be unlocked. |

6. To display MD configuration information, use the following command.

| Command | Function |
|---|---|
| ZXR10#**show md** {**all**\|**(session** *<1-16>***)**} | This displays MD configuration information. |

7. To display MA configuration information, use the following command.

| Command | Function |
|---|---|
| ZXR10#**show ma** {**all** \|**( session** *<1-32>***)**} **md** *<1-16>* | This shows all MAs in one MD or a certain MA once. |

8. To show MP information, use the following command.

| Command | Function |
|---|---|
| ZXR10#**show mp** {**all** \|*<1-64>*}**md** *<1-16>* **ma** *<1-32>* | This shows MP information. Only can show a certain MD, one or all MPs in a certain MA. |

## Enhanced Function Configuration

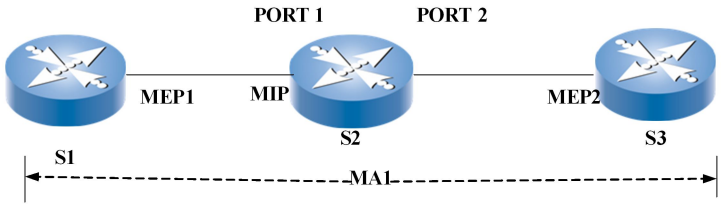
| Step | Command | Function |
|---|---|---|
| 1 | ZXR10#**debug cfm pkt** {**all** \| **(megid md** *<1-16>* **ma** *<1-32>* **mep** *<1-8191>***)** }[{[**direction** {**send** \|**rcv**\|**alll**}],[**pkt-nums** *<10-100>*],[**time-interval** *<4-10000>*]}] | This opens Debug function.<br>In management mode, print all receiving and sending packets of all mep related ports or only print one or several receiving, sending or receiving/sending packets in one mep selectively (default is 10), but the time interval of printing packet can be 4-10000ms. |
| 2 | ZXR10(config-gei_1/x)#**cfm-mac** *<xxxx.xxxx.xxxx>* | This configures port MAC.<br>The mac must be the one that related to the rack. |
| 3 | ZXR10(config)#**clear pbt-cfm** | This clears all CFM configurations.<br>after executing this command, can't see any CFM configuration information by show run. |

## Instance Configuration

The three switches use LT function.

The network figure is shown as Figure 31:

**FIGURE 31 LT FUNCTION CONFIGURATION**




The configurations of S1 are as follows:

```
ZXR10(config)# interface gei_1/1
ZXR10(config—gei_1/1)#switch mode trunk
ZXR10(config—gei_1/1)#exit
ZXR10(config)# vlan 10
ZXR10(config-vlan10)# switchport tag gei_1/1
ZXR10(config-vlan10)# exit
ZXR10(config)# cfm create md session 15 name md15 level 7
ZXR10(config-md)# ma create session 32 name ma1
ZXR10(config-md-ma)#protect vlan
ZXR10(config-md-ma)# primary vlan 10
ZXR10(config-md-ma)# speed slow
ZXR10(config-md-ma)#create mep session 64 1 direction down
ZXR10(config-md-ma)# assign mep 1 to interface gei_1/1
ZXR10(config-md-ma)#  mep 1 state enable
ZXR10(config-md-ma)#create rmep session 2 2 remote-mac
```

```
                        00d0.d052.1200
```

The configurations of S2 are as follows:

```
ZXR10(config)# interface gei_2/1
ZXR10(config—gei_2/1)#switch mode trunk
ZXR10(config—gei_2/1)#exit
ZXR10(config)# interface gei_2/2
ZXR10(config—gei_2/2)#switch mode trunk
ZXR10(config—gei_2/2)#exit
ZXR10(config)# vlan 10
ZXR10(config-vlan10)# switchport tag gei_2/1
ZXR10(config-vlan10)# switchport tag gei_2/2
ZXR10(config-vlan10)# exit
ZXR10(config)# cfm create md session 15 name md15 level 7
ZXR10(config-md)# ma create session 32 name ma1
ZXR10(config-md-ma)#protect vlan
ZXR10(config-md-ma)# primary vlan 10
ZXR10(config-md-ma)# speed slow
ZXR10(config-md-ma)#create mip session 63 name mip63
ZXR10(config-md-ma)# assign mip 63 interface  gei_2/1

ZXR10(config)# cfm enable
```

