Inspur

CN12900 Series

INOS-CN Interfaces Configuration Guide

Inspur-Cisco Networking Technology Co.,Ltd. provides customers with comprehensive technical support and services. For any assistance, please contact our local office or company headquarters.
Website: http://www.inspur.com/
Technical Support Tel: 400-691-1766
Technical Support Email:icnt_service@inspur.com
Technical Document Support Email: icnt_service@inspur.com
Address: 1036 Langchao Road, Lixia District, Jinan City, Shandong Province
Postal code: 250101

Notice

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

Preface

Objectives

This guide describes main functions of the CN12900 Series. To have a quick grasp of the CN12900 Series, please read this manual carefully.

Versions

The following table lists the product versions related to this document.

Product name	Version
CN12900 Series	

Conventions

Symbol conventions

The symbols that may be found in this document are defined as follows.

Symbol	Description
Warning	Indicates a hazard with a medium or low level of risk which, if not avoided, could result in minor or moderate injury.
Caution	Indicates a potentially hazardous situation that, if not avoided, could cause equipment damage, data loss, and performance degradation, or unexpected results.
Note	Provides additional information to emphasize or supplement important points of the main text.
Tip	Indicates a tip that may help you solve a problem or save time.

General conventions

Convention	Description
Boldface	Names of files, directories, folders, and users are in **boldface**. For example, log in as user **root**.
Italic	Book titles are in *italics*.
`Lucida Console`	Terminal display is in `Lucida Console`.

Command conventions

Convention	Description
Boldface	The keywords of a command line are in **boldface**.
Italic	Command arguments are in *italics*.
[]	Items (keywords or arguments) in square brackets [ ] are optional.
{ x | y | ... }	Alternative items are grouped in braces and separated by vertical bars. One is selected.
[x | y | ...]	Optional alternative items are grouped in square brackets and separated by vertical bars. One or none is selected.
{ x | y | ... } *	Alternative items are grouped in braces and separated by vertical bars. A minimum of one or a maximum of all can be selected.
[x | y | ...] *	The parameter before the & sign can be repeated 1 to n times.

GUI conventions

Convention	Description
Boldface	Buttons, menus, parameters, tabs, windows, and dialog titles are in **boldface**. For example, click **OK**.
>	Multi-level menus are in boldface and separated by the ">" signs. For example, choose **File** > **Create** > **Folder**.

Keyboard operation

Format	Description
Key	Press the key. For example, press **Enter** and press **Tab**.

Format	Description
Key 1+Key 2	Press the keys concurrently. For example, pressing **Ctrl+C** means the two keys should be pressed concurrently.
Key 1, Key 2	Press the keys in turn. For example, pressing **Alt**, **A** means the two keys should be pressed in turn.

Mouse operation

Action	Description
Click	Select and release the primary mouse button without moving the pointer.
Double-click	Press the primary mouse button twice continuously and quickly without moving the pointer.
Drag	Press and hold the primary mouse button and move the pointer to a certain position.

Change history

Updates between document versions are cumulative. Therefore, the latest document version contains all updates made to previous versions.

Issue 01 (2020-02-24)

Initial commercial release

Contents

Figure

Table

Preface

This preface includes the following sections:

Audience

This publication is for network administrators who install, configure, and maintain Inspur CN12904 and CN12908 switches.

Document Conventions

Command descriptions use the following conventions:

Convention	Description
bold	Bold text indicates the commands and keywords that you enter literally as shown.
Italic	Italic text indicates arguments for which the user supplies the values.
[x]	Square brackets enclose an optional element (keyword or argument).
[x | y]	Square brackets enclosing keywords or arguments separated by a vertical bar indicate an optional choice.
{x | y}	Braces enclosing keywords or arguments separated by a vertical bar indicate a required choice.
[x {y | z}]	Nested set of square brackets or braces indicate optional or required choices within optional or required elements. Braces and a vertical bar within square brackets indicate a required choice within an optional element.
`variable`	Indicates a variable for which you supply values, in context where italics cannot be used.
string	A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.

Examples use the following conventions:

Convention	Description
`screen font`	Terminal sessions and information the switch displays are in screen font.
`boldface screen font`	Information you must enter is in boldface screen font.
italic screen font	Arguments for which you supply values are in italic screen font.

Convention	Description
< >	Nonprinting characters, such as passwords, are in angle brackets.
[ ]	Default responses to system prompts are in square brackets.
!, #	An exclamation point (!) or a pound sign (#) at the beginning of a line of code indicates a comment line.

CHAPTER 1 New and Changed Information

1.1 New and Changed Information

This table summarizes the new and changed features for the *Inspur CN12900 Series INOS-CN Interfaces Configuration Guide* and where they are documented.

Table 1：New and Changed Features

Feature	Description	Changed in Release	Where Documented
Initial Release	Initial Release	9.2(1i)	
Breakout Support	Added breakout support for CN129-X636Q-R and CN129-X636C-R line cards.	9.2(2)	Notes About Breakout Interfaces.
CWDM4 Optics on 100G Interfaces	Added support for CWDM4 on CN129-X636C-R, CN129-X636Q-R and CN129-X6136YC-R line cards.	9.2(2)	Notes About Breakout Interfaces.

CHAPTER 2 Overview

2.1 About Interfaces

Inspur INOS-CN supports multiple configuration parameters for each of the interface types supported. Most of these parameters are covered in this guide but some are described in other documents.

The following table shows where to get further information on the parameters you can configure for an interface.

Table 2：Interface Parameters

Feature	Parameters	Further Information
Basic parameters	description, duplex, error disable, flow control, MTU, beacon	“Configuring Basic Interface Parameters”
Layer 3	medium, IPv4 and IPv6 addresses	“Configuring Layer 3 Interfaces”
Layer 3	bandwidth, delay, IP routing, VRFs	*Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide* *Inspur CN12900 Series INOS-CN* *Multicast Routing Configuration Guide*
Port Channels	channel group, LACP	“Configuring Port Channels”
Security	EOU	*Inspur CN12900 Series INOS-CN Security Configuration Guide*

2.1.1 Ethernet Interfaces

Access Ports

An access port carries traffic for one VLAN. This type of port is a Layer 2 interface only.

For more information on access ports, see the “Information About Access and Trunk Interfaces” section.

Routed Ports

A routed port is a physical port that can route IP traffic to another device. A routed port is a Layer 3 interface only.

For more information on routed ports, see the “Routed Interfaces” section.

2.1.2 Management Interface

You can use the management Ethernet interface to connect the device to a network for remote management using a Telnet client, the Simple Network Management Protocol (SNMP), or other management agents. The management port (mgmt0) is autosensing and operates in full-duplex mode at a speed of 10/100/1000 Mb/s.

For more information on the management interface, see the Inspur CN12900 Series INOS-CN Fundamentals

Configuration Guide. You will also find information on configuring the IP address and default IP routing for the management interface in this document.

2.1.3 Port-Channel Interfaces

A port channel is a logical interface that is an aggregation of multiple physical interfaces. You can bundle up to 32 individual links to physical ports into a port channel to improve bandwidth and redundancy. You can also use port channeling to load balance traffic across these channeled physical interfaces. For more information about port-channel interfaces, see the "Configuring Port Channels" section.

2.1.4 Subinterfaces

You can create virtual subinterfaces on a parent interface configured as a Layer 3 interface. A parent interface can be a physical port. Subinterfaces divide the parent interface into two or more virtual interfaces on which you can assign unique Layer 3 parameters such as IP addresses and dynamic routing protocols.

2.1.5 Loopback Interfaces

A virtual loopback interface is a virtual interface with a single endpoint that is always up. Any packet that is transmitted over a virtual loopback interface is immediately received by that interface. Loopback interfaces emulate a physical interface. For more information about subinterfaces, see the "Loopback Interfaces" section.

2.1.6 Breakout Interfaces

Inspur INOS-CN supports the breakout of high bandwidth 100G/40G interfaces at the module level or at the per-port level.

Module Level Breakout

For module level breakout, the **interface breakout** command splits the high bandwidth 100G/40G interface of a module into four 25G/10G interfaces. The module is reloaded and the configuration for the interface is removed when the command is executed.

The following is an example of the command:

```
switch# configure terminal
switch(config)# interface breakout module 1
Module will be reloaded. Are you sure you want to continue(yes/no)? yes
```

The **no interface breakout module** *module_number* command undoes the breakout configuration. It puts all interfaces of the module in 40G mode and deletes the configuration for the previous 10G interfaces.

Dynamic Breakout (Per-port Level Breakout)

For dynamic breakout (per-port level breakout), the **interface breakout** command splits high bandwidth 100G/40G ports into four 25G/10G breakout ports. The breakout ports are identified as **Ethernet** <*slot*>/<*front-panel-port*>/<*breakout-port*>. For example, per-port level breakout ports can be identified as

Ethernet 1/2/1, **Ethernet 1/2/2**, **Ethernet 1/2/3**, and **Ethernet 1/2/4**.

When one or more 40G interfaces of the module are broken-out at the per-port level, the configuration for the interfaces are removed when the command is executed.

The following is an example of configuring a breakout port:

```
switch(config)# interface breakout module 1 port 1 map
```

```
10g-4x switch(config)#
```

The following is an example of configuring a multiple breakout ports:

```
switch(config)# interface breakout module 1 port 1-4 map 10g-
4x switch(config)#
```

About the Lane Selector

The lane selector is a push button switch and 4 LEDs located on the Inspur switch (left side of front panel, labeled "LS"). The push button switch and LEDs are used to indicate the status of the ports. The lane selector is supported on Inspur CN12900 series switches.

By default, the LEDs indicate the link/activity status of a 1 x 40G configuration. When the ports are configured as 4 x 10G, you can access the link status of each individual 10G port with the lane selector.

By pressing the lane selector push button, the port LED shows the selected lane's link/activity status. The 1st time the push button is pressed, the first LED displays the status of the first port. Pressing the push button a 2nd time displays the status of the second port, and so on. You can display the status of each of the four ports by pressing the push button in this manner.

For example, if port 60 is configured as 4 x 10G, pressing the lane selector push button once displays the link status of 60/1/1. Pressing the push button a second time displays the link status of 60/1/2.

When you press the push button after displaying the status of the last port, all four of the LEDs should extinguish to indicate that the lane selector has returned to display the status for the default 1 x 40G configuration.

Notes About Breakout Interfaces

Beginning with Inspur INOS-CN, the 36-port 100-Gigabit Ethernet QSFP28 line cards (CN129-X636C-R) and 36-port 40-Gigabit Ethernet QSFP+ line cards (CN129-X636Q-R) provide 4x10-Gigabit support.

Beginning with Inspur INOS-CN Release 9.2(2), CWDM4 is supported on the 36-port 100-Gigabit Ethernet QSFP28 line cards (CN129-X636C-R), the 36-port 40-Gigabit Ethernet QSFP+ line cards (CN129-X636Q-R) and the 52-port 100-Gigabit QSFP28 line cards (CN129-X6136YC-R).

Beginning with Inspur INOS-CN Release 9.2(2), CN129-X636C-R line cards support breaking out 100G ports into 4x25 Gigabits. The default FEC mode on CN129-X636C-R for 25G is FC-FEC because RS-FEC is not supported. The CN129-X6136YC-R line card does not support breakout.

Caveats

- When a break-out port is configured as a part of a port-channel, you need to apply the configuration twice (after write-erase/reload), to ensure the effectiveness of the port-channel.
- Inspur CN12900 Series switches have 40G ports. When you breakout one of the 40G ports into 4x10G ports with a QSFP breakout cable, not all of the subinterfaces can be added to a port channel. The following error message is displayed:

```
switch# channel-group 99 mode active
command failed: port not compatible [Buffer boost]
```

Using the **force** keyword on the port-channel does not enable all the interfaces to be added to the port-channel even though the error message suggests using the keyword.

High Bandwidth Interfaces

The breakout of high bandwidth interfaces (module level or per-port level) are supported only on:

- The CN129-X636Q-R and CN129-X636C-R line cards on an Inspur CN12900 switch.

2.2 Virtual Device Contexts

Inspur INOS-CN can segment operating system and hardware resources into virtual device contexts (VDCs) that emulate virtual devices. The Inspur CN12900 Series switch does not support multiple VDCs. All switch resources are managed in the default VDC.

2.3 High Availability for Interfaces

Interfaces support stateful and stateless restarts. A stateful restart occurs on a supervisor switchover. After the switchover, Inspur INOS-CN applies the runtime configuration.

CHAPTER 3 Configuring Basic Interface Parameters

3.1 About the Basic Interface Parameters

3.1.1 Description

For the Ethernet and management interfaces, you can configure the description parameter to provide a recognizable name for the interface. Using a unique name for each interface allows you to quickly identify the interface when you are looking at a listing of multiple interfaces.

For information about setting the description parameter for port-channel interfaces, see the “Configuring a Port-Channel Description” section. For information about configuring this parameter for other interfaces, see the “Configuring the Description” section.

3.1.2 Beacon

The beacon mode allows you to identify a physical port by flashing its link state LED with a green light. By default, this mode is disabled. To identify the physical port for an interface, you can activate the beacon parameter for the interface.

For information about configuring the beacon parameter, see the “Configuring the Beacon Mode” section.

3.1.3 Error Disabled

A port is in the error-disabled (err-disabled) state when the port is enabled administratively (using the **no shutdown** command) but disabled at runtime by any process. For example, if UDLD detects a unidirectional link, the port is shut down at runtime. However, because the port is administratively enabled, the port status displays as err-disable. Once a port goes into the err-disable state, you must manually reenable it or you can configure a timeout value that provides an automatic recovery. By default, the automatic recovery is not configured, and by default, the err-disable detection is enabled for all causes.

When an interface is in the err-disabled state, use the **errdisable detect cause** command to find information about the error.

You can configure the automatic error-disabled recovery timeout for a particular error-disabled cause and configure the recovery period.

The **errdisable recovery cause** command provides an automatic recovery after 300 seconds.

You can use the **errdisable recovery interval** command to change the recovery period within a range of 30 to 65535 seconds. You can also configure the recovery timeout for a particular err-disable cause.

If you do not enable the error-disabled recovery for the cause, the interface stays in the error-disabled state until you enter the **shutdown** and **no shutdown** commands. If the recovery is enabled for a cause, the interface is brought out of the error-disabled state and allowed to retry operation once all the causes have timed out. Use the **show interface status err-disabled** command to display the reason behind the error.

3.1.4 Interface Status Error Policy

Inspur INOS-CN policy servers such as Access Control List (ACL) Manager and Quality of Service (QoS) Manager, maintain a policy database. A policy is defined through the command-line interface.

Policies are pushed when you configure a policy on an interface to ensure that policies that are pushed are consistent with the hardware policies. To clear the errors and to allow the policy programming to proceed with the running configuration, enter the **no shutdown** command. If the policy programming succeeds, the port is allowed to come up. If the policy programming fails, the configuration is inconsistent with the hardware policies and the port is placed in an error-disabled policy state. The error-disabled policy state remains and the information is stored to prevent the same port from being brought up in the future. This process helps to avoid unnecessary disruption to the system.

3.1.5 Port MTU Size

The maximum transmission unit (MTU) size specifies the maximum frame size that an Ethernet port can process. For transmissions to occur between two ports, you must configure the same MTU size for both ports. A port drops any frames that exceed its MTU size.

By default, each port has an MTU of 1500 bytes, which is the IEEE 802.3 standard for Ethernet frames. Larger MTU sizes are possible for more efficient processing of data with less overhead. The larger frames, called jumbo frames, can be up to 9216 bytes in size, which is also the default system jumbo MTU size.

On a Layer 3 interface, you can configure an MTU size between 576 and 9216 bytes.

For a Layer 2 port, you can configure an MTU size that is either the system default (1500 bytes) or the system jumbo MTU size (initially 9216 bytes).

For information about setting the MTU size, see the “Configuring the MTU Size” section.

3.1.6 Bandwidth

Ethernet ports have a fixed bandwidth of 1,000,000 Kb at the physical layer. Layer 3 protocols use a bandwidth value that you can set for calculating their internal metrics. The value that you set is used for informational purposes only by the Layer 3 protocols—it does not change the fixed bandwidth at the physical layer. For example, the Enhanced Interior Gateway Routing Protocol (EIGRP) uses the minimum path bandwidth to determine a routing metric, but the bandwidth at the physical layer remains at 1,000,000 Kb.

For information about configuring the bandwidth parameter for port-channel interfaces, see the “Configuring the Bandwidth and Delay for Informational Purposes” section. For information about configuring the bandwidth parameter for other interfaces, see the “Configuring the Bandwidth” section.

3.1.7 Throughput Delay

Specifying a value for the throughput-delay parameter provides a value used by Layer 3 protocols; it does not change the actual throughput delay of an interface. The Layer 3 protocols can use this value to make operating decisions. For example, the Enhanced Interior Gateway Routing Protocol (EIGRP) can use the delay setting to set a preference for one Ethernet link over another, if other parameters such as link speed are equal. The delay value that you set is in the tens of microseconds.

For information about configuring the bandwidth parameter for port-channel interfaces, see the “Configuring the Bandwidth and Delay for Informational Purposes” section. For information about configuring the throughput-delay parameter for other interfaces, see the “Configuring the Throughput Delay” section.

3.1.8 Administrative Status

The administrative-status parameter determines whether an interface is up or down. When an interface is administratively down, it is disabled and unable to transmit data. When an interface is administratively up, it is enabled and able to transmit data.

For information about configuring the administrative status parameter for port-channel interfaces, see the “Shutting Down and Restarting the Port-Channel Interface” section. For information about configuring the administrative-status parameter for other interfaces, see the “Shutting Down and Activating the Interface” section.

3.1.9 Unidirectional Link Detection Parameter

UDLD Overview

Unidirectional Link Detection (UDLD) protocol allows devices that are connected through fiber-optic or copper (for example, Category 5 cabling) Ethernet cables to monitor the physical configuration of the cables and detect when a unidirectional link exists. When a device detects a unidirectional link, UDLD shuts down the affected LAN port and alerts the user. Unidirectional links can cause a variety of problems.

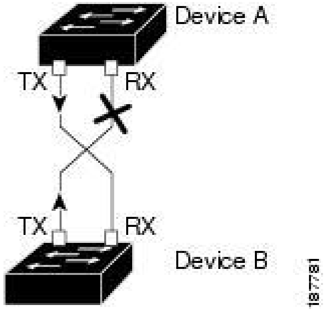
UDLD performs tasks that autonegotiation cannot perform, such as detecting the identities of neighbors and shutting down misconnected LAN ports. When you enable both autonegotiation and UDLD, Layer 1 detections work to prevent physical and logical unidirectional connections and the malfunctioning of other protocols.

A unidirectional link occurs whenever traffic transmitted by the local device over a link is received by the neighbor but traffic transmitted from the neighbor is not received by the local device. If one of the fiber strands in a pair is disconnected, as long as autonegotiation is active, the link does not stay up. In this case, the logical link is undetermined, and UDLD does not take any action. If both fibers are working normally at Layer 1, UDLD determines whether those fibers are connected correctly and whether traffic is flowing bidirectionally between the correct neighbors. This check cannot be performed by autonegotiation, because autonegotiation operates at Layer 1.

The Inspur CN12900 Series device periodically transmits UDLD frames to neighbor devices on LAN ports with UDLD enabled. If the frames are echoed back within a specific time frame and they lack a specific acknowledgment (echo), the link is flagged as unidirectional and the LAN port is shut down. Devices on both ends of the link must support UDLD in order for the protocol to successfully identify and disable unidirectional links. You can configure the transmission interval for the UDLD frames, either globally or for the specified interfaces.

The figure shows an example of a unidirectional link condition. Device B successfully receives traffic from device A on the port. However, device A does not receive traffic from device B on the same port. UDLD detects the problem and disables the port.

Figure 1：Unidirectional Link



Default UDLD Configuration

The following table shows the default UDLD configuration.

Table 3：UDLD Default Configuration

Feature	Default Value
UDLD global enable state	Globally disabled
UDLD per-port enable state for fiber-optic media	Enabled on all Ethernet fiber-optic LAN ports
UDLD per-port enable state for twisted-pair (copper) media	Disabled on all Ethernet 10/100 and 1000BASE-TX LAN ports
UDLD aggressive mode	Disabled

UDLD message interval	15 seconds

For information about configuring the UDLD for the device and its port, see the "Configuring the UDLD Mode" section.

UDLD Aggressive and Nonaggressive Modes

UDLD aggressive mode is disabled by default. You can configure UDLD aggressive mode only on point-to-point links between network devices that support UDLD aggressive mode. If UDLD aggressive mode is enabled, when a port on a bidirectional link that has a UDLD neighbor relationship established stops receiving UDLD frame, UDLD tries to reestablish the connection with the neighbor. After eight failed retries, the port is disabled.

When you enable the UDLD aggressive mode, the following occurs:
One side of a link has a port stuck (both transmission and receive)
One side of a link remains up while the other side of the link is down
In these cases, the UDLD aggressive mode disables one of the ports on the link, which prevents traffic from being discarded.

Tip	When a line card upgrade is being performed during an in-service software upgrade (ISSU) and some of the ports on the line card are members of a Layer 2 port channel and are configured with UDLD aggressive mode, if you shut down one of the remote ports, UDLD puts the corresponding port on the local device into an error-disabled state. This behavior is correct.

To restore service after the ISSU has completed, enter the **shutdown** command followed by the **no shutdown** command on the local port.

3.1.10 Port-Channel Parameters

A port channel is an aggregation of physical interfaces that comprise a logical interface. You can bundle up to 32 individual interfaces into a port channel to provide increased bandwidth and redundancy. Port channeling also load balances traffic across these physical interfaces. The port channel stays operational if at least one physical interface within the port channel is operational.

You can create Layer 3 port channels by bundling compatible Layer 3 interfaces.

Any configuration changes that you apply to the port channel are applied to each interface member of that port channel.

For information about port channels and for information about configuring port channels, see Chapter 6, "Configuring Port Channels."

3.1.11 Port Profiles

You can create a port profile that contains many interface commands and apply that port profile to a range of interfaces. Each port profile can be applied only to a specific type of interface; the choices are as follows:

- Ethernet
- VLAN network interface
- Port channel

When you choose Ethernet or port channel as the interface type, the port profile is in the default mode which is Layer 3. Enter the **switchport** command to change the port profile to Layer 2 mode.

You inherit the port profile when you attach the port profile to an interface or range of interfaces. When you attach, or inherit, a port profile to an interface or range of interfaces, the system applies all the commands in that port profile to the interfaces. Additionally, you can have one port profile inherit the settings from another port profile. Inheriting another port profile allows the initial port profile to assume all of the commands of the second, inherited, port profile that do not conflict with the initial port profile. Four levels of inheritance are supported. The same port profile can be inherited by any number of port profiles.

The system applies the commands inherited by the interface or range of interfaces according to the following guidelines:

- Commands that you enter under the interface mode take precedence over the port profile's commands if there is a conflict. However, the port profile retains that command in the port profile.
- The port profile's commands take precedence over the default commands on the interface, unless the port-profile command is explicitly overridden by the default command.
- When a range of interfaces inherits a second port profile, the commands of the initial port profile override the commands of the second port profile if there is a conflict.
- After you inherit a port profile onto an interface or range of interfaces, you can override individual configuration values by entering the new value at the interface configuration level. If you remove the individual configuration values at the interface configuration level, the interface uses the values in the port profile again.
- There are no default configurations associated with a port profile.

A subset of commands are available under the port-profile configuration mode, depending on which interface type you specify.

To apply the port-profile configurations to the interfaces, you must enable the specific port profile. You can configure and inherit a port profile onto a range of interfaces prior to enabling the port profile. You would then enable that port profile for the configurations to take effect on the specified interfaces.

If you inherit one or more port profiles onto an original port profile, only the last inherited port profile must be enabled; the system assumes that the underlying port profiles are enabled.

When you remove a port profile from a range of interfaces, the system undoes the configuration from the interfaces first and then removes the port-profile link itself. Also, when you remove a port profile, the system checks the interface configuration and either skips the port-profile commands that have been overridden by directly entered interface commands or returns the command to the default value.

If you want to delete a port profile that has been inherited by other port profiles, you must remove the inheritance before you can delete the port profile.

You can also choose a subset of interfaces from which to remove a port profile from among that group of interfaces that you originally applied the profile. For example, if you configured a port profile and configured ten interfaces to inherit that port profile, you can remove the port profile from just some of the specified ten interfaces. The port profile continues to operate on the remaining interfaces to which it is applied.

If you delete a specific configuration for a specified range of interfaces using the interface configuration mode, that configuration is also deleted from the port profile for that range of interfaces only. For example, if you have a channel group inside a port profile and you are in the interface configuration mode and you delete that port channel, the specified port channel is also deleted from the port profile as well.

Just as in the device, you can enter a configuration for an object in port profiles without that object being applied to interfaces yet. For example, you can configure a virtual routing and forward (VRF) instance without it being applied to the system. If you then delete that VRF and related configurations from the port profile, the system is unaffected.

After you inherit a port profile on an interface or range of interfaces and you delete a specific configuration value, that port-profile configuration is not operative on the specified interfaces.

If you attempt to apply a port profile to the wrong type of interface, the system returns an error.

When you attempt to enable, inherit, or modify a port profile, the system creates a checkpoint. If the port-profile configuration fails, the system rolls back to the prior configuration and returns an error. A port profile is never only partially applied.

3.2 Licensing Requirements

The following table shows the licensing requirements for this feature:

Product	**License Requirement**
Inspur INOS-CN	The basic interface parameters require no license. Any feature not included in a license package is bundled with the INOS-CN image and is provided at no extra charge to you.

3.3 Guidelines and Limitations

Basic interface parameters have the following configuration guidelines and limitations:

- MDIX is enabled by default on copper ports. It is not possible to disable it.
- **show** commands with the **internal** keyword are not supported.
- Fiber-optic Ethernet ports must use Inspur-supported transceivers. To verify that the ports are using Inspur -supported transceivers, use the **show interface transceivers** command. Interfaces with Inspur -supported transceivers are listed as functional interfaces.
- A port can be either a Layer 2 or a Layer 3 interface; it cannot be both simultaneously. By default, each port is a Layer 3 interface.

You can change a Layer 3 interface into a Layer 2 interface by using the **switchport** command. You can change a Layer 2 interface into a Layer 3 interface by using the **no switchport** command.

- You usually configure Ethernet port speed and duplex mode parameters to auto to allow the system to negotiate the speed and duplex mode between ports. If you decide to configure the port speed and duplex modes manually for these ports, consider the following:
- Before you configure the speed and duplex mode for an Ethernet or management interface, see the Default Settings section for the combinations of speeds and duplex modes that can be configured at the same time.
- If you set the Ethernet port speed to auto, the device automatically sets the duplex mode to auto.
- If you enter the **no speed** command, the device automatically sets both the speed and duplex parameters to auto (the **no speed** command produces the same results as the **speed auto** command).
- If you configure an Ethernet port speed to a value other than auto (for example, 1G, 10G, or 40G), you must configure the connecting port to match. Do not configure the connecting port to negotiate the speed.
- To configure speed, duplex, and automatic flow control for an Ethernet interface, you can use the **negotiate auto** command. To disable automatic negotiation, use the **no negotiate auto** command.

Caution	Changing the Ethernet port speed and duplex mode configuration might shut down and reenable the interface.

- For BASE-T copper ports, auto-negotiation is enabled even when fixed speed is configured.
- Using a regular expression to address a set of interfaces is supported with the **regex** command option. The **regex** command option is an extension that is available for all interface commands.

Example:

```
switch(config-if-range)# interface ethernet regex [2]/
switch(config-if-range)# where
  conf; interface Ethernet2/1-8 admin@switch%default
switch(config-if-range)# interface ethernet regex [1]/2[2-4]
switch(config-if-range)# where
  conf; interface Ethernet1/22-24      admin@switch%default
```

- The source-interface command option provides support for management applications to configure an IPv4 and/or IPv6 inband or outband source IP address for the copy command and other processes (such as tacacs, ntp, ping/ping6, icmp-error and traceroute).
- Configuration commands

ip *services* **source-interface** *interface* **vrf** *vrf name*

Examples:

```
• ip ftp source-interface ethernet 8/1 vrf management
• ip http source-interface loopback 1 vrf blue
• ip ssh source-interface ethernet ethernet 5/1
  /*This command executes in the VRF context.*/
• ip ping source-interface ethernet 8/1 vrf blue
• ip traceroute source-interface ethernet 8/1 vrf red
• ip icmp-errors source-interface ethernet 8/1
  /*This command executes in the VRF context.*/
```

- Show commands:

show ip copy *services* **source-interface** *interface* **vrf** *vrf name*

```
• show ip ftp source-interface ethernet 8/1 vrf management
• show ip http source-interface loopback 1 vrf blue
• show ip ssh source-interface ethernet ethernet
  5/1 /*This command executes in the VRF context.*/
• show ip ping source-interface ethernet 8/1 vrf blue
• show ip traceroute source-interface ethernet 8/1 vrf red
• show ip icmp-errors source-interface ethernet 8/1
  /*This command executes in the VRF context.*/
```

- Service commands:
  copy *service*://username@hostname/*path* **file source-interface** *interface name*

Examples:

```
• copy ftp://username@hostname/usr/local/bin file source-interface ethernet 8/1
• copy scp://username@hostname/usr/local/bin file source-interface ethernet 8/1
• copy tftp://username@hostname/usr/local/bin file source-interface ethernet 8/1
• copy http://username@hostname/usr/local/bin file source-interface ethernet 8/1
• copy sftp://username@hostname/usr/local/bin file source-interface ethernet 8/1
```

- Port profiles are supported on Inspur CN12900 Series switches.
- Auto-negotiation is not supported on 25-Gigabit Ethernet Transceiver.
- Autonegotiation is not supported on 25G breakout ports.

3.4 Default Settings

The following lists the default settings for the basic interface parameters.

Parameter	**Default**
Description	Blank
Beacon	Disabled
Bandwidth	Data rate of interface
Throughput delay	100 microseconds
Administrative status	Shutdown
MTU	1500 bytes
UDLD global	Globally disabled
UDLD per-port enable state for fiber-optic media	Enabled on all Ethernet fiber-optic LAN ports
UDLD per-port enable state for copper media	Disabled on all Ethernet 1G, 10G, or 40G LAN ports
UDLD message interval	Disabled
UDLD aggressive mode	Disabled
Error disable	Disabled
Error disable recovery	Disabled
Error disable recovery interval	300 seconds
Buffer-boost	Enabled

3.5 Configuring the Basic Interface Parameters

When you configure an interface, you must specify the interface before you can configure its parameters.

3.5.1 Specifying the Interfaces to Configure

Before you begin

Before you can configure the parameters for one or more interfaces of the same type, you must specify the type and the identities of the interfaces.

The following table shows the interface types and identities that you should use for specifying the Ethernet and management interfaces.

Table 4：nformation Needed to Identify an Interface for Configurations

Interface Type	Identity
Ethernet	I/O module slot numbers and port numbers on the module
Management	0 (for port 0)

The interface range configuration mode allows you to configure multiple interfaces with the same configuration parameters. After you enter the interface range configuration mode, all command parameters you enter are attributed to all interfaces within that range until you exit out of the interface range configuration mode.

You enter a range of interfaces using dashes (-) and commas (,). Dashes separate contiguous interfaces and commas separate noncontiguous interfaces. When you enter noncontiguous interfaces, you must enter the media type for each interface.

This example shows how to configure a contiguous interface range:

```
switch(config)# interface ethernet 2/29-30
switch(config-if-range)#
```

This example shows how to configure a noncontiguous interface range:

```
switch(config)# interface ethernet 2/29, ethernet 2/33, ethernet 2/35
switch(config-if-range)#
```

You can specify subinterfaces in a range only when the subinterfaces are on the same port, for example, 2/29.1-2. But you cannot specify the subinterfaces in a range of ports, for example, you cannot enter 2/29.2-2/30.2. You can specify two of the subinterfaces discretely, for example, you can enter 2/29.2, 2/30.2.

This example shows how to configure a a breakout cable:

```
switch(config)# interface ethernet 1/2/1
switch(config-if-range)#
```

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *interface* **Example:**	Specifies the interface that you are configuring. You can specify the interface type and identity. For an Ethernet port,

	Command or Action	Purpose

	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
	Example:
	`switch(config)# interface mgmt0`
	`switch(config-if)#`
		use **ethernet** *slot/port*. For the management interface, use **mgmt0**.
		Examples:
		• The 1st example shows how to specify the slot 2, port 1 Ethernet interface.
		• The 2nd example shows how to specify the management interface.
		Note You do not need to add a space between the interface type and identity (port or slot/port number) For example, for the Ethernet slot 4, port 5 interface, you can specify either “ethernet 4/5” or “ethernet4/5.” The management interface is either “mgmt0” or “mgmt 0.”
		When you are in the interface configuration mode, the commands that you enter configure the interface that you specified for this mode.

3.5.2 Configuring the Description

You can provide textual interface descriptions for the Ethernet and management interfaces.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface*
3. **description** *text*
4. **show interface** *interface*
5. **exit**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *interface*	Specifies the interface that you are configuring. You

	Example:
	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
	Example:
	`switch(config)# interface mgmt0`
	`switch(config-if)#`
		can specify the interface type and identity. For an Ethernet port, use **ethernet** *slot/port*. For the management interface, use **mgmt0**.
		Examples:
		• The 1st example shows how to specify the slot 2, port 1 Ethernet interface.
		• The 2nd example shows how to specify the management interface.

Step 3	**description** *text*
	Example:
	`switch(config-if)# description Ethernet port 3 on module 1`
	`switch(config-if)#`
		Specifies the description for the interface.

Step 4	**show interface** *interface* **Example:** `switch(config)# show interface ethernet 2/1`	(Optional) Displays the interface status, which includes the description parameter.

Step 5	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode.

Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to set the interface description to Ethernet port 24 on module 3:

```
switch# configure terminal
switch(config)# interface ethernet 3/24
switch(config-if)# description
server1 switch(config-if)#
```

The output of the **show interface eth** command is enhanced as shown in the following example:

```
Switch# show version
Software
BIOS: version 05.30
```

```
NOS-CN: version 9.2(2)
BIOS compile time: 07/27/2018
NOS-CN image file is: bootflash:///inos-cn.9.2.2.bin
NOS-CN compile time: 10/30/2018 9:00:00 [10/30/2018 19:09:52]

switch# show interface ethernet 6/36

Ethernet6/36 is up
admin state is up, Dedicated Interface
Hardware: 40000 Ethernet, address: 0022.bdf6.bf91 (bia
0022.bdf8.2bf3) Internet Address is 192.168.100.1/24
MTU 9216 bytes, BW 40000000 Kbit, DLY 10 usec
```

3.5.3 Configuring the Beacon Mode

You can enable the beacon mode for an Ethernet port to flash its LED to confirm its physical location.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/port*
3. [**no**] **beacon**
4. **show interface ethernet** *slot/port*
5. **exit**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *slot/port*
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an interface to configure, and enters interface configuration mode.

Step 3	[**no**] **beacon**
	Example:
	`switch(config)# beacon`
	`switch(config-if)#`
		Enables the beacon mode or disables the beacon mode. The default mode is disabled.

Step 4	**show interface ethernet** *slot/port*
	Example:
	`switch(config)# show interface ethernet 2/1`
	`switch(config-if)#`
		(Optional) Displays the interface status, which includes the beacon mode state.

Step 5	**exit**	Exits the interface mode.

	Command or Action	Purpose

	Example:
	`switch(config-if)# exit`
	`switch(config)#`	

Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to enable the beacon mode for the Ethernet port 3/1:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# beacon
switch(config-if)#
```

This example shows how to disable the beacon mode for the Ethernet port 3/1:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# no
beacon switch(config-if)#
```

This example shows how to configure the dedicated mode for Ethernet port 4/17 in the group that includes ports 4/17, 4/19, 4/21, and 4/23:

```
switch# configure terminal
switch(config)# interface ethernet 4/17, ethernet 4/19, ethernet 4/21, ethernet 4/23
switch(config-if)# shutdown
switch(config-if)# interface ethernet 4/17
switch(config-if)# no shutdown
switch(config-if)#
```

3.5.4 Configuring the Error-Disabled State

You can view the reason an interface moves to the error-disabled state and configure automatic recovery.

Enabling the Error-Disable Detection

You can enable error-disable detection in an application. As a result, when a cause is detected on an interface, the interface is placed in an error-disabled state, which is an operational state that is similar to the link-down state.

SUMMARY STEPS

1. **configure terminal**
2. **errdisable detect cause** {**acl-exception** | **all** | **link-flap** | **loopback**}
3. **shutdown**
4. **no shutdown**
5. **show interface status err-disabled**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**errdisable detect cause** {**acl-exception** | **all** | **link-flap** | **loopback**}
	Example:
	`switch(config)# errdisable detect cause all`
	`switch(config-if)#`
		Specifies a condition under which to place the interface in an error-disabled state. The default is enabled.

Step 3	**shutdown**
	Example:
	`switch(config-if)# shutdown`
	`switch(config)#`
		Brings the interface down administratively. To manually recover the interface from the error-disabled state, enter this command first.

Step 4	**no shutdown**
	Example:
	`switch(config-if)# no shutdown`
	`switch(config)#`
		Brings the interface up administratively and enables the interface to recover manually from the error-disabled state.

Step 5	**show interface status err-disabled** **Example:** `switch(config)# show interface status err-disabled`	(Optional) Displays information about error-disabled interfaces.
Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to enable the error-disabled detection in all cases:

```
switch(config)# errdisable detect cause all
switch(config)#
```

Enabling the Error-Disabled Recovery

You can specify the application to bring the interface out of the error-disabled state and retry coming up. It retries after 300 seconds, unless you configure the recovery timer (see the **errdisable recovery interval** command).

SUMMARY STEPS

1. **configure terminal**
2. **errdisable recovery cause** {**all** | **bpduguard** | **failed-port-state** | **link-flap** | **loopback** | **miscabling** | **psecure-violation** | **security-violation** | **storm-control** | **udld** | **vpc-peerlink**}
3. **show interface status err-disabled**
4. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** switch# **configure terminal** switch(config)#	Enters global configuration mode.
Step 2	**errdisable recovery cause** {**all** | **bpduguard** | **failed-port-state** | **link-flap** | **loopback** | **miscabling** | **psecure-violation** | **security-violation** | **storm-control** | **udld** | **vpc-peerlink**} **Example:** switch(config)# **errdisable recovery cause all** switch(config-if)#	Specifies a condition under which the interface automatically recovers from the error-disabled state, and the device retries bringing the interface up. The device waits 300 seconds to retry. The default is disabled.
Step 3	**show interface status err-disabled** **Example:** switch(config)# **show interface status err-disabled** switch(config-if)#	(Optional) Displays information about error-disabled interfaces.
Step 4	**copy running-config startup-config** **Example:** switch(config)# **copy running-config startup-config**	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to enable error-disabled recovery under all conditions:

```
switch(config)# errdisable recovery cause all
switch(config)#
```

Configuring the Error-Disabled Recovery Interval

You can configure the error-disabled recovery timer value.

SUMMARY STEPS

1. **configure terminal**
2. **errdisable recovery interval** *interval*
3. **show interface status err-disabled**
4. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**errdisable recovery interval** *interval*
	Example:
	`switch(config)# errdisable recovery interval 32`
	`switch(config-if)#`
		Specifies the interval for the interface to recover from the
		error-disabled state. The range is from 30 to 65535 seconds,
		and the default is 300 seconds.

Step 3	**show interface status err-disabled**
	Example:
	`switch(config)# show interface status err-disabled`
	`switch(config-if)#`
		(Optional) Displays information about error-disabled
		interfaces.

Step 4	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure the error-disabled recovery timer to set the interval for recovery to 32 seconds:

```
switch(config)# errdisable recovery interval 32
switch(config)#
```

3.5.5 Configuring the MTU Size

You can configure the maximum transmission unit (MTU) size for Layer 2 and Layer 3 Ethernet interfaces. For Layer 3 interfaces, you can configure the MTU to be between 576 and 9216 bytes (even values are required). For Layer 2 interfaces, you can configure the MTU to be either the system default MTU (1500 bytes) or the system jumbo MTU size (which has the default size of 9216 bytes).

By default, Inspur INOS-CN configures Layer 3 parameters. If you want to configure Layer 2 parameters, you need to switch the port mode to Layer 2.

You can change the port mode by using the **switchport** command.

After changing the port mode to Layer 2, you can return to configuring Layer 3 interfaces by changing the port mode again, by using the **no switchport** command.

Configuring the Interface MTU Size

For Layer 3 interfaces, you can configure an MTU size that is between 576 and 9216 bytes.

For Layer 2 interfaces, you can configure all Layer 2 interfaces to use either the default MTU size (1500 bytes) or the system jumbo MTU size (default size of 9216 bytes).

If you need to use a different system jumbo MTU size for Layer 2 interfaces, see the “Configuring the System Jumbo MTU Size” section.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/port*
3. [**switchport** | **no switchport**]
4. **mtu** *size*
5. **show interface ethernet** *slot/port*
6. **exit**
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *slot/port*
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an Ethernet interface to configure, and enters interface configuration mode.

Step 3	[**switchport** | **no switchport**]
	Example:
	`switch(config-if)# no switchport`
	`switch(config-if)#`
		Specifies to use Layer 3.

Step 4	**mtu** *size*
	Example:
	`switch(config-if)# mtu 9216`
	`switch(config-if)#`
		For a Layer 3 interface, specifies any even number between 576 and 9216.

Step 5	**show interface ethernet** *slot/port* **Example:**	(Optional) Displays the interface status, which includes the MTU size.

	Command or Action	Purpose
	`switch(config)#` **`show interface ethernet 2/1`**	

Step 6	**exit**
	Example:
	`switch(config-if)#` **`exit`**
	`switch(config)#`
		Exits the interface mode.

Step 7	**copy running-config startup-config** **Example:** `switch(config)#` **`copy running-config startup-config`**	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure the Layer 2 Ethernet port 3/1 with the default MTU size (1500):

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# switchport
switch(config-if)# mtu 1500
switch(config-if)#
```

Configuring the System Jumbo MTU Size

You can configure the system jumbo MTU size, which can be used to specify the MTU size for Layer 2 interfaces. You can specify an even number between 1500 and 9216. If you do not configure the system jumbo MTU size, it defaults to 9216 bytes.

SUMMARY STEPS

1. **configure terminal**
2. **system jumbomtu** *size*
3. **show running-config all**
4. **interface** *type slot/port*
5. **mtu** *size*
6. **exit**
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch#` **`configure terminal`**
	`switch(config)#`
		Enters global configuration mode.

Step 2	**system jumbomtu** *size*	Specifies the system jumbo MTU size. Use an even number

	Command or Action	Purpose
	Example: switch(config)# **system jumbomtu 8000** switch(config)#	between 1500 and 9216. **Note** In general accepted practice, a jumbo frame is considered to have an MTU size greater than 9000 bytes.
Step 3	**show running-config all** **Example:** switch(config)# **show running-config all | include jumbomtu**	(Optional) Displays the current operating configuration, which includes the system jumbo MTU size.
Step 4	**interface** *type slot/port* **Example:** switch(config)# **interface ethernet 2/1** switch(config-if)#	Specifies an interface to configure and enters interface configuration mode.
Step 5	**mtu** *size* **Example:** switch(config-if)# **mtu 1500** switch(config-if)#	For a Layer 2 interface, specifies either the default MTU size (1500) or the system jumbo MTU size that you specified earlier. For a Layer 3 interface, specifies any even size between 576 and 9216.
Step 6	**exit** **Example:** switch(config-if)# **exit** switch(config)#	Exits the interface mode.
Step 7	**copy running-config startup-config** **Example:** switch(config)# **copy running-config startup-config**	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure the system jumbo MTU as 8000 bytes and how to change the MTU specification for an interface that was configured with the previous jumbo MTU size:

```
switch# configure terminal
switch(config)# system jumbomtu 8000
switch(config)# show running-config
switch(config)# interface ethernet 2/2
switch(config-if)# switchport
switch(config-if)# mtu 1500
switch(config-if)#
```

3.5.6 Configuring the Bandwidth

You can configure the bandwidth for Ethernet interfaces. The physical layer uses an unchangeable bandwidth of 1G, 10G, or 40G, but you can configure a value of 1 to 100,000,000 KB for Level 3 protocols.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/port*
3. **bandwidth** *kbps*
4. **show interface ethernet** *slot/port*
5. **exit**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *slot/port*
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an Ethernet interface to configure, and enters interface configuration mode.

Step 3	**bandwidth** *kbps*
	Example:
	`switch(config-if)# bandwidth 1000000`
	`switch(config-if)#`
		Specifies the bandwidth as an informational-only value between 1 and 100,000,000.

Step 4	**show interface ethernet** *slot/port* **Example:** `switch(config)# show interface ethernet 2/1`	(Optional) Displays the interface status, which includes the bandwidth value.

Step 5	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode.

Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

	Command or Action	Purpose

Example

This example shows how to configure an informational value of 1,000,000 Kb for the Ethernet slot 3, port 1 interface bandwidth parameter:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# bandwidth
1000000 switch(config-if)#
```

3.5.7 Configuring the Throughput Delay

You can configure the interface throughput delay for Ethernet interfaces. The actual delay time does not change, but you can set an informational value between 1 and 16777215, where the value represents the number of tens of microseconds.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/port*
3. **delay** *value*
4. **show interface ethernet** *slot/port*
5. **exit**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

	Command or Action	**Purpose**

Step 2	**interface ethernet** *slot/port*
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an Ethernet interface to configure, and enters interface configuration mode.

Step 3	**delay** *value*
	Example:
	`switch(config-if)# delay 10000`
	`switch(config-if)#`
		Specifies the delay time in tens of microseconds. You can set an informational value range between 1 and 16777215 tens of microseconds.