The configurations of S3 are as follows:

```
ZXR10(config)# interface gei_3/1
ZXR10(config—gei_3/1)#switch mode trunk
ZXR10(config—gei_3/1)#exit
ZXR10(config)# vlan 10
ZXR10(config-vlan10)# switchport tag gei_3/1
ZXR10(config-vlan10)# exit
ZXR10(config)# cfm create md session 15 name md15 level 7
ZXR10(config-md)# ma create session 32 name ma1
ZXR10(config-md-ma)#protect vlan
ZXR10(config-md-ma)# primary vlan 10
ZXR10(config-md-ma)# speed slow
ZXR10(config-md-ma)#create mep session 62 2 direction down
ZXR10(config-md-ma)# assign mep 2 to interface gei_3/1
ZXR10(config-md-ma)#  mep 2 state enable
ZXR10(config-md-ma)#create rmep session 2 1 remote-mac
                        00d0.d052.2800

ZXR10(config)# cfm enable
```

To enable LT function on S1, use the following command.

```
ZXR10# cfm ltm md 15 ma 32 smep-id 1 dmep-id 2
Interface that S1 views is as follows:
Linktrace to 00d0.d052.2800: timeout 5 seconds, 64 hops, trans-id 1.
Please wait 5 seconds to print the result.
-----------------------------------------------------------------
Hops   MAC ADDRESS Ingress Action   Egress Action   Relay Action
-----------------------------------------------------------------
  1     00d0.d034.5670                          EgrOK           RlyFDB
  2     00d0.d052.2800        IngOK                             RlyHit
Destination 00d0.d052.2800 reached.
```

This page is intentionally blank.

Chapter 15

# Sflow Configuration

**Table of Contents**



## Overview

With the rapid development of network service in business environment application, network scale becomes larger and larger, the number of network devices increases repeatedly, and network flow becomes more complicated, therefore the cost of network maintenance keeps on increasing. How to manage the network devices effectively and how to monitor and analyze real network traffic on real-time have become one of the problem which device carrier pay more attention to. At present, each equipment manufacturer provides various network flow monitor technology, but most of these flow monitor technology are private or need specific hardware support technology. SFLOW is a standard flow monitor technology set by IETF currently. It has low requirements on hardware , low resource consumption on device and good technology commonality. Therefore it is applied by many equipment manufacturers.

SFLOW function is composed of three parts: sFlow packet sampling unit, sFlow agent unit, sFlow collector(analyzer). SFlow packet sampling unit and sFlow agent unit are generally integrated into network device, but sFLOW collection is outside of system, which analyzes multiple sFlow agent packets in the network.

SFlow sampling unit is the base of sFlow technology. Sampling procedure is that sample the packet of network on the interface which supports sFlow and send the sampling packet to sFlow agent device to handle. SFlow Collector is the network device that sFlow manage, monitor, collect and analyze. It is responsible for storing packet sended from each sFlow Agent on the network and then analyzing to give device traffic and various analysis report of service.

# SFlow Sampling Unit

SFlow sampling unit implements on router, switch and other network devices that need monitor. In the network system of router, switch and so on, generally, implement packet sampling function by network processor or ASIC chip.

SFlow sampling module mainly sample the packet according to the user demands, meanwhile send the statistic information in relevant packet forwarding procedure. In this process, the original packet is totally not affected and this simple process mode doesn't affect device original performance of processing packet. The mechanism is simple and easy for processing and implementation on hardware and software.

Firstly, configure a sampling rate for sampling interface, which can be a fixed value such as sampling a packet every N packets , or dynamic sampling rate such as deciding the current sampling rate dynamically according to current port working speed, system resource utility ratio and other information. When setting sampling rate, sFlow works normally. For the packet that need to be collected, system copies the packet or only copies the maximum packet length that sFlow Agent needs, sends the copy of the packet, source port and destination port of the packet, the current total packet counter value, sampling packet counter value and other information to sFlow agent module for processing. After sampling, packet are forwarded according to normal packet forwarding workflow.