Step 4	**show interface ethernet** *slot/port*	(Optional) Displays the interface status, which

	Command or Action	Purpose

	Example:
	`switch(config)#` **`show interface ethernet 3/1`**
	`switch(config-if)#`
		includes the
		throughput-delay time.

Step 5	**exit**
	Example:
	`switch(config-if)#` **`exit`**
	`switch(config)#`
		Exits the interface mode.

Step 6	**copy running-config startup-config** **Example:** `switch(config)#` **`copy running-config startup-config`**	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure the throughput-delay time so that one interface is preferred over another. A lower delay value is preferred over a higher value. In this example, Ethernet 7/48 is preferred over 7/47. The default delay for 7/48 is less than the configured value on 7/47, which is set for the highest value (16777215):

```
switch# configure terminal
switch(config)# interface ethernet
7/47 switch(config-if)# delay 16777215
switch(config-if)# ip address
192.168.10.1/24 switch(config-if)# ip router
eigrp 10 switch(config-if)# no shutdown
switch(config-if)# exit
switch(config)# interface ethernet 7/48
switch(config-if)# ip address
192.168.11.1/24 switch(config-if)# ip router
eigrp 10 switch(config-if)# no shutdown
switch(config-if)#
```

3.5.8 Shutting Down and Activating the Interface

You can shut down and restart Ethernet or management interfaces. When you shut down interfaces, they become disabled and all monitoring displays show them as being down. This information is communicated to other network servers through all dynamic routing protocols. When the interfaces are shut down, the interface is not included in any routing updates. To activate the interface, you must restart the device.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface*
3. **shutdown**
4. **show interface** *interface*
5. **no shutdown**
6. **show interface** *interface*

7. **exit**
8. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# `**`configure terminal`**
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *interface*
	Example:
	`switch(config)# `**`interface ethernet 2/1`**
	`switch(config-if)#`
	`switch(config)# `**`interface mgmt0`**
	`switch(config-if)#`
		Specifies the interface that you are configuring. You can specify the interface type and identity. For an Ethernet port, use *ethernet slot/port*. For the management interface, use *mgmt0*.
		Examples:
		• The 1st example shows how to specify the slot 2, port 1 Ethernet interface.
		• The 2nd example shows how to specify the management interface.

Step 3	**shutdown**
	Example:
	`switch(config-if)# `**`shutdown`**
	`switch(config-if)#`
		Disables the interface.

Step 4	**show interface** *interface*
	Example:
	`switch(config-if)# `**`show interface ethernet 2/1`**
	`switch(config-if)#`
		(Optional) Displays the interface status, which includes the administrative status.

Step 5	**no shutdown**
	Example:
	`switch(config-if)# `**`no shutdown`**
	`switch(config-if)#`
		Reenables the interface.

Step 6	**show interface** *interface*
	Example:
	`switch(config-if)# `**`show interface ethernet 2/1`**
	`switch(config-if)#`
		(Optional) Displays the interface status, which includes the administrative status.

	Command or Action	Purpose

Step 7	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode.

Step 8	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to change the administrative status for Ethernet port 3/1 from disabled to enabled:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# shutdown
switch(config-if)# no
shutdown switch(config-if)#
```

3.5.9 Configuring the UDLD Mode

You can configure normal unidirectional link detection (UDLD) modes for Ethernet interfaces on devices configured to run UDLD.

Before you can enable a UDLD mode for an interface, you must make sure that UDLD is already enabled on the device that includes the interface. UDLD must also be enabled on the other linked interface and its device.

The following table lists CLI details to enable and disable UDLD on different interfaces

Table 5：CLI Details to Enable or Disable UDLD on Different Interfaces

Description	Fiber port	Copper or Nonfiber port
Default setting	Enabled	Disabled
Enable UDLD command	**no udld disable**	**udld enable**
Disable UDLD command	**udld disable**	**no udld enable**

Before you begin

You must enable UDLD for the other linked port and its device.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **feature udld**
3. **udld message-time** *seconds*
4. **udld aggressive**

5. **interface ethernet** *slot/port*
6. **udld** [**enable** | **disable**]
7. **show udld** [**ethernet** *slot/port* | **global** | **neighbors**]
8. **exit**
9. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch#` **`configure terminal`**
	`switch(config)#`
		Enters global configuration mode.

Step 2	[**no**] **feature udld**
	Example:
	`switch(config)#` **`feature udld`**
	`switch(config)#`
	`switch(config)#` **`no feature udld`**
	`switch(config)#`
		Enables/Disables UDLD for the device.

Step 3	**udld message-time** *seconds*
	Example:
	`switch(config)#` **`udld message-time 30`**
	`switch(config)#`
		(Optional) Specifies the interval between sending UDLD messages. The range is from 7 to 90 seconds, and the default is 15 seconds.

Step 4	**udld aggressive**
	Example:
	`switch(config)#` **`udld aggressive`**
	`switch(config)#`
		Optional) Specifies UDLD mode to be aggressive.
		Note For copper interfaces, you enter the interface command mode for those interfaces you want to configure for UDLD aggressive mode and issue this command in interface command model.

Step 5	**interface ethernet** *slot/port*
	Example:
	`switch(config)#` **`interface ethernet 3/1`**
	`switch(config-if)#`
		(Optional) Specifies an interface to configure, and enters interface configuration mode.

Step 6	**udld** [**enable** | **disable**] **Example:** `switch(config-if)#` **`udld enable`**	(Optional) Enables UDLD on the specified copper port or disables UDLD on the specified fiber port. To enable UDLD on copper ports, enter the **udld enable**

	Command or Action	Purpose
	`switch(config-if)#`	command. To enable UDLD on fiber ports, enter the **no** **udld disable** command.

Step 7	**show udld** [**ethernet** *slot/port* | **global** | **neighbors**]
	Example:
	`switch(config)#` **`show udld`**
	`switch(config)#`
		(Optional) Displays the UDLD status.

Step 8	**exit**
	Example:
	`switch(config-if-range)#` **`exit`**
	`switch(config)#`
		Exits the interface mode.

Step 9	**copy running-config startup-config** **Example:** `switch(config)#` **`copy running-config startup-config`**	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to enable the UDLD for the device:

```
switch# configure terminal
switch(config)# feature
udld switch(config)#
```

This example shows how to set the UDLD message interval to 30 seconds:

```
switch# configure terminal
switch(config)# feature udld
switch(config)# udld message-time 30
switch(config)#
```

This example shows how to disable UDLD for Ethernet port 3/1:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if-range)# no udld
enable switch(config-if-range)# exit
```

This example shows how to disable UDLD for the device:

```
switch# configure terminal
switch(config)# no feature
udld switch(config)# exit
```

3.5.10 Configuring the Debounce Timer

You can enable the debounce timer for Ethernet ports by specifying a debounce time (milliseconds) or disable the timer by specifying a debounce time of 0.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/port*
3. **link debounce time** *time*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *slot/port*
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an Ethernet interface to configure, and enters
		interface configuration mode.

Step 3	**link debounce time** *time*
	Example:
	`switch(config-if)# link debounce time 1000`
	`switch(config-if)#`
		Enables the debounce timer for the specified time (1 to 5000
		milliseconds).
		If you specify 0 milliseconds, the debounce timer is
		disabled.

Example

- The following example enables the debounce timer and sets the debounce time to 1000 milliseconds for an Ethernet interface:

```
switch# configure terminal switch(config)#
interface ethernet 1/4 switch(config-if)#
link debounce time 1000
```

- The following example disables the debounce timer for an Ethernet interface:

```
switch# configure terminal
switch(config)# interface ethernet 1/4
switch(config-if)# link debounce time 0
```

3.5.11 Configuring Port Profiles

You can apply several configuration parameters to a range of interfaces simultaneously. All the interfaces in the range must be the same type. You can also inherit the configurations from one port profile into another port profile. The system supports four levels of inheritance.

Creating a Port Profile

You can create a port profile on the device. Each port profile must have a unique name across types and the network.

SUMMARY STEPS

1. **configure terminal**
2. **port-profile** [**type** {**ethernet** | **interface-vlan** | **port-channel**}] *name*
3. **exit**
4. (Optional) **show port-profile**
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal**	Enters the global configuration mode.
Step 2	**port-profile** [**type** {**ethernet** | **interface-vlan** | **port-channel**}] *name*	Creates and names a port profile for the specified type of interface and enters the port-profile configuration mode.
Step 3	**exit**	Exits the port-profile configuration mode.
Step 4	(Optional) **show port-profile**	Displays the port-profile configuration.
Step 5	(Optional) **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to create a port profile named test for ethernet interfaces:

```
switch# configure terminal
switch(config)# port-profile type ethernet test
switch(config-ppm)#
```

Entering Port-Profile Configuration Mode and Modifying a Port Profile

You can enter the port-profile configuration mode and modify a port profile. To modify the port profile, you must be in the port-profile configuration mode.

SUMMARY STEPS

1. **configure terminal**
2. **port-profile** [**type** {**ethernet** | **interface-vlan** | **port-channel**}] *name*
3. **exit**
4. (Optional) **show port-profile**
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal**	Enters the global configuration mode.
Step 2	**port-profile** [**type** {**ethernet** | **interface-vlan** | **port-channel**}] *name*	Enters the port-profile configuration mode for the specified port profile and allows you to add or remove

	Command or Action	Purpose
		configurations to the profile.
Step 3	**exit**	Exits the port-profile configuration mode.
Step 4	(Optional) **show port-profile**	Displays the port-profile configuration.
Step 5	(Optional) **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to enter the port-profile configuration mode for the specified port profile and bring all the interfaces administratively up:

```
switch# configure terminal
switch(config)# port-profile type ethernet test
switch(config-ppm)# no shutdown
switch(config-ppm)#
```

Assigning a Port Profile to a Range of Interfaces

You can assign a port profile to an interface or to a range of interfaces. All the interfaces must be the same type.

SUMMARY STEPS

1. **configure terminal**
2. **interface** [**ethernet** *slot/port* | **interface-vlan** *vlan-id* | **port-channel** *number*]
3. **inherit port-profile** *name*
4. **exit**
5. (Optional) **show port-profile**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal**	Enters the global configuration mode.
Step 2	**interface** [**ethernet** *slot/port* | **interface-vlan** *vlan-id* | **port-channel** *number*]	Selects the range of interfaces.
Step 3	**inherit port-profile** *name*	Assigns the specified port profile to the selected interfaces.
Step 4	**exit**	Exits the port-profile configuration mode.
Step 5	(Optional) **show port-profile**	Displays the port-profile configuration.
Step 6	(Optional) **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to assign the port profile named adam to Ethernet interfaces 7/3 to 7/5, 10/2, and 11/20 to 11/25:

```
switch# configure terminal
switch(config)# interface ethernet7/3-5, ethernet10/2,
ethernet11/20-25 switch(config-if)# inherit port-profile adam
switch(config-if)#
```

Enabling a Specific Port Profile

To apply the port-profile configurations to the interfaces, you must enable the specific port profile. You can configure and inherit a port profile onto a range of interfaces before you enable that port profile. You would then enable that port profile for the configurations to take effect on the specified interfaces.

If you inherit one or more port profiles onto an original port profile, only the last inherited port profile must be enabled; the system assumes that the underlying port profiles are enabled.

You must be in the port-profile configuration mode to enable or disable port profiles.

SUMMARY STEPS

1. **configure terminal**
2. **port-profile** [**type** {**ethernet** | **interface-vlan** | **port-channel**}] *name*
3. **state enabled**
4. **exit**
5. (Optional) **show port-profile**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal**	Enters the global configuration mode.
Step 2	**port-profile** [**type** {**ethernet** | **interface-vlan** | **port-channel**}] *name*	Creates and names a port profile for the specified type of interface and enters the port-profile configuration mode.
Step 3	**state enabled**	Enables that port profile.
Step 4	**exit**	Exits the port-profile configuration mode.
Step 5	(Optional) **show port-profile**	Displays the port-profile configuration.
Step 6	(Optional) **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to enter the port-profile configuration mode and enable the port profile:

```
switch# configure terminal
switch(config)# port-profile type ethernet test
switch(config-ppm)# state
enabled switch(config-ppm)#
```

Inheriting a Port Profile

You can inherit a port profile onto an existing port profile. The system supports four levels of inheritance.

SUMMARY STEPS

1. **configure terminal**
2. **port-profile** *name*
3. **inherit port-profile** *name*
4. **exit**
5. (Optional) **show port-profile**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal**	Enters the global configuration mode.
Step 2	**port-profile** *name*	Enters the port-profile configuration mode for the specified port profile.
Step 3	**inherit port-profile** *name*	Inherits another port profile onto the existing one. The original port profile assumes all the configurations of the inherited port profile.
Step 4	**exit**	Exits the port-profile configuration mode.
Step 5	(Optional) **show port-profile**	Displays the port-profile configuration.
Step 6	(Optional) **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to inherit the port profile named adam onto the port profile named test:

```
switch# configure terminal
switch(config)# port-profile test
switch(config-ppm)# inherit port-profile adam
switch(config-ppm)#
```

Removing a Port Profile from a Range of Interfaces

You can remove a port profile from some or all of the interfaces to which you have applied the profile. You do this configuration in the interfaces configuration mode.

SUMMARY STEPS

1. **configure terminal**
2. **interface** [**ethernet** *slot/port* | **interface-vlan** *vlan-id* | **port-channel** *number*]
3. **no inherit port-profile** *name*

4. **exit**
5. (Optional) **show port-profile**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal**	Enters the global configuration mode.
Step 2	**interface** [**ethernet** *slot/port* | **interface-vlan** *vlan-id* | **port-channel** *number*]	Selects the range of interfaces.
Step 3	**no inherit port-profile** *name*	Un-assigns the specified port profile to the selected interfaces.
Step 4	**exit**	Exits the port-profile configuration mode.
Step 5	(Optional) **show port-profile**	Displays the port-profile configuration.
Step 6	(Optional) **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to unassign the port profile named adam to Ethernet interfaces 7/3 to 7/5, 10/2, and 11/20 to 11/25:

```
switch# configure terminal
switch(config)# interface ethernet 7/3-5, 10/2, 11/20-
25 switch(config-if)# no inherit port-profile adam
switch(config-if)#
```

Removing an Inherited Port Profile

You can remove an inherited port profile. You do this configuration in the port-profile mode.

SUMMARY STEPS

1. **configure terminal**
2. **port-profile** *name*
3. **no inherit port-profile** *name*
4. **exit**
5. (Optional) **show port-profile**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal**	Enters the global configuration mode.
Step 2	**port-profile** *name*	Enters the port-profile configuration mode for the specified port profile.

	Command or Action	Purpose
Step 3	**no inherit port-profile** *name*	Removes an inherited port profile from this port profile.
Step 4	**exit**	Exits the port-profile configuration mode.
Step 5	(Optional) **show port-profile**	Displays the port-profile configuration.
Step 6	(Optional) **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to remove the inherited port profile named adam from the port profile named test:

```
switch# configure terminal
switch(config)# port-profile test
switch(config-ppm)# no inherit port-profile adam
switch(config-ppm)#
```

3.5.12 Configuring 25G Autonegotiation

Autonegotiation allows devices to advertise enhanced modes of operation it possesses via the link segment and to detect corresponding enhanced operational modes that the other devices may be advertising. Autonegotiation provides the means to exchange information between two devices that share a link segment and to automatically configure both devices to take maximum advantage of their abilities.

Guidelines and Limitations for 25G Autonegotiation

- Autonegotiation is not supported on 25G breakout ports.

Enabling Autonegotiation

You can enable autonegotiation using the *negotiate auto* command. To enable autonegotiation, follow these steps:

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *port number*
3. **negotiate auto** *port speed*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *port number* **Example:** `switch# int e1/7`	Selects the interface and enters interface mode.

	Command or Action	Purpose
	`switch(config-if)#`	

Step 3	**negotiate auto** *port speed*
	Example:
	`switch(config-if)#` **`negotiate auto 25000`**
	`switch(config-if)#`
		Enables autonegotiation on the selected interface.
		Note You must apply this command on interfaces at both sides of the 25G native link.

This example shows how to enable autonegotiation on a specified interface:

Example

```
switch# sh int e1/7 st
--------------------------------------------------------------------------------
Port          Name               Status     Vlan      Duplex  Speed   Type
--------------------------------------------------------------------------------
Eth1/7 -- connected routed full 25G SFP-H25GB-CU1M switch# conf
switch(config)# int e1/7
switch(config-if)# negotiate auto
25000
```

Disabling Autonegotiation

You can disable autonegotiation using the *no negotiate auto* command. To disable autonegotiation, follow these steps:

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *port number*
3. **no negotiate auto** *port speed*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch#` **`configure terminal`**
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *port number*
	Example:
	`switch#` **`int e1/7`**
	`switch(config-if)#`
		Selects the interface and enters interface mode.

Step 3	**no negotiate auto** *port speed*
	Example:
	`switch(config-if)#` **`no negotiate auto 25000`**
	`switch(config-if)#`
		Disables autonegotiation on the selected interface.
		Note You must apply this command on interfaces at both sides of the link.

This example shows how to disable autonegotiation on a specified interface.

Example

```
switch# sh int e1/7 st
--------------------------------------------------------------------------------
Port          Name              Status    Vlan      Duplex  Speed   Type
--------------------------------------------------------------------------------
Eth1/7 -- connected routed full 25G SFP-H25GB-CU1M switch# conf
switch(config)# int e1/7 switch(config-
if)# no negotiate auto 25000
```

3.6 Verifying the Basic Interface Parameters

You can verify the basic interface parameters by displaying their values. You can also clear the counters listed when you display the parameter values.

To display basic interface configuration information, perform one of the following tasks:

Command	Purpose
show cdp all	Displays the CDP status.
show interface *interface*	Displays the configured states of one or all interfaces.
show interface brief	Displays a table of interface states.
show interface status err-disabled	Displays information about error-disabled interfaces.
show udld *interface*	Displays the UDLD status for the current interface or all interfaces.
show udld global	Displays the UDLD status for the current device.

3.7 Monitoring the Interface Counters

You can display and clear interface counters using Inspur INOS-CN.

3.7.1 Displaying Interface Statistics

You can set up to three sampling intervals for statistics collections on interfaces.

SUMMARY STEPS

1. **configure terminal**
2. **interface ether** *slot/port*
3. **load-interval counters** [**1** | **2** | **3**] *seconds*
4. **show interface** *interface*
5. **exit**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ether** *slot/port*
	Example:
	`switch(config)# interface ether 4/1`
	`switch(config)#`
		Specifies interface.

Step 3	**load-interval counters** [**1** | **2** | **3**] *seconds*
	Example:
	`switch(config)# load-interval counters 1 100`
	`switch(config)#`
		Sets up to three sampling intervals to collect bit-rate and packet-rate statistics. The default values for each counter is as follows:
		1—30 seconds (60 seconds for VLAN)
		2—300 seconds
		3—not configured

Step 4	**show interface** *interface*
	Example:
	`switch(config)# show interface ethernet 2/2`
	`switch#`
		(Optional) Displays the interface status, which includes the counters.

Step 5	**exit**
	Example:
	`switch(config-if-range)# exit`
	`switch(config)#`
		Exits the interface mode.

Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config` **Example**	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to set the three sample intervals for the Ethernet port 3/1:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# load-interval counter 1 60
switch(config-if)# load-interval counter 2 135
switch(config-if)# load-interval counter 3 225
switch(config-if)#
```

3.7.2 Clearing Interface Counters

You can clear the Ethernet and management interface counters by using the **clear counters interface** command. You can perform this task from the configuration mode or interface configuration mode.

SUMMARY STEPS

1. **clear counters interface** [**all** | **ethernet** *slot/port* | **loopback** *number* | **mgmt** *number* | **port channel** *channel-number*]
2. **show interface** *interface*
3. **show interface** [**ethernet** *slot/port* | **port channel** *channel-number*] **counters**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**clear counters interface** [**all** | **ethernet** *slot/port* | **loopback** *number* | **mgmt** *number* | **port channel** *channel-number*]
	Example:
	`switch# clear counters ethernet 2/1`
	`switch#`
		Clears the interface counters.

Step 2	**show interface** *interface*
	Example:
	`switch# show interface ethernet 2/1`
	`switch#`
		(Optional) Displays the interface status.

Step 3	**show interface** [**ethernet** *slot/port* | **port channel** *channel-number*] **counters**
	Example:
	`switch# show interface ethernet 2/1 counters`
	`switch#`
		(Optional) Displays the interface counters.

Example

This example shows how to clear the counters on Ethernet port 5/5:

```
switch# clear counters interface
ethernet 5/5 switch#
```

CHAPTER 4 Configuring Layer 2 Interfaces

4.1 Information About Access and Trunk Interfaces

4.1.1 About Access and Trunk Interfaces

A Layer 2 port can be configured as an access or a trunk port as follows:

- An access port can have only one VLAN configured on that port; it can carry traffic for only one VLAN.
- A trunk port can have two or more VLANs configured on that port; it can carry traffic for several VLANs simultaneously.

By default, all ports on the device are Layer 3 ports.

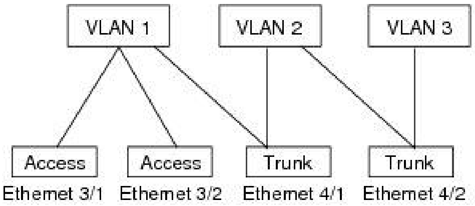
You can make all ports Layer 2 ports using the setup script or by entering the **system default switchport**command. See the Inspur CN12900 Series INOS-CN Fundamentals Configuration Guide for

information about using the setup script. To configure the port as a Layer 2 port using the CLI, use the **switchport** command.

All ports in the same trunk must be in the same VDC, and trunk ports cannot carry VLANs from different VDCs.

The following figure shows how you can use trunk ports in the network. The trunk port carries traffic for two or more VLANs.

Figure 2： Trunk and Access Ports and VLAN Traffic



In order to correctly deliver the traffic on a trunk port with several VLANs, the device uses the IEEE 802.1Q encapsulation, or tagging, method (see the “IEEE 802.1Q Encapsulation” section for more information).

To optimize the performance on access ports, you can configure the port as a host port. Once the port is configured as a host port, it is automatically set as an access port, and channel grouping is disabled. Use the host designation to decrease the time that it takes the designated port to begin to forward packets.

Only an end station can be set as a host port; you will receive an error message if you attempt to configure other ports as hosts.

If an access port receives a packet with an 802.1Q tag in the header other than the access VLAN value, that port drops the packet without learning its MAC source address.

A Layer 2 interface can function as either an access port or a trunk port; it cannot function as both port types simultaneously.

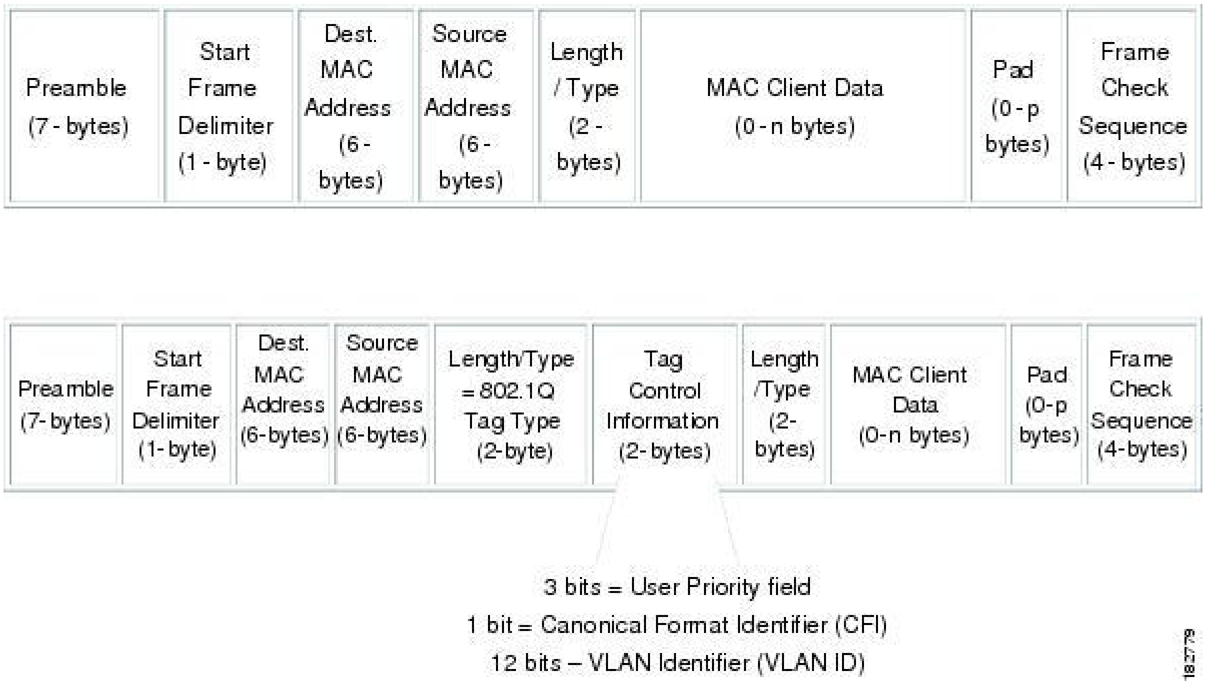
When you change a Layer 2 interface back to a Layer 3 interface, that interface loses all the Layer 2 configuration and resumes the default VLAN configurations.

4.1.2 IEEE 802.1Q Encapsulation

A trunk is a point-to-point link between the switch and another networking device. Trunks carry the traffic of multiple VLANs over a single link and allow you to extend VLANs across an entire network.

To correctly deliver the traffic on a trunk port with several VLANs, the device uses the IEEE 802.1Q encapsulation, or tagging, method that uses a tag that is inserted into the frame header. This tag carries information about the specific VLAN to which the frame and packet belong. This method allows packets that are encapsulated for several different VLANs to traverse the same port and maintain traffic separation between the VLANs. Also, the encapsulated VLAN tag allows the trunk to move traffic end-to-end through the network on the same VLAN.

Figure 3：Header Without and With 802.1Q Tag



Preamble (7 - bytes)	Start Frame Delimiter (1 - byte)	Dest. MAC Address (6 - bytes)	Source MAC Address (6 - bytes)	Length / Type (2 - bytes)	MAC Client Data (0 - n bytes)	Pad (0 - p bytes)	Frame Check Sequence (4 - bytes)

Preamble (7-bytes)	Start Frame Delimiter (1-byte)	Dest. MAC Address (6-bytes)	Source MAC Address (6-bytes)	Length/Type = 802.1Q Tag Type (2-byte)	Tag Control Information (2-bytes)	Length /Type (2-bytes)	MAC Client Data (0-n bytes)	Pad (0-p bytes)	Frame Check Sequence (4-bytes)

3 bits = User Priority field
1 bit = Canonical Format Identifier (CFI)
12 bits – VLAN Identifier (VLAN ID)

4.1.3 Access VLANs

When you configure a port in access mode, you can specify which VLAN will carry the traffic for that interface. If you do not configure the VLAN for a port in access mode, or an access port, the interface carries traffic for the default VLAN (VLAN1).

You can change the access port membership in a VLAN by specifying the new VLAN. You must create the VLAN before you can assign it as an access VLAN for an access port. If you change the access VLAN on an access port to a VLAN that is not yet created, the system shuts that access port down.

If an access port receives a packet with an 802.1Q tag in the header other than the access VLAN value, that port drops the packet without learning its MAC source address.

4.1.4 Native VLAN IDs for Trunk Ports

A trunk port can carry nontagged packets simultaneously with the 802.1Q tagged packets. When you assign a default port VLAN ID to the trunk port, all untagged traffic travels on the default port VLAN ID for the trunk port, and all untagged traffic is assumed to belong to this VLAN. This VLAN is referred to as the native

VLAN ID for a trunk port. That is, the native VLAN ID is the VLAN that carries untagged traffic on trunk ports.

The trunk port sends an egressing packet with a VLAN that is equal to the default port VLAN ID as untagged; all the other egressing packets are tagged by the trunk port. If you do not configure a native VLAN ID, the trunk port uses the default VLAN.

4.1.5 Tagging Native VLAN Traffic

The Inspur software supports the IEEE 802.1Q standard on trunk ports. In order to pass untagged traffic through the trunk ports, you must create a VLAN that does not tag any packets (or you can use the default VLAN). Untagged packets can pass through trunk ports and access ports.

However, all packets that enter the device with an 802.1Q tag that matches the value of the native VLAN on the trunk are stripped of any tagging and egress the trunk port as untagged packets. This situation can cause problems because you may want to retain the tagging on packets on the native VLAN for the trunk port.

You can configure the device to drop all untagged packets on the trunk ports and to retain the tagging of packets entering the device with 802.1Q values that are equal to that of the native VLAN ID. All control traffic still passes on the native VLAN. This configuration is global; trunk ports on the device either do or do not retain the tagging for the native VLAN.

4.1.6 Allowed VLANs

By default, a trunk port sends traffic to and receives traffic from all VLANs. All VLAN IDs are allowed on each trunk. However, you can remove VLANs from this inclusive list to prevent traffic from the specified VLANs from passing over the trunk. Later, you can add any specific VLANs that you may want the trunk to carry traffic for back to the list.

To partition the Spanning Tree Protocol (STP) topology for the default VLAN, you can remove VLAN1 from the list of allowed VLANs. Otherwise, VLAN1, which is enabled on all ports by default, will have a very big STP topology, which can result in problems during STP convergence. When you remove VLAN1, all data traffic for VLAN1 on this port is blocked, but the control traffic continues to move on the port.

4.1.7 Enabling 4K VLAN Configurations with switchport isolated

The switchport isolated feature allows disabling STP on an interface. Using this feature allows a maximum number of 4K VLAN * 48 virtual ports. Configuring the switchport isolated feature places all 4K VLANS in forwarding state for that port. (Removing a VLAN does not bring down the logical port.)

The feature is supported on MSTP modes. It is also supported on physical interfaces and port-channels (including vPC).

In vPC configurations, the Type-1 consistency check is performed between vPC peers. If the check result is inconsistent, the secondary vPC is brought down while the primary continues to stay up.

4.1.8 Default Interfaces

You can use the default interface feature to clear the configured parameters for both physical and logical interfaces such as the Ethernet, loopback, VLAN network, tunnel, and the port-channel interface.

4.1.9 Switch Virtual Interface and Autostate Behavior

In Inspur INOS-CN, a switch virtual interface (SVI) represents a logical interface between the bridging function and the routing function of a VLAN in the device.

The operational state of this interface is governed by the state of the various ports in its corresponding VLAN. An SVI interface on a VLAN comes up when at least one port in that VLAN is in the Spanning Tree Protocol (STP) forwarding state. Similarly, this interface goes down when the last STP forwarding port goes down or goes to another STP state.

SVI Autostate Exclude

Typically, when a VLAN interface has multiple ports in the VLAN, the SVI goes to the down state when all the ports in the VLAN go down. You can use the SVI autostate exclude feature to exclude specific ports and port channels while defining the status of the SVI (up or down) even if it belongs to the same VLAN. For example, even if the excluded port or port channel is in the up state and other ports are in the down state in the VLAN, the SVI state is changed to down.

SVI Autostate Disable

You can configure the autostate disable feature to keep an SVI up even if no interface is up in the corresponding VLAN. You can configure this feature for the system (for all SVIs) or for an individual SVI.

4.1.10 High Availability

The software supports high availability for Layer 2 ports.

4.1.11 Virtualization Support

All ports in the same trunk must be in the same VDC, and trunk ports cannot carry VLANs from different VDCs.

4.1.12 Counter Values

See the following information on the configuration, packet size, incremented counter values, and traffic.

Configuration	Packet Size	Incremented Counters	Traffic
L2 port – without any MTU configuration	6400 and 10000	Jumbo, giant, and input error	Dropped
L2 port – with jumbo MTU 9216 in network-qos configuration	6400	Jumbo	Forwarded
L2 port – with jumbo MTU 9216 in network-qos configuration	10000	Jumbo, giant, and input error	Dropped
Layer 3 port with default Layer 3 MTU and jumbo MTU 9216 in network-qos configuration	6400	Jumbo	Packets are punted to the CPU (subjected to CoPP configs), get fragmented, and then they are forwarded by the software.
Layer 3 port with default Layer 3 MTU and jumbo MTU 9216 in network-qos configuration	6400	Jumbo	Packets are punted to the CPU (subjected to CoPP configs), get fragmented, and then they are forwarded by the software.
Layer 3 port with default Layer 3 MTU and jumbo MTU 9216 in network-qos configuration	10000	Jumbo, giant, and input error	Dropped
Layer 3 port with jumbo	6400	Jumbo	Forwarded without any

Configuration	Packet Size	Incremented Counters	Traffic
Layer 3 MTU and jumbo MTU 9216 in network-qos configuration			fragmentation.
Layer 3 port with jumbo Layer 3 MTU and jumbo MTU 9216 in network-qos configuration	10000	Jumbo, giant, and input error	Dropped
Layer 3 port with jumbo Layer 3 MTU and default L2 MTU configuration	6400 and 10000	Jumbo, giant, and input error	Dropped

4.2 Licensing Requirements for Layer 2 Port Modes

The following table shows the licensing requirements for this feature:

Product	License Requirement
Inspur INOS-CN	Layer 2 port modes require no license. Any feature not included in a license package is bundled with the Inspur INOS-CN system images and is provided at no extra charge to you.

4.3 Prerequisites for Layer 2 Interfaces

Layer 2 interfaces have the following prerequisites:

- You are logged onto the device.
- You must configure the port as a Layer 2 port before you can use the **switchport mode**command. By default, all ports on the device are Layer 3 ports. By default, all ports on the Inspur CN12904 and Inspur CN12908 devices are Layer 2 ports.

4.4 Guidelines and Limitations for Layer 2 Interfaces

VLAN trunking has the following configuration guidelines and limitations:

- Inspur CN12900 Series switches have the **vlan dot1q tag native** command enabled by default. This tags the native VLAN on the configured trunk ports. However, connected switches or third-party switches, probably would not have a similar configuration enabled. This could result in unexpected behaviors. Therefore, it is recommended to have the **vlan dot1q tag native** command disabled in case the connected switch does not have it configured.
- Auto-negotiation is not supported on Inspur CN12908 platform switches with CN129-X636C-R, CN129-X6136YC-R, and CN129-X636Q-R lines cards.
- Beginning with Inspur INOS-CN, the Inspur CN12908 platform switches with CN129-X6136YC-R line cards support 1 Gigabit speed on all 48 ports. However, because the auto negotiation is not supported, 1000BASE-T SFPs links comes up even the cable is removed.
- **show** commands with the **internal** keyword are not supported.

- ASE2 and ASE3 based Inspur CN12900 Series switches acting as transit switches do not preserve the inner tag for double-tagged packets.

The following CLI is mandatory only on LSE based Inspur CN12900 Series switches. For seamless packet forwarding and preservation of all VLAN tags on pure transit boxes in the SP cloud that have no Q-in-Q encapsulation or decapsulation requirement, configure the CLI command, **system dot1q-tunnel transit**. To remove the CLI, use **no system dot1q-tunnel transit** CLI command.

The caveats with the CLI that is executed on the switches are:

- L2 frames that egress out of the trunk ports are tagged even on the native VLAN on the port.
- Any other tunneling mechanism, for example, VXLAN and MPLS does not work with the CLI configured.
- A port can be either a Layer 2 or a Layer 3 interface; it cannot be both simultaneously.
- When you change a Layer 3 port to a Layer 2 port or a Layer 2 port to a Layer 3 port, all layer-dependent configuration is lost. When you change an access or trunk port to a Layer 3 port, all information about the access VLAN, native VLAN, allowed VLANs, and so forth, is lost.
- Do not connect devices with access links because access links may partition a VLAN.
- When connecting Inspur devices through an 802.1Q trunk, make sure that the native VLAN for an 802.1Q trunk is the same on both ends of the trunk link. If the native VLAN on one end of the trunk is different from the native VLAN on the other end, spanning tree loops might result.
- Disabling spanning tree on the native VLAN of an 802.1Q trunk without disabling spanning tree on every VLAN in the network can cause spanning tree loops. You must leave spanning tree enabled on the native VLAN of an 802.1Q trunk. If you cannot leave spanning tree enabled, you must disable spanning tree on every VLAN in the network. Make sure that your network has no physical loops before you disable spanning tree.
- When you connect two Inspur devices through 802.1Q trunks, the devices exchange spanning tree bridge protocol data units (BPDUs) on each VLAN allowed on the trunks. The BPDUs on the native VLAN of the trunk are sent untagged to the reserved IEEE 802.1D spanning tree multicast MAC address (01-80-C2-00-00-00). The BPDUs on all other VLANs on the trunk are sent tagged to the reserved Inspur Shared Spanning Tree (SSTP) multicast MAC address (01-00-0c-cc-cc-cd).
- Non- Inspur 802.1Q devices maintain only a single instance of spanning tree (the Mono Spanning Tree) that defines the spanning tree topology for all VLANs. When you connect a Inspur switch to a non- Inspur switch through an 802.1Q trunk, the Mono Spanning Tree of the non- Inspur switch and the native VLAN spanning tree of the Inspur switch combine to form a single spanning tree topology known as the Common Spanning Tree (CST).
- Because Inspur devices transmit BPDUs to the SSTP multicast MAC address on VLANs other than the native VLAN of the trunk, non- Inspur devices do not recognize these frames as BPDUs and flood them on all ports in the corresponding VLAN. Other Inspur devices connected to the non- Inspur 802.1Q cloud receive these flooded BPDUs. This BPDU reception allows Inspur switches to maintain a per-VLAN spanning tree topology across a cloud of non- Inspur 802.1Q devices. The non- Inspur 802.1Q cloud that separates the Inspur devices is treated as a single broadcast segment between all devices connected to the non- Inspur 802.1Q cloud through 802.1Q trunks.
- Make certain that the native VLAN is the same on all of the 802.1Q trunks that connect the Inspur devices to the non- Inspur 802.1Q cloud.
- If you are connecting multiple Inspur devices to a non- Inspur 802.1Q cloud, all of the connections must be through 802.1Q trunks. You cannot connect Inspur devices to a non- Inspur 802.1Q cloud through access ports because doing so places the access port on the Inspur device into the spanning tree “port inconsistent” state and no traffic will pass through the port.
- You can group trunk ports into port-channel groups, but all trunks in the group must have the same configuration. When a group is first created, all ports follow the parameters set for the first port to be added to the group. If you change the configuration of one of these parameters, the device propagates that setting to all

ports in the group, such as the allowed VLANs and the trunk status. For example, if one port in a port group ceases to be a trunk, all ports cease to be trunks.

- If you try to enable 802.1X on a trunk port, an error message appears, and 802.1X is not enabled. If you try to change the mode of an 802.1X-enabled port to trunk, the port mode is not changed.
- Only ingress unicast packet counters are supported for SVI counters.
- When MAC addresses are cleared on a VLAN with the clear mac address-table dynamic command, the dynamic ARP (Address Resolution Protocol) entries on that VLAN are refreshed.
- If a static ARP entry exists on the VLAN and no MAC address to port mapping is present, the supervisor may generate an ARP request to learn the MAC address. Upon learning the MAC address, the adjacency entry points to the correct physical port.
- Inspur INOS-CN does not support transparent bridging between two VLANs when one of the SVIs is on the Inspur CN12900 using the BIA MAC (burned-in MAC address). This occurs when the BIA MAC is shared between SVIs/VLANs. A MAC, different from the BIA MAC, can be configured under the SVI for transparent bridging to work properly.

4.5 Default Settings for Layer 2 Interfaces

The following table lists the default settings for device access and trunk port mode parameters.

Table 6： Default Access and Trunk Port Mode Parameters

Parameters	**Default**
Switchport mode	Access
Allowed VLANs	1 to 3967, 4048 to 4094
Access VLAN ID	VLAN1
Native VLAN ID	VLAN1
Native VLAN ID tagging	Disabled
Administrative state	Shut
SVI autostate	Enabled

4.6 Configuring Access and Trunk Interfaces

4.6.1 Guidelines for Configuring Access and Trunk Interfaces

All VLANs on a trunk must be in the same VDC.

4.6.2 Configuring a VLAN Interface as a Layer 2 Access Port

You can configure a Layer 2 port as an access port. An access port transmits packets on only one, untagged VLAN. You specify which VLAN traffic that the interface carries, which becomes the access VLAN. If you do not specify a VLAN for an access port, that interface carries traffic only on the default VLAN. The default VLAN is VLAN1.

The VLAN must exist before you can specify that VLAN as an access VLAN. The system shuts down an access port that is assigned to an access VLAN that does not exist.

Before you begin

Ensure that you are configuring a Layer 2 interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** {{*type slot/port*} | {**port-channel** *number*}}
3. **switchport mode** [**access** | **trunk**]
4. **switchport access vlan** *vlan-id*
5. **exit**
6. **show interface**
7. **no shutdown**
8. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** {{*type slot/port*} | {**port-channel** *number*}}
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an interface to configure, and enters interface configuration mode.

Step 3	**switchport mode** [**access** | **trunk**] **Example:** `switch(config-if)# switchport mode access`	Sets the interface as a nontrunking nontagged, single-VLAN Layer 2 interface. An access port can carry traffic in one VLAN only. By default, an access port carries traffic for VLAN1; to set the access port to carry traffic for a different VLAN, use the **switchport access vlan** command.
Step 4	**switchport access vlan** *vlan-id* **Example:** `switch(config-if)# switchport access vlan 5`	Specifies the VLAN for which this access port will carry traffic. If you do not enter this command, the access port carries traffic on VLAN1 only; use this command to change the VLAN for which the access port carries traffic.
Step 5	**exit** **Example:** `switch(config-if)# exit`	Exits the interface configuration mode.

	Command or Action	Purpose
	`switch(config)#`	
Step 6	**show interface** **Example:** `switch# show interface`	(Optional) Displays the interface status and information.

Step 7	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 8	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to set Ethernet 3/1 as a Layer 2 access port that carries traffic for VLAN 5 only:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# switchport mode access
switch(config-if)# switchport access vlan 5
switch(config-if)#
```

4.6.3 Configuring Access Host Ports

You can optimize the performance of access ports that are connected to end stations by simultaneously setting that port as an access port. An access host port handles the STP like an edge port and immediately moves to the forwarding state without passing through the blocking and learning states. Configuring an interface as an access host port also disables port channeling on that interface.

Before you begin

Ensure that you are configuring the correct interface to an interface that is an end station.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *type slot/port*
3. **switchport host**
4. **exit**

5. **show interface**
6. **no shutdown**
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *type slot/port*
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an interface to configure, and enters interface configuration mode.

Step 3	**switchport host** **Example:** switch(config-if)# `switchport host`	Sets the interface to be an access host port, which immediately moves to the spanning tree forwarding state and disables port channeling on this interface. **Note** Apply this command only to end stations.

Step 4	**exit**
	Example:
	`switch(config-if-range)# exit`
	`switch(config)#`
		Exits the interface mode.

Step 5	**show interface** **Example:** `switch# show interface`	(Optional) Displays the interface status and information.

Step 6	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 7	**copy running-config startup-config** **Example:**	(Optional) Copies the running configuration to the startup configuration.

	Command or Action	Purpose
	switch(config)# **copy running-config startup-config**	

Example

This example shows how to set Ethernet 3/1 as a Layer 2 access port with PortFast enabled and port channel disabled:

```
switch# configure terminal
switch(config)# interface ethernet
3/1 switch(config-if)# switchport
host switch(config-if)#
```

4.6.4 Configuring Trunk Ports

You can configure a Layer 2 port as a trunk port. A trunk port transmits untagged packets for one VLAN plus encapsulated, tagged, packets for multiple VLANs. (See the “IEEE 802.1Q Encapsulation” section for information about encapsulation.)

Before you begin

Before you configure a trunk port, ensure that you are configuring a Layer 2 interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface** {*type slot/port* | **port-channel** *number*}
3. **switchport mode** [**access** | **trunk**]
4. **exit**
5. **show interface**
6. **no shutdown**
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** switch# **configure terminal** switch(config)#	Enters global configuration mode.
Step 2	**interface** {*type slot/port* | **port-channel** *number*} **Example:** switch(config)# **interface ethernet 3/1** switch(config-if)#	Specifies an interface to configure, and enters interface configuration mode.
Step 3	**switchport mode** [**access** | **trunk**] **Example:**	Sets the interface as a Layer 2 trunk port. A trunk port can carry traffic in one or more VLANs on the same

	Command or Action	Purpose
	`switch(config-if)# switchport mode trunk`	physical link (VLANs are based on the trunk-allowed VLANs list). By default, a trunk interface can carry traffic for all VLANs. To specify that only certain VLANs are allowed on the specified trunk, use the **switchport trunk allowed vlan** command.

Step 4	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode.

Step 5	**show interface** **Example:** `switch# show interface`	(Optional) Displays the interface status and information.

Step 6	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs
		where policies correspond with hardware policies. This
		command allows policy programming to continue and the
		port to come up. If policies do not correspond, the errors
		are placed in an error-disabled policy state.

Step 7	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to set Ethernet 3/1 as a Layer 2 trunk port:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# switchport mode
trunk switch(config-if)#
```

4.6.5 Configuring the Native VLAN for 802.1Q Trunking Ports

You can configure the native VLAN for 802.1Q trunk ports. If you do not configure this parameter, the trunk port uses the default VLAN as the native VLAN ID.

SUMMARY STEPS

1. **configure terminal**
2. **interface** {{*type slot/port*} | {**port-channel** *number*}}
3. **switchport trunk native vlan** *vlan-id*
4. **exit**
5. **show vlan**
6. **no shutdown**
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** {{*type slot/port*} | {**port-channel** *number*}}
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an interface to configure, and enters interface configuration mode.

Step 3	**switchport trunk native vlan** *vlan-id* **Example:** `switch(config-if)# switchport trunk native vlan 5`	Sets the native VLAN for the 802.1Q trunk. Valid values are from 1 to 4094, except those VLANs reserved for internal use. The default value is VLAN1.

Step 4	**exit**
	Example:
	`switch(config-if-range)# exit`
	`switch(config)#`
		Exits interface configuration mode.

Step 5	**show vlan** **Example:** `switch# show vlan`	(Optional) Displays the status and information of VLANs.
Step 6	**no shutdown** **Example:** `switch# configure terminal`	(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors

	Command or Action	Purpose
	switch(config)# **int e3/1** switch(config-if)# **no shutdown**	are placed in an error-disabled policy state.
Step 7	**copy running-config startup-config** **Example:** switch(config)# **copy running-config startup-config**	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to set the native VLAN for the Ethernet 3/1, Layer 2 trunk port to VLAN 5:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# switchport trunk native vlan
5 switch(config-if)#
```

4.6.6 Configuring the Allowed VLANs for Trunking Ports

You can specify the IDs for the VLANs that are allowed on the specific trunk port.