# SFlow Agent Unit

The main function of SFlow Agent is to analyze sampling packet , send encapsulated sFlow packet according to protocol to sFlow collector device, meanwhile read the statistic information on the interface and send them to sFlow collector device.

The location of sFlow agent module can be in network device itself or outside of device. For most network devices supporting sFlow, sFlow agent unit is integrated into management module of network device. SFlow agent unit runs on a part of device network management software module. It integrates interface count information and sampling packet information into sFlow management packet which is sent to sFlow collector.

# SFlow Collector

SFlow Collector is the network device that sFlow manage, monitor, collect and analyze. It is responsible for storing packet sended from each sFlow Agent on the network and then analyzing to give device traffic and various analysis report of service. Meanwhile, some collector software that have MIB function can configure sFlow agent.

# Configuring sFlow

1. To enable/disable sflow module, use the following command.

| Command | Function |
|---|---|
| ZXR10(config)#**sflow** { **enable** \| **disable**} | This enables/disables sflow module. |

2. To configure sflow Agent, use the following command.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**sflow agent-config ipv4-address A.B.C.D** [**udp_port**] | This configures sFlow Agent IP address. |
| 2 | ZXR10(config)#**sflow agent-config ipv6-address X:X::X:X** [**udp_port**] | This configures sFlow Agent IPv6 address. |

3. To configure sFlow Collector, use the following command.

| Step | Command | Function |
|---|---|---|
| 1 | ZXR10(config)#**sflow collector-config ipv4-address A.B.C.D** [**udp_port**] | This configures sFlow Collector IP address. |
| 2 | ZXR10(config)#**sflow collector-config ipv6-address X:X::X:X** [**udp_port**] | This configures sFlow Collector IPv6 address. |

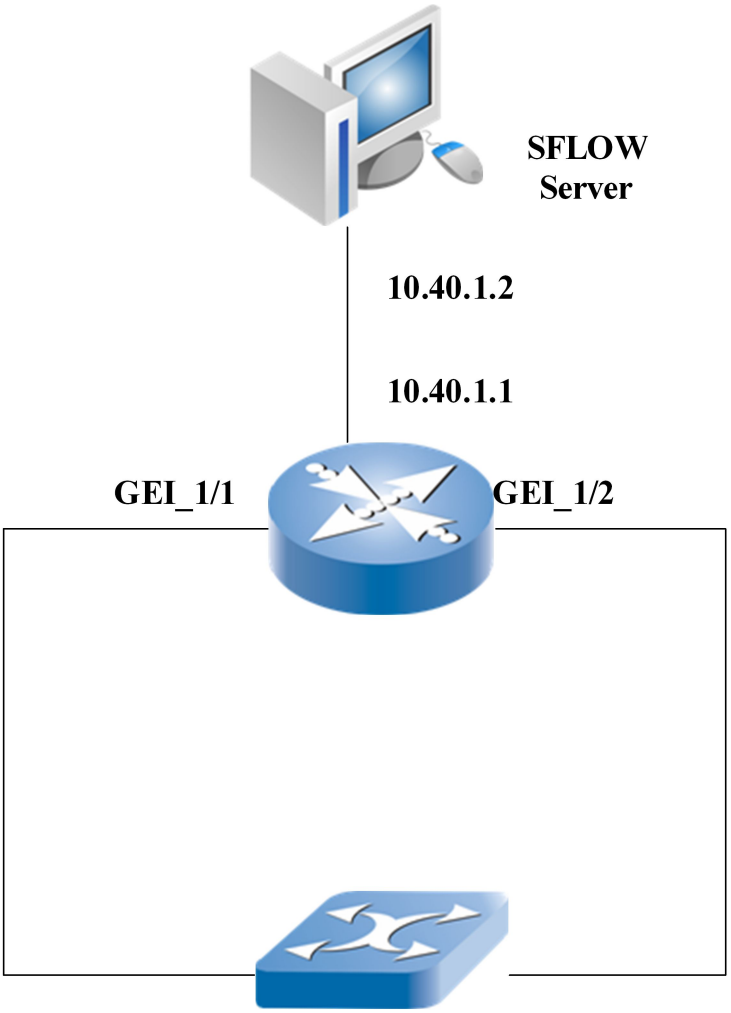
4. To configure sFlow sampling rate, use the following command.