Before you begin

Before you configure the allowed VLANs for the specified trunk ports, ensure that you are configuring the correct interfaces and that the interfaces are trunks.

SUMMARY STEPS

1. **configure terminal**
2. **interface** {**ethernet** *slot/port* | **port-channel** *number*}
3. **switchport trunk allowed vlan** {*vlan-list* **add** *vlan-list* | **all** | **except** *vlan-list* | **none** | **remove** *vlan-list*}
4. **exit**
5. **show vlan**
6. **no shutdown**
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** switch# **configure terminal** switch(config)#	Enters global configuration mode.
Step 2	**interface** {**ethernet** *slot/port* | **port-channel** *number*} **Example:** switch(config)# interface ethernet 3/1	Specifies an interface to configure, and enters interface configuration mode.

	Command or Action	Purpose

Step 3	**switchport trunk allowed vlan** {*vlan-list* **add** *vlan-list* |
	all | **except** *vlan-list* | **none** | **remove** *vlan-list*}
	Example:
	`switch(config-if)# switchport trunk allowed vlan`
	`add 15-20#`
		Sets the allowed VLANs for the trunk interface. The default is to allow all VLANs on the trunk interface: 1 to 3967 and 4048 to 4094. VLANs 3968 to 4047 are the default VLANs reserved for internal use by default. By default, all VLANs are allowed on all trunk interfaces.
		The default reserved VLANs are 3968 to 4094, and you can change the block of reserved VLANs.
		See the Inspur CN12900 Series INOS-CN Layer 2 Switching Configuration Guide for more information.
		Note
		You cannot add internally allocated VLANs as allowed VLANs on trunk ports. The system returns a message if you attempt to list an internally allocated VLAN as an allowed VLAN.

Step 4	**exit**
	Example:
	`switch(config-if)# ` **`exit`**
	`switch(config)#`
		Exits the interface mode.

Step 5	**show vlan** **Example:** `switch# ` **`show vlan`**	(Optional) Displays the status and information for VLANs.

Step 6	**no shutdown**
	Example:
	`switch# ` **`configure terminal`**
	`switch(config)# ` **`int e3/1`**
	`switch(config-if)# ` **`no shutdown`**
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 7	**copy running-config startup-config** **Example:** `switch(config)# ` **`copy running-config startup-config`**	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to add VLANs 15 to 20 to the list of allowed VLANs on the Ethernet 3/1, Layer 2 trunk port:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# switchport trunk allowed vlan 15-
20 switch(config-if)#
```

4.6.7 Configuring MAC addresses Limitation on a VLAN

Inspur CN12900 Series switches with CN129-X636C-R, CN129-X636Q-R, CN129-X6136YC-R line cards provides the ability to set an upper limit for the number of MAC addresses that can reside inside MAC address table of a Line-card Expansion-module (LEM). You can configure the limitations at System, VLAN, port, trunk and tunnel levels. For instance, if the specified VLAN limitation is 2000 MACs, the Layer 2 Forwarding Manager (L2FM) accepts the first 2000 MACs it receives and reject the remaining MACs. To configure MAC address limitation on VLAN, complete the following steps:

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **mac address-table limit system** *value*
3. switch(config)# **mac address-table limit vlan** *value*
4. switch(config)# **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **mac address-table limit system** *value*	Specifies an upper limit for MAC learning at system level.
Step 3	switch(config)# **mac address-table limit vlan** *value*	Specifies an upper limit for MAC learning at VLAN level.
Step 4	switch(config)# **exit**	Exits configuration mode.

Example

This example shows how to configure the upper limit for MAC learning at system and VLAN levels:

```
switch# configure terminal
switch(config)# mac address-table limit system 10000
Configuring Mac address limit will result in flushing existing Macs in the
specified VLAN/System.Proceed(yes/no)? [no] yes
switch(config)# mac address-table limit vlan 30 3000
Configuring Mac address limit will result in flushing existing Macs in the
specified VLAN/System.Proceed(yes/no)? [no] yes
switch(config)# exit
```

This example shows how to display the MAC address limitations:

```
switch# configure terminal

switch(config)# sh mac address-table limit
```

```
System Limit: 10000


Vlan    Learning Limit
----    ----------
1       196000
20      196000
30      3000

100     196000
switch(config)# exit
```

4.6.8 Configuring switchport isolated

The switchport isolated feature is supported.

SUMMARY STEPS

1. **configure terminal**
2. **interface** {{**ethernet** *slot/port*} | {**port-channel** *number*}}
3. **switchport isolated**
4. **show running-config interface port-channel** *port-channel-number*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** {{**ethernet** *slot/port*} | {**port-channel** *number*}}
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an interface to configure, and enters interface configuration mode.

Step 3	**switchport isolated** **Example:** `switch(config-if)# switchport isolated`	Enables the switchport isolated feature.
Step 4	**show running-config interface port-channel** *port-channel-number*	(Optional) Displays the interface status and information.

4.6.9 Configuring a Default Interface

The default interface feature allows you to clear the existing configuration of multiple interfaces such as Ethernet, loopback, VLAN network, port-channel, and tunnel interfaces. All user configuration under a specified interface will be deleted. You can optionally create a checkpoint before clearing the interface configuration so that you can later restore the deleted configuration.

SUMMARY STEPS

1. **configure terminal**
2. **default interface** *int-if* [**checkpoint** *name*]
3. **exit**
4. **show interface**
5. **no shutdown**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**default interface** *int-if* [**checkpoint** *name*] **Example:** `switch(config)# default interface ethernet 3/1 checkpoint test8`	Deletes the configuration of the interface and restores the default configuration. Use the **?** keyword to display the supported interfaces. Use the **checkpoint** keyword to store a copy of the running configuration of the interface before clearing the configuration.

Step 3	**exit**
	Example:
	`switch(config)# exit`
	`switch(config)#`
		Exits global configuration mode.

Step 4	**show interface** **Example:** `switch# show interface`	(Optional) Displays the interface status and information.

Step 5	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Example

This example shows how to delete the configuration of an Ethernet interface while saving a checkpoint of the running configuration for rollback purposes:

```
switch# configure terminal
```

```
switch(config)# default interface ethernet 3/1 checkpoint test8
.......Done
switch(config)#
```

4.6.10 Configuring SVI Autostate Exclude

You can configure the SVI Autostate Exclude feature on an Ethernet interface or a port channel. You can use the Autostate Exclude option to enable or disable the port from bringing up or down the SVI calculation and applying it to all VLANs that are enabled on the selected port. You can also use the SVI Autostate Exlude VLAN feature to exclude a VLAN from the autostate excluded interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface** {{*type slot/port*} | {**port-channel** *number*}}
3. **switchport**
4. [**no**] **switchport autostate exclude**
5. [**no**] **switchport autostate exclude vlan** {*vlan id* | **all** | **except}**
6. **exit**
7. **show running-config interface** {{*type slot/port*} | {**port-channel** *number*}}
8. **no shutdown**
9. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** {{*type slot/port*} | {**port-channel** *number*}}
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an interface to configure, and enters interface configuration mode.

Step 3	**switchport** **Example:** `switch(config-if)# switchport`	Sets the interface as a Layer 2 interface.
Step 4	[**no**] **switchport autostate exclude** **Example:** `switch(config-if)# switchport autostate exclude`	Excludes this port from the VLAN interface link-up calculation when there are multiple ports in the VLAN. To revert to the default settings, use the no form of this command.
Step 5	[**no**] **switchport autostate exclude vlan** {*vlan id* | **all** | **except}**	(Optional) Excludes a vlan or a set of vlans from the autostate-excluded interface. This will help to minimize

	Command or Action	Purpose

	Example:
	`switch(config-if)# switchport autostate exclude`
	`vlan 10`
		any disruption to the system.
		To revert to the default settings, use the **no** form of this command.

Step 6	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface configuration mode.

Step 7	**show running-config interface** {{*type slot/port*} | {**port-channel** *number*}}
	Example:
	`switch(config)# show running-config interface ethernet 3/1`
		(Optional) Displays configuration information about the specified interface.

Step 8	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 9	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to exclude a port from the VLAN interface link-up calculation on the Inspur INOS-CN device:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# switchport
switch(config-if)# switchport autostate exclude
```

This example shows how to exclude a VLAN from the auto-excluded interface:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# switchport
switch(config-if)# switchport autostate exclude
switch(config-if)# switchport autostate exclude vlan 10
```

4.6.11 Configuring SVI Autostate Disable for the System

You can manage an SVI with the SVI autostate feature. You can configure the SVI autostate disable feature to keep an SVI up even if no interface is up in the corresponding VLAN. (Similarly, configure the SVI autostate enable feature so an SVI goes down when no interface is up in the corresponding VLAN). Use this procedure to configure this feature for the entire system.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **system default interface-vlan autostate**
3. **no shutdown**
4. **show running-config** [**all**]

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	[**no**] **system default interface-vlan autostate**
	Example:
	`switch(config)# no system default interface-vlan`
	`autostate`
		Disables the default autostate behavior for the device.
		Note Use the **system default interface-vlan autostate** command to enable the autostate behavior for the device.

Step 3	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 4	**show running-config** [**all**] **Example:** `switch(config)# show running-config`	(Optional) Displays the running configuration. To display the default and configured information, use the **all** keyword.

Example

This example shows how to disable the default autostate behavior on the Inspur INOS-CN device:

```
switch# configure terminal
switch(config)# no system default interface-vlan
autostate switch(config)# show running-config
```

4.6.12 Configuring SVI Autostate Disable Per SVI

You can configure SVI autostate enable or disable on individual SVIs. The SVI-level setting overrides the system-level SVI autostate configuration for that particular SVI.

SUMMARY STEPS

1. **configure terminal**
2. **feature interface-vlan**
3. **interface vlan** *vlan-id*
4. [**no**] **autostate**
5. **exit**
6. **show running-config interface vlan** *vlan-id*
7. **no shutdown**
8. **show startup-config interface vlan** *vlan-id*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature interface-vlan** **Example:** `switch(config)# feature interface-vlan`	Enables VLAN interface mode.

Step 3	**interface vlan** *vlan-id*
	Example:
	`switch(config-if)# interface vlan10`
	`switch(config)#`
		Creates a VLAN interface and enters interface configuration mode. The range is from 1 and 4094.

Step 4	[**no**] **autostate** **Example:** `switch(config-if)# no autostate`	By default, enables the SVI autostate feature on specified interface. To disable the default settings, use the **no** form of this command.
Step 5	**exit** **Example:**	Exits the interface configuration mode.

	Command or Action	Purpose

	`switch(config-if)# exit`
	`switch(config)#`	

Step 6	**show running-config interface vlan** *vlan-id* **Example:** `switch(config)# show running-config interface vlan10`	(Optional) Displays the running configuration for the specified VLAN interface.

Step 7	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 8	**show startup-config interface vlan** *vlan-id* **Example:** `switch(config)# show startup-config interface vlan10`	(Optional) Displays the VLAN configuration in the startup configuration.

Example

This example shows how to disable the default autostate behavior on an individual SVI:

```
switch# configure terminal
switch(config)# feature interface-vlan
switch(config)# interface vlan10
witch(config-if)# no autostate
```

4.6.13 Configuring the Device to Tag Native VLAN Traffic

When you are working with 802.1Q trunked interfaces, you can maintain the tagging for all packets that enter with a tag that matches the value of the native VLAN ID and drops all untagged traffic (you will still carry control traffic on that interface). This feature applies to the entire device; you cannot apply it to selected VLANs on a device.

The **vlan dot1q tag native global** command changes the behavior of all native VLAN ID interfaces on all trunks on the device.

SUMMARY STEPS

1. **configure terminal**
2. **vlan dot1q tag native**
3. **exit**
4. **show vlan**
5. **no shutdown**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vlan dot1q tag native** **Example:** `switch(config)# vlan dot1q tag native`	Modifies the behavior of a 802.1Q trunked native VLAN ID interface. The interface maintains the taggings for all packets that enter with a tag that matches the value of the native VLAN ID and drops all untagged traffic. The control traffic is still carried on the native VLAN. The default is disabled.

Step 3	**exit**
	Example:
	`switch(config-if-range)# exit`
	`switch(config)#`
		Exits the interface configuration mode.

Step 4	**show vlan** **Example:** `switch# show vlan`	(Optional) Displays the status and information for VLANs.

Step 5	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to change the behavior of the native VLAN on an 802.1Q trunked interface to maintain the tagged packets and drop all untagged traffic (except control traffic):

```
switch# configure terminal
switch(config)# vlan dot1q tag native
switch#
```

4.6.14 Changing the System Default Port Mode to Layer 2

You can set the system default port mode to Layer 2 access ports.

SUMMARY STEPS

1. **configure terminal**
2. **system default switchport** [**shutdown**]
3. **exit**
4. **show interface brief**
5. **no shutdown**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**system default switchport** [**shutdown**] **Example:** switch(config-if)# **`system default switchport`**	Sets the default port mode for all interfaces on the system to Layer 2 access port mode and enters interface configuration mode. By default, all the interfaces are Layer 3. **Note** When the **system default switchport shutdown** command is issued: • Any Layer 2 port that is not specifically configured with **no shutdown** are shutdown. To avoid the shutdown, configure the Layer 2 port with **no shut**

Step 3	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface configuration mode.

Step 4	**show interface brief**	(Optional) Displays the status and information for interfaces.

	Command or Action	Purpose
	Example: `switch# show interface brief`	

Step 5	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs
		where policies correspond with hardware policies. This
		command allows policy programming to continue and the
		port to come up. If policies do not correspond, the errors
		are placed in an error-disabled policy state.

Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to set the system ports to be Layer 2 access ports by default:

```
switch# configure terminal switch(config-
if)# system default switchport
switch(config-if)#
```

4.7 Verifying the Interface Configuration

To display access and trunk interface configuration information, perform one of the following tasks.

Command	Purpose
show interface ethernet *slot/port* [**brief** | | **counters** | **debounce** | **description** | **flowcontrol** | **mac-address** | **status** | **transceiver**]	Displays the interface configuration.
show interface brief	Displays interface configuration information, including the mode.
show interface switchport	Displays information, including access and trunk interface, information for all Layer 2 interfaces.
show interface trunk [**module** *module-number* | **vlan** *vlan-id*]	Displays trunk configuration information.
show interface capabilities	Displays information about the capabilities of the interfaces.
show running-config [**all**]	Displays information about the current configuration.

Command	Purpose
	The **all** command displays the default and current configurations.
show running-config interface ethernet *slot/port*	Displays configuration information about the specified interface.
show running-config interface port-channel *slot/port*	Displays configuration information about the specified port-channel interface.
show running-config interface vlan *vlan-id*	Displays configuration information about the specified VLAN interface.

4.8 Monitoring the Layer 2 Interfaces

Use the following commands to display Layer 2 interfaces:

Command	Purpose
clear counters interface [**interface**]	Clears the counters.
load- interval {**interval** *seconds* {**1** | **2** | **3**}}	Inspur CN12900 Series devices set three different sampling intervals to bit-rate and packet-rate statistics.
show interface counters [**module** *module*]	Displays input and output octets unicast packets, multicast packets, and broadcast packets.
show interface counters detailed [**all**]	Displays input packets, bytes, and multicast as well as output packets and bytes.
show interface counters errors [**module** *module*]	Displays information on the number of error packets.

4.9 Configuration Examples for Access and Trunk Ports

This example shows how to configure a Layer 2 access interface and assign the access VLAN mode for that interface:

```
switch# configure terminal
switch(config)# interface ethernet 2/30
switch(config-if)# switchport
switch(config-if)# switchport mode access
switch(config-if)# switchport access vlan 5
switch(config-if)#
```

This example shows how to configure a Layer 2 trunk interface, assign the native VLAN and the allowed VLANs, and configure the device to tag the native VLAN traffic on the trunk interface:

```
switch# configure terminal switch(config)#
interface ethernet 2/35 switch(config-if)#
```

```
switchport switch(config-if)# switchport
mode trunk
switch(config-if)# switchport trunk native vlan 10
switch(config-if)# switchport trunk allowed vlan 5, 10
switch(config-if)# exit
switch(config)# vlan dot1q tag native
switch(config)#
```

4.10 Related Documents

Related Documents	Document Title
Configuring Layer 3 interfaces	Configuring Layer 2 Interfaces section
Port Channels	Configuring Port Channels section
VLANs, private VLANs, and STP	*Inspur CN12900 Series INOS-CN Layer 2 Switching Configuration Guide*
System management	*Inspur CN12900 Series INOS-CN System Management Configuration Guide*
High availability	*Inspur CN12900 Series INOS-CN High Availability and Redundancy Guide*
Release Notes	*Inspur CN12900 Series INOS-CN Release Notes*

CHAPTER 5 Configuring Layer 3 Interfaces

5.1 About Layer 3 Interfaces

Layer 3 interfaces forward IPv4 and IPv6 packets to another device using static or dynamic routing protocols. You can use Layer 3 interfaces for IP routing and inter-VLAN routing of Layer 2 traffic.

5.1.1 Routed Interfaces

You can configure a port as a Layer 2 interface or a Layer 3 interface. A routed interface is a physical port that can route IP traffic to another device. A routed interface is a Layer 3 interface only and does not support Layer 2 protocols, such as the Spanning Tree Protocol (STP).

All Ethernet ports are routed interfaces by default. You can change this default behavior with the CLI setup script.

You can assign an IP address to the port, enable routing, and assign routing protocol characteristics to this routed interface.

You can also create a Layer 3 port channel from routed interfaces. For more information about port channels, see the “Configuring Port Channels” section.

Routed interfaces and subinterfaces support exponentially decayed rate counters. Inspur INOS-CN tracks the following statistics with these averaging counters:

- Input packets/sec
- Output packets/sec
- Input bytes/sec
- Output bytes/sec

5.1.2 Subinterfaces

You can create virtual subinterfaces on a parent interface configured as a Layer 3 interface. A parent interface can be a physical port.

Subinterfaces divide the parent interface into two or more virtual interfaces on which you can assign unique Layer 3 parameters such as IP addresses and dynamic routing protocols. The IP address for each subinterface should be in a different subnet from any other subinterface on the parent interface.

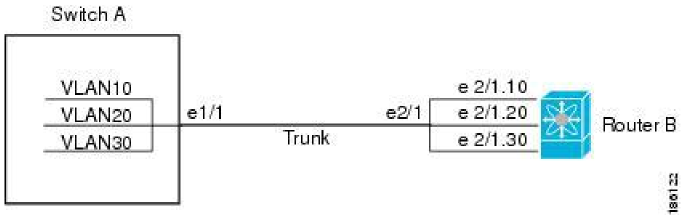
You create a subinterface with a name that consists of the parent interface name (for example, Ethernet 2/1) followed by a period and then by a number that is unique for that subinterface. For example, you could create a subinterface for Ethernet interface 2/1 named Ethernet 2/1.1 where .1 indicates the subinterface.

Inspur INOS-CN enables subinterfaces when the parent interface is enabled. You can shut down a subinterface independent of shutting down the parent interface. If you shut down the parent interface, Inspur INOS-CN shuts down all associated subinterfaces as well.

One use of subinterfaces is to provide unique Layer 3 interfaces to each virtual local area network (VLAN) supported by the parent interface. In this scenario, the parent interface connects to a Layer 2 trunking port on another device. You configure a subinterface and associate the subinterface to a VLAN ID using 802.1Q trunking.

The following figure shows a trunking port from a switch that connects to router B on interface E 2/1. This interface contains three subinterfaces that are associated with each of the three VLANs carried by the trunking port.

Figure 4：Subinterfaces for VLANs



For more information about VLANs, see the Inspur CN12900 Series INOS-CN Layer 2 Switching Configuration Guide.

Limitations for Subinterfaces

The following are the limitations for subinterfaces:

- Statistics for subinterfaces are not supported.

5.1.3 VLAN Interfaces

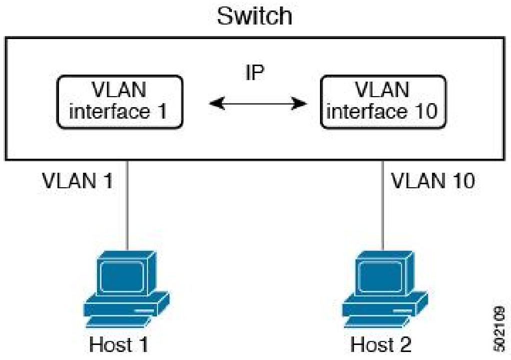
A VLAN interface, or switch virtual interface (SVI), is a virtual routed interface that connects a VLAN on the device to the Layer 3 router engine on the same device. Only one VLAN interface can be associated with a VLAN, but you need to configure a VLAN interface for a VLAN only when you want to route between VLANs or to provide IP host connectivity to the device through a virtual routing and forwarding (VRF) instance that is not the management VRF. When you enable VLAN interface creation, Inspur INOS-CN creates a VLAN interface for the default VLAN (VLAN 1) to permit remote switch administration.

You must enable the VLAN network interface feature before you can see configure it. The system automatically takes a checkpoint prior to disabling the feature, and you can roll back to this checkpoint. See the Inspur CN12900 Series INOS-CN System Management Configuration Guide for information on rollbacks and checkpoints.

You can route across VLAN interfaces to provide Layer 3 inter-VLAN routing by configuring a VLAN interface for each VLAN that you want to route traffic to and assigning an IP address on the VLAN interface. For more information about IP addresses and IP routing, see the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide.

The following figure shows two hosts connected to two VLANs on a device. You can configure VLAN interfaces for each VLAN that allows Host 1 to communicate with Host 2 using IP routing between the VLANs. VLAN 1 communicates at Layer 3 over VLAN interface 1 and VLAN 10 communicates at Layer 3 over VLAN interface 10.

Figure 5：Connecting Two VLANs with VLAN interfaces



5.1.4 Changing VRF Membership for an Interface

When you enter the **vrf member** command under an interface, you receive an alert regarding the deletion of interface configurations and to notify the clients/listeners (such as CLI-Server) to delete configurations with respect to the interface.

Entering the **system vrf-member-change retain-l3-config** command enables the retention of the Layer 3 configuration when the VRF member changes on the interface. It does this by sending notification to the clients/listeners to store (buffer) the existing configurations, delete the configurations from the old vrf context, and reapply the stored configurations under the new VRF context.

You can disable the retention of the Layer 3 configuration with the **no system vrf-member-change retain-l3-config** command. In this mode, the Layer 3 configuration is not retained when the VRF member changes.

Notes About Changing VRF Membership for an Interface

- Momentary traffic loss may occur when changing the VRF name.
- Only the configurations under the interface level are processed when the **system vrf-member-change retain-l3-config** command is enabled. You must manually process any configurations at the router level to accommodate routing protocols after a VRF change.
- The **system vrf-member-change retain-l3-config** command supports interface level configurations with:
- Layer 3 configurations maintained by the CLI Server, such as **ip address** and **ipv6 address** (secondary) and all OSPF/ISIS/EIGRP CLIs available under the interface configuration.
- HSRP
- DHCP Relay Agent CLIs, such as **ip dhcp relay address** [**use-vrf**] and **ipv6 dhcp relay address** [**use-vrf**].
- For DHCP:
- As a best practice, the client and server interface VRF should be changed one at a time. Otherwise, the DHCP packets cannot be exchanged on the relay agent.
- When the client and server are in different VRFs, use the **ip dhcp relay address** [**use-vrf**] command to exchange the DHCP packets in the relay agent over the different VRFs.

5.1.5 Loopback Interfaces

A loopback interface is a virtual interface with a single endpoint that is always up. Any packet transmitted over a loopback interface is immediately received by this interface. Loopback interfaces emulate a physical interface. You can configure up to 1024 loopback interfaces, numbered 0 to 1023.

You can use loopback interfaces for performance analysis, testing, and local communications. Loopback interfaces can act as a termination address for routing protocol sessions. This loopback configuration allows routing protocol sessions to stay up even if some of the outbound interfaces are down.

5.1.6 IP Unnumbered

The IP unnumbered feature enables the processing of IP packets on a point to point (p2p) interface without explicitly configuring a unique IP address on it. This approach borrows an IP address from another interface and conserves address space on point to point links.

Any interface which conforms to the point to point mode can be used as an IP unnumbered interface. The IP unnumbered feature is supported only on Ethernet interfaces and sub-interfaces. The borrowed interface can only be a loopback interface and is known as the numbered interface.

A loopback interface is ideal as a numbered interface in that it is always functionally up. However, because loopback interfaces are local to a switch/router, the reachability of unnumbered interfaces first needs to be established through static routes or by using an interior gateway protocol, such as OSPF or ISIS.

Configuring IP unnumbered interfaces for port channels is supported on all Inspur CN12900 Series switches.

5.1.7 MAC-Embedded IPv6 Address

BGP allows an IPv4 prefix to be carried over an IPv6 next hop. The IPv6 next hop is leveraged to remove neighbor discovery (ND)-related traffic from the network. To do this , the MAC address is embedded in the IPv6 address. Such an address is called a MAC-embedded IPv6 (MEv6) address. The router extracts the MAC address directly from the MEv6 address instead of going through ND. Local interface and next-hop MAC addresses are extracted from the IPv6 addresses.

On MEv6-enabled IPv6 interfaces, the same MEv6-extracted MAC address is used for IPv4 traffic as well. MEv6 is supported on all Layer 3-capable interfaces except switch virtual interfaces (SVIs).

Important	When MEv6 is enabled on an interface, ping6 to the IPv6 link local address, OSPFv3, and BFDv6 are not supported on that interface.

5.1.8 High Availability

Layer 3 interfaces support stateful and stateless restarts. After the switchover, Inspur INOS-CN applies the runtime configuration after the switchover.

See the Inspur CN12900 Series INOS-CN High Availability and Redundancy Guide for complete information about high availability.

5.1.9 Virtualization Support

Layer 3 interfaces support Virtual Routing and Forwarding instances (VRFs). VRFs exist within virtual device contexts (VDCs). By default, Inspur INOS-CN places you in the default VDC and default VRF .

5.1.10 DHCP Client

Inspur INOS-CN supports DHCP client for IPv4 and IPv6 addresses on SVIs, physical Ethernet, and management interfaces. You can configure the IP address of a DHCP client by using the **ip address dhcp** or **ipv6 address dhcp** command. These commands send a request from the DHCP client to the DHCP server soliciting an IPv4 or IPv6 address from the DHCP server. The DHCP client on the Inspur switch identifies itself to the DHCP server. The DHCP server uses this identifier to send the IP address back to the DHCP client.

When a DHCP client is configured on the SVI with the DHCP server sending router and DNS options, the **ip route** 0.0.0.0/0 *router-ip* and **ip name-server** *dns-ip* commands are configured on the switch automatically.

Limitations for Using DHCP Client on Interfaces

The following are the limitations for using DHCP client on interfaces:

- This feature is supported only on physical Ethernet interfaces, management interfaces, and SVIs.
- This feature is supported on non-default virtual routing and forwarding (VRF) instances.
- The DNS server and default router option-related configurations are saved in the startup configuration when you enter the **copy running-config startup-config** command. When you reload the switch, if this configuration is not applicable, you might have to remove it.
- You can configure a maximum of six DNS servers on the switch, which is a switch limitation. This maximum number includes the DNS servers configured by the DHCP client and the DNS servers configured manually.

If the number of DNS servers configured on the switch is more than six, and if you get a DHCP offer for an SVI with DNS option set, the IP address is not assigned to the SVI.

- An Inspur CN12900 Series switch supports a maximum of 10 IPv4 and 10 IPv6 DHCP clients.
- DHCP relay and DHCP client configurations are incompatible and are not supported on the same switch. You must remove the DHCP relay configuration before configuring the DHCP Client on an interface.
- When DHCP snooping is enabled on the VLAN whose SVI is configured with the DHCP client, the DHCP snooping is not enforced on the SVI DHCP client.
- When configuring the IPv6 DHCP client, you must configure with the **ipv6 address use-link-local-only**

command before the **ipv6 address dhcp** command.

5.2 Licensing Requirements for Layer 3 Interfaces

The following table shows the licensing requirements for this feature:

Product	License Requirement
Inspur INOS-CN	Layer 3 interfaces require no license. Any feature not included in a license package is bundled with the Inspur INOS-CN image and is provided at no extra charge to you.

5.3 Prerequisites for Layer 3 Interfaces

Layer 3 interfaces have the following prerequisites:

- You are familiar with IP addressing and basic configuration. See the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide for more information about IP addressing.

5.4 Guidelines and Limitations for Layer 3 Interfaces

Layer 3 interfaces have the following configuration guidelines and limitations:

- **show** commands with the **internal** keyword are not supported.
- Configuring a subinterface on a port-channel interface is not supported.
- If you change a Layer 3 interface to a Layer 2 interface, Inspur INOS-CN shuts down the interface, reenables the interface, and removes all configuration specific to Layer 3.
- If you change a Layer 2 interface to a Layer 3 interface, Inspur INOS-CN shuts down the interface, reenables the interface, and deletes all configuration specific to Layer 2.
- The Dynamic Host Configuration Protocol (DHCP) option is not supported when configuring a subinterface on a port-channel interface.
- When an IP unnumbered interface is configured, a loopback interface should be in the same VRF as the IP unnumbered interface.
- An **admin-shutdown** command on a loopback interface that is a numbered interface does not bring down the IP unnumbered interface. This means that the routing protocols running over the IP unnumbered interface continue to be up.
- Static routes running over the IP unnumbered interface should use pinned static routes.
- An IP unnumbered interface is supported only on physical and sub-interfaces.
- Only loopback interfaces can use unnumbered interfaces as numbered interfaces.
- OSPF over an IP unnumbered interface is supported.
- ISIS over an IP unnumbered interface is supported.
- BGP over a loopback interface with an IP unnumbered interface as an overlay interface is supported.
- The default and non-default VRF is supported by IP unnumbered interfaces.
- The switch has a limit of 16 user-defined MAC addresses (MEv6/static). Configuring beyond this limit might result in issues documented in CSCux84428.

5.5 Default Settings

The following table lists the default settings for Layer 3 interface parameters.

Table 7：Default Layer 3 Interface Parameters

Parameters	Default
Admin state	Shut

5.6 Configuring Layer 3 Interfaces

5.6.1 Configuring a Routed Interface

You can configure any Ethernet port as a routed interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/port*
3. **no switchport**
4. [**ip address** *ip-address/length* | **ipv6 address** *ipv6-address/length*]
5. **show interfaces**
6. **no shutdown**
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *slot/port*
	Example:
	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 interface.

Step 4	[**ip address** *ip-address/length* | **ipv6 address** *ipv6-address/length*]
	Example:
	`switch(config-if)# ip address 192.0.2.1/8`
	Example:
	`switch(config-if)# ipv6 address 2001:0DB8::1/8`
		• Configures an IP address for this interface. See the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide for more information about IP addresses.
		• Configures an IPv6 address for this interface. See the Inspur CN12900 Series INOS-CN Unicast

	Command or Action	Purpose
		Routing Configuration Guide for more information about IPv6 addresses.
Step 5	**show interfaces** **Example:** `switch(config-if)# show interfaces ethernet 2/1`	(Optional) Displays the Layer 3 interface statistics.

Step 6	**no shutdown**
	Example:
	`switch#`
	`switch(config-if)# int e2/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces where policies
		correspond with hardware policies. This command allows
		policy programming to continue and the port to come up.
		If policies do not correspond, the errors are placed in an
		error-disabled policy state.

Step 7	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves the configuration change.

Example

- Use the **medium** command to set the interface medium to either point to point or broadcast.

Command	Purpose

medium {**broadcast** | **p2p**}
Example:
`switch(config-if)# medium p2p medium p2p`
	Configures the interface medium as either point to point or broadcast.

switchport Example: `switch(config-if)# switchportswitchport`	Configures the interface as a Layer 2 interface and deletes any configuration specific to Layer 3 on this interface.

- This example shows how to configure a routed interface:

```
switch# configure terminal
switch(config)# interface ethernet 2/1
switch(config-if)# no switchport
switch(config-if)# ip address 192.0.2.1/8
switch(config-if)# copy running-config startup-config
```

The default setting for interfaces is routed. If you want to configure an interface for Layer 2, enter the **switchport** command. Then, if you change a Layer 2 interface to a routed interface, enter the **no switchport** command.

5.6.2 Configuring a Subinterface on a Routed Interface

You can configure one or more subinterfaces on a routed interface made from routed interfaces.

Before you begin

Configure the parent interface as a routed interface.
See the "Configuring a Routed Interface" section.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/port.number*
3. [**ip address** *ip-address/length* | **ipv6 address** *ipv6-address/length*]
4. **encapsulation dot1Q** *vlan-id*
5. **show interfaces**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *slot/port.number*
	Example:
	`switch(config)# interface ethernet 2/1.1`
	`switch(config-subif)#`
		Creates a subinterface and enters subinterface configuration mode. The number range is from 1 to 4094.
		Note A subinterface greater than 511 is not supported on physical interfaces.

Step 3	[**ip address** *ip-address/length* | **ipv6 address** *ipv6-address/length*]
	Example:
	`switch(config-subif)# ip address 192.0.2.1/8`
	Example:
	`switch(config-subif)# ipv6 address 2001:0DB8::1/8`
		• Configures an IP address for this subinterface. See the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide for more information on IP addresses.
		• Configures an IPv6 address for this subinterface. See the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide for more information on IPv6 addresses.

Step 4	**encapsulation dot1Q** *vlan-id*	Configures IEEE 802.1Q VLAN encapsulation on

	Command or Action	Purpose
	Example: `switch(config-subif)# encapsulation dot1Q 33`	the subinterface. The range is from 2 to 4093.
Step 5	**show interfaces** **Example:** `switch(config-subif)# show interfaces ethernet 2/1.1`	(Optional) Displays the Layer 3 interface statistics.
Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves the configuration change.

Example

- This example shows how to create a subinterface:

```
switch# configure terminal
switch(config)# interface ethernet 2/1.1
switch(config-if)# ip address 192.0.2.1/8
switch(config-if)# encapsulation dot1Q 33
switch(config-if)# copy running-config startup-config
```

- The output of the **show interface eth** command is enhanced for the subinterfaces as shown in the following :

```
switch# show interface ethernet 1/2.1
Ethernet1/2.1 is down (Parent Interface Admin down)
admin state is down, Dedicated Interface, [parent interface is
Ethernet1/2] Hardware: 40000 Ethernet, address: 0023.ac67.9bc1 (bia
4055.3926.61d4) Internet Address is 10.10.10.1/24
MTU 1500 bytes, BW 40000000 Kbit, DLY 10 usec
reliability 255/255, txload 1/255, rxload
1/255 Auto-mdix is turned off
EtherType is 0x8100
L3 in Switched:
 ucast: 0 pkts, 0 bytes - mcast: 0 pkts, 0
bytes L3 out Switched:
    ucast: 0 pkts, 0 bytes - mcast: 0 pkts, 0 bytes
```

5.6.3 Configuring a Subinterface on a Port-Channel Interface

You can configure one or more subinterfaces on a port-channel interface.

Before you begin

Configure the parent interface as a port-channel interface.
See the “Configuring Port Channels” chapter.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *channel-id.number*
3. [**ip address** *ip-address/length* | **ipv6 address** *ipv6-address/length*]
4. **encapsulation dot1Q** *vlan-id*
5. **show interfaces**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *channel-id.number*
	Example:
	`switch(config)# interface port-channel 100.1`
	`switch(config-subif)#`
		Creates a subinterface and enters subinterface configuration mode.

Step 3	[**ip address** *ip-address/length* | **ipv6 address** *ipv6-address/length*]
	Example:
	`switch(config-subif)# ip address 192.0.2.1/8`
	Example:
	`switch(config-subif)# ipv6 address 2001:0DB8::1/8`
		• Configures an IP address for this subinterface. See the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide for more information on IP addresses.
		• Configures an IPv6 address for this subinterface. See the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide for more information on IPv6 addresses.

Step 4	**encapsulation dot1Q** *vlan-id* **Example:** `switch(config-subif)# encapsulation dot1Q 33`	Configures IEEE 802.1Q VLAN encapsulation on the subinterface. The range is from 2 to 4093.
Step 5	**show interfaces** **Example:** `switch(config-subif)# show interfaces ethernet 2/1.1`	(Optional) Displays the Layer 3 interface statistics.
Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves the configuration change.

Example

This example shows how to create a subinterface:

```
switch# configure terminal
switch(config)# interface port-channel 115.3
switch(config-subif)# ip address 141.143.101.2/24
switch(config-subif)# encapsulation dot1q 3
switch(config-subif)# copy running-config startup-config
```

5.6.4 Configuring a VLAN Interface

You can create VLAN interfaces to provide inter-VLAN routing.

SUMMARY STEPS

1. **configure terminal**
2. **feature interface-vlan**
3. **interface vlan** *number*
4. [**ip address** *ip-address/length* | **ipv6 address** *ipv6-address/length*]
5. **show interface vlan** *number*
6. **no shutdown**
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	**feature interface-vlan** **Example:** `switch(config)# feature interface-vlan`	Enables VLAN interface mode.

Step 3	**interface vlan** *number*
	Example:
	`switch(config)# interface vlan 10`
	`switch(config-if)#`
		Creates a VLAN interface. The number range is from 1 to 4094.

Step 4	[**ip address** *ip-address/length* | **ipv6 address** *ipv6-address/length*]
	Example:
	`switch(config-if)# ip address 192.0.2.1/8`
	Example:
	`switch(config-if)# ipv6 address 2001:0DB8::1/8`
		• Configures an IP address for this VLAN interface. See the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide for more information on IP addresses.
		• Configures an IPv6 address for this VLAN interface. See the Inspur CN12900 Series INOS-CN Unicast

		Routing Configuration Guide for more information on IPv6 addresses.
Step 5	**show interface vlan** *number* **Example:** `switch(config-if)# show interface vlan 10`	(Optional) Displays the Layer 3 interface statistics.

Step 6	**no shutdown**
	Example:
	`switch(config)# int e3/1`
	`switch(config)# no shutdown`
		(Optional) Clears the errors on the interfaces where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up.
		If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 7	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves the configuration change.

Example

This example shows how to create a VLAN interface:

```
switch# configure terminal
switch(config)# feature interface-vlan
switch(config)# interface vlan 10
switch(config-if)# ip address 192.0.2.1/8 switch(config-
if)# copy running-config startup-config
```

5.6.5 Enabling Layer 3 Retention During VRF Membership Change

The following steps enable the retention of the Layer 3 configuration when changing the VRF membership on the interface.

SUMMARY STEPS

1. **configure terminal**
2. **system vrf-member-change retain-l3-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

	Command or Action	Purpose

Step 2	**system vrf-member-change retain-l3-config**
	Example:
	`switch(config)# system vrf-member-change retain-l3-config`
	`Warning: Will retain L3 configuration when vrf member change on interface.`
		Enables Layer 3 configuration retention during VRF membership change.
		Note To disable the retention of the Layer 3 configuration, use the **no system vrf-member-change retain-l3-config** command.

5.6.6 Configuring a Loopback Interface

You can configure a loopback interface to create a virtual interface that is always up.

Before you begin

Ensure that the IP address of the loopback interface is unique across all routers on the network.

SUMMARY STEPS

1. **configure terminal**
2. **interface loopback** *instance*
3. [**ip address** *ip-address/length* | **ipv6 address** *ipv6-address/length*]
4. **show interface loopback** *instance*
5. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	**interface loopback** *instance*
	Example:
	`switch(config)# interface loopback 0`
	`switch(config-if)#`
		Creates a loopback interface. The range is from 0 to 1023.

Step 3	[**ip address** *ip-address/length* | **ipv6 address** *ipv6-address/length*]
	Example:
	`switch(config-if)# ip address 192.0.2.1/8`
	Example:
	`switch(config-if)# ipv6 address 2001:0DB8::1/8`
		• Configures an IP address for this interface. See the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide for more information about IP addresses.
		• Configures an IPv6 address for this interface. See the Inspur CN12900 Series INOS-CN Unicast

	Command or Action	Purpose
		Routing Configuration Guide for more information about IPv6 addresses.
Step 4	**show interface loopback** *instance* **Example:** `switch(config-if)# show interface loopback 0`	(Optional) Displays the loopback interface statistics.
Step 5	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves the configuration change.

Example

This example shows how to create a loopback interface:

```
switch# configure terminal switch(config)#
interface loopback 0 switch(config-if)# ip
address 192.0.2.1/8
switch(config-if)# copy running-config startup-config
```

5.6.7 Configuring IP Unnumbered on an Ethernet Interface

You can configure the IP unnumbered feature on an ethernet interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/port*
3. **medium p2p**
4. **ip unnumbered** *type number*

ETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *slot/port*
	Example:
	`switch(config)# interface ethernet 1/1`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	**medium p2p** **Example:**	Configures the interface medium as point to point.

	Command or Action	Purpose
	`switch(config-if)# medium p2p`	
Step 4	**ip unnumbered** *type number* **Example:** `switch(config-if)# ip unnumbered loopback 100`	Enables IP processing on an interface without assigning an explicit IP address to the interface. *type* and *number* specify another interface on which the router has an assigned IP address. The interface specified cannot be another unnumbered interface. **Note** *type* is limited to **loopback**.

5.6.8 Configuring OSPF for an IP Unnumbered Interface

You can configure OSPF for an IP unnumbered loopback interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/port*
3. **encapsulation dot1Q** *vlan-id*
4. **medium p2p**
5. **ip unnumbered** *type number*
6. (Optional) **ip ospf authentication**
7. (Optional) **ip ospf authentication-key** *password*
8. **ip router ospf** *instance* **area** *area-number*
9. **no shutdown**
10. **interface loopback** *instance*
11. **ip address** *ip-address/length*
12. **ip router ospf** *instance* **area** *area-number*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *slot/port*
	Example:
	`switch(config)# interface ethernet 1/20.1`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	**encapsulation dot1Q** *vlan-id* **Example:**	Configures IEEE 802.1Q VLAN encapsulation on the subinterface. The range is from 2 to 4093.

	Command or Action	Purpose
	`switch(config-if)# encapsulation dot1Q 100`	
Step 4	**medium p2p** **Example:** `switch(config-if)# medium p2p`	Configures the interface medium as point to point.
Step 5	**ip unnumbered** *type number* **Example:** `switch(config-if)# ip unnumbered loopback 101`	Enables IP processing on an interface without assigning an explicit IP address to the interface. *type* and *number* specify another interface on which the router has an assigned IP address. The interface specified cannot be another unnumbered interface. **Note** *type* is limited to **loopback**.
Step 6	(Optional) **ip ospf authentication** **Example:** `switch(config-if)# ip ospf authentication`	Specifies the authentication type for interface.
Step 7	(Optional) **ip ospf authentication-key** *password* **Example:** `switch(config-if)# ip ospf authentication 3 b7bdf15f62bbd250`	Specifies the authentication password for OSPF authentication.
Step 8	**ip router ospf** *instance* **area** *area-number* **Example:** `switch(config-if)#  ip router ospf 100 area 0.0.0.1`	Configures routing process for IP on an interface and specifies an area. **Note** The **ip router ospf** command is required for both the unnumbered and the numbered interface.
Step 9	**no shutdown** **Example:** `switch(config-if)# no shutdown`	Brings up the interface (administratively).
Step 10	**interface loopback** *instance* **Example:** `switch(config)# interface loopback 101`	Creates a loopback interface. The range is from 0 to 1023.
Step 11	**ip address** *ip-address/length* **Example:** `switch(config-if)# 192.168.101.1/32`	Configures an IP address for the interface.

	Command or Action	Purpose
Step 12	**ip router ospf** *instance* **area** *area-number* **Example:** `switch(config-if)# ip router ospf 100 area 0.0.0.1`	Configures routing process for IP on an interface and specifies an area. **Note** The **ip router ospf** command is required for both the unnumbered and the numbered interface.

5.6.9 Configuring ISIS for an IP Unnumbered Interface

You can configure ISIS for an IP unnumbered loopback interface.

SUMMARY STEPS

1. **configure terminal**
2. **feature isis**
3. **router isis** *area-tag*
4. **net** *network-entity-title*
5. **end**
6. **interface ethernet** *slot/port*
7. **encapsulation dot1Q** *vlan-id*
8. **medium p2p**
9. **ip unnumbered** *type number*
10. **ip router isis** *area-tag*
11. **no shutdown**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature isis** **Example:** `Switch(config)# feature isis`	Enables ISIS.
Step 3	**router isis** *area-tag* **Example:** `Switch(config)# router isis 100`	Assigns a tag to an IS-IS process and enters router configuration mode.
Step 4	**net** *network-entity-title* **Example:** `Switch(config-router)# net 49.0001.0100.0100.1001.00`	Configures the network entity title (NET) on the device.

	Command or Action	Purpose

Step 5	**end**
	Example:
	`Switch(config-router)# end`
		Exit router configuration mode.

Step 6	**interface ethernet** *slot/port* **Example:** `switch(config)# interface ethernet 1/20.1`	Enters interface configuration mode.
Step 7	**encapsulation dot1Q** *vlan-id* **Example:** `switch(config-subif)# encapsulation dot1Q 100`	Configures IEEE 802.1Q VLAN encapsulation on the subinterface. The range is from 2 to 4093.
Step 8	**medium p2p** **Example:** `switch(config-subif)# medium p2p`	Configures the interface medium as point to point.
Step 9	**ip unnumbered** *type number* **Example:** `switch(config-if)# ip unnumbered loopback 101`	Enables IP processing on an interface without assigning an explicit IP address to the interface. *type* and *number* specify another interface on which the router has an assigned IP address. The interface specified cannot be another unnumbered interface. **Note** *type* is limited to **loopback**.
Step 10	**ip router isis** *area-tag* **Example:** `switch(config-subif)# ip router isis 100`	Enables ISIS on the unnumbered interface.
Step 11	**no shutdown** **Example:** `switch(config-subif)# no shutdown`	Brings up the interface (administratively).

5.6.10 Configuring PBR on SVI on the Gateway

This procedure configures PBR on the primary SVI interface in the gateway.

SUMMARY STEPS

1. **configure terminal**
2. **ip access-list** *list-name*
3. **permit tcp host** *ipaddr* **host** *ipaddr* **eq** *port-number*
4. **exit**
5. **route-map** *route-map-name*

6. **match ip address** *access-list-name*
7. **set ip next-hop** *addr1*
8. **exit**
9. **interface vlan** *vlan-id*
10. **ip address** *ip-addr*
11. **no ip redirects**
12. (Optional) **ip policy route-map pbr-sample**
13. **exit**
14. **hsrp version 2**
15. **hsrp***group-num*
16. **name** *name-val*
17. **ip** *ip-addr*
18. **no shutdown**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enter global configuration mode.
Step 2	**ip access-list** *list-name* **Example:** `switch(config)# ip access-list pbr-sample`	Configure access list.
Step 3	**permit tcp host** *ipaddr* **host** *ipaddr* **eq** *port-number* **Example:** `switch(config-acl)# permit tcp host 10.1.1.1 host 192.168.2.1 eq 80`	Specify the packets to forward on a specific port.
Step 4	**exit** **Example:** `switch(config-acl)# exit`	Exit configuration mode.
Step 5	**route-map** *route-map-name* **Example:** `switch(config)# route-map pbr-sample`	Create a route-map or enter route-map command mode.
Step 6	**match ip address** *access-list-name* **Example:** `switch(config-route-map)# match ip address pbr-sample`	Match values from the routing table.
Step 7	**set ip next-hop** *addr1* **Example:**	Set IP address of the next hop.

	Command or Action	Purpose
	`switch(config-route-map)# set ip next-hop 192.168.1.1`	
Step 8	**exit** **Example:** `switch(config-route-map)# exit`	Exit command mode.
Step 9	**interface vlan** *vlan-id* **Example:** `switch(config)# interface vlan 2003`	Creates a VLAN interface and enters interface configuration mode. The range is from 1 and 4094.This is the primary VLAN.
Step 10	**ip address** *ip-addr* **Example:** `switch(config-if)# ip address 10.0.0.1/8`	Configures an IP address for the interface.
Step 11	**no ip redirects** **Example:** `switch(config-if)# no ip redirects`	Needs to be configured on all unnumbered primary and secondary VLAN interfaces.
Step 12	(Optional) **ip policy route-map pbr-sample** **Example:** `switch(config-if)# ip policy route-map pbr-sample`	Enter this command if you want to apply a PBR policy on the unnumbered Primary/Secondary VLAN interface.
Step 13	**exit** **Example:** `switch(config-if)# exit`	Exit command mode.
Step 14	**hsrp version 2** **Example:** `switch(config-if)# hsrp version 2`	Set the HSRP version.
Step 15	**hsrp***group-num* **Example:** `switch(config-if)# hsrp 200`	Set the HSRP group number.
Step 16	**name** *name-val* **Example:** `switch(config-if-hsrp)# name master`	Configure the redundancy name string.
Step 17	**ip** *ip-addr* **Example:**	Configures an IP address.

	Command or Action	Purpose
	`switch(config-if-hsrp)# ip 10.0.0.100`	
Step 18	**no shutdown** **Example:** `switch(config-if-hsrp)# no shutdown`	Negates shutdown.

5.6.11 Configuring IP Unnumbered on SVI Secondary VLAN on the Gateway

This procedure configures IP unnumbered on the secondary SVI in the gateway.

SUMMARY STEPS

1. **configure terminal**
2. **interface vlan** *vlan-list*
3. **ip unnumbered vlan** *primary-vlan-id*
4. (Optional) **ip policy route-map pbr-sample**
5. **no ip redirects**
6. **hsrp version 2**
7. **hsrp** *group-num*
8. **follow** *name*
9. **ip** *ip-addr*
10. **no shutdown**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enter configuration mode.
Step 2	**interface vlan** *vlan-list* **Example:** `switch(config)# interface vlan 2001`	Creates a VLAN interface and enters interface configuration mode. The range is from 1 to 4094. This is the secondary VLAN.
Step 3	**ip unnumbered vlan** *primary-vlan-id* **Example:** `switch(config-if)# ip unnumbered vlan 2003`	Enables IP processing on an interface without assigning an explicit IP address to an interface.
Step 4	(Optional) **ip policy route-map pbr-sample** **Example:** `switch(config-if)# ip policy route-map pbr-sample`	Enter this command if you want to apply a PBR policy on the unnumbered Primary/Secondary VLAN interface.
Step 5	**no ip redirects**	Needs to be configured on all unnumbered primary and

	Command or Action	Purpose
	Example: `switch(config-if)# no ip redirects`	secondary VLAN interfaces.
Step 6	**hsrp version 2** **Example:** `switch(config-if)# hsrp version 2`	Set the HSRP version.
Step 7	**hsrp** *group-num* **Example:** `switch(config-if)# hsrp 200`	Set the HSRP group number.
Step 8	**follow** *name* **Example:** `switch(config-if-hsrp)# follow master`	Configure the group to be followed.
Step 9	**ip** *ip-addr* **Example:** `switch(config-if-hsrp)# ip 10.0.0.100`	Enters HRSP IPv4 and sets the virtual IP address.
Step 10	**no shutdown** **Example:** `switch(config-if-hsrp)# no shutdown`	Negate shutdown.

5.6.12 Assigning an Interface to a VRF

You can add a Layer 3 interface to a VRF.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface-type number*
3. **vrf member** *vrf-name*
4. **ip address** *ip-prefix/length*
5. **show vrf** [*vrf-name*] **interface** *interface-type number*
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	**interface** *interface-type number* **Example:**	Enters interface configuration mode.

	Command or Action	Purpose

	`switch(config)# `**`interface loopback 0`**
	`switch(config-if)#`	

Step 3	**vrf member** *vrf-name* **Example:** `switch(config-if)# `**`vrf member RemoteOfficeVRF`**	Adds this interface to a VRF.
Step 4	**ip address** *ip-prefix/length* **Example:** `switch(config-if)# `**`ip address 192.0.2.1/16`**	Configures an IP address for this interface. You must do this step after you assign this interface to a VRF.
Step 5	**show vrf** [*vrf-name*] **interface** *interface-type number* **Example:** `switch(config-vrf)# `**`show vrf Enterprise interface loopback 0`**	(Optional) Displays VRF information.
Step 6	**copy running-config startup-config** **Example:** `switch(config-if)# `**`copy running-config startup-config`**	(Optional) Saves the configuration change.

Example

This example shows how to add a Layer 3 interface to the VRF:

```
switch# configure terminal
switch(config)# interface loopback 0
switch(config-if)# vrf member RemoteOfficeVRF
switch(config-if)# ip address 209.0.2.1/16
switch(config-if)# copy running-config startup-config
```

5.6.13 Configuring a MAC-Embedded IPv6 Address

You can configure a MAC-embedded IPv6 (MEv6) address.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*
3. **no switchport**
4. **mac-address ipv6-extract**
5. **ipv6 address** *ip-address/length*
6. **ipv6 nd mac-extract** [**exclude nud-phase**]
7. (Optional) **show ipv6 icmp interface** *type slot/port*
8. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet 1/3`
	`switch(config-if)#`
		Enters the interface configuration mode for the specified
		interface.

Step 3	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 interface and deletes any configuration specific to Layer 2 on this interface. **Note** To convert a Layer 3 interface back into a Layer 2 interface, use the **switchport** command.
Step 4	**mac-address ipv6-extract** **Example:** `switch(config-if)# mac-address ipv6-extract`	Extracts the MAC address embedded in the IPv6 address configured on the interface. **Note** The MEv6 configuration is currently not supported with the EUI-64 format of IPv6 address.
Step 5	**ipv6 address** *ip-address/length* **Example:** `switch(config-if)# ipv6 address 2002:1::10/64`	Configures an IPv6 address for this interface.
Step 6	**ipv6 nd mac-extract** [**exclude nud-phase**] **Example:** `switch(config-if)# ipv6 nd mac-extract`	Extracts the next-hop MAC address embedded in a next-hop IPv6 address. The **exclude nud-phase** option blocks packets during the ND phase only. When the **exclude nud-phase** option is not specified, packets are blocked during both ND and neighbor unreachability detection (NUD) phases.
Step 7	(Optional) **show ipv6 icmp interface** *type slot/port* **Example:** `switch(config-if)# show ipv6 icmp interface ethernet 1/3`	Displays IPv6 Internet Control Message Protocol version 6 (ICMPv6) interface information.

	Command or Action	Purpose
Step 8	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Example

This example shows how to configure a MAC-embedded IPv6 address with ND mac-extract enabled:

```
switch# configure terminal
switch(config)# interface ethernet
1/3 switch(config-if)# no switchport
switch(config-if)# mac-address ipv6-extract
switch(config-if)# ipv6 address 2002:1::10/64
switch(config-if)# ipv6 nd mac-extract switch(config-
if)# show ipv6 icmp interface ethernet 1/3
ICMPv6 Interfaces for VRF "default"
Ethernet1/3, Interface status: protocol-up/link-up/admin-up
  IPv6 address: 2002:1::10
  IPv6 subnet:  2002:1::/64
  IPv6 interface DAD state:  VALID
  ND mac-extract : Enabled
  ICMPv6 active timers:
      Last Neighbor-Solicitation sent: 00:01:39
      Last Neighbor-Advertisement sent: 00:01:40
      Last Router-Advertisement sent: 00:01:41
      Next Router-Advertisement sent in: 00:03:34
  Router-Advertisement parameters:
      Periodic interval: 200 to 600 seconds
      Send "Managed Address Configuration" flag: false
      Send "Other Stateful Configuration" flag: false
      Send "Current Hop Limit" field: 64
      Send "MTU" option value: 1500
      Send "Router Lifetime" field: 1800 secs
      Send "Reachable Time" field: 0 ms
      Send "Retrans Timer" field: 0 ms
      Suppress RA: Disabled
      Suppress MTU in RA: Disabled
  Neighbor-Solicitation parameters:
      NS retransmit interval: 1000 ms
  ICMPv6 error message parameters:
      Send redirects: true
      Send unreachables: false
  ICMPv6-nd Statisitcs (sent/received):
      RAs: 3/0, RSs: 0/0, NAs: 2/0, NSs: 7/0, RDs: 0/0
      Interface statistics last reset: never
```

This example shows how to configure a MAC-embedded IPv6 address with ND mac-extract (excluding NUD phase) enabled:

```
switch# configure terminal
switch(config)# interface ethernet
1/5 switch(config-if)# no switchport
switch(config-if)# mac-address ipv6-extract
switch(config-if)# ipv6 address 2002:2::10/64
switch(config-if)# ipv6 nd mac-extract exclude nud-phase
switch(config-if)# show ipv6 icmp interface ethernet 1/5
```

```
ICMPv6 Interfaces for VRF "default"
Ethernet1/5, Interface status: protocol-up/link-up/admin-up
  IPv6 address: 2002:2::10
  IPv6 subnet:  2002:2::/64
  IPv6 interface DAD state:  VALID
  ND mac-extract : Enabled (Excluding NUD Phase)
  ICMPv6 active timers:
      Last Neighbor-Solicitation sent: 00:06:45
      Last Neighbor-Advertisement sent: 00:06:46
      Last Router-Advertisement sent: 00:02:18
      Next Router-Advertisement sent in: 00:02:24
  Router-Advertisement parameters:
      Periodic interval: 200 to 600 seconds
      Send "Managed Address Configuration" flag: false
      Send "Other Stateful Configuration" flag: false
      Send "Current Hop Limit" field: 64
      Send "MTU" option value: 1500
      Send "Router Lifetime" field: 1800 secs
      Send "Reachable Time" field: 0 ms
      Send "Retrans Timer" field: 0 ms
      Suppress RA: Disabled
      Suppress MTU in RA: Disabled
  Neighbor-Solicitation parameters:
      NS retransmit interval: 1000 ms
  ICMPv6 error message parameters:
      Send redirects: true
      Send unreachables: false
  ICMPv6-nd Statisitcs (sent/received):
      RAs: 6/0, RSs: 0/0, NAs: 2/0, NSs: 7/0, RDs: 0/0
      Interface statistics last reset: never
```

5.6.14 Configuring a DHCP Client on an Interface

You can configure the DHCP client on an SVI, a management interface, or a physical Ethernet interface for IPv4 or IPv6 address

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **interface ethernet** *type slot/port* | **mgmt** *mgmt-interface-number* | **vlan** *vlan id*
3. switch(config-if)# [**no**] **ipv6 address use-link-local-only**
4. switch(config-if)# [**no**] [**ip** | **ipv6] address dhcp**
5. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface ethernet** *type slot/port* | **mgmt** *mgmt-interface-number* | **vlan** *vlan id*	Creates a physical Ethernet interface, a management interface, or a VLAN interface. The range of *vlan id* is from 1 to 4094.
Step 3	switch(config-if)# [**no**] **ipv6 address use-link-local-only**	Prepares for request to the DHCP server. **Note** This command is only required for an IPv6 address.

	Command or Action	Purpose
Step 4	switch(config-if)# [**no**] [**ip** | **ipv6] address dhcp**	Requests the DHCP server for an IPv4 or IPv6 address. The **no** form of this command removes any address that was acquired.
Step 5	(Optional) switch(config)# **copy running-config startup-config**	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Example

This example shows how to configure the IP address of a DHCP client on an SVI:

```
switch# configure terminal
switch(config)# interface vlan 15
switch(config-if)# ip address dhcp
```

This example shows how to configure an IPv6 address of a DHCP client on a management interface:

```
switch# configure terminal
switch(config)# interface mgmt 0
switch(config-if)# ipv6 address use-link-local-only
switch(config-if)# ipv6 address dhcp
```

5.7 Verifying the Layer 3 Interfaces Configuration

To display the Layer 3 configuration, perform one of the following tasks:

Command	Purpose
show interface ethernet *slot/port*	Displays the Layer 3 interface configuration, status, and counters (including the 5-minute exponentially decayed moving average of inbound and outbound packet and byte rates).
show interface ethernet *slot/port* **brief**	Displays the Layer 3 interface operational status.
show interface ethernet *slot/port* **capabilities**	Displays the Layer 3 interface capabilities, including port type, speed, and duplex.
show interface ethernet *slot/port* **description**	Displays the Layer 3 interface description.
show interface ethernet *slot/port* **status**	Displays the Layer 3 interface administrative status, port mode, speed, and duplex.
show interface ethernet *slot/port.number*	Displays the subinterface configuration, status, and counters (including the f-minute exponentially decayed moving average of inbound and outbound

Command	Purpose
	packet and byte rates).
show interface port-channel *channel-id.number*	Displays the port-channel subinterface configuration, status, and counters (including the 5-minute exponentially decayed moving average of inbound and outbound packet and byte rates).
show interface loopback *number*	Displays the loopback interface configuration, status, and counters.
show interface loopback *number* **brief**	Displays the loopback interface operational status.
show interface loopback *number* **description**	Displays the loopback interface description.
show interface loopback *number* **status**	Displays the loopback interface administrative status and protocol status.
show interface vlan *number*	Displays the VLAN interface configuration, status, and counters.
show interface vlan *number* **brief**	Displays the VLAN interface operational status.
show interface vlan *number* **description**	Displays the VLAN interface description.
show interface vlan *number* **status**	Displays the VLAN interface administrative status and protocol status.
show ip interface brief	Displays interface address and interface status (numbered/unnumbered).
show ip route	Displays routes learned via OSPF or ISIS. (Includes addresses for best unicast and multicast next-hop.)

5.8 Monitoring the Layer 3 Interfaces

Use the following commands to display Layer 3 statistics:

Command	Purpose
load- interval {**interval** *seconds* {**1** | **2** | **3**}}	Inspur CN12900 Series devices set three different sampling intervals to bit-rate and packet-rate statistics. The range for VLAN network interface is 60 to 300 seconds, and the range for Layer interfaces is 30 to 300 seconds.
show interface ethernet *slot/port* **counters**	Displays the Layer 3 interface statistics (unicast, multicast, and broadcast).
show interface ethernet *slot/port* **counters brief**	Displays the Layer 3 interface input and output counters.

Command	Purpose
show interface ethernet errors *slot/port* **detailed** [**all**]	Displays the Layer 3 interface statistics. You can optionally include all 32-bit and 64-bit packet and byte counters (including errors).
show interface ethernet errors *slot/port* **counters errors**	Displays the Layer 3 interface input and output errors.
show interface ethernet errors *slot/port* **counters snmp**	Displays the Layer 3 interface counters reported by SNMP MIBs.
show interface ethernet *slot/port.number* **counters**	Displays the subinterface statistics (unicast, multicast, and broadcast).
show interface port-channel *channel-id.number* **counters**	Displays the port-channel subinterface statistics (unicast, multicast, and broadcast).
show interface loopback *number* **counters**	Displays the loopback interface input and output counters (unicast, multicast, and broadcast).
show interface loopback *number* **detailed** [**all**]	Displays the loopback interface statistics. You can optionally include all 32-bit and 64-bit packet and byte counters (including errors).
show interface loopback *number* **counters errors**	Displays the loopback interface input and output errors.
show interface vlan *number* **counters**	Displays the VLAN interface input and output counters (unicast, multicast, and broadcast).
show interface vlan *number* **counters detailed** [**all**]	Displays the VLAN interface statistics. You can optionally include all Layer 3 packet and byte counters (unicast and multicast).
show interface vlan *number* **counters snmp**	Displays the VLAN interface counters reported by SNMP MIBs.

5.9 Configuration Examples for Layer 3 Interfaces

This example shows how to configure Ethernet subinterfaces:

```
interface ethernet 2/1.10
description Layer 3
ip address 192.0.2.1/8
```

This example shows how to configure a loopback interface:

```
interface loopback 3 ip
address 192.0.2.2/32
```

5.9.1 Example of Changing VRF Membership for an Interface

- Enable Layer 3 configuration retention when changing VRF membership.

```
switch# configure terminal
switch(config)# system vrf-member-change retain-l3-config
Warning: Will retain L3 configuration when vrf member change on interface.
```

- Verify Layer 3 retention.

```
switch# show running-config | include vrf-member-change
system vrf-member-change retain-l3-config
```

- Configure the SVI interface with Layer 3 configuration as VRF "blue".

```
switch# configure terminal
switch(config)# show running-config interface vlan 2002
interface Vlan2002
description TESTSVI
no shutdown
mtu 9192
vrf member blue
no ip redirects
ip address 192.168.211.2/27
ipv6 address 2620:10d:c041:12::2/64
ipv6 link-local fe80::1
ip router ospf 1 area 0.0.0.0
ipv6 router ospfv3 1 area 0.0.0.0
hsrp version 2
hsrp 2002
preempt delay minimum 300 reload 600
priority 110 forwarding-threshold lower 1 upper 110
ip 192.168.211.1
hsrp 2002 ipv6
preempt delay minimum 300 reload 600
priority 110 forwarding-threshold lower 1 upper 110
ip 2620:10d:c041:12::1
```

- Change the SVI interface VRF to "red".

```
switch# configure terminal

Enter configuration commands, one per line. End with
CNTL/Z. switch(config)# interface vlan 2002
switch(config-if)# vrf member red
Warning: Retain-L3-config is on, deleted and re-added L3 config on interface Vlan2002
```

- Verify SVI interface after VRF change.

```
switch# configure terminal
switch(config)# show running-config interface vlan 2002
interface Vlan2002
description TESTSVI
no shutdown
mtu 9192
```

```
vrf member red
no ip redirects
ip address 192.168.211.2/27
ipv6 address 2620:10d:c041:12::2/64
ipv6 link-local fe80::1
ip router ospf 1 area 0.0.0.0
ipv6 router ospfv3 1 area
0.0.0.0 hsrp version 2
hsrp 2002
preempt delay minimum 300 reload 600
priority 110 forwarding-threshold lower 1 upper 110
ip 192.168.211.1
hsrp 2002 ipv6
preempt delay minimum 300 reload 600
priority 110 forwarding-threshold lower 1 upper 110
ip 2620:10d:c041:12::1
```

5.10 Related Documents

Related Documents	**Document Title**
IP	*Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide*
VLANs	*Inspur CN12900 Series INOS-CN Layer 2 Switching Configuration Guide*

CHAPTER 6 Configuring Bidirectional Forwarding Detection

6.1 About BFD

BFD is a detection protocol designed to provide fast forwarding-path failure detection times for media types, encapsulations, topologies, and routing protocols. You can use BFD to detect forwarding path failures at a uniform rate, rather than the variable rates for different protocol hello mechanisms. BFD makes network profiling and planning easier and reconvergence time consistent and predictable.

BFD provides subsecond failure detection between two adjacent devices and can be less CPU-intensive than protocol hello messages because some of the BFD load can be distributed onto the data plane on supported modules.

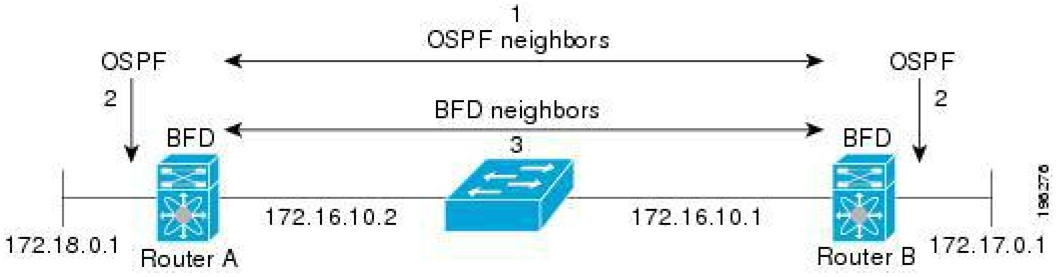
6.1.1 Asynchronous Mode

Inspur INOS-CN supports the BFD asynchronous mode, which sends BFD control packets between two adjacent devices to activate and maintain BFD neighbor sessions between the devices. You configure BFD on both devices (or BFD neighbors). Once BFD has been enabled on the interfaces and on the appropriate protocols, Inspur INOS-CN creates a BFD session, negotiates BFD session parameters, and begins to send BFD control packets to each BFD neighbor at the negotiated interval. The BFD session parameters include the following:

- Desired minimum transmit interval—The interval at which this device wants to send BFD hello messages.
- Required minimum receive interval—The minimum interval at which this device can accept BFD hello messages from another BFD device.
- Detect multiplier—The number of missing BFD hello messages from another BFD device before this local device detects a fault in the forwarding path.

The following figure shows how a BFD session is established. The figure shows a simple network with two routers running Open Shortest Path First (OSPF) and BFD. When OSPF discovers a neighbor (1), it sends a request to the local BFD process to initiate a BFD neighbor session with the OSPF neighbor router (2). The BFD neighbor session with the OSPF neighbor router is now established (3).

Figure 6： Establishing a BFD Neighbor Relationship



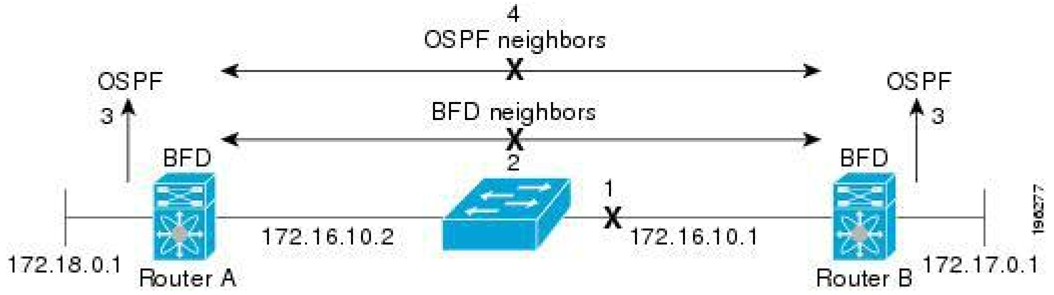
6.1.2 BFD Detection of Failures

Once a BFD session has been established and timer negotiations are complete, BFD neighbors send BFD control packets that act in the same manner as an IGP hello protocol to detect liveliness, except at a more accelerated rate. BFD detects a failure, but the protocol must take action to bypass a failed peer.

BFD sends a failure detection notice to the BFD-enabled protocols when it detects a failure in the forwarding path. The local device can then initiate the protocol recalculation process and reduce the overall network convergence time.

The following figure shows what happens when a failure occurs in the network (1). The BFD neighbor session with the OSPF neighbor router is torn down (2). BFD notifies the local OSPF process that the BFD neighbor is no longer reachable (3). The local OSPF process tears down the OSPF neighbor relationship (4). If an alternative path is available, the routers immediately start converging on it.

Figure 7：Tearing Down an OSPF Neighbor Relationship



6.1.3 Distributed Operation

Inspur INOS-CN can distribute the BFD operation to compatible modules that support BFD. This process offloads the CPU load for BFD packet processing to the individual modules that connect to the BFD neighbors. All BFD session traffic occurs on the module CPU. The module informs the supervisor when a BFD failure is detected.

6.1.4 BFD Echo Function

The BFD echo function sends echo packets from the forwarding engine to the remote BFD neighbor. The BFD neighbor forwards the echo packet back along the same path in order to perform detection; the BFD neighbor does not participate in the actual forwarding of the echo packets. The echo function and the forwarding engine are responsible for the detection process. BFD can use the slow timer to slow down the asynchronous session when the echo function is enabled and reduce the number of BFD control packets that are sent between two BFD neighbors. Also, the forwarding engine tests the forwarding path on the remote (neighbor) system without involving the remote system, so there is less interpacket delay variability and faster failure detection times.

The echo function is without asymmetry when both BFD neighbors are running echo function.

6.1.5 Security

Inspur INOS-CN uses the packet Time to Live (TTL) value to verify that the BFD packets came from an adjacent BFD peer. For all asynchronous and echo request packets, the BFD neighbor sets the TTL value to 255 and the local BFD process verifies the TTL value as 255 before processing the incoming packet. For the echo response packet, BFD sets the TTL value to 254.

You can configure SHA-1 authentication of BFD packets.

6.1.6 High Availability

BFD supports stateless restarts. After a reboot or supervisor switchover, Inspur INOS-CN applies the running configuration and BFD immediately sends control packets to the BFD peers.

6.1.7 Virtualization Support

BFD supports virtual routing and forwarding instances (VRFs). VRFs exist within virtual device contexts (VDCs). By default, Inspur INOS-CN places you in the default VDC and default VRF.

6.2 Licensing Requirements for BFD

The following table shows the licensing requirements for this feature:

Product	License Requirement
Inspur INOS-CN	BFD requires no license. Any feature not included in a license package is bundled with the INOS-CN image and is provided at no extra charge to you.

6.3 Prerequisites for BFD

BFD has the following prerequisites:

- You must enable the BFD feature.
- Disable Internet Control Message Protocol (ICMP) redirect messages on BFD-enabled interfaces.
- Disable the IP packet verification check for identical IP source and destination addresses.
- See other detailed prerequisites that are listed with the configuration tasks.

6.4 Guidelines and Limitations

BFD has the following configuration guidelines and limitations:

- The QSFP 40/100-G BiDi comes up in the highest possible speed available on the port. If you need to connect to 40-G SR4 BiDi, the speed on the 40/100-G BiDi needs to be set to 40 G.
- Beginning with Inspur INOS-CN, QSFP-40/100-SRBD comes up in the speed of 100-G and interoperate with other QSFP-40/100-SRBD at either 40-G or 100-G speed on Inspur CN12900 Switches with the CN129-X6136YC-R line card. The QSFP-40/100-SRBD can also interoperate with QSFP-40G-SR-BD at 40G speeds. However to operate at 40G speed, you must configure the speed as 40G.
- The dual rate BiDi does not support auto speed sensing. With speed auto, the QSFP always defaults to 100 G rather than the speed of the link partner.
- **show** commands with the **internal** keyword are not supported.
- Inspur CN12900 Series switches supports BFD per-member link.
- BFD per-member link support is added on Inspur CN12900 Series switches.
- BFD supports BFD version 1.
- BFD supports IPv4 and IPv6.
- BFD supports OSPFv3.
- BFD supports IS-ISv6.
- BFD supports BGPv6.
- BFD supports EIGRPv6.
- BFD supports only one session per address family, per interface.
- BFD supports single-hop BFD.
- BFD for BGP supports single-hop EBGP and iBGP peers.
- BFD supports keyed SHA-1 authentication.
- BFD supports the following Layer 3 interfaces—physical interfaces, port channels, subinterfaces, and VLAN interfaces.
- BFD depends on a Layer 3 adjacency information to discover topology changes, including Layer 2 topology changes. A BFD session on a VLAN interface (SVI) may not be up after the convergence of the Layer 2 topology if there is no Layer 3 adjacency information available.
- For BFD on a static route between two devices, both devices must support BFD. If one or both of the devices

do not support BFD, the static routes are not programmed in the Routing Information Base (RIB).

- Port channel configuration limitations:
- For Layer 3 port channels used by BFD, you must enable LACP on the port channel.
- For Layer 2 port channels used by SVI sessions, you must enable LACP on the port channel.
- SVI limitations:
- An ASIC reset causes traffic disruption for other ports and it can cause the SVI sessions on the other ports to flap. For example, if the carrier interface is a virtual port channel (vPC), BFD is not supported over the SVI interface and it could cause a trigger for an ASIC reset. When a BFD session is over

SVI using virtual port channel (vPC) peer-link, the BFD echo function is not supported. You must disable the BFD echo function for all sessions over SVI between vPC peer nodes.

An SVI on the Inspur series switches should not be configured to establish a BFD neighbor adjacency with a device connected to it via a vPC. This is because the BFD keepalives from the neighbour, if sent over the vPC member link connected to the vPC peer-switch, do not reach this SVI causing the BFD adjacency to fail.

- When you change the topology (for example, add or delete a link into a VLAN, delete a member from a Layer 2 port channel, and so on), the SVI session could be affected. It may go down first and then come up after the topology discovery is finished.
- When a BFD session is over SVI using virtual port-channel (vPC) peer-link (either BCM or GEM based ports), the BFD echo function is not supported. You must disable the BFD echo function for all sessions over SVI between vPC peer nodes using the **no bfd echo** command at the SVI configuration level.

Tip	If you do not want the SVI sessions to flap and you need to change the topology, you can disable the BFD feature before making the changes and reenable BFD after the changes have been made. You can also configure the BFD timer to be a large value (for example, 5 seconds), and change it back to a fast timer after the above events complete.

- When you configure the BFD Echo function on the distributed Layer 3 port channels, reloading a member module flaps the BFD session hosted on that module, which results in a packet loss.

If you connect the BFD peers directly without a Layer 2 switch in between, you can use the BFD per-link mode as an alternative solution.

- When you specify a BFD neighbor prefix in the **clear** {**ip** | **ipv6**} **route** *prefix* command, the BFD echo session will flap.
- The **clear** {**ip** | **ipv6**} **route *** command causes BFD echo sessions to flap.
- HSRP for IPv4 is supported with BFD.
- BFD packets generated by the Inspur INOS-CN device linecards are sent with COS 6/DSCP CS6. The DSCP/COS values for BFD packets are not user configurable.
- When configuring BFDv6 in no-bfd-echo mode, it is recommended to run with timers of 150 ms with a multiplier of 3.
- BFDv6 is not supported for VRRPv3 and HSRP for v6.
- IPv6 **eigrp bfd** cannot be disabled on an interface.
- Port channel configuration notes:

When the BFD per-link mode is configured, the BFD echo function is not supported. You must disable the BFD echo function using the **no bfd echo** command before configuring the **bfd per-link** command.

- Configuring BFD per-link with link-local is not supported.
- Forming BFD neighbours on a vPC VLAN through an orphan port is not supported on Inspur CN12900 Series Switches

6.5 Default Settings

The following table lists the default settings for BFD parameters.

Table 8：Default BFD Parameters

Parameters	Default
BFD feature	Disabled
Required minimum receive interval	50 milliseconds
Desired minimum transmit interval	50 milliseconds
Detect multiplier	3
Echo function	Enabled
Mode	Asynchronous
Port-channel	Logical mode (one session per source-destination pair address)
Slow timer	2000 milliseconds
Startup timer	5 seconds

6.6 Configuring BFD

6.6.1 Configuration Hierarchy

You can configure BFD at the global level and at the interface level. The interface configuration overrides the global configuration.

For physical ports that are members of a port channel, the member port inherits the master port channel BFD configuration.

6.6.2 Task Flow for Configuring BFD

Follow these steps in the following sections to configure BFD:

- Enabling the BFD Feature.
- Configuring Global BFD Parameters or Configuring BFD on an Interface.

6.6.3 Enabling the BFD Feature

You must enable the BFD feature before you can configure BFD on an interface and protocol.

SUMMARY STEPS

1. **configure terminal**
2. **feature bfd**
3. **show feature | include bfd**
4. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	**feature bfd** **Example:** `switch(config)# feature bfd`	Enables the BFD feature.
Step 3	**show feature | include bfd** **Example:** `switch(config)# show feature | include bfd`	(Optional) Displays enabled and disabled features.
Step 4	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves the configuration change.

6.6.4 Configuring Global BFD Parameters

You can configure the BFD session parameters for all BFD sessions on the device. The BFD session parameters are negotiated between the BFD peers in a three-way handshake.

See the Configuring BFD on an Interface section to override these global session parameters on an interface.

Before you begin

Enable the BFD feature.

SUMMARY STEPS

1. **configure terminal**
2. **bfd interval** *mintx* **min_rx** *msec* **multiplier** *value*
3. **bfd slow-timer** [*interval*]
4. [**no**] **bfd startup-timer** [*seconds*]
5. **bfd echo-interface loopback** *interface number*
6. **show running-config bfd**
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	**bfd interval** *mintx* **min_rx** *msec* **multiplier** *value*	Configures the BFD session parameters for all BFD sessions

	Command or Action	Purpose

	Example:
	`switch(config)# bfd interval 50 min_rx`
	`50 multiplier 3`
		on the device. This command overrides these values by configuring the BFD session parameters on an interface. The *mintx* and *msec* range is from 50 to 999 milliseconds and the default is 50. The multiplier range is from 1 to 50. The multiplier default is 3.

Step 3	**bfd slow-timer** [*interval*] **Example:** `switch(config)# bfd slow-timer 2000`	Configures the slow timer used in the echo function. This value determines how fast BFD starts up a new session and at what speed the asynchronous sessions use for BFD control packets when the echo function is enabled. The slow-timer value is used as the new control packet interval, while the echo packets use the configured BFD intervals. The echo packets are used for link failure detection, while the control packets at the slower rate maintain the BFD session. The range is from 1000 to 30000 milliseconds. The default is 2000.
Step 4	[**no**] **bfd startup-timer** [*seconds*] **Example:** `switch(config)# bfd startup-timer 20`	Configures the BFD startup timer, which delays the startup time for BFD sessions in order to give the routes that are being used by local and remote routers time to settle down in the hardware. Using this feature can prevent BFD flaps in higher scale scenarios. The range is from 0 to 30 seconds. The default is 5 seconds. The **bfd startup-timer 0** command disables the BFD startup timer. The **no bfd startup-timer** command sets the BFD startup timer to 5 seconds (the default value).
Step 5	**bfd echo-interface loopback** *interface number*	Configures the interface used for Bidirectional Forwarding

	Command or Action	Purpose
	Example: `switch(config)# bfd echo-interface loopback 1 3`	Detection (BFD) echo frames. This command changes the source address for the echo packets to the one configured on the specified loopback interface. The interface number range is from 0 to 1023.
Step 6	**show running-config bfd** **Example:** `switch(config)# show running-config bfd`	(Optional) Displays the BFD running configuration.
Step 7	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves the configuration change.

6.6.5 Configuring BFD on an Interface

You can configure the BFD session parameters for all BFD sessions on an interface. The BFD session parameters are negotiated between the BFD peers in a three-way handshake.

This configuration overrides the global session parameters for the configured interface.

Before you begin

Ensure that Internet Control Message Protocol (ICMP) redirect messages are disabled on BFD-enabled interfaces. Use the **no ip redirects** command or the **no ipv6 redirects** command on the interface.

Enable the BFD feature. See the Enabling the BFD Feature section.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *int-if*
3. **bfd interval** *mintx* **min_rx** *msec* **multiplier** *value*
4. **bfd authentication keyed-sha1 keyid** *id* **key** *ascii_key*
5. **show running-config bfd**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	**interface** *int-if* **Example:**	Enters interface configuration mode. Use the ? keyword to display the supported interfaces.

	Command or Action	Purpose

	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`	

Step 3	**bfd interval** *mintx* **min_rx** *msec* **multiplier** *value*
	Example:
	`switch(config-if)# bfd interval 50`
	`min_rx 50 multiplier 3`
		Configures the BFD session parameters for all BFD sessions on the device. This command overrides these values by configuring the BFD session parameters on an interface.
		The *mintx* and *msec* range is from 50 to 999 milliseconds and the default is 50. The multiplier range is from 1 to 50.
		The multiplier default is 3.

Step 4	**bfd authentication keyed-sha1 keyid** *id* **key** *ascii_key*
	Example:
	`switch(config-if)# bfd authentication`
	`keyed-sha1 keyid 1 ascii_key Inspur123`
		(Optional) Configures SHA-1 authentication for all BFD sessions on the interface. The *ascii_key* string is a secret key shared among BFD peers. The *id* value, a number between 0 and 255, is assigned to this particular *ascii_key*.
		BFD packets specify the key by *id*, allowing the use of multiple active keys.
		To disable SHA-1 authentication on the interface, use the **no** form of the command.

Step 5	**show running-config bfd** **Example:** `switch(config-if)# show running-config bfd`	(Optional) Displays the BFD running configuration.

Step 6	**copy running-config startup-config**
	Example:
	`switch(config-if)# copy running-config`
	`startup-config`
		(Optional) Saves the configuration change.

6.6.6 Configuring BFD on a Port Channel

You can configure the BFD session parameters for all BFD sessions on a port channel. If per-link mode is used for Layer 3 port channels, BFD creates a session for each link in the port channel and provides an aggregate result to client protocols. For example, if the BFD session for one link on a port channel is up, BFD informs client protocols, such as OSPF, that the port channel is up. The BFD session parameters are negotiated between the BFD peers in a three-way handshake.

This configuration overrides the global session parameters for the configured port channel. The member ports of the port channel inherit the port channel BFD session parameters.

Before you begin

Ensure that you enable LACP on the port channel before you enable BFD.

Ensure that Internet Control Message Protocol (ICMP) redirect messages are disabled on BFD-enabled interfaces. Use the **no ip redirects** command on the interface.

Enable the BFD feature. See the Enabling the BFD Feature section.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *number*
3. **bfd per-link**
4. **bfd interval** *mintx* **min_rx** *msec* **multiplier** *value*
5. **bfd authentication keyed-sha1 keyid** *id* **key** *ascii_key*
6. **show running-config bfd**
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *number*
	Example:
	`switch(config)# interface port-channel 2`
	`switch(config-if)#`
		Enters port-channel configuration mode. Use the **?** keyword to display the supported number range.

Step 3	**bfd per-link** **Example:** `switch(config-if)# bfd per-link`	Configures the BFD sessions for each link in the port channel.

Step 4	**bfd interval** *mintx* **min_rx** *msec* **multiplier** *value*
	Example:
	`switch(config-if)# bfd interval 50`
	`min_rx 50 multiplier 3`
		(Optional) Configures the BFD session parameters for all BFD sessions on the port channel. This command overrides these values by configuring the BFD session parameters.
		The *mintx* and *msec* range is from 50 to 999 milliseconds and the default is 50. The multiplier range is from 1 to 50.
		The multiplier default is 3.

Step 5	**bfd authentication keyed-sha1 keyid** *id* **key** *ascii_key* **Example:**	(Optional) Configures SHA-1 authentication for all BFD sessions on the interface. The *ascii_key* string is a secret

	Command or Action	Purpose

	`switch(config-if)# bfd authentication`
	`keyed-sha1 keyid 1 ascii_key Inspur123`
		key shared among BFD peers. The *id* value, a number
		between 0 and 255, is assigned to this particular *ascii_key*.
		BFD packets specify the key by *id*, allowing the use of
		multiple active keys.
		To disable SHA-1 authentication on the interface, use the
		no form of the command.

Step 6	**show running-config bfd** **Example:** `switch(config-if)# show running-config bfd`	(Optional) Displays the BFD running configuration.
Step 7	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves the configuration change.

6.6.7 Configuring the BFD Echo Function

You can configure the BFD echo function on one or both ends of a BFD-monitored link. The echo function slows down the required minimum receive interval, based on the configured slow timer. The RequiredMinEchoRx BFD session parameter is set to zero if the echo function is disabled. The slow timer becomes the required minimum receive interval if the echo function is enabled.

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Configure the BFD session parameters. See the Configuring Global BFD Parameters section on or the Configuring BFD on an Interface section.

Ensure that Internet Control Message Protocol (ICMP) redirect messages are disabled on BFD-enabled interfaces. Use the **no ip redirects** command on the interface.

Ensure that the IP packet verification check for identical IP source and destination addresses is disabled. Use the **no hardware ip verify address identical** command. See the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide for more information about this command.

SUMMARY STEPS

1. **configure terminal**
2. **bfd slow-timer** *echo-interval*
3. **interface** *int-if*
4. **bfd echo**
5. **show running-config bfd**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	**bfd slow-timer** *echo-interval* **Example:** `switch(config)# bfd slow-timer 2000`	Configures the slow timer used in the echo function. This value determines how fast BFD starts up a new session and is used to slow down the asynchronous sessions when the BFD echo function is enabled. This value overwrites the required minimum receive interval when the echo function is enabled. The range is from 1000 to 30000 milliseconds. The default is 2000.

Step 3	**interface** *int-if*
	Example:
	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode. Use the ? keyword to
		display the supported interfaces.

Step 4	**bfd echo** **Example:** `switch(config-if)# bfd echo`	Enables the echo function. The default is enabled.
Step 5	**show running-config bfd** **Example:** `switch(config-if)# show running-config bfd`	(Optional) Displays the BFD running configuration.
Step 6	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves the configuration change.

6.6.8 Configuring Per-Member Link BFD Sessions

BFD per-member link support is added on Inspur CN12900 Series switches. See the following sections for more information.

BFD Enhancement to Address Per-link Efficiency

The Bidirectional Forwarding (BFD) enhancement to address per-link efficiency, called as IETF Micro BFD, lets you configure the individual BFD sessions on every Link Aggregation Group (LAG) member interfaces (as defined in RFC 7130).

With this enhancement, the BFD sessions run on each member link of the port-channel. If BFD detects a link failure, the member link is removed from the forwarding table. This mechanism delivers faster failure detection as the BFD sessions are created on an individual port-channel interface.

The BFD sessions running on member links of the port-channel are called as Micro BFD sessions. You can configure RFC 7130 BFD over main port-channel interface, that performs bandwidth monitoring over LAG by having one Micro BFD session over each member. If any of the member port goes down, the port is removed from the forwarding table and this prevents traffic disruption on that member.

Micro BFD sessions are supported for both LACP and non-LACP based-port channels. For more information on how to configure Micro BFD sessions, see *Configuring Micro BFD Sessions*.

Limitations of the IETF Bidirectional Forwarding Detection

See the following limitations of the IETF Bidirectional Forwarding Detection:

- BFD Limitations
- It cannot co-exist with BFD over logical port-channels or proprietary BFD per-member links. BFD IPv6 logical/proprietary per-link session is also not supported when BFD IETF IPv4 is configured on PC.
- IETF BFD IPv6 is not supported.
- Echo functionality is not supported for Micro-BFD sessions.
- Port-channel interfaces should be directly connected between two switches that are running the BFD sessions. No intermediate Layer 2 switches are expected.
- EthPCM/LACP Limitations
- If a LACP port-channel has members in hot-standby state, BFD failure in one of the active links may not cause the hot-standby link to come up directly. Once the active link with BFD failure goes down, the hot-standby member becomes active. However, it may not be able to prevent the port-channel from going down before the hot-standby link comes up, in cases where port-channel min-link condition is hit.
- General Limitations:
- It is supported only on Layer 3 port-channels.
- It is not supported on the following:
- vPC
- Layer 3 sub-interfaces
- Layer 2 port-channels/Layer 2 Fabric Path
- FPC/HIF PC
- Layer 3 sub-interfaces
- VSI over port-channels

Guidelines for Migration/Configuration of IETF Per-Member Sessions:

See the following guidelines for migration/configuration of IETF per-member sessions:

- The logical BFD sessions that are created using the routing protocols over port-channel sub-interfaces (where RFC 7130 cannot run) are still supported. The main port-channel interface however does not support both logical and RFC 7130 sessions that co-exist. It can support only either of them.
- You can configure RFC 7130 BFD over the main port-channel interface that perform bandwidth monitoring over the LAG by having one Micro-BFD session over each member. If any of the member port goes down, BFD notifies it to the port-channel manager that removes the port from the LTL, thereby preventing blackholing of the traffic on that member.
- If the minimum number of links required to have the port-channel operationally *up* is not met in the above case, the port-channel is brought down by the port-channel manager. This in turn brings down the port-channel sub-interfaces if they are configured and thereby the logical BFD session also comes down notifying the routing protocol.
- When you are using RFC 7130 on the main port-channel and logical BFD on the sub-interfaces, the logical BFD session should be run with lesser aggressive timers than the RFC 7130 BFD session. You can have RFC 7130 configured on the port-channel interface or you can have it configured in conjunction with the logical BFD sessions on the port-channel sub-interfaces.

- When a proprietary per-link is configured, enabling IETF Micro-BFD sessions is not allowed on a port channel and vice-versa. You have to remove the proprietary per-link configuration. Current implementation of proprietary per-link does not allow changing the configuration (no per-link), if there is any BFD session that is bootstrapped by the applications. You need to remove the BFD tracking on the respective applications and remove per-link configuration. The migration path from the proprietary per-link to IETF Micro-BFD is as follows:
- Remove the BFD configuration on the applications.
- Remove the per-link configuration.
- Enable the IETF Micro-BFD command.
- Enable BFD on the applications.

The same migration path can be followed for proprietary BFD to IETF Micro-BFD on the main port-channel interface.

Configuring Port Channel Interface

Before you begin

Ensure that the BFD feature is enabled.

SUMMARY STEPS

1. switch(config)# **interface port-channel** *port-number*
2. switch(config)# **no switchport**

DETAILED STEPS

Step 1 switch(config)# **interface port-channel** *port-number*

Configures interface port-channel.

Step 2 switch(config)# **no switchport**

Configures interface as Layer 3 port-channel.

What to do next

- Configuring BFD Start Timer
- Enabling IETF Per-link BFD

Configuring BFD Start Timer

Complete the following steps to configure the BFD start timer:

SUMMARY STEPS

1. switch(config)# **interface port-channel***port-number*

DETAILED STEPS

switch(config)# **interface port-channel***port-number*

Configures the BFD start timer for a port-channel.

Note The default value is infinite (that is no timer is running). For start timer to work, configure start timer value before completing the port-channel BFD configurations (for example, before the port-channel bfd track-member-link and port-channel bfd destination are configured for Layer 3

What to do next

- Enabling IETF Per-link BFD
- Configuring BFD Destination IP Address

Enabling IETF Per-link BFD

SUMMARY STEPS

1. switch(config-if)# **port-channel bfd track-member-link**

DETAILED STEPS

switch(config-if)# **port-channel bfd track-member-link**
Enables IETF BFD on port-channel interface.

What to do next

- Configuring BFD Destination IP Address
- Verifying Micro BFD Session Configurations

Configuring BFD Destination IP Address

Complete the following steps to configure the BFD destination IP address:

SUMMARY STEPS

1. switch(config-if)# **port-channel bfd destination***ip-address*

DETAILED STEPS

switch(config-if)# **port-channel bfd destination***ip-address*
Configures an IPv4 address to be used for the BFD sessions on the member links.

What to do next

- Verifying Micro BFD Sessions Configuration

Verifying Micro BFD Session Configurations

Use the following commands to verify the Micro BFD session configurations.

SUMMARY STEPS

1. Displays the port-channel and port-channel member operational state.
2. switch# **show bfd neighbors**
3. switch# **show bfd neighbors details**
4. switch# **show tech-support bfd**
5. switch# **show tech-support lacp all**
6. switch# **show running-config interface port-channel** *port-channel-number*

DETAILED STEPS

Step 1 Displays the port-channel and port-channel member operational state. switch# **show port-channel summary**

Step 2 switch# **show bfd neighbors**
Displays Micro BFD sessions on port-channel members.

Step 3 switch# **show bfd neighbors details**
Displays BFD session for a port channel interface and the associated Micro BFD sessions on members.

Step 4 switch# **show tech-support bfd**

Displays the technical support information for BFD.

Step 5 switch# **show tech-support lacp all**
Displays the technical support information for Ethernet Port Manager, Ethernet Port-channel Manager, and LACP.

Step 6 switch# **show running-config interface port-channel** *port-channel-number*
Displays the running configuration information of the port-channel interface.

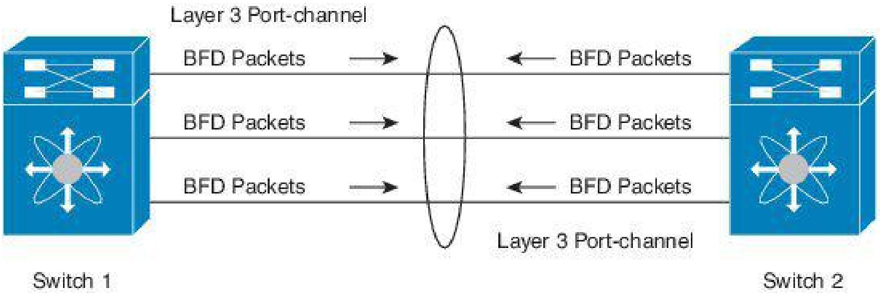
Examples: Configuring Micro BFD Sessions

See the following examples for configuring Micro BFD sessions.

Configuring Micro BFD Sessions

In this example, the following topology is used.

Figure 8：Configuring Micro BFD Session



The sample configuration of switch 1 is as follows:

```
feature bfd
configure terminal
    interface port-channel 10 port-channel
       bfd track-member-link
       port-channel bfd destination 10.1.1.2
       port-channel bfd start 60
       ip address 10.1.1.1/24
```

The sample configuration of switch 2 is as follows:

```
feature bfd
configure terminal
     interface port-channel 10
           port-channel bfd track-member-link
           port-channel bfd destination 10.1.1.1
           port-channel bfd start 60
             ip address 10.1.1.2/24
```

Verifying Micro BFD Sessions Configuration

The following example displays the show output of the **show running-config interface port-channel**<*port-channel*>, **show port-channel summary**, **show bfd neighbors vrf internet_routes**, and **show bfd neighbors interface port-channel** <*port-channel*> **vrf internet_routes details** commands.

```
switch# show running-config interface port-channel 1001
```

```
!Command: show running-config interface port-channel1001
!Time: Fri Oct 21 09:08:00 2016

version 9.2(1i)

interface port-channel1001
  no switchport
  vrf member internet_routes
  port-channel bfd track-member-link
  port-channel bfd destination 40.4.1.2
  ip address 40.4.1.1/24

  ipv6 address 2001:40:4:1::1/64

switch# show por
port-channel   port-profile
switch# show port-channel summary
Flags:  D - Down        P - Up in port-channel (members)
        I - Individual  H - Hot-standby (LACP only)
        s - Suspended   r - Module-removed
        b - BFD Session Wait
        S - Switched    R - Routed
        U - Up (port-channel)
        p - Up in delay-lacp mode (member)
        M - Not in use. Min-links not met
--------------------------------------------------------------------------------
Group Port-       Type      Protocol  Member Ports
      Channel
--------------------------------------------------------------------------------
1001  Po1001(RU)  Eth      LACP       Eth1/11/1(P) Eth1/11/2(P)  Eth1/12/1(P)
                                      Eth1/12/2(P)
switch# show bfd neighbors vrf internet_routes

OurAddr         NeighAddr       LD/RD                RH/RS          Holdown(mult)
State       Int                   Vrf
40.4.1.1        40.4.1.2        1090519041/0         Up             N/A(3)          Up
          Po1001                internet_routes
40.4.1.1        40.4.1.2        1090519042/1090519051 Up            819(3)          Up
          Eth1/12/1             internet_routes
40.4.1.1        40.4.1.2        1090519043/1090519052 Up            819(3)          Up
          Eth1/12/2             internet_routes
40.4.1.1        40.4.1.2        1090519044/1090519053 Up            819(3)          Up
          Eth1/11/1             internet_routes
40.4.1.1        40.4.1.2        1090519045/1090519054 Up            819(3)          Up

          Eth1/11/2             internet_routes
switch#

switch# show bfd neighbors interface port-channel 1001 vrf internet_routes details

OurAddr         NeighAddr       LD/RD                RH/RS          Holdown(mult)
State       Int                   Vrf
40.4.1.1        40.4.1.2        1090519041/0         Up             N/A(3)          Up

          Po1001                internet_routes


    Session state is
    Up Local Diag: 0
    Registered protocols:  eth_port_channel
    Uptime: 1 days 11 hrs 4 mins 8 secs
    Hosting LC: 0, Down reason: None, Reason not-hosted: None
    Parent session, please check port channel config for member
    info switch#
```

```
switch# show bfd neighbors interface ethernet 1/12/1 vrf internet_routes details
```

```
OurAddr         NeighAddr       LD/RD                RH/RS           Holdown(mult)

State       Int                  Vrf                                 604(3)            Up
40.4.1.1        40.4.1.2        1090519042/1090519051 Up
          Eth1/12/1             internet_routes

Session state is Up and not using echo function

Local Diag: 0, Demand mode: 0, Poll bit: 0, Authentication: None
MinTxInt: 100000 us, MinRxInt: 100000 us, Multiplier: 3
Received MinRxInt: 300000 us, Received Multiplier: 3
Holdown (hits): 900 ms (0), Hello (hits): 300 ms (458317)
Rx Count: 427188, Rx Interval (ms) min/max/avg: 19/1801/295 last: 295 ms ago
Tx Count: 458317, Tx Interval (ms) min/max/avg: 275/275/275 last: 64 ms ago
Registered protocols:  eth_port_channel
Uptime: 1 days 11 hrs 4 mins 24 secs   - Diagnostic: 0
Last packet: Version: 1
             State bit: Up             - Demand bit: 0
             Poll bit: 0               - Final bit: 0
             Multiplier: 3             - Length: 24
             My Discr.: 1090519051     - Your Discr.: 1090519042
             Min tx interval: 300000   - Min rx interval: 300000
             Min Echo interval: 300000 - Authentication bit: 0
Hosting LC: 1, Down reason: None, Reason not-hosted: None
Member session under parent interface Po1001

switch# show bfd neighbors interface ethernet 1/12/2 vrf internet_routes details


OurAddr         NeighAddr       LD/RD                RH/RS           Holdown(mult)

State       Int                  Vrf                                 799(3)            Up
40.4.1.1        40.4.1.2        1090519043/1090519052 Up
          Eth1/12/2             internet_routes

Session state is Up and not using echo function

Local Diag: 0, Demand mode: 0, Poll bit: 0, Authentication: None
MinTxInt: 100000 us, MinRxInt: 100000 us, Multiplier: 3
Received MinRxInt: 300000 us, Received Multiplier: 3
Holdown (hits): 900 ms (0), Hello (hits): 300 ms (458336)
Rx Count: 427207, Rx Interval (ms) min/max/avg: 19/1668/295 last: 100 ms ago
Tx Count: 458336, Tx Interval (ms) min/max/avg: 275/275/275 last: 251 ms ago
Registered protocols:  eth_port_channel
Uptime: 1 days 11 hrs 4 mins 30 secs   - Diagnostic: 0
Last packet: Version: 1
             State bit: Up             - Demand bit: 0
             Poll bit: 0               - Final bit: 0
             Multiplier: 3             - Length: 24
             My Discr.: 1090519052     - Your Discr.: 1090519043
             Min tx interval: 300000   - Min rx interval: 300000
             Min Echo interval: 300000 - Authentication bit: 0
                  Hosting LC: 1, Down reason: None, Reason not-hosted:
                  None Member session under parent interface Po1001
                  switch#
```

6.7 Configuring BFD Support for Routing Protocols

6.7.1 Configuring BFD on BGP

You can configure BFD for the Border Gateway Protocol (BGP).

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Configure the BFD session parameters. See the Configuring Global BFD Parameters section or the Configuring BFD on an Interface section.

Enable the BGP feature. See the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide for more information.

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *as-number*
3. **neighbor** (*ip-address* | *ipv6-address*) **remote-as** *as-number*
4. **bfd**
5. **update-source** *interface*
6. **show running-config bgp**
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	**router bgp** *as-number*
	Example:
	`switch(config)# router bgp 64496`
	`switch(config-router)#`
		Enables BGP and assigns the AS number to the local BGP speaker. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.

Step 3	**neighbor** (*ip-address* | *ipv6-address*) **remote-as** *as-number*
	Example:
	`switch(config-router)# neighbor 209.165.201.1 remote-as 64497`
	`switch(config-router-neighbor)#`
		Configures the IPv4 or IPv6 address and AS number for a remote BGP peer. The *ip-address* format is x.x.x.x. The *ipv6-address* format is A:B::C:D.

Step 4	**bfd** **Example:** `switch(config-router-neighbor)# bfd`	Enables BFD for this BGP peer.
Step 5	**update-source** *interface* **Example:** `switch(config-router-neighbor)# update-source`	Allows BGP sessions to use the primary IP address from a particular interface as the local address when forming a BGP session with a neighbor and enables BGP to register as a client with BFD.

	Command or Action	Purpose
	`ethernet 2/1`	
Step 6	**show running-config bgp** **Example:** `switch(config-router-neighbor)# show running-config bgp`	(Optional) Displays the BGP running configuration.
Step 7	**copy running-config startup-config** **Example:** `switch(config-router-neighbor)# copy running-config startup-config`	(Optional) Saves the configuration change.

6.7.2 Configuring BFD on EIGRP

You can configure BFD for the Enhanced Interior Gateway Routing Protocol (EIGRP).

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Configure the BFD session parameters. See the Configuring Global BFD Parameters section or the Configuring BFD on an Interface section.

Enable the EIGRP feature. See the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide for more information.

SUMMARY STEPS

1. **configure terminal**
2. **router eigrp** *instance-tag*
3. **bfd** [**ipv4** | **ipv6**]
4. **interface** *int-if*
5. **ip eigrp** *instance-tag* **bfd**
6. **show ip eigrp** [**vrf** *vrf-name*] [**interfaces** *if*]
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	**router eigrp** *instance-tag*
	Example:
	`switch(config)# router eigrp Test1`
	`switch(config-router)#`
		Creates a new EIGRP process with the configured instance tag. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
		If you configure an instance-tag that does not qualify as an AS number, you must use the **autonomous-system** to

		configure the AS number explicitly or this EIGRP instance will remain in the shutdown state.
Step 3	**bfd** [**ipv4** | **ipv6**] **Example:** `switch(config-router-neighbor)# bfd ipv4`	(Optional) Enables BFD for all EIGRP interfaces.

Step 4	**interface** *int-if*
	Example:
	`switch(config-router-neighbor)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode. Use the **?** keyword to display the supported interfaces.

Step 5	**ip eigrp** *instance-tag* **bfd** **Example:** `switch(config-if)# ip eigrp Test1 bfd`	(Optional) Enables or disables BFD on an EIGRP interface. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters. The default is disabled.
Step 6	**show ip eigrp** [**vrf** *vrf-name*] [**interfaces** *if*] **Example:** `switch(config-if)# show ip eigrp`	(Optional) Displays information about EIGRP. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters.
Step 7	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves the configuration change.

6.7.3 Configuring BFD on OSPF

You can configure BFD for the Open Shortest Path First.

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Configure the BFD session parameters. See the Configuring Global BFD Parameters section or the Configuring BFD on an Interface section.

Enable the OSPF feature. See the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide for more information.

SUMMARY STEPS

1. **configure terminal**
2. **router ospf** *instance-tag*
3. **bfd** [**ipv4** | **ipv6**]

4. **interface** *int-if*
5. **ip ospf bfd**
6. **show ip ospf** [**vrf** *vrf-name*] [**interfaces** *if*]
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**router ospf** *instance-tag*
	Example:
	`switch(config)# router ospf 200`
	`switch(config-router)#`
		Creates a new OSPF instance with the configured instance tag. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.

Step 3	**bfd** [**ipv4** | **ipv6**] **Example:** `switch(config-router)# bfd`	(Optional) Enables BFD for all OSPF interfaces.

Step 4	**interface** *int-if*
	Example:
	`switch(config-router)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode. Use the **?** keyword to display the supported interfaces.

Step 5	**ip ospf bfd** **Example:** `switch(config-if)# ip ospf bfd`	(Optional) Enables or disables BFD on an OSPF interface. The default is disabled.
Step 6	**show ip ospf** [**vrf** *vrf-name*] [**interfaces** *if*] **Example:** `switch(config-if)# show ip ospf`	(Optional) Displays information about OSPF. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters.
Step 7	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves the configuration change.

Example Configurations for BFD

- Example configuration for IS-IS where BFD is enabled under IPv4 and an IPv6 address family.

```
configure terminal
  router isis isis-1
    bfd
    address-family ipv6
    unicast bfd
```

- Example configuration where BFD is enabled under a non-default VRF (OSPFv3 neighbors in vrf3).

```
configure
  terminal router
  ospfv3 10
    vrf vrf3
    bfd
```

- Example configuration where BFD is disabled per interface.

```
configure terminal interface
  port-channel 10
    no ip redirects
    ip address 22.1.10.1/30 ipv6
    address 22:1:10::1/120 no
    ipv6 redirects
    ip router ospf 10 area 0.0.0.0
    ip ospf bfd disable        /*** disables IPv4 BFD session for OSPF
    ospfv3 bfd disable         /*** disables IPv6 BFD session for OSPFv3
```

- Example configuration where BFD is enabled for interface static BFD neighbors.

```
configure terminal
  interface
  Ethernet1/15
    ip address 25.7.1.1/30 ipv6
    address 25:7:1::1/120 no ip
    redirects
    no ipv6 redirects
    bfd neighbor src-ip 25.7.1.1 dest-ip 25.7.1.2 /*** simulates IPv4 BFD client
    bfd neighbor src-ip 25:7:1::1 dest-ip 25:7:1::2 /*** simulates IPv6 BFD
    client no shutdown
```

6.7.4 Configuring BFD on IS-IS

You can configure BFD for the Intermediate System-to-Intermediate System (IS-IS) protocol.

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Configure the BFD session parameters. See the Configuring Global BFD Parameters section or the Configuring BFD on an Interface section.

Enable the IS-IS feature. See the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide for more information.

SUMMARY STEPS

1. **configure terminal**
2. **router isis** *instance-tag*
3. **bfd** [**ipv4** | **ipv6**]
4. **interface** *int-if*
5. **isis bfd**

6. **show isis** [**vrf** *vrf-name*] [**interface** *if*]
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**router isis** *instance-tag*
	Example:
	`switch(config)# router isis 100`
	`switch(config-router)# net 49.0001.1720.1600.1001.00`
	`switch(config-router)# address-family ipv6 unicast`
		Creates a new IS-IS instance with the configured *instance* *tag*.

Step 3	**bfd** [**ipv4** | **ipv6**] **Example:** `switch(config-router)# bfd`	(Optional) Enables BFD for all OSPF interfaces.

Step 4	**interface** *int-if*
	Example:
	`switch(config-router)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode. Use the **?** keyword to display the supported interfaces.

Step 5	**isis bfd** **Example:** `switch(config-if)# isis bfd`	(Optional) Enables or disables BFD on an IS-IS interface. The default is disabled.
Step 6	**show isis** [**vrf** *vrf-name*] [**interface** *if*] **Example:** `switch(config-if)# show isis`	(Optional) Displays information about IS-IS. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters.
Step 7	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves the configuration change.

6.7.5 Configuring BFD on HSRP

You can configure BFD for the Hot Standby Router Protocol (HSRP). The active and standby HSRP routers track each other through BFD. If BFD on the standby HSRP router detects that the active HSRP router is down, the standby HSRP router treats this event as an active time rexpiry and takes over as the active HSRP router.

The **show hsrp detail** command shows this event as BFD@Act-down or BFD@Sby-down.

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Configure the BFD session parameters. See the Configuring Global BFD Parameters section or the Configuring BFD on an Interface section.

Enable the HSRP feature. See the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide for more information.

SUMMARY STEPS

1. **configure terminal**
2. **hsrp bfd all-interfaces**
3. **interface** *int-if*
4. **hsrp bfd**
5. **show running-config hsrp**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**hsrp bfd all-interfaces** **Example:** `switch# hsrp bfd all-interfaces`	(Optional) Enables or disables BFD on all HSRP interfaces. The default is disabled.

Step 3	**interface** *int-if*
	Example:
	`switch(config-router)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode. Use the **?** keyword to
		display the supported interfaces.

Step 4	**hsrp bfd** **Example:** `switch(config-if)# hsrp bfd`	(Optional) Enables or disables BFD on an HSRP interface. The default is disabled.
Step 5	**show running-config hsrp**	(Optional) Displays the HSRP running configuration.

	Command or Action	Purpose
	Example: `switch(config-if)# show running-config hsrp`	
Step 6	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves the configuration change.

6.7.6 Configuring BFD on VRRP

You can configure BFD for the Virtual Router Redundancy Protocol (VRRP). The active and standby VRRP routers track each other through BFD. If BFD on the standby VRRP router detects that the active VRRP router is down, the standby VRRP router treats this event as an active time rexpiry and takes over as the active VRRP router.

The **show vrrp detail** command shows this event as BFD@Act-down or BFD@Sby-down.

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Configure the BFD session parameters. See the Configuring Global BFD Parameters section or the Configuring BFD on an Interface section.

Enable the VRRP feature. See the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide for more information.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *int-if*
3. **vrrp** *group-no*
4. **vrrp bfd** *address*
5. **show running-config vrrp**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *int-if*
	Example:
	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode. Use the **?** keyword to display the supported interfaces.

Step 3	**vrrp** *group-no* **Example:**	Specifies the VRRP group number.

	Command or Action	Purpose
	`switch(config-if)# vrrp 2`	
Step 4	**vrrp bfd** *address* **Example:** `switch(config-if)# vrrp bfd`	Enables or disables BFD on a VRRP interface. The default is disabled.
Step 5	**show running-config vrrp** **Example:** `switch(config-if)# show running-config vrrp`	(Optional) Displays the VRRP running configuration.
Step 6	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves the configuration change.

6.7.7 Configuring BFD on PIM

You can configure BFD for the Protocol Independent Multicast (PIM) protocol.

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Enable the PIM feature. See the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide for more information.

SUMMARY STEPS

1. **configure terminal**
2. **ip pim bfd**
3. **interface** *int-if*
4. **ip pim bfd-instance** [**disable**]
5. **show running-config pim**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**ip pim bfd** **Example:** `switch(config)# ip pim bfd`	Enables BFD for PIM.
Step 3	**interface** *int-if*	Enters interface configuration mode. Use the **?**

	Command or Action	Purpose

	Example:
	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
		keyword to
		display the supported interfaces.

Step 4	**ip pim bfd-instance** [**disable**] **Example:** `switch(config-if)# ip pim bfd-instance`	(Optional) Enables or disables BFD on a PIM interface. The default is disabled.
Step 5	**show running-config pim** **Example:** `switch(config)# show running-config pim`	(Optional) Displays the PIM running configuration.
Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves the configuration change.

6.7.8 Configuring BFD on Static Routes

You can configure BFD for static routes on an interface. You can optionally configure BFD on a static route within a virtual routing and forwarding (VRF) instance.

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

SUMMARY STEPS

1. **configure terminal**
2. **vrf context** *vrf-name*
3. **ip route** *route interface* {*nh-address* | *nh-prefix*}
4. **ip route static bfd** *interface* {*nh-address* | *nh-prefix*}
5. **show ip route static** [**vrf** *vrf-name*]
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vrf context** *vrf-name* **Example:**	(Optional) Enters VRF configuration mode.

	Command or Action	Purpose
	switch(config)# **vrf context Red** switch(config-vrf)#	
Step 3	**ip route** *route interface* {*nh-address* | *nh-prefix*} **Example:** switch(config-vrf)# **ip route 192.0.2.1 ethernet 2/1 192.0.2.4**	Creates a static route Use the **?** keyword to display the supported interfaces.
Step 4	**ip route static bfd** *interface* {*nh-address* | *nh-prefix*} **Example:** switch(config-vrf)# **ip route static bfd ethernet 2/1 192.0.2.4**	Enables BFD for all static routes on an interface. Use the**?** keyword to display the supported interfaces.
Step 5	**show ip route static** [**vrf** *vrf-name*] **Example:** switch(config-vrf)# **show ip route static vrf Red**	(Optional) Displays the static routes.
Step 6	**copy running-config startup-config** **Example:** switch(config-vrf)# **copy** **running-config startup-config**	(Optional) Saves the configuration change.

6.7.9 Disabling BFD on an Interface

You can selectively disable BFD on an interface for a routing protocol that has BFD enabled at the global or VRF level.

To disable BFD on an interface, use one of the following commands in interface configuration mode:

Command	Purpose
ip eigrp *instance-tag* **bfd disable** **Example:** switch(config-if)# ip eigrp Test1 bfd disable	Disables BFD on an EIGRP interface. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
ip ospf bfd disable **Example:** switch(config-if)# ip ospf bfd disable	Disables BFD on an OSPFv2 interface.
isis bfd disable **Example:**	Disables BFD on an IS-IS interface.

`switch(config-if)# isis bfd disable`	

6.8 Configuring BFD Interoperability

6.8.1 Configuring BFD Interoperability in Inspur INOS-CN Devices in a Point-to-Point Link

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *int-if*
3. **ip ospf bfd**
4. **no ip redirects**
5. **bfd interval** *mintx* **min_rx** *msec* **multiplier** *value*
6. **exit**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *int-if* **Example:** `switch(config-if)# interface ethernet 2/1`	Enters interface configuration mode. Use the **?** keyword to display the supported interfaces.
Step 3	**ip ospf bfd** **Example:** `switch(config-if)# ip ospf bfd`	Enables BFD on an OSPFv2 interface. The default is disabled. OSPF is used as an example. You can enable BFD of any of the supported protocols.
Step 4	**no ip redirects** **Example:** `switch(config-if)# no ip redirects`	Prevents the device from sending redirects.
Step 5	**bfd interval** *mintx* **min_rx** *msec* **multiplier** *value* **Example:** `switch(config-if)# bfd interval 50`	Configures the BFD session parameters for all BFD sessions on the port channel. This command overrides these values by configuring the BFD session parameters. The *mintx* and *msec* range is from 50 to 999 milliseconds and the default

	Command or Action	Purpose
	`min_rx 50 multiplier 3`	is 50. The multiplier range is from 1 to 50. The multiplier default is 3.
Step 6	**exit** **Example:** `switch(config-if)# exit`	Exits interface configuration mode and returns to EXEC mode.

6.8.2 Configuring BFD Interoperability in Inspur INOS-CN Devices in a Switch Virtual Interface

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *vlan vlan-id*
3. **bfd interval** *mintx* **min_rx** *msec* **multiplier** *value*
4. **no ip redirects**
5. **ip address** *ip-address/length*
6. **ip ospf bfd**
7. **exit**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *vlan vlan-id*
	Example:
	`switch(config)# interface vlan 998`
	`switch(config-if)#`
		Creates a dynamic Switch Virtual Interface (SVI).

Step 3	**bfd interval** *mintx* **min_rx** *msec* **multiplier** *value*
	Example:
	`switch(config-if)# bfd interval 50`
	`min_rx 50 multiplier 3`
		Configures the BFD session parameters for all BFD sessions on the device. The *mintx* and *msec* range is from 50 to 999 milliseconds and the default is 50. The multiplier range is from 1 to 50. The multiplier default is 3.

Step 4	**no ip redirects** **Example:** `switch(config-if)# no ip redirects`	Prevents the device from sending redirects.

	Command or Action	Purpose
Step 5	**ip address** *ip-address/length* **Example:** `switch(config-if)# ip address 10.1.0.253/24`	Configures an IP address for this interface.
Step 6	**ip ospf bfd** **Example:** `switch(config-if)# ip ospf bfd`	Enables BFD on an OSPFv2 interface. The default is disabled.
Step 7	**exit** **Example:** `switch(config-if)# exit`	Exits interface configuration mode and returns to EXEC mode.

6.8.3 Configuring BFD Interoperability in Inspur INOS-CN Devices in Logical Mode

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *type number.subinterface-id*
3. **bfd interval** *mintx* **min_rx** *msec* **multiplier** *value*
4. **no ip redirects**
5. **ip ospf bfd**
6. **exit**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *type number.subinterface-id* **Example:** `switch(config-if)# interface port-channel 50.2`	Enters port channel configuration mode. Use the **?** keyword to display the supported number range.
Step 3	**bfd interval** *mintx* **min_rx** *msec* **multiplier** *value* **Example:** `switch(config-if)# bfd interval 50`	Configures the BFD session parameters for all BFD sessions on the port channel. The *mintx* and *msec* range is from 50 to 999 milliseconds and the default is 50. The multiplier range is from 1 to 50. The multiplier default is 3.

	Command or Action	Purpose
	`min_rx 50 multiplier 3`	
Step 4	**no ip redirects** **Example:** `switch(config-if)# no ip redirects`	Prevents the device from sending redirects.
Step 5	**ip ospf bfd** **Example:** `switch(config-if)# ip ospf bfd`	Enables BFD on an OSPFv2 interface. The default is disabled. OSPF is used as an example. You can enable BFD of any of the supported protocols.
Step 6	**exit** **Example:** `switch(config-if)# exit`	Exits interface configuration mode and returns to EXEC mode.

6.8.4 Verifying BFD Interoperability in an Inspur CN12900 Series Device

The following example shows how to verify BFD interoperability in an Inspur CN12900 Series device.

```
switch# show bfd neighbors details

OurAddr NeighAddr LD/RD RH/RS Holdown(mult) State
Int Vrf
10.1.1.1 10.1.1.2 1140850707/2147418093 Up 6393(4) Up
Vlan2121 default
Session state is Up and using echo function with 50 ms
interval Local Diag: 0, Demand mode: 0, Poll bit: 0,
Authentication: None MinTxInt: 50000 us, MinRxInt: 2000000
us, Multiplier: 3 Received MinRxInt: 2000000 us, Received
Multiplier: 4
Holdown (hits): 8000 ms (0), Hello (hits): 2000 ms (108)
Rx Count: 92, Rx Interval (ms) min/max/avg: 347/1996/1776 last: 1606 ms ago
Tx Count: 108, Tx Interval (ms) min/max/avg: 1515/1515/1515 last: 1233 ms ago
Registered protocols: ospf
Uptime: 0 days 0 hrs 2 mins 44 secs
Last packet: Version: 1 - Diagnostic: 0
State bit: Up - Demand bit: 0
Poll bit: 0 - Final bit: 0
Multiplier: 4 - Length: 24
My Discr.: 2147418093 - Your Discr.: 1140850707
Min tx interval: 2000000 - Min rx interval: 2000000
Min Echo interval: 1000 - Authentication bit: 0
Hosting LC: 10, Down reason: None, Reason not-hosted: None


switch# show bfd neighbors details

OurAddr NeighAddr LD/RD RH/RS Holdown(mult) State
Int Vrf
10.0.2.1 10.0.2.2 1140850695/131083 Up 270(3) Up
Po14.121 default
Session state is Up and not using echo function
Local Diag: 0, Demand mode: 0, Poll bit: 0, Authentication: None
```

```
MinTxInt: 50000 us, MinRxInt: 50000 us, Multiplier: 3
Received MinRxInt: 100000 us, Received Multiplier: 3
Holdown (hits): 300 ms (0), Hello (hits): 100 ms (3136283)
Rx Count: 2669290, Rx Interval (ms) min/max/avg: 12/1999/93 last: 29 ms ago
Tx Count: 3136283, Tx Interval (ms) min/max/avg: 77/77/77 last: 76 ms ago



Registered protocols: ospf
Uptime: 2 days 21 hrs 41 mins 45 secs
Last packet: Version: 1 - Diagnostic: 0
State bit: Up - Demand bit: 0
Poll bit: 0 - Final bit: 0
Multiplier: 3 - Length: 24
My Discr.: 131083 - Your Discr.: 1140850695
Min tx interval: 100000 - Min rx interval: 100000
Min Echo interval: 0 - Authentication bit: 0
Hosting LC: 8, Down reason: None, Reason not-hosted: None
```

6.9 Verifying the BFD Configuration

To display BFD configuration information, perform one of the following:

Command	**Purpose**
show running-config bfd	Displays the running BFD configuration.
show startup-config bfd	Displays the BFD configuration that will be applied on the next system startup.

6.10 Monitoring BFD

Use the following commands to display BFD:

Command	**Purpose**
show bfd neighbors [**application** *name*] [**details**]	Displays information about BFD for a supported application, such as BGP or OSPFv2.
show bfd neighbors [**interface** *int-if*] [**details**]	Displays information about BGP sessions on an interface.
show bfd neighbors [**dest-ip** *ip-address*] [**src-ip** *ip-address*][**details**]	Displays information about the specified BGP session on an interface.
show bfd neighbors [**vrf** *vrf-name*] [**details**]	Displays information about BFD for a VRF.
show bfd [**ipv4** | **ipv6**] [**neighbors**]	Displays information about IPv4 neighbors or IPv6 neighbors.

6.11 BFD Multihop

Beginning with Inspur INOS-CN, BFD Multihop is supported on Inspur CN12900 switches in compliance with RFC5883. You can now configure IPv4 BFD sessions over multihop routes. BFD Multihop session is setup between a unique source and destination address pair. A multihop BFD session is associated with the link between a source-destination rather than an interface as in Single hop BFD sessions. BFD Multihop is not supported on IPv6.

6.11.1 BFD Multihop Number of hops

BFD multihop will set the TTL field to the maximum limit and it does not check the value on reception. The BFD code has no impact on the number of hops a BFD multihop packet can traverse. However, in most of the systems it limits the number of hops to 255

6.11.2 Guidelines and Limitations for BFD Multihop

BFD Multihop has the following configuration guidelines and limitations:

- BFD Multihop is supported only on Inspur CN12900 platform switches.
- Multihop BFD is identified with UDP Destination port 4784
- Echo mode is not supported for Multihop BFD.
- The minimum supported timers and number of sessions may vary. The default timer is 250 ms.
- Multi hop with segment routing underlay is not supported
- The existing BFD authentication support is extended for Multihop sessions also.
- Multihop BFD supports only IPv4 addresses.
- Maximum number of Multihop BFD sessions supported is platform dependent.

6.11.3 Configuring BFD Multihop Session Global Interval Parameters

You can configure the BFD session global parameters for all BFD sessions on the device. Different BFD session parameters for each session can be achieved using the per session configuration commands .

Before you begin

Enable the BFD feature.

SUMMARY STEPS

1. **configure terminal**
2. **[no] bfd multihop interval** *milliseconds* **min_rx** *milliseconds* **multiplier** *interval-multiplier*
3. **end**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	**[no] bfd multihop interval** *milliseconds* **min_rx** *milliseconds* **multiplier** *interval-multiplier* **Example:** `switch(config)# bfd multihop interval 250 min_rx`	Configures the BFD multihop session global parameters for all BFD sessions on the device. This command overrides the default values. The *Required Minimum Receive Internval* and *Desired Minimum Transmit Interval* are 250. The

	Command or Action	Purpose
	`250 multiplier 3`	multiplier default is 3.

Step 3	**end**
	Example:
	`switch(config)# end`
		Saves the configuration change and ends the configuration session.

6.11.4 Configuring Per Multihop Session BFD Parameters

You can configure per multihop session BFD parameters.

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *as-number*
3. **neighbor** (*ip-address* | *ipv6-address*) **remote-as** *as-number*
4. **update-source** *interface*
5. **bfd**
6. **bfd multihop interval** *mintx* **min_rx** *msec* **multiplier** *value*
7. **bfd multihop authentication keyed-sha1 keyid** *id* **key** *ascii_key*
8. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	**router bgp** *as-number*
	Example:
	`switch(config)# router bgp 64496`
	`switch(config-router)#`
		Enables BGP and assigns the AS number to the local BGP speaker. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.

Step 3	**neighbor** (*ip-address* | *ipv6-address*) **remote-as** *as-number*
	Example:
	`switch(config-router)# neighbor 209.165.201.1 remote-as 64497`
	`switch(config-router-neighbor)#`
		Configures the IPv4 or IPv6 address and AS number for a remote BGP peer. The *ip-address* format is x.x.x.x. The *ipv6-address* format is A:B::C:D.

	Command or Action	Purpose

Step 4	**update-source** *interface*
	Example:
	`switch(config-router-neighbor)# update-source Ethernet1/4`
	`switch(config-router-neighbor)#`
		Retrives the source IP address of the BFD session from the interface.

Step 5	**bfd** **Example:** `switch(config-router-neighbor)# bfd`	Enables BFD for this BGP peer.

Step 6	**bfd multihop interval** *mintx* **min_rx** *msec* **multiplier** *value*
	Example:
	`switch(config-router-neighbor)# bfd multihop`
	`interval 250`
	`min_rx 250 multiplier 3`
		Configures Multihop BFD interval values for this neighbor.
		The *mintx* and *msec* range is from 250 to 999 milliseconds and the default is 250. The multiplier range is from 1 to 50.
		The multiplier default is 3.

Step 7	**bfd multihop authentication keyed-sha1 keyid** *id* **key** *ascii_key*
	Example:
	`switch(config-router-neighbor)# bfd multihop`
	`authentication`
	`keyed-sha1 keyid 1 ascii_key Inspur123`
		(Optional) Configures SHA-1 authentication for BFDs on Multihop BFD session over this neighbor. The *ascii_key* string is a secret key shared among BFD peers. The *id* value, a number between 0 and 255, is assigned to this particular *ascii_key*. BFD packets specify the key by *id*, allowing the use of multiple active keys.
		To disable SHA-1 authentication on the interface, use the **no** form of the command.

Step 8	**copy running-config startup-config** **Example:** `switch(config-router-neighbor)# copy running-config startup-config`	(Optional) Saves the configuration change.

6.12 Configuration Examples for BFD

This example shows how to configure BFD for OSPFv2 on Ethernet 2/1, using the default BFD session parameters:

```
feature bfd
feature ospf
```

```
router ospf Test1
interface ethernet
2/1 ip ospf bfd
no shutdown
```

This example shows how to configure BFD for all EIGRP interfaces, using the default BFD session parameters:

```
feature bfd
feature eigrp
bfd interval 100 min_rx 100 multiplier 4


router eigrp
Test2 bfd
```

This example shows how to configure BFDv6:

```
feature bfd feature
ospfv3 router ospfv3
Test1
interface Ethernet2/7
  ipv6 router ospfv3 Test1 area
  0.0.0.0 ospfv3 bfd
  no shutdown
```

6.12.1 Show Example for BFD

This example shows results of the **show bfd ipv6 neighbors details** command.

6.13 Related Documents

```
#show bfd ipv6 neighbors details
OurAddr                          NeighAddr
 LD/RD                 RH/RS           Holdown(mult)     State       Int
 Vrf
cc:10::2                         cc:10::1
 1090519335/1090519260 Up              5692(3)           Up          Po1
 default

Session state is Up and using echo function with 250 ms interval
Local Diag: 0, Demand mode: 0, Poll bit: 0, Authentication: None
MinTxInt: 250000 us, MinRxInt: 2000000 us, Multiplier: 3
Received MinRxInt: 2000000 us, Received Multiplier: 3
Holdown (hits): 6000 ms (4), Hello (hits): 2000 ms (205229)
Rx Count: 227965, Rx Interval (ms) min/max/avg: 124/1520/1510 last: 307 ms ago
Tx Count: 205229, Tx Interval (ms) min/max/avg: 1677/1677/1677 last: 587 ms ago
Registered protocols:  bgp
Uptime: 3 days 23 hrs 31 mins 13 secs
Last packet: Version: 1               - Diagnostic: 0
             State bit: Up            - Demand bit: 0
             Poll bit: 0              - Final bit: 0
             Multiplier: 3            - Length: 24
             My Discr.: 1090519260    - Your Discr.: 1090519335
             Min tx interval: 250000  - Min rx interval: 2000000
```