| Command | Function |
|---|---|
| ZXR10(config-gei_1/x)#**sflow-sample-rate** { **ingress**\| **egress**}<*ratevalue*> | This configures sFlow sampling rate on the interface. |

# SFlow Configuration Example

The networking figure is as shown in Figure 32.

FIGURE 32 SFLOW CONFIGURATION EXAMPLE




Configure sampling port gei_2/12 sending and receiving data flow which is monitored on sflow server.

```
ZXR10(config)#sflow enable
ZXR10(config)#sflow agent-config ipv4-address 10.40.1.1
ZXR10(config)#sflow collector-config ip-address 10.40.1.2
ZXR10(config)#interface gei_1/1
ZXR10(config-gei_1/1)#sflow-sample-rate egress  1000
ZXR10(config-gei_1/1)#sflow-sample-rate ingress  1000
ZXR10(config-gei_1/1)#exit
```

# SFlow Maintenance and Diagnosis

SFlow provides the following commands for maintenance and diagnosis. The viewed information is the configuration in the instance.

```
ZXR10 (config)#show sflow
```

```
sflow enable
sflowagent ip-addr:         10.40.1.1
sflwcollector ip-addr:       10.40.1.2

portname     egress_sample_rate    ingress_sample_rate
gei_2/12           1000                  1000
```

This page is intentionally blank.

Chapter 16

# IPFIX Configuration

**Table of Contents**



# IPFIX Overview

## IPFIX Overview

IPFIX (IP Flow Information Export) is used to analyze and perform statistics to communication traffic and flow direction in network. In 2003, IETF select Netflow V9 as IPFIX standard from 5 candidate schemes.

To analyze and perform statistics to data flow in network, it is needed to distinguish types of packets transmitted in network.

Due to non-connection oriented characteristics of IP network, the communication of different types of services in network can be a series of IP packets sent from one terminal device to another terminal device. This series of packets actually forms one data flow of a service in carrier network. If management system can distinguish all flows in the entire network and correctly record transmit time of each flow, occupied network port, transmit source/destination address and size of data flows, traffic and flow direction of all communications in the entire carrier network can be analyzed and performed with statistics.

By telling differences among different flows in network, it is available to judge if two IP packets belong to the same one flow. This can be realized by analyzing 7 attributes of IP packet: source IP address, destination IP address, source port id, destination id, L3 protocol type, TOS byte (DSCP), ifIndex for network device input (or output).

With above 7 attributes of IP packet, flows of different service types transmitted in network can be rapidly distinguished. Each distinguished data flow can be traced separately and counted accurately, its flow direction characteristics such as transmit direction and destination can be recorded, and the start time, end time, ser-

vice type, contained packet number, byte number and other traffic information can be performed statistics.

As a macro analysis tool for network communication, Netflow technology doesn't analyze the specific data contained in each packet in network, instead it tests characteristics of transmitted data flow, which enables Netflow technology with good scalability: supporting high-speed network port and large-scale telecom network.

As for processing mechanism, IPFIX introduces multi-level processing procedures:

- In preprocessing stage, IPFIX can filter data flow of a specific level or perform sampling to packets on high-speed network interface based on demands of network management. With IPFIX, processing load of network device can be relieved and scalability of system can be enhanced while the needed management information is collected and performed statistics.
- In postprocessing stage, IPFIX can select to output all collected original statistics of data flow to upper-layer server for data sorting and summary; alternatively, network device can perform data aggregation to original statistics in various modes and send the summary statistics result to upper layer management server. The latter one can reduce the data quantity output by network device, thus decreasing requirement to configuration of upper layer management server and promoting scalability and working efficiency of upper layer management system.

IPFIX outputs data in format of template. Network device will send packet template and data flow records respectively to upper layer management server when outputting data in IPFIX format. Packet template specifies format and length of packet in subsequently sent data flow record for management server processing subsequent packets. Meanwhile to avoid packet loss and errors in packet transmission, network device repeats sending packet template to upper layer management server regularly.

# Sampling

IPFIX supports packet number-based sampling as well as time-based sampling. Sampling rate can be configured on each interface separately.

# Timeout Management

As for collected flow data,

- In case data are not updated within the inactive time, data will be output to NM server;
- As for long time active flow, the data will also be output to NM server after active time.

## Data Output

After collecting data flows in network, network device always outputs them to NM server. IPFIX supports to output data to multiple NM servers. Generally, data are output to two servers: master server and slave server.

IPFIX adopts template-based data output mode. IFPIX supports to send template every a few packets or at a certain interval. Packet template specifies the format and length of packets in subsequent data flows, and server resolves subsequent data flows according to template.

# Configuring IPFIX

## Basic Configuration

### Enabling/Disabling IPFIX Module

| Command | Functions |
|---|---|
| ZXR10(config)#**ip stream** {**enable**\|**disable**} | This enables/disables IPFIX module. |

### Setting IPFIX Memory Entries

| Command | Functions |
|---|---|
| ZXR10(config)#**ip stream cache entries** <*number*> | This sets the number of data flow entries stored in IPFIX module, 4096 by default. |

### Setting Aging Time of Active Stream

| Command | Functions |
|---|---|
| ZXR10(config)#**ip stream cache timeout active**<*number*> | This sets aging time of active stream. |

As for long time active stream, in case it exceeds the set aging time, this data flow will age out, in minutes, 30 minutes by default.

## Setting Aging Time of Inactive Stream

| Command | Functions |
|---|---|
| ZXR10(config)#**ip stream cache timeout inactive**<*number*> | This sets aging time of inactive stream. |

If data of a flow are not updated within the specified time, the aging information will be notified to stream record, in seconds, 15 seconds by default.

## Setting Sampling Rate

| Step | Command | Functions |
|---|---|---|
| 1 | ZXR10(config)#**interface** < *interface-name*> | This enters interface configuration mode. |
| 2 | ZXR10(config-if)#**netflow-sample-rate**{ **ingress**\| **egress** } | This configures packet number-based IPFIX sampling rate. |

## Setting NM Server Address and L4 Port ID

| Command | Functions |
|---|---|
| ZXR10(config)#**ip stream export destination** <*ip-address*> **udp-port** | This sets the address and port id of NM server, to which packets are sent. |

## Setting Source Address for Network Device Sending Packets

| Command | Functions |
|---|---|
| ZXR10(config)#**ip stream export source** <*ip-address*> | This sets source address for network device sending packets. |

## Configuring TOPN

| Command | Functions |
|---|---|
| ZXR10(config)#**ip stream topn N sort-by** {**bytes**\|**packets**} | This sets size and sorting behavior of TOPN (by packet number or byte number). |

# Template Configuration

## Setting Template

| Command | Functions |
|---|---|
| ZXR10(config)#**ip stream templat** <*template-name*> | This sets template. |

## Setting Data Field Contained in Template Packet

| Command | Functions |
|---|---|
| ZXR10(config-stream-template)#**match field** | This sets data field contained in template packet. |

Server resolves data contained in subsequent data flow according to these fields. The fields include source IP, destination IP, source port, destination port, the number of bytes contained in data flow, the number of packets contained in data flow, type of L3 protocol, TOS field, start time of data flow, end time of data flow, data flow ingress index, data flow egress index and TCP flag.