```
                         Min Echo interval: 250000 - Authentication bit: 0
                Hosting LC: 1, Down reason: None, Reason not-hosted: None
```

6.14 Related Documents

Related Topic	Document Title
BFD commands	*Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide*

6.15 RFCs

RFC	Title
RFC 5880	*Bidirectional Forwarding Detection (BFD)*
RFC 5881	*BFD for IPv4 and IPv6 (Single Hop)*
RFC 7130	*Bidirectional Forwarding Detection (BFD) on Link Aggregation Group (LAG) Interfaces*

CHAPTER 7 Configuring Port Channels

7.1 About Port Channels

A port channel is an aggregation of multiple physical interfaces that creates a logical interface. You can bundle up to 32 individual active links into a port channel to provide increased bandwidth and redundancy. Port channeling also load balances traffic across these physical interfaces. The port channel stays operational as long as at least one physical interface within the port channel is operational.

You can create a Layer 2 port channel by bundling compatible Layer 2 interfaces, or you can create Layer 3 port channels by bundling compatible Layer 3 interfaces. You cannot combine Layer 2 and Layer 3 interfaces in the same port channel.

You can also change the port channel from Layer 3 to Layer 2. See the Configuring Layer 2 Interfaces chapter for information about creating Layer 2 interfaces.

Any configuration changes that you apply to the port channel are applied to each member interface of that port channel. For example, if you configure Spanning Tree Protocol (STP) parameters on the port channel, the Inspur INOS-CN software applies those parameters to each interface in the port channel.

You can use static port channels, with no associated aggregation protocol, for a simplified configuration.

For more flexibility, you can use the Link Aggregation Control Protocol (LACP), which is defined in IEEE 802.3ad. When you use LACP, the link passes protocol packets. You cannot configure LACP on shared interfaces.

See the LACP Overview section for information about LACP.

7.2 Port Channels

A port channel bundles physical links into a channel group to create a single logical link that provides the aggregate bandwidth of up to 32 physical links. If a member port within a port channel fails, the traffic previously carried over the failed link switches to the remaining member ports within the port channel.

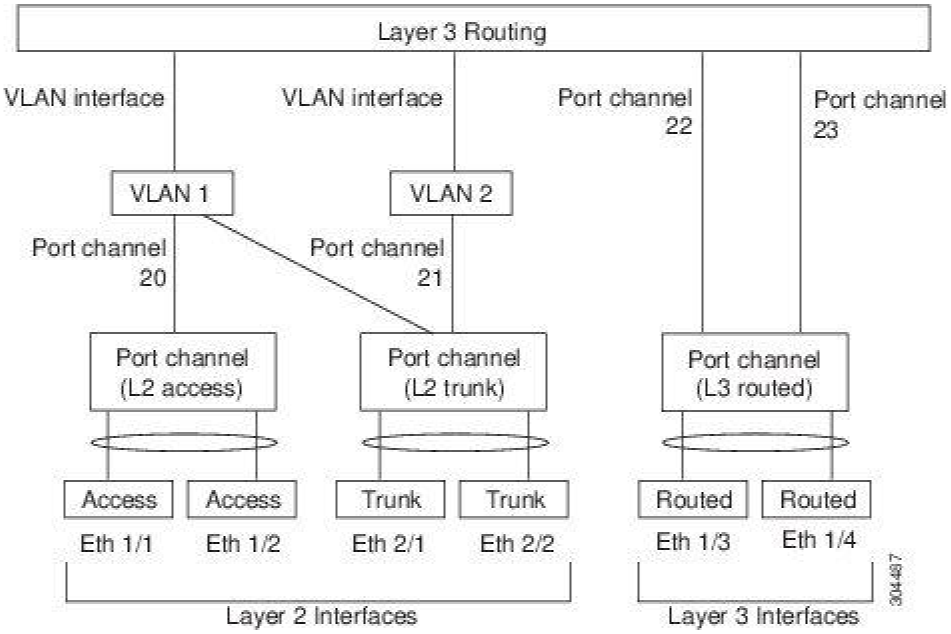
However, you can enable the LACP to use port channels more flexibly. Configuring port channels with LACP and static port channels require a slightly different procedure (see the “Configuring Port Channels” section).

Each port can be in only one port channel. All the ports in a port channel must be compatible; they must use the same speed and duplex mode (see the “Compatibility Requirements” section). When you run static port channels with no aggregation protocol, the physical links are all in the on channel mode; you cannot change this mode without enabling LACP (see the “Port-Channel Modes” section).

You can create port channels directly by creating the port-channel interface, or you can create a channel group that acts to aggregate individual ports into a bundle. When you associate an interface with a channel group, the software creates a matching port channel automatically if the port channel does not already exist. In this instance, the port channel assumes the Layer 2 or Layer 3 configuration of the first interface. You can also create the port channel first. In this instance, the Inspur INOS-CN software creates an empty channel group with the same channel number as the port channel and takes the default Layer 2 or Layer 3 configuration, as well as the compatibility configuration (see the “Compatibility Requirements” section).

7.3 Port-Channel Interfaces

Figure 9 : Port-Channel Interfaces



You can classify port-channel interfaces as Layer 2 or Layer 3 interfaces. In addition, you can configure Layer 2 port channels in either access or trunk mode. Layer 3 port-channel interfaces have routed ports as channel members.

You can configure a Layer 3 port channel with a static MAC address. If you do not configure this value, the Layer 3 port channel uses the router MAC of the first channel member to come up. See the Inspur CN12900 Series INOS-CN Layer 2 Switching Configuration Guide for information about configuring static MAC addresses on Layer 3 port channels.

See the "Configuring Layer 2 Interfaces" chapter for information about configuring Layer 2 ports in access or trunk mode and the "Configuring Layer 3 Interfaces" chapter for information about configuring Layer 3 interfaces and subinterfaces.

7.4 Basic Settings

You can configure the following basic settings for the port-channel interface:

- Bandwidth—Use this setting for informational purposes only; this setting is to be used by higher-level protocols.
- Delay—Use this setting for informational purposes only; this setting is to be used by higher-level protocols.
- Description
- Duplex
- IP addresses
- Maximum Transmission Unit (MTU)
- Shutdown

7.5 Compatibility Requirements

When you add an interface to a channel group, the software checks certain interface attributes to ensure that the interface is compatible with the channel group. For example, you cannot add a Layer 3 interface to a Layer 2 channel group. The Inspur INOS-CN software also checks a number of operational attributes for an interface before allowing that interface to participate in the port-channel aggregation.

The compatibility check includes the following operational attributes:

- Network layer
- (Link) speed capability
- Speed configuration
- Duplex capability
- Duplex configuration
- Port mode
- Access VLAN
- Trunk native VLAN
- Tagged or untagged
- Allowed VLAN list
- MTU size
- SPAN—Cannot be a SPAN source or a destination port
- Storm control
- Flow-control capability
- Flow-control configuration
- Media type, either copper or fiber

Use the **show port-channel compatibility-parameters** command to see the full list of compatibility checks that the Inspur INOS-CN uses.

You can only add interfaces configured with the channel mode set to on to static port channels, and you can only add interfaces configured with the channel mode as active or passive to port channels that are running LACP. You can configure these attributes on an individual member port. If you configure a member port with an incompatible attribute, the software suspends that port in the port channel.

Alternatively, you can force ports with incompatible parameters to join the port channel if the following parameters are the same:

- (Link) speed capability
- Speed configuration
- Duplex capability
- Duplex configuration
- Flow-control capability
- Flow-control configuration

When the interface joins a port channel, some of its individual parameters are removed and replaced with the values on the port channel as follows:

- Bandwidth
- Delay
- Extended Authentication Protocol over UDP
- VRF
- IP address
- MAC address
- Spanning Tree Protocol
- NAC
- Service policy
- Access control lists (ACLs)

Many interface parameters remain unaffected when the interface joins or leaves a port channel as follows:

- Beacon
- Description
- CDP
- LACP port priority
- Debounce
- UDLD
- MDIX

- Rate mode
- Shutdown
- SNMP trap

7.6 Load Balancing Using Port Channels

The Inspur INOS-CN software load balances traffic across all operational interfaces in a port channel by hashing the addresses in the frame to a numerical value that selects one of the links in the channel. Port channels provide load balancing by default. Port-channel load balancing uses MAC addresses, IP addresses, or Layer 4 port numbers to select the link. Port-channel load balancing uses either source or destination addresses or ports, or both source and destination addresses or ports.

You can configure the load- balancing mode to apply to all port channels that are configured on the entire device. You can configure one load-balancing mode for the entire device. You cannot configure the load-balancing method per port channel.

You can configure the type of load-balancing algorithm used. You can choose the load-balancing algorithm that determines which member port to select for egress traffic by looking at the fields in the frame.

The default load-balancing mode for Layer 3 interfaces is the source and destination IP L4 ports, and the default load-balancing mode for non-IP traffic is the source and destination MAC address. Use the **port-channel load-balance** command to set the load-balancing method among the interfaces in the channel-group bundle. The default method for Layer 2 packets is src-dst-mac. The default method for Layer 3 packets is src-dst ip-l4port.

You can configure the device to use one of the following methods to load balance across the port channel:

- Destination MAC address
- Source MAC address
- Source and destination MAC address
- Destination IP address
- Source IP address
- Source and destination IP address
- Source TCP/UDP port number
- Destination TCP/UDP port number
- Source and destination TCP/UDP port number

Non-IP and Layer 3 port channels both follow the configured load-balancing method, using the source, destination, or source and destination parameters. For example, when you configure load balancing to use the source IP address, all non-IP traffic uses the source MAC address to load balance the traffic while the Layer 3 traffic load balances the traffic using the source IP address. Similarly, when you configure the destination MAC address as the load-balancing method, all Layer 3 traffic uses the destination IP address while the non-IP traffic load balances using the destination MAC address.

The unicast and multicast traffic is load-balanced across port-channel links based on configured load-balancing algorithm displayed in **show port-channel load-balancing** command output.

The multicast traffic uses the following methods for load balancing with port channels:

- Multicast traffic with Layer 4 information—Source IP address, source port, destination IP address, destination port
- Multicast traffic without Layer 4 information—Source IP address, destination IP address
- Non-IP multicast traffic—Source MAC address, destination MAC address

7.7 Symmetric Hashing

To be able to effectively monitor traffic on a port channel, it is essential that each interface connected to a port channel receives both forward and reverse traffic flows. Normally, there is no guarantee that the forward and reverse traffic flows will use the same physical interface. However, when you enable symmetric hashing on the port channel, bidirectional traffic is forced to use the same physical interface and each physical interface in the port channel is effectively mapped to a set of flows.

When symmetric hashing is enabled, the parameters used for hashing, such as the source and destination IP address, are normalized before they are entered into the hashing algorithm. This process ensures that when the parameters are reversed (the source on the forward traffic becomes the destination on the reverse traffic), the hash output is the same. Therefore, the same interface is chosen.

Only the following load-balancing algorithms support symmetric hashing:

- src-dst ip
- src-dst ip-l4port

7.8 Guidelines and Limitations for ECMP

You might observe that load balancing with Layer 2/Layer 3 GW flows are not load balanced equally among all links when the switch comes up initially after reload. There are two CLIs to change the ECMP hash configuration in the hardware. The two CLI commands are mutually exclusive.

- Enter the **port-channel load-balance [src | src-dst | dst] mac** command for MAC-based only hash.
- For hash based on IP/Layer 4 ports, enter either the **ip load-share** or **port-channel load-balance** command.
- The port**-channel load-balance** command can overwrite the **ip load-share** command. It is better to enter the **port-channel load-balance** command which helps to set both the IP and MAC parameters.

There are no options to force the hashing algorithm based on the IP/Layer 4 port. The default MAC

7.9 Resilient Hashing

With the exponential increase in the number of physical links used in data centers, there is also the potential for an increase in the number of failed physical links. In static hashing systems that are used for load balancing flows across members of port channels or Equal Cost Multipath (ECMP) groups, each flow is hashed to a link. If a link fails, all flows are rehashed across the remaining working links. This rehashing of flows to links results in some packets being delivered out of order even for those flows that were not hashed to the failed link.

This rehashing also occurs when a link is added to the port channel or Equal Cost Multipath (ECMP) group. All flows are rehashed across the new number of links, which results in some packets being delivered out of order. Resilient hashing supports only unicast traffic.

Resilient hashing maps flows to physical ports. In case a link fails, the flows assigned to the failed link are redistributed uniformly among the working links. The existing flows through the working links are not rehashed and their packets are not delivered out of order.

Resilient hashing is supported only for ECMP groups and not on part channel interfaces. Resiliency is guaranteed only upon deletion of a member in an ECMP group, not on adding a member to the ECMP group. When a link is added to the port channel or ECMP group, some of the flows hashed to the existing links are rehashed to the new link, but not across all existing links.

Resilient hashing also occurs when a link is added to the port channel or Equal Cost Multipath (ECMP) group. Resilient hashing is supported on both Layer 2 port-channel member links and Layer 3 ECMP paths on the routing table.

Resilient hashing supports IPv4 and IPv6 known unicast traffic, but it does not support IPv4 multicast traffic.

7.10 GTP Tunnel Load Balancing

GPRS Tunneling Protocol (GTP) is used mainly to deliver mobile data on wireless networks via Inspur CN12900 Series switches as the core router. When two routers carrying GTP traffic are connected with link bundling, the traffic is required to be distributed evenly between all bundle members.

To achieve load balancing, Inspur CN12900 Series switches use 5-tuple load balancing mechanism. The load balancing mechanism takes into account the source IP, destination IP, protocol, Layer 4 resource and destination port (if traffic is TCP or UDP) fields from the packet. In the case of GTP traffic, a limited number of unique values for these fields restrict the equal distribution of traffic load on the tunnel.

In order to avoid polarization for GTP traffic in load balancing, a tunnel endpoint identifier (TEID) in the GTP header is used instead of a UDP port number. Since the TEID is unique per tunnel, traffic can be evenly load balanced across multiple links in the bundle.

This feature overrides the source and destination port information with the 32-bit TEID value that is present in GTPU packets.

GTP tunnel load balancing feature adds support for:

- GTP with IPv4/IPv6 transport header on physical interface
- GTP traffic over TE tunnel
- GTPU with UDP port 2152

The **ip load-sharing address source-destination gtpu** command enables the GTP tunnel load balancing.

To know the egress interface for GTP traffic after load balancing, use **show cef {ipv4 | ipv6} exact-route** command with TEID in place of L4 protocol source and destination port number. Use 16MSBist of TEID in source port and 16LSBits of TEID in destination port.

7.11 LACP

LACP allows you to configure up to 16 interfaces into a port channel.

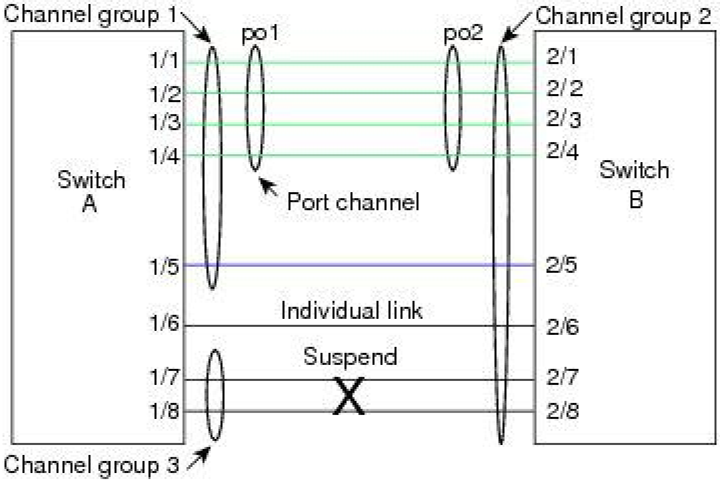
7.11.1 LACP Overview

The Link Aggregation Control Protocol (LACP) for Ethernet is defined in IEEE 802.1AX and IEEE 802.1ad. This protocol controls how physical ports are bundled together to form one logical channel.

The system automatically takes a checkpoint before disabling the feature, and you can roll back to this checkpoint. See the Inspur CN12900 Series INOS-CN System Management Configuration Guide for information about rollbacks and checkpoints.

The following figure shows how individual links can be combined into LACP port channels and channel groups as well as function as individual links.

Figure 10： Individual Links Combined into a Port Channel



With LACP, you can bundle up to 32 interfaces in a channel group.
You cannot disable LACP while any LACP configurations are present.

7.11.2 Port-Channel Modes

Individual interfaces in port channels are configured with channel modes. When you run static port channels with no aggregation protocol, the channel mode is always set to **on**.

After you enable LACP globally on the device, you enable LACP for each channel by setting the channel mode for each interface to **active** or **passive**. You can configure either channel mode for individual links in the LACP channel group when you are adding the links to the channel group.

The following table describes the channel modes.

Table 9：Channel Modes for Individual Links in a Port Channel

Channel Mode	Description
passive	LACP mode that places a port into a passive negotiating state in which the port responds to LACP packets that it receives but does not initiate LACP negotiation.
active	LACP mode that places a port into an active negotiating state in which the port initiates negotiations with other ports by sending LACP packets.
on	All static port channels (that are not running LACP) remain in this mode. If you attempt to change the channel mode to active or passive before enabling LACP, the device displays an error message. You enable LACP on each channel by configuring the interface in that channel for the channel mode as either **active** or **passive**. When an LACP attempts to negotiate with an interface in the on state, it does not receive any LACP packets and becomes an individual link with that interface; it does not join the LACP channel group. The default port-channel mode is **on**.

Both the passive and active modes allow LACP to negotiate between ports to determine if they can form a port channel based on criteria such as the port speed and the trunking state.The passive mode is useful when you do not know whether the remote system, or partner, supports LACP.

Ports can form an LACP port channel when they are in different LACP modes if the modes are compatible as in the following examples:

- A port in **active** mode can form a port channel successfully with another port that is in **active** mode.
- A port in **active** mode can form a port channel with another port in **passive** mode.
- A port in **passive** mode cannot form a port channel with another port that is also in **passive** mode, because neither port will initiate negotiation.
- A port in **on** mode is not running LACP and cannot form a port channel with another port that is in **active** or

passive mode.

7.11.3 LACP ID Parameters

This section describes the LACP parameters.

LACP System Priority

Each system that runs LACP has an LACP system priority value. You can accept the default value of 32768 for this parameter, or you can configure a value between 1 and 65535. LACP uses the system priority with the MAC address to form the system ID and also uses the system priority during negotiation with other devices. A higher system priority value means a lower priority.

LACP Port Priority

Each port that is configured to use LACP has an LACP port priority. You can accept the default value of 32768 for the LACP port priority, or you can configure a value between 1 and 65535. LACP uses the port priority with the port number to form the port identifier.

LACP uses the port priority to decide which ports should be put in standby mode when there is a limitation that prevents all compatible ports from aggregating and which ports should be put into active mode. A higher port priority value means a lower priority for LACP. You can configure the port priority so that specified ports have a lower priority for LACP and are most likely to be chosen as active links, rather than hot-standby links.

LACP Administrative Key

LACP automatically configures an administrative key value equal to the channel-group number on each port configured to use LACP. The administrative key defines the ability of a port to aggregate with other ports. A port’s ability to aggregate with other ports is determined by these factors:

- Port physical characteristics, such as the data rate and the duplex capability
- Configuration restrictions that you establish

7.11.4 LACP Marker Responders

You can dynamically redistribute the data traffic by using port channels. This redistribution might result from a removed or added link or a change in the load-balancing scheme. Traffic redistribution that occurs in the middle of a traffic flow can cause misordered frames.

LACP uses the Marker Protocol to ensure that frames are not duplicated or reordered due to this redistribution. The Marker Protocol detects when all the frames of a given traffic flow are successfully received at the remote end. LACP sends Marker PDUs on each of the port-channel links. The remote system responds to the Marker PDU once it receives all the frames received on this link prior to the Marker PDU. The remote system then sends a Marker Responder. Once the Marker Responders are received by the local system on all member links of the port channel, the local system can redistribute the frames in the traffic flow with no chance of misordering. The software supports only Marker Responders.

7.11.5 LACP-Enabled and Static Port Channels Differences

The following table summarizes the major differences between port channels with LACP enabled and static port channels.

Table 10：Port Channels with LACP Enabled and Static Port Channels

Configurations	Port Channels with LACP Enabled	Static Port Channels
Protocol applied	Enable globally	Not applicable

Configurations	Port Channels with LACP Enabled	Static Port Channels
Channel mode of links	Can be either: • Active • Passive	Can only be On
Maximum number of links in channel	32	32

7.11.6 LACP Compatibility Enhancements

When a Inspur CN12900 Series device is connected to a non-Inspur peer, its graceful failover defaults may delay the time taken for a disabled port to be brought down or cause traffic from the peer to be lost. To address these conditions, the **lacp graceful-convergence** command was added.

By default, LACP sets a port to the suspended state if it does not receive an LACP PDU from the peer. In some cases, although this feature helps in preventing loops created due to misconfigurations, it can cause servers to fail to boot up because they require LACP to logically bring up the port. You can put a port into an individual state by using the **lacp suspend-individual**.

7.11.7 Delayed LACP

LACP port-channels exchange LACP PDUs for quick bundling of links when connecting a server and a switch. However, the links go into suspended state when the PDUs are not received.

The delayed LACP feature enables one port-channel member, the delayed-LACP port, to come up first as a member of a regular port-channel before LACP PDUs are received. After it is connected in LACP mode, other members, the auxiliary LACP ports, are brought up. This avoids having the links becoming suspended when PDUs are not received.

7.11.8 LACP Port-Channel Minimum Links and MaxBundle

A port channel aggregates similar ports to provide increased bandwidth in a single manageable interface.

The introduction of the minimum links and maxbundle feature further refines LACP port-channel operation and provides increased bandwidth in one manageable interface.

The LACP port-channel minimum links feature does the following:

- Configures the minimum number of ports that must be linked up and bundled in the LACP port channel.
- Prevents the low-bandwidth LACP port channel from becoming active.
- Causes the LACP port channel to become inactive if there are few active members ports to supply the required minimum bandwidth.

The LACP MaxBundle defines the maximum number of bundled ports allowed in a LACP port channel.

The LACP MaxBundle feature does the following:

- Defines an upper limit on the number of bundled ports in an LACP port channel.
- Allows hot-standby ports with fewer bundled ports. (For example, in an LACP port channel with five ports, you can designate two of those ports as hot-standby ports.)

7.11.9 LACP Fast Timers

You can change the LACP timer rate to modify the duration of the LACP timeout. Use the lacp rate command to set the rate at which LACP control packets are sent to an LACP-supported interface. You can change the timeout rate from the default rate (30 seconds) to the fast rate (1 second). This command is supported only on LACP-enabled interfaces. To configure the LACP fast time rate, see the “Configuring the LACP Fast Timer Rate” section.

ISSU and stateful switchover cannot be guaranteed with LACP fast timers.

7.11.10 Virtualization Support

You must configure the member ports and other port channel-related configuration from the virtual device context (VDC) that contains the port channel and member ports. You can use the numbers from 1 to 4096 in each VDC to number the port channels.

All ports in one port channel must be in the same VDC. When you are using LACP, all possible 8 active ports and all possible 8 standby ports must be in the same VDC.

7.11.11 High Availability

Port channels provide high availability by load balancing traffic across multiple ports. If a physical port fails, the port channel is still operational if there is an active member in the port channel. You can bundle ports from different modules and create a port channel that remains operational even if a module fails because the settings are common across the module.

Port channels support stateful and stateless restarts. A stateful restart occurs on a supervisor switchover. After the switchover, the Inspur INOS-CN software applies the runtime configuration after the switchover.

The port channel goes down if the operational ports fall below the configured minimum links number.

7.12 Licensing Requirements for Port Channeling

The following table shows the licensing requirements for this feature:

Product	**License Requirement**
Inspur INOS-CN	Port channeling requires no license. Any feature not included in a license package is bundled with the Inspur INOS-CN image and is provided at no extra charge to you.

7.13 Prerequisites for Port Channeling

Port channeling has the following prerequisites:

- You must be logged onto the device.
- All ports for a single port channel must be either Layer 2 or Layer 3 ports.
- All ports for a single port channel must meet the compatibility requirements. See the “Compatibility Requirements” section for more information about the compatibility requirements.
- You must configure load balancing from the default VDC.

7.14 Guidelines and Limitations

Port channeling has the following configuration guidelines and limitations:

- **show** commands with the **internal** keyword are not supported.
- The LACP port-channel minimum links and maxbundle feature is not supported for host interface port channels.
- You must enable LACP before you can use that feature.
- You can configure multiple port channels on a device.

- Do not put shared and dedicated ports into the same port channel. (See the “Configuring Basic Interface Parameters” chapter for information about shared and dedicated ports.)
- For Layer 2 port channels, ports with different STP port path costs can form a port channel if they are compatibly configured with each other. See the “Compatibility Requirements” section for more information about the compatibility requirements.
- In STP, the port-channel cost is based on the aggregated bandwidth of the port members.
- After you configure a port channel, the configuration that you apply to the port channel interface affects the port channel member ports. The configuration that you apply to the member ports affects only the member port where you apply the configuration.
- LACP does not support half-duplex mode. Half-duplex ports in LACP port channels are put in the suspended state.
- Do not configure ports that belong to a port channel group as private VLAN ports. While a port is part of the private VLAN configuration, the port channel configuration becomes inactive.
- Channel member ports cannot be a source or destination SPAN port.
- Port-channels are supported on devices with generation 2 (and later) 100G interfaces.
- The port channel might be affected by the limitations of the Application Leaf Engine (ALE) uplink ports on CN12900 Series devices: Limitations for ALE Uplink Ports
- Resilient hashing (port-channel load-balancing resiliency) and VXLAN configurations are not compatible with VTEPs using ALE uplink ports.
- The maximum number of subinterfaces for a port is 511.
- For a given set of ports which are in the same quadrant, you must use a transceiver of the same speed. You should not mix speeds within a set of ports in the same quadrant. The port numbers that share same quadrent are as follows:
- 1,4,7,10
- 2,5,8,11
- 3,6,9,12
- 13,16,19,22
- 14,17,20,23
- 15,18,21,24
- 25,28,31,34
- 26,29,32,35
- 27,30,33,36
- 37,40,43,46
- 38,41,44,47
- 39,42,45,48

7.15 Default Settings

The following table lists the default settings for port-channel parameters.

Table 11： Default Port-Channel Parameters

Parameters	**Default**
Port channel	Admin up
Load balancing method for Layer 3 interfaces	Source and destination IP address
Load balancing method for Layer 2 interfaces	Source and destination MAC address
Load balancing per module	Disabled

Parameters	Default
LACP	Disabled
Channel mode	on
LACP system priority	32768
LACP port priority	32768
Minimum links for LACP	1
Maxbundle	32

7.16 Configuring Port Channels

7.16.1 Creating a Port Channel

You can create a port channel before you create a channel group. The software automatically creates the associated channel group.

This is required only when the channel group members are Layer 2 ports (switchport) and trunks (switchport mode trunk).

Before you begin

Enable LACP if you want LACP-based port channels.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *channel-number*
3. **show port-channel summary**
4. **no shutdown**
5. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *channel-number*
	Example:
	`switch(config)# interface port-channel 1`
	`switch(config-if)`
		Specifies the port-channel interface to configure, and enters the interface configuration mode. The range is from 1 to 4096. The Inspur INOS-CN software automatically creates the channel group if it does not already exist.

Step 3	**show port-channel summary**	(Optional) Displays information about the port channel.

	Command or Action	Purpose
	Example: `switch(config-router)# show port-channel summary`	

Step 4	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 5	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to create a port channel:

```
switch# configure terminal
switch (config)# interface port-channel 1
```

See the “Compatibility Requirements” section for details on how the interface configuration changes when you delete the port channel.

7.16.2 Adding a Layer 2 Port to a Port Channel

You can add a Layer 2 port to a new channel group or to a channel group that already contains Layer 2 ports. The software creates the port channel associated with this channel group if the port channel does not already exist.

Before you begin

Enable LACP if you want LACP-based port channels.
All Layer 2 member ports must run in full-duplex mode and at the same speed

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*
3. **switchport**
4. **switchport mode trunk**
5. **switchport trunk** {**allowed vlan** *vlan-id* | **native** *vlan-id*}
6. **channel-group** *channel-number* [**force**] [**mode** {**on** | **active** | **passive**}]
7. **show interface** *type slot/port*
8. **no shutdown**
9. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch#` **`configure terminal`**
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *type slot/port*
	Example:
	`switch(config)#` **`interface ethernet 1/4`**
	`switch(config-if)#`
		Specifies the interface that you want to add to a channel group, and enters the interface configuration mode.

Step 3	**switchport** **Example:** `switch(config)#` **`switchport`**	Configures the interface as a Layer 2 access port.
Step 4	**switchport mode trunk** **Example:** `switch(config)#` **`switchport mode trunk`**	(Optional) Configures the interface as a Layer 2 trunk port.

Step 5	**switchport trunk** {**allowed vlan** *vlan-id* | **native** *vlan-id*}
	Example:
	`switch(config)#` **`switchport trunk native 3`**
	`switch(config-if)#`
		(Optional) Configures necessary parameters for a Layer 2 trunk port.

Step 6	**channel-group** *channel-number* [**force**] [**mode** {**on** | **active** | **passive**}] **Example:** • `switch(config-if)#` **`channel-group 5`** • `switch(config-if)#` **`channel-group 5 force`**	Configures the port in a channel group and sets the mode. The channel-number range is from 1 to 4096. This command creates the port channel associated with this channel group if the port channel does not already exist. All static port-channel interfaces are set to mode **on**. You must set all LACP-enabled port-channel interfaces to **active** or **passive**. The default mode is **on**. (Optional) Forces an interface with some incompatible configurations to join the channel. The forced interface must have the same speed, duplex, and flow control settings as the channel group. **Note** The **force** option fails if the port has a QoS policy mismatch with the other members of

	Command or Action	Purpose
		the port channel.
Step 7	**show interface** *type slot/port* **Example:** `switch# show interface port channel 5`	(Optional) Displays interface information.

Step 8	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 9	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to add a Layer 2 Ethernet interface 1/4 to channel group 5:

```
switch# configure terminal
switch (config)# interface ethernet 1/4
switch(config-if)# switchport
switch(config-if)# channel-group 5
```

7.16.3 Adding a Layer 3 Port to a Port Channel

You can add a Layer 3 port to a new channel group or to a channel group that is already configured with Layer 3 ports. The software creates the port channel associated with this channel group if the port channel does not already exist.

If the Layer 3 port that you are adding has a configured IP address, the system removes that IP address before adding the port to the port channel. After you create a Layer 3 port channel, you can assign an IP address to the port-channel interface.

Before you begin

Enable LACP if you want LACP-based port channels.
Remove any IP addresses configured on the Layer 3 interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*

3. **no switchport**
4. **channel-group** *channel-number* [**force**] [**mode** {**on** | **active** | **passive**}]
5. **show interface** *type slot/port*
6. **no shutdown**
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet 1/4`
	`switch(config-if)#`
		Specifies the interface that you want to add to a channel
		group, and enters the interface configuration mode.

Step 3	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 port.
Step 4	**channel-group** *channel-number* [**force**] [**mode** {**on** | **active** | **passive**}] **Example:** • `switch(config-if)# channel-group 5` • `switch(config-if)# channel-group 5 force`	Configures the port in a channel group and sets the mode. The channel-number range is from 1 to 4096. The Inspur INOS-CN software creates the port channel associated with this channel group if the port channel does not already exist. (Optional) Forces an interface with some incompatible configurations to join the channel. The forced interface must have the same speed, duplex, and flow control settings as the channel group.
Step 5	**show interface** *type slot/port* **Example:** `switch# show interface ethernet 1/4`	(Optional) Displays interface information.
Step 6	**no shutdown** **Example:** `switch# configure terminal`	(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors

	Command or Action	Purpose

	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		are placed in an error-disabled policy state.

Step 7	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to add a Layer 3 Ethernet interface 1/5 to channel group 6 in on mode:

```
switch# configure terminal
switch (config)# interface ethernet 1/5
switch(config-if)# switchport
switch(config-if)# channel-group 6
```

This example shows how to create a Layer 3 port-channel interface and assign the IP address:

```
switch# configure terminal
switch (config)# interface port-channel 4
switch(config-if)# ip address 192.0.2.1/8
```

7.16.4 Configuring the Bandwidth and Delay for Informational Purposes

The bandwidth of the port channel is determined by the number of total active links in the channel. You configure the bandwidth and delay on port-channel interfaces for informational purposes.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *channel-number*
3. **bandwidth** *value*
4. **delay** *value*
5. **exit**
6. **show interface port-channel** *channel-number*
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *channel-number*
	Example:
	`switch(config)# interface port-channel 2`
	`switch(config-if)#`
		Specifies the port-channel interface that you want to configure, and enters the interface mode.

Step 3	**bandwidth** *value*	Specifies the bandwidth, which is used for

	Command or Action	Purpose

	Example:
	`switch(config-if)# bandwidth 60000000`
	`switch(config-if)#`
		informational
		purposes. The range is from 1 to 3,200,000,000 kbs. The
		default value depends on the total active interfaces in the
		channel group.

Step 4	**delay** *value*
	Example:
	`switch(config-if)# delay 10000`
	`switch(config-if)#`
		Specifies the throughput delay, which is used for informational purposes. The range is from 1 to 16,777,215
		tens of microseconds. The default value is 10 microseconds.

Step 5	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode and returns to the configuration
		mode.

Step 6	**show interface port-channel** *channel-number* **Example:** `switch# show interface port-channel 2`	(Optional) Displays interface information for the specified port channel.
Step 7	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure the informational parameters of the bandwidth and delay for port channel 5:

```
switch# configure terminal
switch (config)# interface port-channel 5
switch(config-if)# bandwidth
60000000 switch(config-if)# delay
10000 switch(config-if)#
```

7.16.5 Shutting Down and Restarting the Port-Channel Interface

You can shut down and restart the port-channel interface. When you shut down a port-channel interface, no traffic passes and the interface is administratively down.

SUMMARY STEPS

1. **configure terminal**

2. **interface port-channel** *channel-number*
3. **shutdown**
4. **exit**
5. **show interface port-channel** *channel-number*
6. **no shutdown**
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *channel-number*
	Example:
	`switch(config)# interface port-channel 2`
	`switch(config-if)#`
		Specifies the port-channel interface that you want to configure, and enters the interface mode.

Step 3	**shutdown**
	Example:
	`switch(config-if)# shutdown`
	`switch(config-if)#`
		Shuts down the interface. No traffic passes and the interface displays as administratively down. The default is no shutdown.
		Note Use the **no shutdown** command to open the interface.
		The interface displays as administratively up. If there are no operational problems, traffic passes.
		The default is no shutdown.

Step 4	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode and returns to the configuration mode.

Step 5	**show interface port-channel** *channel-number* **Example:** `switch(config-router)# show interface port-channel 2`	(Optional) Displays interface information for the specified port channel.
Step 6	**no shutdown** **Example:**	(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This

	Command or Action	Purpose

	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 7	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to bring up the interface for port channel 2:

```
switch# configure terminal
switch (config)# interface port-channel 2
switch(config-if)# no shutdown
```

7.16.6 Configuring a Port-Channel Description

You can configure a description for a port channel.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *channel-number*
3. **description**
4. **exit**
5. **show interface port-channel** *channel-number*
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *channel-number*
	Example:
	`switch(config)# interface port-channel 2`
	`switch(config-if)#`
		Specifies the port-channel interface that you want to configure, and enters the interface mode.

Step 3	**description** **Example:** `switch(config-if)# description engineering`	Allows you to add a description to the port-channel interface. You can use up to 80 characters in the description. By default, the description does not display; you

	Command or Action	Purpose
	`switch(config-if)#`	must configure this parameter before the description displays in the output.

Step 4	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode and returns to the configuration
		mode.

Step 5	**show interface port-channel** *channel-number* **Example:** `switch#` **`show interface port-channel 2`**	(Optional) Displays interface information for the specified port channel.
Step 6	**copy running-config startup-config** **Example:** `switch(config)#` **`copy running-config startup-config`**	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to add a description to port channel 2:

```
switch# configure terminal
switch (config)# interface port-channel 2
switch(config-if)# description engineering
```

You can configure the speed and duplex settings for a port-channel interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *channel-number*
3. **speed** {**10** | **100** | **1000** | **auto**}
4. **duplex** {**auto** | **full** | **half**}
5. **exit**
6. **show interface port-channel** *channel-number*
7. **copy running-config startup-config**

7.17 Configuring the Speed and Duplex Settings for a Port-Channel Interface

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose

	`switch# configure terminal`
	`switch(config)#`	

Step 2	**interface port-channel** *channel-number*
	Example:
	`switch(config)# interface port-channel 2`
	`switch(config-if)#`
		Specifies the port-channel interface that you want to configure, and enters the interface mode.

Step 3	**speed** {**10** | **100** | **1000** | **auto**}
	Example:
	`switch(config-if)# speed auto`
	`switch(config-if)#`
		Sets the speed for the port-channel interface. The default is auto for autonegotiation.

Step 4	**duplex** {**auto** | **full** | **half**}
	Example:
	`switch(config-if)# speed auto`
	`switch(config-if)#`
		Sets the duplex for the port-channel interface. The default is auto for autonegotiation.

Step 5	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode and returns to the configuration mode.

Step 6	**show interface port-channel** *channel-number* **Example:** `switch# show interface port-channel 2`	(Optional) Displays interface information for the specified port channel.
Step 7	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to set port channel 2 to 100 Mb/s:

```
switch# configure terminal
switch (config)# interface port-channel 2
switch(config-if)# speed 100
```

7.17.1 Configuring Load Balancing Using Port Channels

You can configure the load-balancing algorithm for port channels that applies to the entire device.

Before you begin

Enable LACP if you want LACP-based port channels.

SUMMARY STEPS

1. **configure terminal**
2. **port-channel load-balance** *method* {**dst ip** | **dst ip-gre** | **dst ip-l4port** | **dst ip-l4port-vlan** | **dst ip-vlan** | **dst l4port** | **dst mac** | **src ip** | **src ip-gre** | **src ip-l4port** | **src ip-l4port-vlan** | **src ip-vlan** | **src l4port** | **src mac** | **src-dst ip** | **src-dst ip-gre** | **src-dst ip-l4port** [**symmetric**] | **src-dst ip-l4port-vlan** | **src-dst ip-vlan** | **src-dst l4port** | **src-dst mac**} [**rotate** *rotate*]
3. **show port-channel load-balance**
4. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**port-channel load-balance** *method* {**dst ip** | **dst ip-gre** | **dst ip-l4port** | **dst ip-l4port-vlan** | **dst ip-vlan** | **dst l4port** | **dst mac** | **src ip** | **src ip-gre** | **src ip-l4port** | **src ip-l4port-vlan** | **src ip-vlan** | **src l4port** | **src mac** | **src-dst ip** | **src-dst ip-gre** | **src-dst ip-l4port** [**symmetric**] | **src-dst ip-l4port-vlan** | **src-dst ip-vlan** | **src-dst l4port** | **src-dst mac**} [**rotate** *rotate*] **Example:** • `switch(config)# port-channel load-balance src-dst mac` `switch(config)#` • `switch(config)# no port-channel load-balance src-dst mac` `switch(config)#`	Specifies the load-balancing algorithm for the device. The range depends on the device. The default for Layer 3 is **src-dst ip-l4port** for both IPv4 and IPv6, and the default for non-IP is **src-dst mac**. **Note** Only the following load-balancing algorithms support symmetric hashing: • src-dst ip • src-dst ip-l4port
Step 3	**show port-channel load-balance** **Example:** `switch(config-router)# show port-channel load-balance`	(Optional) Displays the port-channel load-balancing algorithm.
Step 4	**copy running-config startup-config** **Example:**	(Optional) Copies the running configuration to the startup configuration.

	Command or Action	Purpose
	switch(config)# **copy running-config startup-config**	

7.17.2 Enabling LACP

LACP is disabled by default; you must enable LACP before you begin LACP configuration. You cannot disable LACP while any LACP configuration is present.

LACP learns the capabilities of LAN port groups dynamically and informs the other LAN ports. Once LACP identifies correctly matched Ethernet links, it group the links into a port channel. The port channel is then added to the spanning tree as a single bridge port.

To configure LACP, you must do the following:

- Enable LACP globally by using the **feature lacp** command.
- You can use different modes for different interfaces within the same LACP-enabled port channel. You can change the mode between **active** and **passive** for an interface only if it is the only interface that is designated to the specified channel group.

SUMMARY STEPS

1. **configure terminal**
2. **feature lacp**
3. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** switch# **configure terminal** switch(config)#	Enters global configuration mode.
Step 2	**feature lacp** **Example:** switch(config)# **feature lacp**	Enables LACP on the device.
Step 3	**copy running-config startup-config** **Example:** switch(config)# **copy running-config startup-config**	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to enable LACP:

```
switch# configure terminal
switch (config)# feature lacp
```

7.17.3 Configuring LACP Port-Channel Port Modes

After you enable LACP, you can configure the channel mode for each individual link in the LACP port channel as **active** or **passive**. This channel configuration mode allows the link to operate with LACP.

When you configure port channels with no associated aggregation protocol, all interfaces on both sides of the link remain in the **on** channel mode.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*
3. **channel-group** *number* **mode** {**active** | **on** | **passive**}
4. **show port-channel summary**
5. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet 1/4`
	`switch(config-if)#`
		Specifies the interface that you want to add to a channel group, and enters the interface configuration mode.

Step 3	**channel-group** *number* **mode** {**active** | **on** | **passive**}
	Example:
	`switch(config-if)# channel-group 5 mode active`
		Specifies the port mode for the link in a port channel. After LACP is enabled, you configure each link or the entire channel as active or passive.
		When you run port channels with no associated aggregation protocol, the port-channel mode is always on.
		The default port-channel mode is **on**.

Step 4	**show port-channel summary** **Example:** `switch(config-if)# show port-channel summary`	(Optional) Displays summary information about the port channels.
Step 5	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to set the LACP-enabled interface to the active port-channel mode for Ethernet interface 1/4 in channel group 5:

```
switch# configure terminal
switch (config)# interface ethernet 1/4
switch(config-if)# channel-group 5 mode active
```

7.17.4 Configuring LACP Port-Channel Minimum Links

You can configure the LACP minimum links feature. Although minimum links and maxbundles work only in LACP, you can enter the CLI commands for these features for non-LACP port channels, but these commands are nonoperational.

Before you begin

Ensure that you are in the correct port-channel interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *number*
3. **lacp min-links** *number*
4. **show running-config interface port-channel** *number*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *number*
	Example:
	`switch(config)# interface port-channel 3`
	`switch(config-if)#`
		Specifies the interface to configure, and enters the interface
		configuration mode.

Step 3	**lacp min-links** *number* **Example:** `switch(config-if)# lacp min-links 3`	Specifies the port-channel interface to configure the number of minimum links. The range is from 1 to 16.

Step 4	**show running-config interface port-channel** *number*
	Example:
	`switch(config-if)# show running-config interface`
	`port-channel 3`
		(Optional) Displays the port-channel minimum links
		configuration.

Example

This example shows how to configure the minimum number of port-channel member interfaces to be up/active for the port-channel to be up/active:

```
switch# configure terminal switch(config)#
interface port-channel 3
switch(config-if)# lacp min-links 3
```

7.17.5 Configuring the LACP Port-Channel MaxBundle

You can configure the LACP maxbundle feature. Although minimum links and maxbundles work only in LACP, you can enter the CLI commands for these features for non-LACP port channels, but these commands are nonoperational.

Before you begin

Ensure that you are in the correct port-channel interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *number*
3. **lacp max-bundle** *number*
4. **show running-config interface port-channel** *number*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *number*
	Example:
	`switch(config)# interface port-channel 3`
	`switch(config-if)#`
		Specifies the interface to configure, and enters the interface configuration mode.

Step 3	**lacp max-bundle** *number* **Example:** `switch(config-if)# lacp max-bundle`	Specifies the port-channel interface to configure max-bundle. The default value for the port-channel max-bundle is 16. The allowed range is from 1 to 32. **Note** Even if the default value is 16, the number of active members in a port channel is the minimum of the pc_max_links_config and pc_max_active_members that is allowed in the port channel.
Step 4	**show running-config interface port-channel** *number* **Example:**	(Optional) Displays the port-channel max-bundle configuration.

	Command or Action	Purpose
	switch(config-if)# **show running-config** **interface** **port-channel 3**	

Example

This example shows how to configure the port channel interface max-bundle:

```
switch# configure terminal switch(config)#
interface port-channel 3
switch(config-if)# lacp max-bundle 3
```

7.17.6 Configuring the LACP Fast Timer Rate

You can change the LACP timer rate to modify the duration of the LACP timeout. Use the **lacp rate** command to set the rate at which LACP control packets are sent to an LACP-supported interface. You can change the timeout rate from the default rate (30 seconds) to the fast rate (1 second). This command is supported only on LACP-enabled interfaces.

Before you begin

Ensure that you have enabled the LACP feature.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*
3. **lacp rate fast**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** switch# **configure terminal** switch(config)#	Enters global configuration mode.
Step 2	**interface** *type slot/port* **Example:** switch(config)# **interface ethernet 1/4** switch(config-if)#	Specifies the interface to configure and enters the interface configuration mode.
Step 3	**lacp rate fast** **Example:** switch(config-if)# **lacp rate fast**	Configures the fast rate (one second) at which LACP control packets are sent to an LACP-supported interface. To reset the timeout rate to its default, use the **no** form of the command.

Example

This example shows how to configure the LACP fast rate on Ethernet interface 1/4:

```
switch# configure terminal
switch (config)# interface ethernet 1/4
switch(config-if)# lacp rate fast
```

This example shows how to restore the LACP default rate (30 seconds) on Ethernet interface 1/4.

```
switch# configure terminal
switch (config)# interface ethernet 1/4
switch(config-if)# no lacp rate fast
```

7.17.7 Configuring the LACP System Priority

The LACP system ID is the combination of the LACP system priority value and the MAC address.

Before you begin

Enable LACP.

SUMMARY STEPS

1. **configure terminal**
2. **lacp system-priority** *priority*
3. **show lacp system-identifier**
4. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**lacp system-priority** *priority* **Example:** `switch(config)# lacp system-priority 40000`	Configures the system priority for use with LACP. Valid values are from 1 through 65535, and higher numbers have a lower priority. The default value is 32768. **Note** Each VDC has a different LACP system ID because the software adds the MAC address to this configured value.
Step 3	**show lacp system-identifier** **Example:** `switch(config-if)# show lacp system-identifier`	(Optional) Displays the LACP system identifier.
Step 4	**copy running-config startup-config**	(Optional) Copies the running configuration to the

	Command or Action	Purpose
	Example: `switch(config)# copy running-config startup-config`	startup configuration.

Example

This example shows how to set the LACP system priority to 2500:

```
switch# configure terminal switch(config)#

lacp system-priority 2500
```

7.17.8 Configuring the LACP Port Priority

When you enable LACP, you can configure each link in the LACP port channel for the port priority.

Before you begin

Enable LACP.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*
3. **lacp port-priority** *priority*
4. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet 1/4`
	`switch(config-if)#`
		Specifies the interface that you want to add to a channel
		group, and enters the interface configuration mode.

Step 3	**lacp port-priority** *priority* **Example:** `switch(config-if)# lacp port-priority 40000`	Configures the port priority for use with LACP. Valid values are from 1 through 65535, and higher numbers have a lower priority. The default value is 32768.
Step 4	**copy running-config startup-config**	(Optional) Copies the running configuration to the startup

	Command or Action	Purpose
	Example: `switch(config-if)# copy running-config startup-config`	configuration.

Example

This example shows how to set the LACP port priority for Ethernet interface 1/4 to 40000:

```
switch# configure terminal
switch (config)# interface ethernet 1/4
switch(config-if)# lacp port-priority 40000
```

7.17.9 Disabling LACP Graceful Convergence

By default, LACP graceful convergence is enabled. In situations where you need to support LACP interoperability with devices where the graceful failover defaults may delay the time taken for a disabled port to be brought down or cause traffic from the peer to be lost, you can disable convergence. If the downstream access switch is not an Inspur device, disable the LACP graceful convergence option.

Before you begin

Enable LACP.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *number*
3. **shutdown**
4. **no lacp graceful-convergence**
5. **no shutdown**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *number*
	Example:
	`switch(config)# interface port-channel 1`
	`switch(config-if)#`
		Specifies the port channel interface to configure and enters the interface configuration mode.

Step 3	**shutdown** **Example:** `switch(config-if) shutdown`	Administratively shuts down the port channel.

	Command or Action	Purpose
Step 4	**no lacp graceful-convergence** **Example:** `switch(config-if)# no lacp graceful-convergence`	Disables LACP graceful convergence on the port channel.
Step 5	**no shutdown** **Example:** `switch(config-if) no shutdown`	Brings the port channel administratively up.
Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to disable LACP graceful convergence on a port channel:

```
switch# configure terminal
switch (config)# interface port-channel 1
switch(config-if)# shutdown
switch(config-if)# no lacp graceful-convergence
switch(config-if)# no shutdown
```

Reenabling LACP Graceful Convergence

If the default LACP graceful convergence is once again required, you can reenable convergence.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *number*
3. **shutdown**
4. **lacp graceful-convergence**
5. **no shutdown**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *number* **Example:** `switch(config)# interface port-channel 1`	Specifies the port channel interface to configure and enters the interface configuration mode.

	`switch(config-if)#`	
Step 3	**shutdown** **Example:** `switch(config-if) shutdown`	Administratively shuts down the port channel.
Step 4	**lacp graceful-convergence** **Example:** `switch(config-if)# lacp graceful-convergence`	Enables LACP graceful convergence on the port channel.
Step 5	**no shutdown** **Example:** `switch(config-if) no shutdown`	Brings the port channel administratively up.
Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to enable LACP graceful convergence on a port channel:

```
switch# configure terminal
switch (config)# interface port-channel 1
switch(config-if)# shutdown
switch(config-if)# lacp graceful-convergence
switch(config-if)# no shutdown
```

7.17.10 Disabling LACP Suspend Individual

LACP sets a port to the suspended state if it does not receive an LACP PDU from the peer. This process can cause some servers to fail to boot up as they require LACP to logically bring up the port.

Before you begin

Enable LACP.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *number*
3. **shutdown**
4. **no lacp suspend-individual**
5. **no shutdown**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.

	Command or Action	Purpose

	Example:
	`switch# configure terminal`
	`switch(config)#`	

Step 2	**interface port-channel** *number*
	Example:
	`switch(config)# interface port-channel 1`
	`switch(config-if)#`
		Specifies the port channel interface to configure and enters
		the interface configuration mode.

Step 3	**shutdown** **Example:** `switch(config-if) shutdown`	Administratively shuts down the port channel.
Step 4	**no lacp suspend-individual** **Example:** `switch(config-if)# no lacp suspend-individual`	Disables LACP individual port suspension behavior on the port channel.
Step 5	**no shutdown** **Example:** `switch(config-if) no shutdown`	Brings the port channel administratively up.
Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to disable LACP individual port suspension on a port channel:

```
switch# configure terminal
switch (config)# interface port-channel 1
switch(config-if)# shutdown
switch(config-if)# no lacp suspend-individual
switch(config-if)# no shutdown
```

7.17.11 Reenabling LACP Suspend Individual

You can reenable the default LACP individual port suspension.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *number*
3. **shutdown**
4. **lacp suspend-individual**

5. **no shutdown**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *number*
	Example:
	`switch(config)# interface port-channel 1`
	`switch(config-if)#`
		Specifies the port channel interface to configure and enters
		the interface configuration mode.

Step 3	**shutdown** **Example:** `switch(config-if) shutdown`	Administratively shuts down the port channel.
Step 4	**lacp suspend-individual** **Example:** `switch(config-if)# lacp suspend-individual`	Enables LACP individual port suspension behavior on the port channel.
Step 5	**no shutdown** **Example:** `switch(config-if) no shutdown`	Brings the port channel administratively up.
Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to reenable the LACP individual port suspension on a port channel:

```
switch# configure terminal
switch (config)# interface port-channel 1
switch(config-if)# shutdown
switch(config-if)# lacp suspend-individual
switch(config-if)# no shutdown
```

7.17.12 Configuring Delayed LACP

You configure the delayed LACP with the **lacp mode delay**command followed by configuring the LACP port priority.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *number*
3. **lacp mode delay**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**interface port-channel** *number*	Specifies the port channel interface to configure and enters the interface configuration mode.
Step 3	**lacp mode delay**	Enables delayed LACP. **Note** To disable delayed LACP, use the **no lacp mode delay** command. Complete the configuration of the delayed LACP by configuring the LACP port priority. See the "Configuring the LACP Port Priority" section for details. The priority of a LACP port determines the election of the delayed LACP port. The port with the lowest numerical priority is elected. When two or more ports have the same best priority, the VDC system MAC is used to determine which vPC is used. Then within a non-vPC switch or the elected vPC switch, the smallest of the ethernet port names is used. When the delayed LACP feature is configured and made effective with a port channel flap, the delayed LACP port operates as a member of a regular port channel, allowing data to be exchanged between the server and switch. After receiving the first LACP PDU, the delayed LACP port transitions from a regular port member to a LACP port member. **Note** The election of the delayed LACP port is not

	Command or Action	Purpose
		complete or effective until the port channel flaps on the switch or at a remote server.

Example

The following example configures delayed LACP.