## Deleting Template

| Command | Functions |
|---|---|
| ZXR10(config)#**no ip stream template template-name** | This deletes one template. |

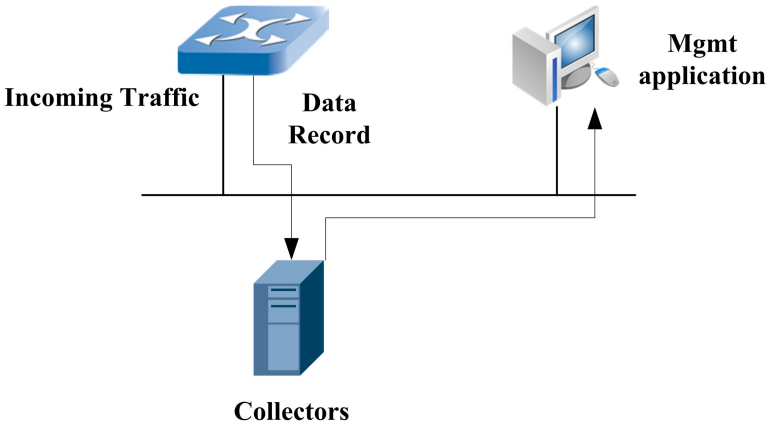
## Running Template

| Command | Functions |
|---|---|
| ZXR10(config)#**ip stream template template-name** | This runs template. |

# IPFIX Configuration Example

An IPFIX configuration example is given here with network topology as shown in Figure 33.

**FIGURE 33 IPFIX CONFIGURATION EXAMPLE**




```
ZXR10_R1(config)#ip stream enable
ZXR10_R1(config)#interface gei_2/12
ZXR10_R1(config-if)#netflow-sample-rate ingress unicast 1024
ZXR10_R1(config-if)#netflow-sample-rate egress unicast 1024
ZXR10_R1(config)#ip strem export destination 192.168.1.1 2055
ZXR10_R1(config)#ip strem export destination 192.168.1.2 2055
ZXR10_R1(config)#ip stream export source 192.168.1.244
ZXR10_R1(config)#ip stream export version 9
ZXR10_R1(config)#ip stream topn 10 sort-by packets
ZXR10_R1(config)#ip stream template test
ZXR10_R1(config-stream-tempalte)#match src-ip
ZXR10_R1(config-stream-tempalte)#match dst-ip
ZXR10_R1(config-stream-tempalte)#match src-port
ZXR10_R1(config-stream-tempalte)#match dst-port
ZXR10_R1(config-stream-tempalte)#exit
ZXR10_R1(config)#ip stream run template test
```

# IPFIX Maintenance and Diagnosis

For the convenience of IPFIX maintenance and diagnosis, IPFIX provides related view commands.

1. To show IPFIX-related configurations, execute the following command:

   **show ip stream-config**

   This includes whether to enable IPFIX module, size of memory entries, server address, port configuration, source address configuration, template refresh rate and refresh time configuration.

2. To show TOPN, execute the following command:

   **show ip stream-topn**

   This shows information of N data flows according to set TOPN display mode. The information includes data flow ingress, egress, source address, destination address, source port, destination port, L3 protocol type, the number of packets or the number of bytes (corresponding to TOPNS setting).

3. To show template configuration, execute the following command:

   **show ipstream-template**

   This shows configuration of template, that is, fields contained in template.

This page is intentionally blank.

# Figures

# Glossary

**BPDU**
- Bridge Protocol Data Unit

**CIST**
- Common and Internal Spanning Tree

**CST**
- Common Spanning Tree

**HMAC-MD5**
- Hashed Message Authentication Code with MD5

**IGMP**
- Internet Group Management Protocol

**ISP**
- Internet Service Provider

**IST**
- Internal Spanning Tree

**LACP**
- Link Aggregation Control Protocol

**MAC**
- Medium Access Control

**MD5**
- Message Digest 5 Algorithm

**MSTP**
- Multiple Spanning Tree Protocol

**OAM**
- Operation, Administration and Maintenance

**PE**
- Provider Edge

**PVLAN**
- Private Virtual Local Area Network

**RSTP**
- Rapid Spanning Tree Protocol

**STP**
- Spanning Tree Protocol

**UDLD**
- UniDirectional Link Detection

**VID**
- VLAN Identifier

**VLAN**
- Virtual Local Area Network

**ZESR**
- ZTE Ethernet Switch Ring

**ZESS**
- ZTE Ethernet Smart Switch