```
switch# config terminal
switch(config)# interface po 1
switch(config-if)# lacp mode delay

switch# config terminal switch(config)#
interface ethernet 1/1 switch(config-if)#
lacp port-priority 1
switch(config-if)# channel-group 1 mode active
```

The following example disables delayed LACP.

```
switch# config terminal
switch(config)# interface po 1
switch(config-if)# no lacp mode delay
```

7.17.13 Configuring Port Channel Hash Distribution

Inspur INOS-CN supports the adaptive and fixed hash distribution configuration for both global and port-channel levels. This option minimizes traffic disruption by minimizing Result Bundle Hash (RBH) distribution changes when members come up or go down so that flows that are mapped to unchange RBH values continue to flow through the same links. The port-channel level configuration overrules the global configuration. The default configuration is adaptive globally, and there is no configuration for each port channel, so there is no change during an ISSU. No ports are flapped when the command is applied, and the configuration takes effect at the next member link change event. Both modes work with RBH module or non-module schemes.

During an ISSD to a lower version that does not support this feature, you must disable this feature if the fixed mode command is being used globally or if there is a port-channel level configuration.

Configuring Port Channel Hash Distribution at the Global Level

SUMMARY STEPS

1. **configure terminal**
2. **no port-channel hash-distribution** {**adaptive** | **fixed**}
3. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**no port-channel hash-distribution** {**adaptive** |	Specifies the port-channel hash distribution at the

	Command or Action	Purpose

	fixed}
	Example:
	`switch(config)# port-channel hash-distribution adaptive`
	`switch(config)#`
		global
		level.
		The default is adaptive mode.
		The command does not take effect until the next member
		link event (link down/up/no shutdown/shutdown).
		(`Do you still want to continue(y/n)? [yes]`)

Step 3	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure hash distribution at the global level:

```
switch# configure terminal
switch(config)# no port-channel hash-distribution fixed
```

Configuring Port Channel Hash Distribution at the Port Channel Level

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** {*channel-number* | *range*}
3. **no port-channel port hash-distribution** {**adaptive** | **fixed**}
4. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** {*channel-number* | *range*}
	Example:
	`switch# interface port-channel 4`
	`switch(config-if)#`
		Specifies the interface to configure, and enters the interface
		configuration mode.

Step 3	**no port-channel port hash-distribution** {**adaptive** | **fixed**} **Example:**	Specifies the port-channel hash distribution at the port channel level. There is no default

	Command or Action	Purpose
	switch(config-if)# **port-channel port hash-distribution adaptive** switch(config-if)	The command does not take effect until the next member link event (link down/up/no shutdown/shutdown). (Do you still want to continue(y/n)? [yes])
Step 4	**copy running-config startup-config** **Example:** switch(config)# **copy running-config startup-config**	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure hash distribution as a global-level command:

```
switch# configure terminal
switch(config)# no port-channel hash-distribution fixed
```

7.17.14 Enabling ECMP Resilient Hashing

Resilient ECMP ensures minimal impact to the existing flows when members are deleted from an ECMP group. This is achieved by replicating the existing members in a round-robin fashion at the indices that were previously occupied by the deleted members.

SUMMARY STEPS

1. **configure terminal**
2. **hardware profile ecmp resilient**
3. **copy running-config startup-config**
4. **reload**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** switch# **configure terminal**	Enters global configuration mode.
Step 2	**hardware profile ecmp resilient** **Example:** switch(config)# **hardware profile ecmp resilient**	Enables ECMP resilient hashing and displays the following: **Warning: The command will take effect after next reload.**
Step 3	**copy running-config startup-config** **Example:** switch(config)# **copy running-config startup-config**	Copies the running configuration to the startup configuration.

	Command or Action	Purpose
Step 4	**reload** **Example:** `switch(config)# reload`	Reboots the switch.

7.17.15 Disabling ECMP Resilient Hashing

Before you begin

ECMP resilient hashing is enabled.

SUMMARY STEPS

1. **configure terminal**
2. **no hardware profile ecmp resilient**
3. **copy running-config startup-config**
4. **reload**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**no hardware profile ecmp resilient** **Example:** `switch(config)# no hardware profile ecmp resilient`	Disables ECMP resilient hashing and displays the following: **Warning: The command will take effect after next reload.**
Step 3	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.
Step 4	**reload** **Example:** `switch(config)# reload`	Reboots the switch.

7.17.16 Verifying the ECMP Resilient Hashing Configuration

To display ECMP Resilient Hashing configuration information, perform one of the following tasks:

Command	Purpose

`switch(config)# show running-config | grep "hardware profile ecmp resilient`
`hardware profile ecmp resilient`
`switch(config)#`
	Displays the enabled status.

switch(config)# **show running-config | grep "hardware profile ecmp resilient** switch(config)#	Displays the disabled status.

7.17.17 Verifying the Port-Channel Configuration

To display port-channel configuration information, perform one of the following tasks:

Command	Purpose
show interface port-channel *channel-number*	Displays the status of a port-channel interface.
show feature	Displays enabled features.
load- interval {**interval** *seconds* {**1** | **2** | **3**}}	Sets three different sampling intervals to bit-rate and packet-rate statistics.
show port-channel compatibility-parameters	Displays the parameters that must be the same among the member ports in order to join a port channel.
show port-channel database [**interface port-channel** *channel-number*]	Displays the aggregation state for one or more port-channel interfaces.
show port-channel load-balance	Displays the type of load balancing in use for port channels.
show port-channel summary	Displays a summary for the port-channel interfaces.
show port-channel traffic	Displays the traffic statistics for port channels.
show port-channel usage	Displays the range of used and unused channel numbers.
show lacp {**counters** [**interface port-channel** *channel-number*] | [**interface** *type/slot*] | **neighbor** [**interface port-channel** *channel-number*] | **port-channel** [**interface port-channel** *channel-number*] | **system-identifier**]]}	Displays information about LACP.
show running-config interface port-channel *channel-number*	Displays information about the running configuration of the port-channel.

7.17.18 Monitoring the Port-Channel Interface Configuration

Use the following commands to display port-channel interface configuration information.

Command	Purpose
clear counters interface port-channel *channel-number*	Clears the counters.

Command	Purpose
clear lacp counters [**interface port-channel** *channel-number*]	Clears the LACP counters.
load- interval {**interval** *seconds* {**1** | **2** | **3**}}	Sets three different sampling intervals to bit-rate and packet-rate statistics.
show interface counters [**module** *module*]	Displays input and output octets unicast packets, multicast packets, and broadcast packets.
show interface counters detailed [**all**]	Displays input packets, bytes, and multicast and output packets and bytes.
show interface counters errors [**module** *module*]	Displays information about the number of error packets.
show lacp counters	Displays statistics for LACP.

7.17.19 Example Configurations for Port Channels

This example shows how to create an LACP port channel and add two Layer 2 interfaces to that port channel:

```
switch# configure terminal
switch (config)# feature lacp
switch (config)# interface port-channel 5
switch (config-if)# interface ethernet 1/4
switch(config-if)# switchport switch(config-
if)# channel-group 5 mode active
switch(config-if)# lacp port priority 40000
switch(config-if)# interface ethernet 1/7
switch(config-if)# switchport switch(config-
if)# channel-group 5 mode
```

This example shows how to add two Layer 3 interfaces to a channel group. The Inspur INOS-CN software automatically creates the port channel:

```
switch# configure terminal
switch (config)# interface ethernet 1/5
switch(config-if)# no switchport
switch(config-if)# no ip address
switch(config-if)# channel-group 6 mode active
switch (config)# interface ethernet 2/5
switch(config-if)# no switchport
switch(config-if)# no ip address
switch(config-if)# channel-group 6 mode active
switch (config)# interface port-channel 6
switch(config-if)# ip address 192.0.2.1/8
```

7.17.20 Related Documents

Related Topic	Document Title

System management	*Inspur CN12900 Series INOS-CN System Management Configuration Guide*
High availability	*Inspur CN12900 Series INOS-CN High Availability and Redundancy Guide*

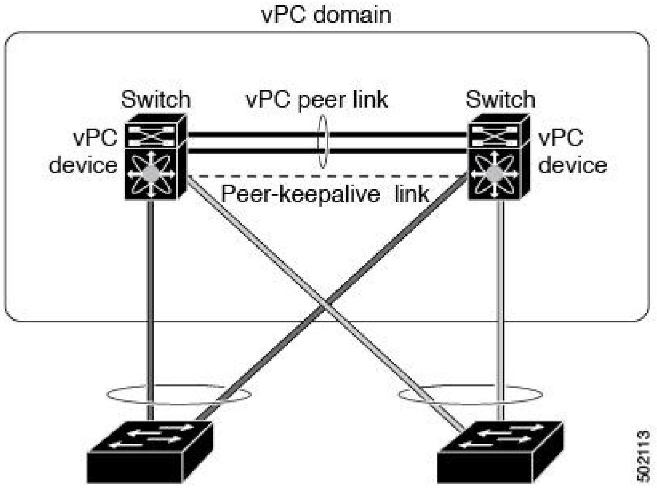
CHAPTER 8 Configuring vPCs

8.1 Information About vPCs

8.1.1 vPC Overview

A virtual port channel (vPC) allows links that are physically connected to two different Inspur CN12900 Series devices to appear as a single port channel by a third device (see figure). The third device can be a switch, server, or any other networking device that supports port channels. A vPC can provide Layer 2 multipathing, which allows you to create redundancy and increase the bisectional bandwidth by enabling multiple parallel paths between nodes and allowing load balancing traffic.

Figure 11：vPC Architecture



You can use only Layer 2 port channels in the vPC. You configure the port channels by using one of the following:

- No protocol
- Link Aggregation Control Protocol (LACP)

When you configure the port channels in a vPC—including the vPC peer link channel—without using LACP, each device can have up to eight active links in a single port channel. When you configure the port channels in a vPC—including the vPC peer link channels—using LACP, each device can have eight active links and eight standby links in a single port channel. (See the “vPC Interactions with Other Features” section for more information on using LACP and vPCs.)

The system automatically takes a checkpoint prior to disabling the feature, and you can roll back to this checkpoint.

After you enable the vPC functionality, you create the peer-keepalive link, which sends heartbeat messages between the two vPC peer devices.

You can create a vPC peer link by configuring a port channel on one Inspur CN12900 Series chassis by using two or more 10-Gigabit Ethernet ports or 40-Gigabit Ethernet ports. To ensure that you have the correct hardware to enable and run a vPC, enter the **show hardware feature-capability** command. If you see an X across from the vPC in your command output, your hardware cannot enable the vPC feature.

We recommend that you configure the vPC peer link Layer 2 port channels as trunks. On another Inspur CN12900 Series chassis, you configure another port channel again using two or more 10-Gigabit Ethernet ports or 40-Gigabit Ethernet ports in the dedicated port mode. Connecting these two port channels creates a vPC peer link in which the two linked Inspur devices appear as one device to a third device. The third device, or downstream device, can be a switch, server, or any other networking device that uses a regular port channel to connect to the vPC. If you are not using the correct module, the system displays an error message.

We recommend that you configure the vPC peer links on dedicated ports of different modules to reduce the possibility of a failure. For the best resiliency scenario, use at least two modules.

If you must configure all the vPC peer links and core-facing interfaces on a single module, you should configure a track object that is associated with the Layer 3 link to the core and on all the links on the vPC peer link on both vPC peer devices. Once you configure this feature and if the primary vPC peer device fails, the system automatically suspends all the vPC links on the primary vPC peer device. This action forces all the vPC traffic to the secondary vPC peer device until the system stabilizes.

You can create a track object and apply that object to all links on the primary vPC peer device that connect to the core and to the vPC peer link. See the *Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide* for information about the **track interface** command.

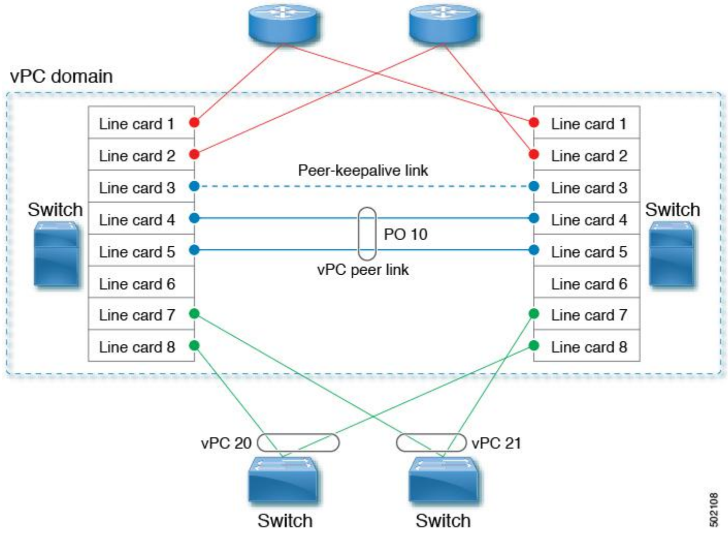
The vPC domain includes both vPC peer devices, the vPC peer-keepalive link, the vPC peer link, and all of the port channels in the vPC domain connected to the downstream device. You can have only one vPC domain ID on each device.

In this version, you can connect each downstream device to a single vPC domain ID using a single port channel.

A vPC (see figure) provides the following benefits:

- Allows a single device to use a port channel across two upstream devices
- Eliminates Spanning Tree Protocol (STP) blocked ports
- Provides a loop-free topology
- Uses all available uplink bandwidth
- Provides fast convergence if either the link or a device fails
- Provides link-level resiliency
- Assures high availability

Figure 12：vPC Interfaces



8.1.2 Hitless vPC Role Change

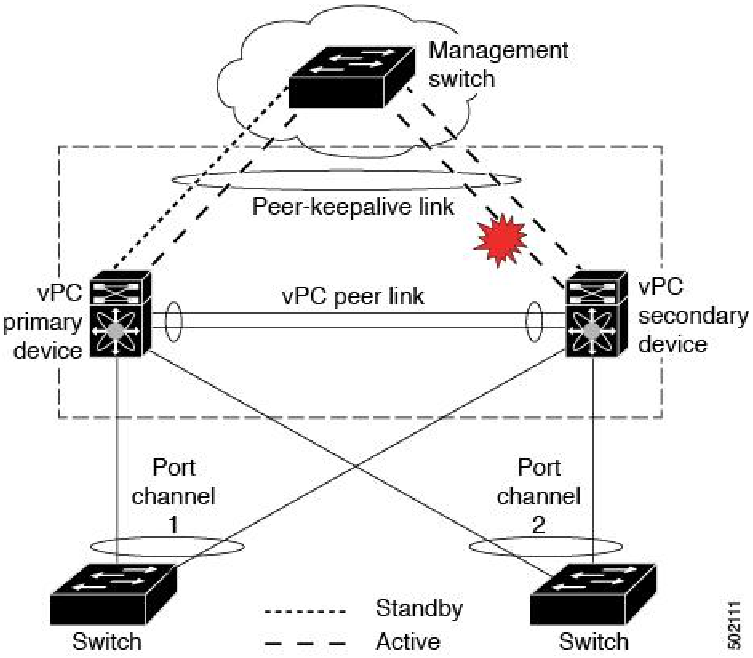
A virtual port channel (vPC) allows links that are physically connected to two different Inspur CN12900 Series devices to appear as a single port channel. The vPC role change feature enables you switch vPC roles between vPC peers without impacting traffic flow. The vPC role switching is done based on the role priority value of the device under the vPC domain. A vPC peer device with lower role priority is selected as the primary vPC device during the vPC Role switch. You can use the vpc role preempt command to switch vPC role between peers.

8.1.3 vPC Terminology

The terminology used in vPCs is as follows:

- vPC—The combined port channel between the vPC peer devices and the downstream device.
- vPC peer device—One of a pair of devices that are connected with the special port channel known as the vPC peer link.
- vPC peer link—The link used to synchronize states between the vPC peer devices. Both ends must be on 10-Gigabit Ethernet or 40-Gigabit Ethernet interfaces.
- vPC member port—An interface that belongs to a vPC.
- Host vPC port—A Fabric Extender host interfaces that belongs to a vPC.
- vPC domain—This domain includes both vPC peer devices, the vPC peer-keepalive link, and all of the port channels in the vPC connected to the downstream devices. It is also associated to the configuration mode that you must use to assign vPC global parameters.
- vPC peer-keepalive link—The peer-keepalive link monitors the vitality of a vPC peer Inspur CN12900 Series device. The peer-keepalive link sends configurable, periodic keepalive messages between vPC peer devices.
  We recommend that you associate a peer-keepalive link to a separate virtual routing and forwarding (VRF) instance that is mapped to a Layer 3 interface in each vPC peer device. If you do not configure a separate VRF, the system uses the management VRF by default. However, if you use the management interfaces for the peer-keepalive link, you must put a management switch connected to both the active and standby management ports on each vPC peer device (see figure).

Figure 13： Separate Switch Required to Connect Management Ports for vPC Peer-Keepalive Link



No data or synchronization traffic moves over the vPC peer-keepalive link; the only traffic on this link is a message that indicates that the originating switch is operating and running a vPC.

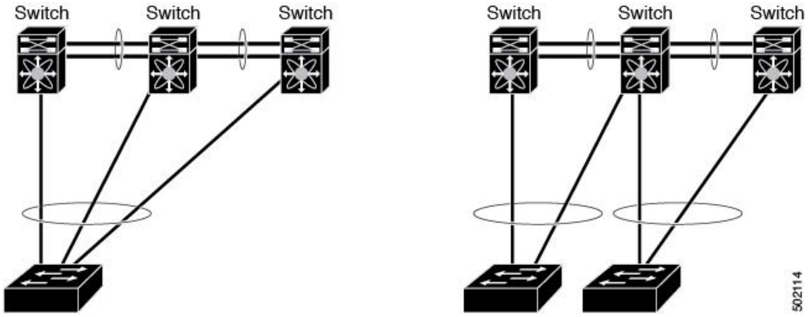
- vPC member port—Interfaces that belong to the vPCs.
- Dual-active— Both vPC peers act as primary. This situation occurs when the peer-keepalive and peer-link go down when both the peers are still active. In this case, the secondary vPC assumes that the primary vPC is inactive and acts as the primary vPC.
- Recovery—When the peer-keepalive and the peer-link come up, one switch becomes the secondary vPC. On the switch that becomes the secondary vPC, the vPC links go down and come back up.

vPC Peer Link Overview

You can have only two devices as vPC peers; each device can serve as a vPC peer to only one other vPC peer. The vPC peer devices can also have non-vPC links to other devices.

See the following figure for invalid vPC peer configurations.

Figure 14： vPC Peer Configurations That Are Not Allowed



To make a valid configuration, you first configure a port channel on each device and then configure the vPC domain. You assign the port channel on each device as a peer link, using the same vPC domain ID. For redundancy, we recommend that you should configure at least two of the dedicated ports into the port channel because if one of the interfaces in the vPC peer link fails, the device automatically falls back to use another interface in the peer link.

Many operational parameters and configuration parameters must be the same in each device connected by a vPC peer link (see the “Compatibility Parameters for vPC Interfaces” section). Because each device is completely independent on the management plane, you must ensure that the devices are compatible on the critical parameters. vPC peer devices have separate control planes. After configuring the vPC peer link, you should display the configuration on each vPC peer device to ensure that the configurations are compatible.

When you configure the vPC peer link, the vPC peer devices negotiate that one of the connected devices is the primary device and the other connected device is the secondary device (see the “Configuring vPCs” section). The Inspur INOS-CN software uses the lowest MAC address to elect the primary device. The software takes different actions on each device—that is, the primary and secondary—only in certain failover conditions. If the primary device fails, the secondary device becomes the new primary device when the system recovers, and the previously primary device is now the secondary device.

You can also configure which of the vPC devices is the primary device. Changing the priority of the vPC peer devices can cause the interfaces in your network to go up and down. If you want to configure the role priority again to make one vPC device the primary device, configure the role priority on both the primary vPC device with a lower priority value and the secondary vPC device with the higher value. Then, shut down the port channel that is the vPC peer link on both devices by entering the **shutdown** command, and finally reenable the port channel on both devices by entering the **no shutdown** command.

The software keeps all traffic that forwards across the vPC peer devices as local traffic. A packet that ingresses the port channel uses one of the local links rather than moving across the vPC peer link. Unknown unicast, multicast, and broadcast traffic (including STP BPDUs) are flooded across the vPC peer link. The software keeps the multicast forwarding state synchronized on both of the vPC peer devices.

You can configure any of the standard load-balancing schemes on both the vPC peer link devices and the downstream device (see the “Configuring Port Channels” chapter for information about load balancing).

If the vPC peer link fails, the software checks the status of the remote vPC peer device using the peer-keepalive link, which is a link between vPC peer devices that ensures that both devices are up. If the vPC peer device is up, the secondary vPC device disables all vPC ports on its device, to prevent loops and disappearing or flooding traffic. The data then forwards down the remaining active links of the port channel.

The software learns of a vPC peer device failure when the keepalive messages are not returned over the peer-keepalive link.

Use a separate link (vPC peer-keepalive link) to send configurable keepalive messages between the vPC peer devices. The keepalive messages on the vPC peer-keepalive link determines whether a failure is on the vPC peer link only or on the vPC peer device. The keepalive messages are used only when all the links in the peer link fail. See the “Peer-Keepalive Link and Messages” section for information about the keepalive message.

Features That You Must Manually Configure on the Primary and Secondary Devices

You must manually configure the following features to conform to the primary/secondary mapping of each of the vPC peer devices:

- STP root—Configure the primary vPC peer device as the STP primary root device and configure the vPC secondary device to be the STP secondary root device. See the “vPC Peer Links and STP” section for more information about vPCs and STP.
- We recommend that you configure the vPC peer link interfaces as STP network ports so that Bridge Assurance is enabled on all vPC peer links.
- We recommend that you configure Rapid per VLAN Spanning Tree plus (PVST+) so that the primary device is the root for all VLANs and configure Multiple Spanning Tree (MST) so that the primary device is the root for all instances.
- Layer 3 VLAN network interface—Configure Layer 3 connectivity from each vPC peer device by configuring

a VLAN network interface for the same VLAN from both devices.

- HSRP active—If you want to use Hot Standby Router Protocol (HSRP) and VLAN interfaces on the vPC peer devices, configure the primary vPC peer device with the HSRP active highest priority. Configure the secondary device to be the HSRP standby and ensure that you have VLAN interfaces on each vPC device that are in the same administrative and operational mode. (See the “vPC Peer Links and Routing” section for more information on vPC and HSRP.)

While you configure Unidirectional Link Detection (UDLD), note the following recommendations:

- If LACP is used as port-channel aggregation protocol, UDLD is not required in a vPC domain.
- If LACP is not used as the port-channel aggregation protocol (static port-channel), use UDLD in normal mode on vPC member ports.
- If STP is used without Bridge Assurance and if LACP is not used, use UDLD in normal mode on vPC orphan ports.

Configuring Layer 3 Backup Routes on a vPC Peer Link

You can use VLAN network interfaces on the vPC peer devices to link to Layer 3 of the network for such applications as HSRP and PIM. Ensure that you have a VLAN network interface configured on each peer device and that the interface is connected to the same VLAN on each device. Also, each VLAN interface must be in the same administrative and operational mode. For more information about configuring VLAN network interfaces, see the “Configuring Layer 3 Interfaces” chapter.

If a failover occurs on the vPC peer link, the VLAN interfaces on the vPC peer devices are also affected. If a vPC peer link fails, the system brings down associated VLAN interfaces on the secondary vPC peer device.

You can ensure that specified VLAN interfaces do not go down on the vPC secondary device when the vPC peer link fails.

Use the **dual-active exclude interface-vlan** command to configure this feature.

8.1.4 Peer-Keepalive Link and Messages

The Inspur INOS-CN software uses the peer-keepalive link between the vPC peers to transmit periodic, configurable keepalive messages. You must have Layer 3 connectivity between the peer devices to transmit these messages; the system cannot bring up the vPC peer link unless the peer-keepalive link is already up and running.

If one of the vPC peer devices fails, the vPC peer device on the other side of the vPC peer link senses the failure by not receiving any peer-keepalive messages. The default interval time for the vPC peer-keepalive message is 1 second, and you can configure the interval between 400 milliseconds and 10 seconds.

You can configure a hold-timeout value with a range of 3 to 10 seconds; the default hold-timeout value is 3 seconds. This timer starts when the vPC peer link goes down. During this hold-timeout period, the secondary

vPC peer device ignores vPC peer-keepalive messages, which ensures that network convergence occurs before a vPC action takes place. The purpose of the hold-timeout period is to prevent false-positive cases.

You can also configure a timeout value with a range of 3 to 20 seconds; the default timeout value is 5 seconds. This timer starts at the end of the hold-timeout interval. During the timeout period, the secondary vPC peer device checks for vPC peer-keepalive hello messages from the primary vPC peer device. If the secondary vPC peer device receives a single hello message, that device disables all vPC interfaces on the secondary vPC peer device.

The difference between the hold-timeout and the timeout parameters is as follows:

- During the hold-timeout, the vPC secondary device does not take any action based on any keepalive messages received, which prevents the system taking action when the keepalive might be received just temporarily, such as if a supervisor fails a few seconds after the peer link goes down.
- During the timeout, the vPC secondary device takes action to become the vPC primary device if no keepalive message is received by the end of the configured interval.

See the “Configuring vPCs” section for information about configuring the timer for the keepalive messages.

Use the command-line interface (CLI) to configure the interfaces you are using the vPC peer-keepalive messages as trusted ports. Leave the precedence at the default (6) or configure it higher. The following is an example of configuring an interface as a trusted port:

```
(config)# class-map type qos match-all trust-map
(config-cmap-qos)# match cos 4-7
(config)# policy-map type qos ingresspolicy
(config-pmap-qos)# class trust-map
(config)# interface Ethernet 8/11
(config-if)# service-policy type qos input ingresspolicy
```

8.1.5 vPC Peer-Gateway

You can configure vPC peer devices to act as the gateway even for packets that are destined to the vPC peer device’s MAC address.

Use the **peer-gateway** command to configure this feature.

Some network-attached storage (NAS) devices or load balancers might have features that help to optimize the performances of particular applications. These features enable the device to avoid a routing-table lookup when responding to a request that originated from a host that is not locally attached to the same subnet. Such devices might reply to traffic using the MAC address of the sender Inspur CN12900 Series device rather than the common HSRP gateway. This behavior is noncomplaint with some basic Ethernet RFC standards.

Packets that reach a vPC device for the nonlocal router MAC address are sent across the peer link and could be dropped by the built in vPC loop avoidance mechanism if the final destination is behind another vPC.

The vPC peer-gateway capability allows a vPC switch to act as the active gateway for packets that are addressed to the router MAC address of the vPC peer. This feature enables local forwarding of packets without the need to cross the vPC peer link. In this scenario, the feature optimizes use of the peer link and avoids potential traffic loss.

Configuring the peer-gateway feature must be done on both primary and secondary vPC peers and is nondisruptive to the operations of the device or to the vPC traffic. The vPC peer-gateway feature can be configured globally under the vPC domain submode.

When you enable this feature, Inspur INOS-CN automatically disables IP redirects on all interface VLANs mapped over a vPC VLAN to avoid generation of IP redirect messages for packets switched through the peer gateway router.

Packets that arrive at the peer-gateway vPC device have their Time to Live (TTL) decremented, so that packets carrying a TTL of 1 might get dropped in transit due to TTL expiration. You should take this situation into account when the peer-gateway feature is enabled and particular network protocols that source packets with a TTL of 1 operate on a vPC VLAN.

8.1.6 vPC Domain

You can use the vPC domain ID to identify the vPC peer links and the ports that are connected to the vPC downstream devices.

The vPC domain is also a configuration mode that you use to configure the keepalive messages and other vPC peer link parameters rather than accept the default values. See the “Configuring vPCs” section for more information about configuring these parameters.

To create a vPC domain, you must first create a vPC domain ID on each vPC peer device using a number from 1 to 1000. You can have only one vPC domain per vPC peer.

You must explicitly configure the port channel that you want to act as the peer link on each device. You associate the port channel that you made a peer link on each device with the same vPC domain ID to form a single vPC domain. Within this domain, the system provides a loop-free topology and Layer 2 multipathing.

You can only configure these port channels and vPC peer links statically. All ports in the vPC on each of the vPC peer devices must be in the same VDC. You can configure the port channels and vPC peer links either using LACP or no protocol. We recommend that you use LACP with the interfaces in active mode to configure port channels in each vPC, which ensures an optimized, graceful recovery in a port-channel failover scenario and provides configuration checks against configuration mismatches among the port channels themselves.

The vPC peer devices use the vPC domain ID that you configure to automatically assign a unique vPC system MAC address. Each vPC domain has a unique MAC address that is used as a unique identifier for the specific vPC-related operations, although the devices use the vPC system MAC addresses only for link-scope operations, such as LACP. We recommend that you create each vPC domain within the contiguous Layer 2 network with a unique domain ID. You can also configure a specific MAC address for the vPC domain, rather than having the Inspur INOS-CN software assign the address.

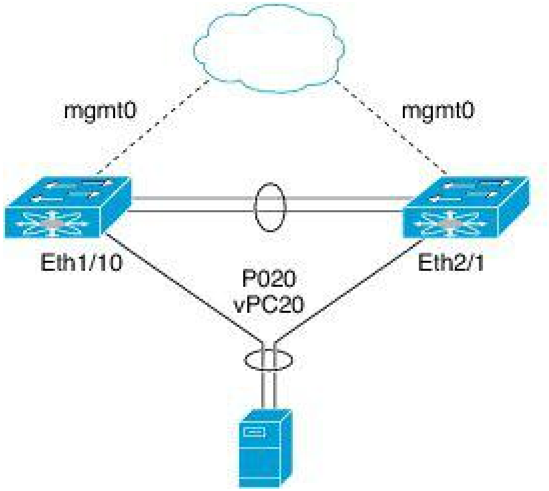
See the “vPC and Orphan Ports” section for more information about displaying the vPC MAC table.

After you create a vPC domain, the Inspur INOS-CN software creates a system priority for the vPC domain. You can also configure a specific system priority for the vPC domain.

8.1.7 vPC Topology

The following figure shows a basic configuration in which the Inspur CN12900 Series device ports are directly connected to another switch or host and are configured as part of a port channel that becomes part of a vPC.

Figure 15： witch vPC Topology



In the figure, vPC 20 is configured on port channel 20, which has Eth1/10 on the first device and Eth2/1 on the second as member ports.

8.1.8 Compatibility Parameters for vPC Interfaces

Many configuration and operational parameters must be identical on all interfaces in the vPC. We recommend that you configure the Layer 2 port channels that you use for the vPC peer link in trunk mode.

The compatibility check process for vPCs differs from the compatibility check for regular port channels.

See the “Configuring Port Channels” chapter for information about regular port channels.

Configuration Parameters That Must Be Identical

The configuration parameters in this section must be configured identically on both devices of the vPC peer link; otherwise, the vPC moves fully or partially into a suspended mode.

The devices automatically check for compatibility for some of these parameters on the vPC interfaces. The per-interface parameters must be consistent per interface, and the global parameters must be consistent globally:

- Port-channel mode: on, off, or active (port-channel mode can, however, be active/passive on each side of the vPC peer)

- Link speed per channel
- Duplex mode per channel
- Trunk mode per channel:
- Native VLAN
- VLANs allowed on trunk
- Tagging of native VLAN traffic
- Spanning Tree Protocol (STP) mode
- STP region configuration for Multiple Spanning Tree
- Enable/disable state per VLAN
- STP global settings:
- Bridge Assurance setting
- Port type setting
- Loop Guard settings
- STP interface settings:
- Port type setting
- Loop Guard
- Root Guard
- Maximum Transmission Unit (MTU)

If any of these parameters are not enabled or defined on either device, the vPC consistency check ignores those parameters.

Configuration Parameters That Should Be Identical

When any of the following parameters are not configured identically on both vPC peer devices, a misconfiguration might cause undesirable behavior in the traffic flow:

- MAC aging timers
- Static MAC entries
- VLAN interface—Each device on the end of the vPC peer link must have a VLAN interface configured for the same VLAN on both ends and they must be in the same administrative and operational mode. Those VLANs configured on only one device of the peer link do not pass traffic using the vPC or peer link. You must create all VLANs on both the primary and secondary vPC devices, or the VLAN will be suspended.
- All ACL configurations and parameters
- Quality of Service (QoS) configuration and parameters
- STP interface settings:
- BPDU Filter
- BPDU Guard
- Cost
- Link type
- Priority
- VLANs (Rapid PVST+)
- Port security
- Dynamic Host Configuration Protocol (DHCP) snooping
- Network Access Control (NAC)
- Dynamic ARP Inspection (DAI)
- IP source guard (IPSG)
- Internet Group Management Protocol (IGMP) snooping
- Hot Standby Routing Protocol (HSRP)
- Protocol Independent Multicast (PIM)
- All routing protocol configurations

To ensure that all the configuration parameters are compatible, we recommend that you display the configurations for each vPC peer device once you configure the vPC.

Consequences of Parameter Mismatches

You can configure the graceful consistency check feature, which suspends only the links on the secondary peer device when a mismatch is introduced in a working vPC. This feature is configurable only in the CLI and is enabled by default.

Use the graceful consistency-check command to configure this feature.

As part of the consistency check of all parameters from the list of parameters that must be identical, the system checks the consistency of all VLANs.

The vPC remains operational, and only the inconsistent VLANs are brought down. This per-VLAN consistency check feature cannot be disabled and does not apply to Multiple Spanning Tree (MST) VLANs.

8.1.9 vPC Number

Once you have created the vPC domain ID and the vPC peer link, you create port channels to attach the downstream device to each vPC peer device. That is, you create one port channel to the downstream device from the primary vPC peer device and you create another port channel to the downstream device from the secondary peer device.

On each vPC peer device, you assign a vPC number to the port channel that connects to the downstream device. You will experience minimal traffic disruption when you are creating vPCs. To simplify the configuration, you can assign the vPC ID number to every port channel to be the same as the port channel itself (that is, vPC ID 10 for port channel 10).

8.1.10 Moving Other Port Channels into a vPC

To connect to the downstream device, you create a port channel to the downstream device from the primary vPC peer device and you create another port channel to the downstream device from the secondary peer device.

On each vPC peer device, you assign a vPC number to the port channel that connects to the downstream device. You will experience minimal traffic disruption when you are creating vPCs.

Configuring vPC Peer Links and Links to the Core on a Single Module

If you must configure all the vPC peer links and core-facing interfaces on a single module, you should configure, using the command-line interface, a track object and a track list that is associated with the Layer 3 link to the core and on all vPC peer links on both vPC peer devices. You use this configuration to avoid dropping traffic if that particular module goes down because when all the tracked objects on the track list go down, the system does the following:

- Stops the vPC primary peer device sending peer-keepalive messages, which forces the vPC secondary peer device to take over.
- Brings down all the downstream vPCs on that vPC peer device, which forces all the traffic to be rerouted in the access switch toward the other vPC peer device.

Once you configure this feature and if the module fails, the system automatically suspends all the vPC links on the primary vPC peer device and stops the peer-keepalive messages. This action forces the vPC secondary device to take over the primary role and all the vPC traffic to go to this new vPC primary device until the system stabilizes.

You should create a track list that contains all the links to the core and all the vPC peer links as its object. Enable tracking for the specified vPC domain for this track list. Apply this same configuration to the other vPC peer device. See the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide for information about configuring object tracking and track lists.

To configure a track list to switch over a vPC to the remote peer when all related interfaces on a single module fail, follow these steps:

1. Configure track objects on an interface (Layer 3 to core) and on a port channel (vPC peer link).

```
switch(config-if)# track 35 interface ethernet 8/35 line-protocol
switch(config-track)# track 23 interface ethernet 8/33 line-protocol
switch(config)# track 55 interface port-channel 100 line-protocol
```

2. Create a track list that contains all the interfaces in the track list using the Boolean OR to trigger when all objects fail.

```
switch(config)# track 44 list boolean OR
switch(config-track)# object 23


switch(config-track)# object 35
switch(config-track)# object 55
switch(config-track)# end
```

3. Add this track object to the vPC domain:

```
switch(config)# vpc domain 1

switch(config-vpc-domain)# track 44
```

4. Display the track object:

```
switch# show vpc brief
Legend:
(*) - local vPC is down, forwarding via vPC peer-link
vPC domain id : 1
Peer status : peer adjacency formed ok
vPC keep-alive status : peer is alive
Configuration consistency status: success
vPC role : secondary
Number of vPCs configured : 52
Track object : 44 vPC
Peer-link status
---------------------------------------------------------------------
id Port Status Active vlans
-- ---- ------ --------------------------------------------------
1 Po100 up 1-5,140
vPC status
----------------------------------------------------------------------
id Port Status Consistency Reason Active vlans
-- ---- ------ ----------- -------------------------- ------------
1 Po1 up success success 1-5,140
```

This example shows how to display information about the track objects:

```
switch# show track brief
Track Type Instance Parameter State Last
Change
23 Interface Ethernet8/33 Line Protocol UP 00:03:05
35 Interface Ethernet8/35 Line Protocol UP 00:03:15
44 List ----- Boolean
or UP 00:01:19
55 Interface port-channel100 Line Protocol UP 00:00:34
```

8.1.11 vPC Interactions with Other Features

vPC and LACP

LACP uses the system MAC address of the vPC domain to form the LACP Aggregation Group (LAG) ID for the vPC. (See the “Configuring Port Channels” chapter for information about LAG-ID and LACP.)

You can use LACP on all the vPC port channels, including those channels from the downstream device. We recommend that you configure LACP with active mode on the interfaces on each port channel on the vPC peer devices. This configuration allows you to more easily detect compatibility between devices, unidirectional links, and multihop connection, and provides dynamic reaction to run-time changes and link failures.

We recommend that you manually configure the system priority on the vPC peer link devices to ensure that the vPC peer link devices have a higher LACP priority than the downstream connected devices. A lower numerical value system priority means a higher LACP priority.

vPC Peer Links and STP

Although vPCs provide a loop-free Layer 2 topology, STP is still required to provide a fail-safe mechanism to protect against any incorrect or defective cabling or possible misconfiguration. When you first bring up a vPC, STP reconverges. STP treats the vPC peer link as a special link and always includes the vPC peer link in the STP active topology.

We recommend that you set all the vPC peer link interfaces to the STP network port type so that Bridge Assurance is automatically enabled on all vPC peer links. We also recommend that you do not enable any of the STP enhancement features on vPC peer links. If the STP enhancements are already configured, they do not cause any problems for the vPC peer links..

When you are running both MST and Rapid PVST+, ensure that the PVST simulation feature is correctly configured.

See the Inspur CN12900 Series INOS-CN Layer 2 Switching Configuration Guide for information about STP enhancement features and PVST simulation.

STP is distributed; that is, the protocol continues running on both vPC peer devices. However, the configuration on the vPC peer device elected as the primary device controls the STP process for the vPC interfaces on the secondary vPC peer device.

The STP process for vPC also relies on the periodic keepalive messages to determine when one of the connected devices on the peer link fails. See the "Peer-Keepalive Link and Messages" section for information about these messages.

The vPC manager performs a proposal/handshake agreement between the vPC peer devices that set the primary and secondary devices and coordinates the two devices for STP. The primary vPC peer device then controls the STP protocol on both the primary and secondary devices. We recommend that you configure the primary vPC peer device as the STP primary root device and configure the secondary VPC device to be the STP secondary root device.

If the primary vPC peer device fails over to the secondary vPC peer device, there is no change in the STP topology.

The BPDUs uses the MAC address set for the vPC for the STP bridge ID in the designated bridge ID field. The vPC primary device sends these BPDUs on the vPC interfaces.

You must configure both ends of vPC peer link with the identical STP configuration for the following parameters:

- STP global settings:
- STP mode
- STP region configuration for MST
- Enable/disable state per VLAN
- Bridge Assurance setting
- Port type setting
- Loop Guard settings
- STP interface settings:
- Port type setting
- Loop Guard
- Root Guard

Ensure that the following STP interface configurations are identical on both sides of the vPC peer links or you may see unpredictable behavior in the traffic flow:

- BPDU Filter
- BPDU Guard
- Cost
- Link type
- Priority
- VLANs (PVRST+)

You can use the **show spanning-tree** command to display information about the vPC when that feature is enabled. See the Inspur CN12900 Series INOS-CN Layer 2 Switching Configuration Guide for an example.

vPC Peer Switch

The vPC peer switch feature was added to Inspur INOS-CN to address performance concerns around STP convergence. This feature allows a pair of Inspur CN12900 Series devices to appear as a single STP root in the Layer 2 topology. This feature eliminates the need to pin the STP root to the vPC primary switch and improves vPC convergence if the vPC primary switch fails.

To avoid loops, the vPC peer link is excluded from the STP computation. In vPC peer switch mode, STP BPDUs are sent from both vPC peer devices to avoid issues related to STP BPDU timeout on the downstream switches, which can cause traffic disruption.

This feature can be used with the pure peer switch topology in which the devices all belong to the vPC.

See the Inspur CN12900 Series INOS-CN Layer 2 Switching Configuration Guide for information about STP enhancement features and Rapid PVST+.

vPC Multicast—PIM, IGMP, and IGMP Snooping

The Inspur INOS-CN software for the CN12900 Series devices supports the following on a vPC:

- PIM Any Source Multicast (ASM).
- PIM Source-Specific Multicast (SSM) .

The software keeps the multicast forwarding state synchronized on both of the vPC peer devices. The IGMP snooping process on a vPC peer device shares the learned group information with the other vPC peer device through the vPC peer link; the multicast states are always synchronized on both vPC peer devices. The PIM process in vPC mode ensures that only one of the vPC peer devices forwards the multicast traffic to the receivers.

Each vPC peer is a Layer 2 or Layer 3 device. Multicast traffic flows from only one of the vPC peer devices. You might see duplicate packets in the following scenarios:

- Orphan hosts
- When the source and receivers are in the Layer 2 vPC cloud in different VLANs with multicast routing enabled and a vPC member link goes down.

You might see negligible traffic loss in the following scenarios:

- When you reload the vPC peer device that is forwarding the traffic.
- When you restart PIM on the vPC peer device that is forwarding the traffic.

Ensure that you dual-attach all Layer 3 devices to both vPC peer devices. If one vPC peer device goes down, the other vPC peer device continues to forward all multicast traffic normally.

The following outlines vPC PIM and vPC IGMP/IGMP snooping:

- vPC PIM—The PIM process in vPC mode ensures that only one vPC peer device forwards multicast traffic. The PIM process in vPC mode synchronizes the source state with both vPC peer devices and elects which vPC peer device forwards the traffic.
- vPC IGMP/IGMP snooping—The IGMP process in vPC mode synchronizes the designated router (DR) information on both vPC peer devices. Dual DRs are available for IGMP when you are in vPC mode. Dual DRs are not available when you are not in vPC mode, because both vPC peer devices maintain the multicast group information between the peers.

You should enable or disable IGMP snooping identically on both vPC peer devices, and all the feature configurations should be identical. IGMP snooping is on by default.

See the *Inspur CN12900 Series INOS-CN Multicast Routing Configuration Guide* for more information about multicasting.

Multicast PIM Dual DR (Proxy DR)

By default, a multicast router sends PIM joins upstream only if it has interested receivers. These interested receivers can either be IGMP hosts (they communicate through IGMP reports) or other multicast routers (they communicate through PIM joins).

In the Inspur INOS-CN vPC implementation, PIM works in dual designated router (DR) mode. That is, if a vPC device is a DR on a vPC SVI outgoing interface (OIF), its peer automatically assumes the proxy DR role. IGMP adds an OIF (the report is learned on that OIF) to the forwarding if the OIF is a DR. With dual DRs, both vPC devices have an identical (*,G) entry with respect to the vPC SVI OIFs as shown in this example:

```
VPC Device1:
------------
(*,G)
oif1 (igmp)
VPC Device2:
------------
(*,G)
oif1 (igmp)
```

IP PIM PRE-BUILD SPT

When the multicast source is in a Layer 3 cloud (outside the vPC domain), one vPC peer is elected as the forwarder for the source. This forwarder election is based on the metrics to reach the source. If there is a tie, the vPC primary is chosen as the forwarder. Only the forwarder has the vPC OIFs in its associated (S,G) and the nonforwarder (S,G) has 0 OIFs. Therefore, only the forwarder sends PIM (S,G) joins toward the source as shown in this example:

```
VPC Device1 (say this is Forwarder for Source 'S'):
------------
(*,G)
oif1 (igmp)
(S,G)
oif1 (mrib)
VPC Device2:
------------
(*,G)
oif1 (igmp)
(S,G)
NULL
```

In the case of a failure (for example, a Layer 3 Reverse Path Forwarding (RPF) link on the forwarder becomes inoperable or the forwarder gets reloaded), if the current nonforwarder ends up becoming the forwarder, it has to start sending PIM joins for (S,G) toward the source to pull the traffic. Depending upon the number of hops to reach the source, this operation might take some time (PIM is a hop-by-hop protocol).

To eliminate this issue and get better convergence, use the **ip pim pre-build-spt** command. This command enables PIM send joins even if the multicast route has 0 OIFs. In a vPC device, the nonforwarder sends PIM (S,G) joins upstream toward the source. The downside is that the link bandwidth upstream from the nonforwarder gets used for the traffic that is ultimately dropped by it. The benefits that result with better convergence far outweigh the link bandwidth usage. Therefore, we recommend that you use this command if you use vPCs.

vPC Peer Links and Routing

The First Hop Redundancy Protocols (FHRPs) interoperate with vPCs. The Hot Standby Routing Protocol

(HSRP), and Virtual Router Redundancy Protocol (VRRP) all interoperate with vPCs. We recommend that you dual-attach all Layer 3 devices to both vPC peer devices.

The primary FHRP device responds to ARP requests, even though the secondary vPC device forwards the data traffic.

To simplify initial configuration verification and vPC/HSRP troubleshooting, you can configure the primary vPC peer device with the FHRP active router highest priority.

In addition, you can use the priority command in the if-hsrp configuration mode to configure failover thresholds for when a group state enabled on a vPC peer link is in standby or in listen state. You can configure lower and upper thresholds to prevent the interface from going up and down.

VRRP acts similarly to HSRP when running on vPC peer devices. You should configure VRRP the same way that you configure HSRP.

When the primary vPC peer device fails over to the secondary vPC peer device, the FHRP traffic continues to flow seamlessly.

We recommend that you configure routing adjacency between the two vPC peer devices to act as a backup routing path. If one vPC peer device loses Layer 3 uplinks, the vPC can redirect the routed traffic to the other vPC peer device and leverage its active Layer 3 uplinks.

You can configure the inter-switch link for a backup routing path in the following ways:

- Create a Layer 3 link between the two vPC peer devices.
- Use the non-VPC VLAN trunk with a dedicated VLAN interface.
- Use a vPC peer link with a dedicated VLAN interface.

We do not recommend that you configure the burnt-in MAC address option (use-bia) for HSRP or manually configure virtual MAC addresses for any FHRP protocol in a vPC environment because these configurations can adversely affect vPC load balancing. The HSRP use-bia option is not supported on vPCs. When you are configuring custom MAC addresses, you must configure the same MAC address on both vPC peer devices.

You can use the **delay restore** command to configure a restore timer that delays the vPC coming back up until after the peer adjacency forms and the VLAN interfaces are back up. This feature enables you to avoid packet drops when the routing tables might not be converged before the vPC is once again passing traffic. Use the **delay restore** command to configure this feature.

To delay the VLAN interfaces on the restored vPC peer device from coming up, use the **interfaces-vlan** option of the **delay restore** command.

See the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide for more information about FHRPs and routing.

Best Practices for Layer 3 and vPC Configuration

This section describes best practices for using and configuring Layer 3 with vPC.

Layer 3 and vPC Configuration Overview

When a Layer 3 device is connected to a vPC domain through a vPC, it has the following views:

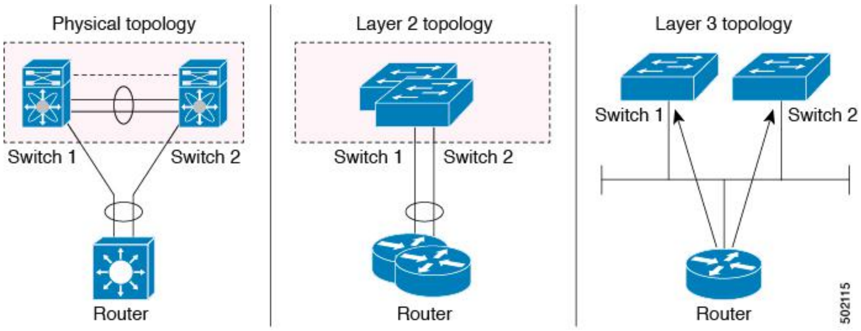
- At Layer 2, the Layer 3 device sees a unique Layer 2 switch presented by the vPC peer devices.
- At Layer 3, the Layer 3 device sees two distinct Layer 3 devices (one for each vPC peer device).

vPC is a Layer 2 virtualization technology, so at Layer 2, both vPC peer devices present themselves as a unique logical device to the rest of the network.

There is no virtualization technology at Layer 3, so each vPC peer device is seen as a distinct Layer 3 device by the rest of the network.

The following figure illustrates the two different Layer 2 and Layer 3 views with vPC.

Figure 16： Different Views for vPC Peer Devices



Guidelines for Layer 3 and vPC Configurations

To connect Layer 3 devices to a vPC domain, use Layer 3 links from Layer 3 devices to connect each vPC peer device.

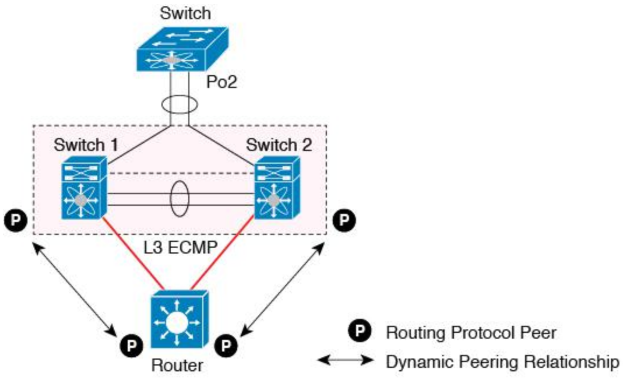
Layer 3 devices are able to initiate Layer 3 routing protocol adjacencies with both vPC peer devices.

One or multiple Layer 3 links can be used to connect a Layer 3 device to each vPC peer device. Inspur CN12900 series devices support Layer 3 Equal Cost Multipathing (ECMP) with up to 16 hardware load-sharing paths per prefix. Traffic from a vPC peer device to a Layer 3 device can be load-balanced across all the Layer 3 links interconnecting the two devices together.

Using Layer 3 ECMP on the Layer 3 device can effectively use all Layer 3 links from the device to the vPC domain. Traffic from a Layer 3 device to the vPC domain can be load-balanced across all the Layer 3 links interconnecting the two entities together.

The supported connection model for a Layer 3 device to the vPC domain is illustrated in the following figure.

Figure 17： Using Separate Layer 3 Links to Connect L3 Device to a vPC Domain



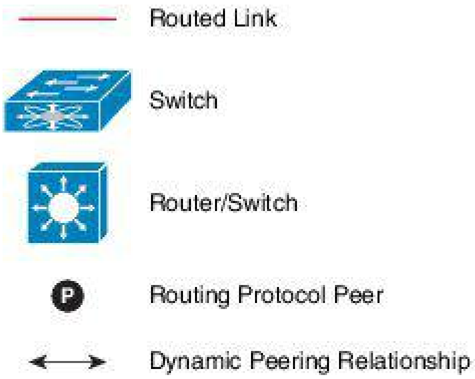
Follow these guidelines when connecting a Layer 3 device to the vPC domain:

- Use separate Layer 3 links to connect Layer 3 devices to the vPC domain.
- Do not use a Layer 2 vPC to attach a Layer 3 device to a vPC domain unless the Layer 3 device can statically route to the HSRP address configured on the vPC peer devices.
- When both routed and bridged traffic are required, use individual Layer 3 links for routed traffic and a separate Layer 2 port-channel for bridged traffic when both routed and bridged traffic are required.
- Enable Layer 3 connectivity between vPC peer device by configuring a VLAN network interface for the same VLAN from both devices or by using a dedicated Layer 3 link between the two peer devices (for Layer 3 backup routing path purposes).

Example Topologies for Layer 3 and vPC

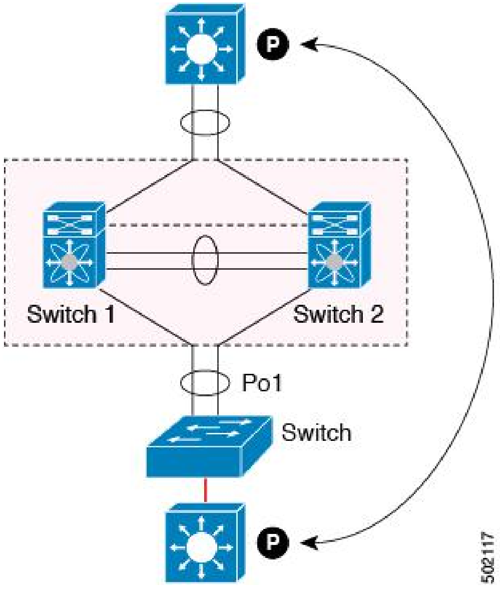
This section contains examples of network topologies for Layer 3 and vPC.

Figure 18： Legend



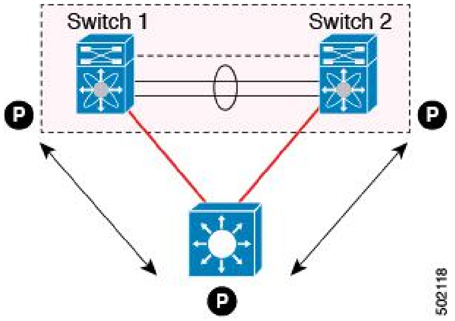
In this example, vPC is used as a Layer 2 transit path. Because there is no direct routing protocol peering adjacency from the Layer 3 device to any vPC peer device, this topology is supported.

Figure 19： Peering Between Routers



This example shows a topology that uses Layer 3 links to connect a Layer 3 device to the vPC domain.

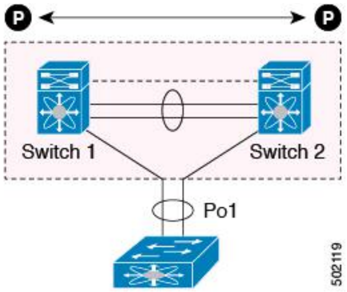
Figure 20： Peering with an External Router Using Layer 3 Links



This example shows peering between the two vPC peer devices with a Layer 3 backup routed path. If the Layer 3 uplinks on vPC peer device 1 or vPC peer device 2 fail, the path between the two peer devices is used to redirect traffic to the switch that has the Layer 3 uplinks in the up state.

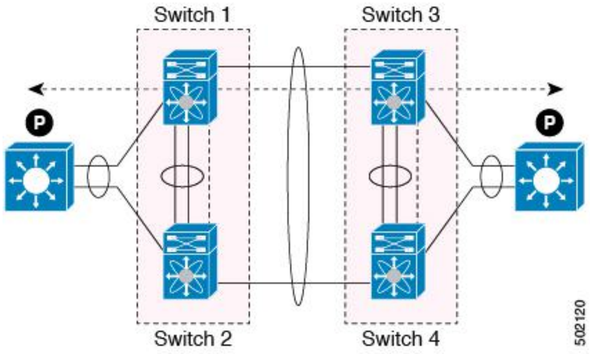
The Layer 3 backup routing path can be implemented using a dedicated interface VLAN (such as SVI) over the vPC peer-link or by using dedicated Layer 2 or Layer 3 links across the two vPC peer devices.

Figure 21： Peering Between vPC Devices for a Backup Routing Path



This example is similar to the Peering between Routers topology. The difference here is that the vPC domains are only used as Layer 2 transit paths.

Figure 22： Peering Between Two Routers with vPC Devices as Transit Switches



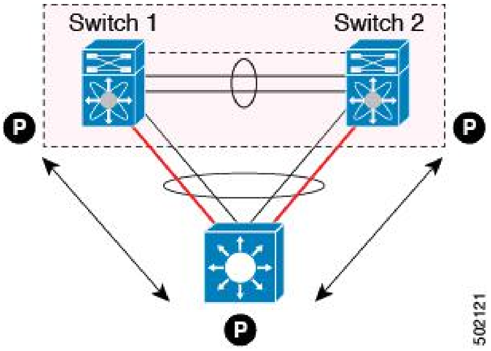
This example shows the Layer 3 device attached to the vPC domain through two different types of links, Layer 2 links and Layer 3 links.

The Layer 2 links are used for bridged traffic (traffic staying in the same VLAN) or inter-VLAN traffic (assuming vPC domain hosts the interface VLAN and associated HSRP configuration).

The Layer 3 links are used for routing protocol peering adjacency with each vPC peer device.

The purpose of this topology is to attract specific traffic to go through the Layer 3 device. Layer 3 links are also used to carry routed traffic from a Layer 3 device to the vPC domain.

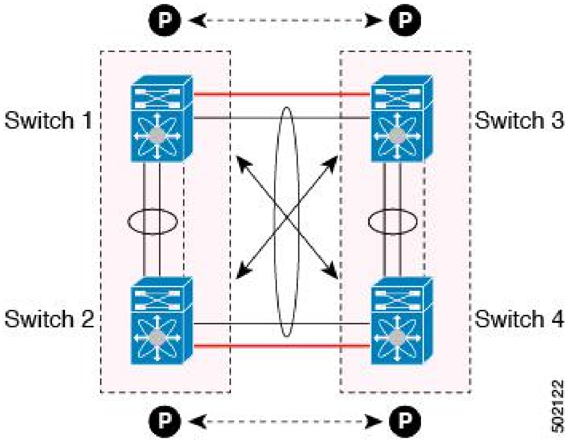
Figure 23 : Peering with an External Router on Parallel Interconnected Routed Ports



When routing protocol peering adjacency is required to be established between the two data centers, a best practice is to add dedicated Layer 3 links between the two sites as shown in this example.

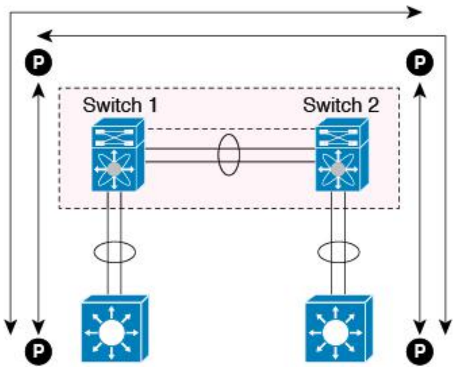
The vPC link between the two data centers carry bridged traffic or inter-VLAN traffic while the dedicated Layer 3 links carry the routed traffic across the two sites.

Figure 24 : Peering Over a vPC Interconnection on Parallel Interconnected Routed Ports



This example shows when the Layer 3 device is single-attached to the vPC domain, you can use a non-vPC VLAN with a dedicated inter-switch link to establish the routing protocol peering adjacency between the Layer 3 device and each vPC peer device. However, the non-vPC VLAN must be configured to use a static MAC that is different than the vPC VLAN.

Figure 25：Peering Over a PC Interconnection and Dedicated Interswitch Link Using non-vPC VLAN



vPC and Orphan Ports

When a device that is not vPC-capable connects to each peer, the connected ports are known as orphan ports because they are not members of a vPC. The device's link to one peer will be active (forwarding) and the other link will be standby (blocking) due to STP.

If a peer link failure or restoration occurs, an orphan port's connectivity might be bound to the vPC failure or restoration process. For example, if a device's active orphan port connects to the secondary vPC peer, the device loses any connections through the primary peer if a peer link failure occurs and the vPC ports are suspended by the secondary peer. If the secondary peer were to also suspend the active orphan port, the device's standby port becomes active, provides a connection to the primary peer, and restores connectivity. You can configure in the CLI that specific orphan ports are suspended by the secondary peer when it suspends its vPC ports and are restored when the vPC is restored.

8.1.12 Virtualization Support

All ports in a given vPC must be in the same VDC. This version of the software supports only one vPC domain per VDC. You can use the numbers from 1 to 4096 in each VDC to number the vPC.

8.1.13 vPC Recovery After an Outage

In a data center outage, both of the Inspur CN12900 Series devices that include a vPC get reloaded. Occasionally only one peer can be restored. With no functioning peer-keepalive or peer link, the vPC cannot function normally, but depending on your Inspur INOS-CN release, a method might be available to allow vPC services to use only the local ports of the functional peer.

Autorecovery

You can configure the Inspur CN12900 Series device to restore vPC services when its peer fails to come online by using the **auto-recovery** command. You must save this setting in the startup configuration. On reload, if the peer link is down and three consecutive peer-keepalive messages are lost, the secondary device assumes the primary STP role and the primary LACP role. The software reinitializes the vPCs, bringing up its local ports. Because there are no peers, the consistency check is bypassed for the local vPC ports. The device elects itself to be the STP primary regardless of its role priority and also acts as the master for LACP port roles.

vPC Peer Roles After a Recovery

When the other peer device completes its reload and adjacency forms, the following process occurs:

1. The first vPC peer maintains its current role to avoid any transition reset to other protocols. The peer accepts the other available role.

2. When an adjacency forms, consistency checks are performed and appropriate actions are taken.

8.1.14 vPC Forklift Upgrade Scenario

The following describes a scenario for migrating from a pair of Inspur CN12900 Series switches in a vPC topology to a different pair of Inspur CN12900 Series switches.

Considerations for a vPC forklift upgrade:

- vPC Role Election and Sticky-bit

When the two vPC systems are joined to form a vPC domain, priority decides which device is the vPC primary and which is the vPC secondary. When the primary device is reloaded, the system comes back online and connectivity to the vPC secondary device (now the operational primary) is restored. The operational role of the secondary device (operational primary) does not change (to avoid unnecessary disruptions). This behavior is achieved with a sticky-bit, where the sticky information is not saved in the startup configuration. This method makes the device that is up and running win over the reloaded device. Hence, the vPC primary becomes the vPC operational secondary. Sticky-bit is also set when a vPC node comes up with peer-link and peer-keepalive down and it becomes primary after the auto recovery period.

- vPC Delay Restore

The delay restore timer is used to delay the vPC from coming up on the restored vPC peer device after a reload when the peer adjacency is already established.

To delay the VLAN interfaces on the restored vPC peer device from coming up, use the **interfaces-vlan** option of the **delay restore** command.

- vPC Auto-Recovery

During a data center power outage when both vPC peer switches go down, if only one switch is restored, the auto-recovery feature allows that switch to assume the role of the primary switch and the vPC links come up after the auto-recovery time period. The default auto-recovery period is 240 seconds.

The following example is a migration scenario that replaces vPC peer nodes Node1 and Node2 with New_Node1 and New_Node2.

	Migration Step	Expected Behavior	Node1 Configured role (Ex: role priority 100)	Node1 Operational role	Node2 Configured role (Ex: role priority 200)	Node2 Operational role
1	Initial state	Traffic is forwarded by both vPC peers – Node1 and Node2. Node1 is primary and Node2 is secondary.	primary	Primary Sticky bit: False	secondary	Secondary Sticky bit: False
2	Node2 replacement – Shut all vPCs and uplinks on Node2. Peer-link and vPC peer-keepalive are in administrative up state.	Traffic converged on Primary vPC peer Node1.	primary	Primary Sticky bit: False	secondary	Secondary Sticky bit: False

	Migration Step	**Expected Behavior**	**Node1 Configured role (Ex: role priority 100)**	**Node1 Operational role**	**Node2 Configured role (Ex: role priority 200)**	**Node2 Operational role**
3	Remove Node2.	Node1 will continue to forward traffic.	primary	Primary Sticky bit: False	n/a	n/a
4	Configure New_Node2. Copy the configuration to startup config. vPC peer-link and peer-keepalive in administrative up state. Power off New_Node2. Make all connections. Power on New_Node2.	New_Node2 will come up as secondary. Node1 continue to be primary. Traffic will continue to be forwarded on Node01.	primary	Primary Sticky bit: False	secondary	Secondary Sticky bit: False
	Migration Step	**Expected Behavior**	**Node1 Configured role (Ex: role priority 100)**	**Node1 Operational role**	**Node2 Configured role (Ex: role priority 200)**	**Node2 Operational role**
5	Bring up all vPCs and uplink ports on New_Node2.	Traffic will be forwarded by both Node 1 and New_Node2.	primary	Primary Sticky bit: False	secondary	Secondary Sticky bit: False
6	Node1 replacement - Shut vPCs and uplinks on Node1.	Traffic will converge on New_Node2.	primary	Primary Sticky bit: False	secondary	Secondary Sticky bit: False
7	Remove Node1.	New_Node2 will become secondary, operational primary and sticky bit will be set to True.	n/a	n/a	secondary	Primary Sticky bit: True
8	Configure New_Node1. Copy running to startup.	New_Node1 will come up as primary,	primary	Secondary Sticky bit: False	secondary	Primary Sticky bit: True

	Migration Step	Expected Behavior	Node1 Configured role (Ex: role priority 100)	Node1 Operational role	Node2 Configured role (Ex: role priority 200)	Node2 Operational role
	Power off the new Node1. Make all connections. Power on New_Node1.	operational secondary.				
9	Bring up all vPCs and uplink ports on New_Node1.	Traffic will be forwarded by both New Node1 and new Node2.	primary	Secondary Sticky bit: False	secondary	Primary Sticky bit: True

8.2 Licensing Requirements for vPCs

The following table shows the licensing requirements for this feature:

Product	License Requirement
Inspur INOS-CN	Inspur INOS-CN vPCs require no license. Any feature not included in a license package is bundled with the Inspur INOS-CN system images and is provided at no extra charge to you.

8.3 Guidelines and Limitations

vPCs have the following configuration guidelines and limitations:

- Make sure that both vPC peers are in the same mode (regular mode or enhanced mode) before performing a non-disruptive upgrade.
- **show** commands with the **internal** keyword are not supported.
- Inspur CN12900 Series switches do not support NAT on vPC topology.
- The **show vpc consistency-checker** command is not available on Inspur CN12900 switches starting from Inspur INOS-CN.
- The **delay restore interface-bridge-domain** and **peer-gateway exclude-bridge-domain** commands are not available on Inspur CN12900-R platform switches starting from Inspur INOS-CN.
- vPC peers must run the same Inspur INOS-CN release. During a software upgrade, make sure to upgrade the primary vPC peer first.
- All ports for a given vPC must be in the same VDC.
- You must enable vPCs before you can configure them.
- You must configure the peer-keepalive link and messages before the system can form the vPC peer link.
- Only Layer 2 port channels can be in vPCs.
- You must configure both vPC peer devices; the configuration is not sent from one device to the other.
- To configure multilayer (back-to-back) vPCs, you must assign unique vPC domain ID for each respective vPC.
- Check that the necessary configuration parameters are compatible on both sides of the vPC peer link. See the

"Compatibility Parameters for vPC Interfaces" section for information about compatibility recommendations.

- You might experience minimal traffic disruption while configuring vPCs.
- The software does not support BIDR PIM on vPCs.
- The software does not support DHCP snooping, DAI, or IPSG in a vPC environment; DHCP Relay is supported.
- Port security is not supported on port channels.
- When **peer-switch** features are configured under **vpc domain** configuration mode on two Inspur CN12900 Series switches, the spanning-tree root changes even for VLANs that are not enabled on the vPC peer-link. Both the switches act as one system with one MAC address as the bridge address. This is true even for non-vPC mst-instance or VLANs. Therefore, a non vPC peer-link between the two switches gets blocked as a backup link. This is an expected behavior.
- We recommend that you configure all the port channels in the vPC using LACP with the interfaces in active mode.
- Back-to-back, multilayer vPC topologies require unique domain IDs on each respective vPC.
- Having the same Hot Standby Router Protocol (HSRP)/Virtual Router Redundancy Protocol (VRRP) group on all nodes on a double sided vPC is supported..
- When migrating from a pair of spine nodes to a pair of Inspur CN12900 devices, the HSRP priority should be configured so that the Inspur CN12900 vPC peers are in Active/Standby state. There is no support for Inspur CN12900 vPC peers in HSRP state to be in Active/Listen state, or Standby/Listen state.
- When using vPCs, we recommend that you use default timers for FHRP (HSRP, VRRP), and PIM configurations. There is no advantage in convergence times when using aggressive timers in vPC configurations.
- If you configure open shortest path first (OSPF) in a vPC environment, use the following timer commands in router configuration mode on the core switch to ensure fast OSPF convergence when a vPC peer link is shut down:

```
switch (config-router)# timers throttle spf 1 50 50

switch (config-router)# timers lsa-arrival 10
```

See the *Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide* for further details about OSPF.

- BFD for VRRP/HSRP is not supported in a vPC environment.
- The STP port cost is fixed to 200 in a vPC environment.
- Jumbo frames are enabled by default on the vPC peer link.
- To accommodate increased traffic when the vPC goes down and traffic needs to cross the peer-link, it is a best practice to use multiple high bandwidth interfaces (such as the 40G interfaces for the Inspur CN12900) across linecards for the peer-link.
- The **vpc orphan-ports suspend** command also applies to ports in non-vPC VLANs and Layer 3 ports. However, it is recommended to be used with ports in VPC VLANs.
- When configuring vPCs, the behavior previously provided by using the ip pim pre-build-spt command has now been enabled automatically by default and cannot be disabled.
- A vPC port channel member link that is operating in Individual state will be flapped while checking for VLAN inconsistencies. To avoid having the link flapped during server provisioning, disable the VPC graceful consistency check with the **no graceful consistency-check** command.

The following example disables the VPC graceful consistency check:

```
switch# conf t
Enter configuration commands, one per line. End with CNTL/Z.
switch(config)# vpc domain 1
switch(config-vpc-domain)# no graceful consistency-check
```

- vPC STP hitless role change feature is supported.
- vPC role change can be performed from either of the peer devices.
- If the original secondary device has higher role priority value than the original primary device, role swapping cannot be performed. Change the role priority on either vPC device so that the value of the original secondary device is lower than the original primary one. To view the existing role of a device, use the show vpc role command on local and peer switch.
- Always check the existing configured role priority before configuring vPC hitless role change feature
- In a vPC domain, enable the peer-switch command, where both vPC peers have same STP priorities, and ensure it is operational before issuing a role change. If you do not enable the peer-switch command, it can lead to convergence issues. Use **show spanning-tree summary | grep peer** command to verify whether the peer vPC switch is operational or not.

8.4 Default Settings

The following table lists the default settings for vPC parameters.

Table 12： Default vPC Parameters

Parameters	**Default**
vPC system priority	32667
vPC peer-keepalive message	Disabled
vPC peer-keepalive interval	1 second
vPC peer-keepalive timeout	5 seconds
vPC peer-keepalive UDP port	3200

8.5 Configuring vPCs

This section escribes how to configure vPCs using the command-line interface (CLI).

8.5.1 Enabling vPCs

You must enable the vPC functionality before you can configure and use vPCs.

SUMMARY STEPS

1. **configure terminal**
2. **feature vpc**
3. **exit**
4. **show feature**
5. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	**Purpose**
Step 1	**configure terminal**	Enters global configuration mode.

	Example:
	`switch# configure terminal`
	`switch(config)#`	

Step 2	**feature vpc** **Example:** `switch(config)# feature vpc`	Enables vPCs on the device.

Step 3	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		Exits global configuration mode.

Step 4	**show feature** **Example:** `switch# show feature`	(Optional) Displays which features are enabled on the device.
Step 5	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to enable the vPC feature:

```
switch# configure terminal
switch(config)# feature vpc
switch(config)# exit
switch(config)#
```

8.5.2 Disabling vPCs

SUMMARY STEPS

1. **configure terminal**
2. **no feature vpc**
3. **exit**
4. **show feature**
5. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
	`switch(config)#`	
Step 2	**no feature vpc** **Example:** `switch(config)#` **`no feature vpc`**	Disables vPCs on the device.

Step 3	**exit**
	Example:
	`switch(config)#` **`exit`**
	`switch#`
		Exits global configuration mode.

Step 4	**show feature** **Example:** `switch#` **`show feature`**	(Optional) Displays which features are enabled on the device.
Step 5	**copy running-config startup-config** **Example:** `switch#` **`copy running-config startup-config`**	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to disable the vPC feature:

```
switch# configure terminal
switch(config)# no feature vpc
switch(config)# exit
switch#
```

8.5.3 Creating a vPC Domain and Entering vpc-domain Mode

You can create a vPC domain and put the vPC peer link port channels into the identical vPC domain on both vPC peer devices. Use a unique vPC domain number throughout a single vPC domain . This domain ID is used to automatically to form the vPC system MAC address.

You can also use this command to enter vpc-domain command mode.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id* [**shut** | **no shut**]
3. **exit**
4. **show vpc brief**
5. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.

	Command or Action	Purpose

	Example:
	`switch# configure terminal`
	`switch(config)#`	

Step 2	**vpc domain** *domain-id* [**shut** | **no shut**]
	Example:
	`switch(config)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Creates a vPC domain on the device, and enters vpc-domain configuration mode for configuration purposes. There is no default; the range is from 1 to 1000.

Step 3	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 4	**show vpc brief** **Example:** `switch# show vpc brief`	(Optional) Displays brief information about each vPC domain.
Step 5	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to enter the vpc-domain command mode to configure an existing vPC domain:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-vpc-domain)# exit
switch(config)#
```

8.5.4 Configuring a vPC Keepalive Link and Messages

You can configure the destination IP for the peer-keepalive link that carries the keepalive messages. Optionally, you can configure other parameters for the keepalive messages.

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id* [**shut** | **no shut**]
3. **peer-keepalive destination** *ipaddress* [**hold-timeout** *secs* | **interval** *msecs* {**timeout** *secs*} | {**precedence**

{*prec-value* | **network** | **internet** | **critical** | **flash-override** | **flash** | **immediate priority** | **routine**}} | **tos** {*tos-value* | **max-reliability** | **max-throughput** | **min-delay** | **min-monetary-cost** | **normal**}} |**tos-byte** *tos-byte-value*} | **source** *ipaddress* | **vrf** {*name* | **management vpc-keepalive**}]

4. **exit**
5. **show vpc statistics**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vpc domain** *domain-id* [**shut** | **no shut**]
	Example:
	`switch(config)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Creates a vPC domain on the device, and enters vpc-domain configuration mode.

Step 3	**peer-keepalive destination** *ipaddress* [**hold-timeout** *secs* | **interval** *msecs* {**timeout** *secs*} | {**precedence** {*prec-value* | **network** | **internet** | **critical** | **flash-override** | **flash** | **immediate priority** | **routine**}} | **tos** {*tos-value* | **max-reliability** | **max-throughput** | **min-delay** | **min-monetary-cost** | **normal**}} |**tos-byte** *tos-byte-value*} | **source** *ipaddress* | **vrf** {*name* | **management** **vpc-keepalive**}]
	Example:
	`switch(config-vpc-domain)# peer-keepalive destination 172.28.230.85`
	`switch(config-vpc-domain)#`
		Configures the IPv4 address for the remote end of the vPC peer-keepalive link.
		Note The system does not form the vPC peer link until you configure a vPC peer-keepalive link.
		The management ports and VRF are the defaults.
		Note We recommend that you configure a separate VRF and use a Layer 3 port from each vPC peer device in that VRF for the vPC peer-keepalive link. For more information about creating and configuring VRFs, see the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide.

Step 4	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		Exits global configuration mode.

Step 5	**show vpc statistics**	(Optional) Displays information about the configuration

	Command or Action	Purpose
	Example: `switch# show vpc statistics`	for the keepalive messages.
Step 6	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

For more information about configuring VRFs, see the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide.

This example shows how to configure the destination and source IP address and VRF for the vPC-peer-keepalive link:

```
switch# configure terminal
switch(config)# vpc domain 100
switch(config-vpc-domain)# peer-keepalive destination 172.168.1.2 source 172.168.1.1 vrf
vpc-keepalive
switch(config-vpc-domain)# exit
switch#
```

8.5.5 Creating a vPC Peer Link

You create the vPC peer link by designating the port channel that you want on each device as the peer link for the specified vPC domain. We recommend that you configure the Layer 2 port channels that you are designating as the vPC peer link in trunk mode and that you use two ports on separate modules on each vPC peer device for redundancy.

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *channel-number*
3. **switchport mode trunk**
4. **switchport trunk allowed vlan** *vlan-list*
5. **vpc peer-link**
6. **exit**
7. **show vpc brief**
8. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

	Command or Action	Purpose

Step 2	**interface port-channel** *channel-number*
	Example:
	`switch(config)# `**`interface port-channel 20`**
	`switch(config-if)#`
		Selects the port channel that you want to use as the vPC peer link for this device, and enters interface configuration mode.

Step 3	**switchport mode trunk** **Example:** `switch(config-if)# `**`switchport mode trunk`**	(Optional) Configures this interface in trunk mode.
Step 4	**switchport trunk allowed vlan** *vlan-list* **Example:** `switch(config-if)# `**`switchport trunk allowed vlan 1-120,201-3967`**	(Optional) Configures the permitted VLAN list.

Step 5	**vpc peer-link**
	Example:
	`switch(config-if)# `**`vpc peer-link`**
	`switch(config-vpc-domain)#`
		Configures the selected port channel as the vPC peer link, and enters vpc-domain configuration mode.

Step 6	**exit**
	Example:
	`switch(config)# `**`exit`**
	`switch#`
		Exits vpc-domain configuration mode.

Step 7	**show vpc brief** **Example:** `switch# `**`show vpc brief`**	(Optional) Displays information about each vPC, including information about the vPC peer link.
Step 8	**copy running-config startup-config** **Example:** `switch# `**`copy running-config startup-config`**	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure a vPC peer link:

```
switch# configure terminal
switch(config)# interface port-channel 20
switch(config-if)# switchport mode
switch(config-if)# switchport mode trunk
switch(config-if)# switchport trunk allowed vlan 1-120,201-3967
switch(config-if)# vpc peer-link
switch(config-vpc-domain)# exit
switch(config)#
```

8.5.6 Configuring a vPC Peer-Gateway

You can configure vPC peer devices to act as the gateway for packets that are destined to the vPC peer device's MAC address.

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id* [**shut** | **no shut**]
3. **peer-gateway**
4. **exit**
5. **show vpc brief**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vpc domain** *domain-id* [**shut** | **no shut**]
	Example:
	`switch(config-if)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Creates a vPC domain if it does not already exist, and enters vpc-domain configuration mode.

Step 3	**peer-gateway** **Example:** `switch(config-vpc-domain)# peer-gateway` **Note** Disable IP redirects on all interface-vlans of this vPC domain for correct operation of this feature.	Enables Layer 3 forwarding for packets destined to the peer's gateway MAC address.

Step 4	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 5	**show vpc brief**	(Optional) Displays information about each vPC,

	Command or Action	Purpose
	Example: `switch# show vpc brief`	including information about the vPC peer link.
Step 6	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

8.5.7 Configuring Fast Convergence

Fast convergence feature is supported on Inspur CN12900 Series platforms. You can enable or disable the vPC optimizations using this command. To achieve faster convergence, you must enable **[no] fast-convergence** on both vPC peers to achieve fast-convergence.

SUMMARY STEPS

1. **configure terminal**
2. **switch(config) # vpc domain <domain>**
3. **switch(config) # peer-switch**
4. **switch(config) # show vpc peer-keepalive**
5. **switch(config) # delay restore { time }**
6. **switch(config) # peer-gateway**
7. **switch(config) # delay restore orphan-port**
8. **switch(config-vpc-domain)# fast-convergence**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**switch(config) # vpc domain <domain>**	Configure the VPC domain number.
Step 3	**switch(config) # peer-switch**	Define the peer switch.
Step 4	**switch(config) # show vpc peer-keepalive**	Displays information about the peer keepalive messages
Step 5	**switch(config) # delay restore { time }**	Number of seconds to delay bringing up the restored vPC peer device. The range is from 1 to 3600.
Step 6	**switch(config) # peer-gateway**	To enable Layer 3 forwarding for packets destined to the gateway MAC address of the virtual Port Channel (vPC), use the peer-gateway command. To disable Layer 3 forwarding packets, use the no form of this command.

	Command or Action	Purpose
		.
Step 7	**switch(config) # delay restore orphan-port**	Number of seconds to delay bringing up the restored device's orphan port
Step 8	**switch(config-vpc-domain)# fast-convergence**	Configure vPC fast convergence.

8.5.8 Configuring LACP vPC Convergence

Link Aggregation Control Protocol (LACP) vPC convergence feature is supported on Inspur CN12900 Series platforms. You can configure LACP vPC convergence feature for more efficient use of port channels.

SUMMARY STEPS

1. **configure terminal**
2. **switch(config) # interface** {*type/slot* | **portchannel** *number*}
3. **switch(config-if) # lacp vpc-convergence**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**switch(config) # interface** {*type/slot* | **portchannel** *number*}	Specifies an interface to configure, and enters interface configuration mode.
Step 3	**switch(config-if) # lacp vpc-convergence**	Configure LACP convergence.

8.5.9 Configuring Layer 3 over vPC

Layer 3 over vPC is supported on Inspur CN12900 Series switches.

Before you begin

Ensure that the peer-gateway feature is enabled and it is configured on both the peers and both the peers run an image that supports Layer 3 over vPC. If you enter the **layer3 peer-router** command without enabling the peer-gateway feature, a syslog message is displayed recommending you to enable the peer-gateway feature.

Ensure that the peer link is up.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **vpc domain** *domain-id*
3. switch(config-vpc-domain)#**layer3 peer-router**
4. switch(config-vpc-domain)# **exit**
5. (Optional) switch#**show vpc brief**
6. (Optional) switch#**copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	switch# **configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	switch(config)# **vpc domain** *domain-id*
	Example:
	`switch(config)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Creates a vPC domain if it does not already exist, and enters the vpc-domain configuration mode. There is no default; the range is from <1 to 1000>.

Step 3	switch(config-vpc-domain)#**layer3 peer-router**	Enables the Layer 3 device to form peering adjacency with both the peers. **Note** Configure this command in both the peers. If you configure this command only on one of the peers or you disable it on one peer, the operational state of layer 3 peer-router gets disabled. You get a notification when there is a change in the operational state.
Step 4	switch(config-vpc-domain)# **exit**	Exits the vpc-domain configuration mode.
Step 5	(Optional) switch#**show vpc brief**	Displays brief information about each vPC domain.
Step 6	(Optional) switch#**copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

The following example shows how to configure Layer 3 over vPC feature:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-vpc-domain)# layer3 peer-router
switch(config-vpc-domain)# exit
switch(config)#
```

This example shows how to verify if the Layer 3 over vPC feature is configured. The **Operational Layer3 Peer** is enabled or disabled depending up on how the operational state of Layer 3 over vPC is configured.

```
switch# show vpc brief
vPC domain id : 5
Peer status : peer adjacency formed ok
vPC keep-alive status : peer is alive
Configuration consistency status : success
```

```
Per-vlan consistency status : failed
Type-2 consistency status : success
vPC role : secondary
Number of vPCs configured : 2
Peer Gateway : Enabled
Peer gateway excluded VLANs : -
Dual-active excluded VLANs : -
Graceful Consistency Check : Enabled


Auto-recovery status : Enabled (timeout = 240 seconds)
Operational Layer3 Peer : Enabled
```

8.5.10 Layer 3 over vPC Supported Topologies

See the following Layer 3 over vPC supported topologies:

Figure 26 : Supported: Peering Over a vPC Interconnection Where the Router Peers with Both the vPC Peers.

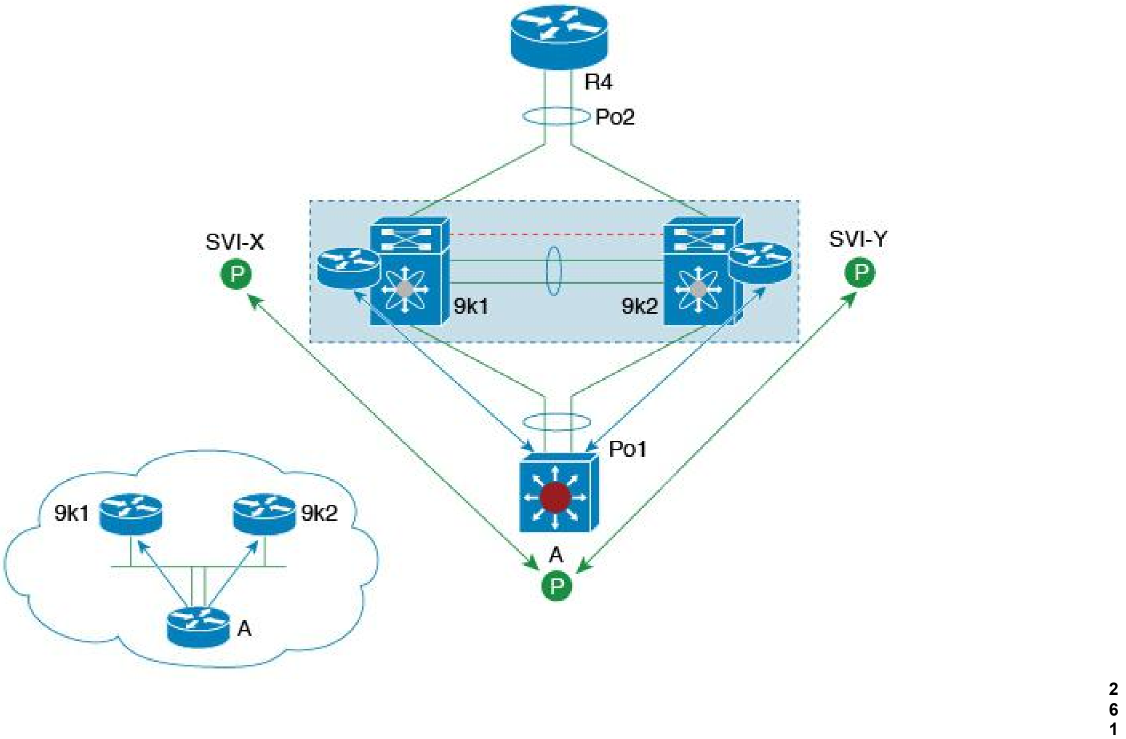


Figure 27：Supported: Peering Over an STP Interconnection Using a vPC VLAN Where the Router Peers with Both the vPC Peers.

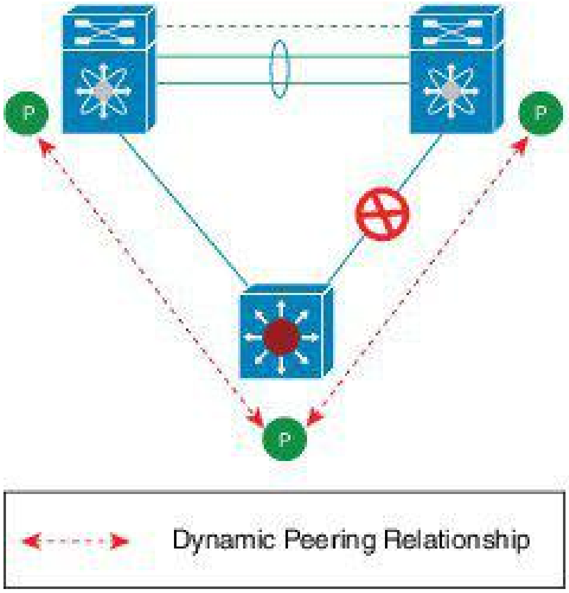


Figure 28：Supported: Peering Over an Orphan Device with Both the vPC Peers.

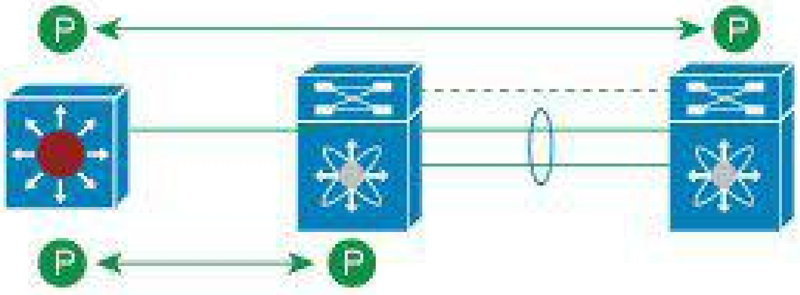
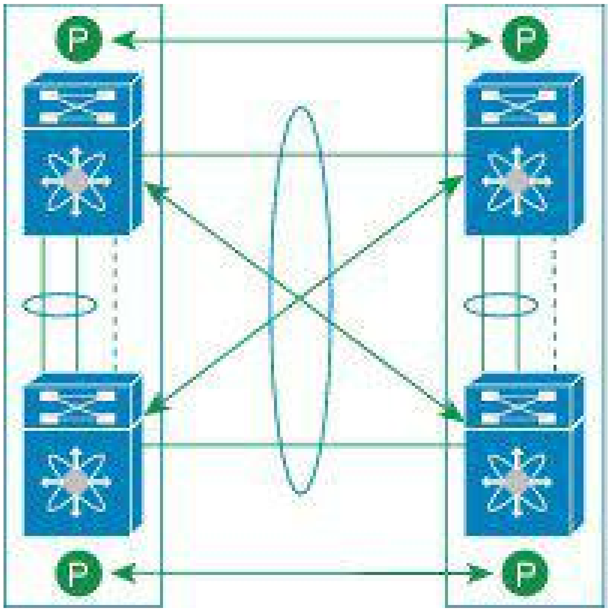


Figure 29 : Supported: Peering Over a vPC Interconnection Where Each Inspur Device Peers with Two vPC Peers.



8.5.11 Configuring a Graceful Consistency Check

You can configure the graceful consistency check feature, which is enabled by default. Unless this feature is enabled, the vPC is completely suspended when a mismatch in a mandatory compatibility parameter is introduced in a working vPC. When this feature is enabled, only the links on the secondary peer device are suspended. See the "Compatibility Parameters for vPC Interfaces" section for information about consistent configurations on the vPCs.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id* [**shut** | **no shut**]
3. **graceful consistency-check**
4. **exit**
5. **show vpc brief**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vpc domain** *domain-id* [**shut** | **no shut**]
	Example:
	`switch(config-if)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Creates a vPC domain if it does not already exist, and enters
		vpc-domain configuration mode.

	Command or Action	Purpose
Step 3	**graceful consistency-check** **Example:** `switch(config-vpc-domain)# graceful consistency-check`	Specifies that only the links on the secondary peer device are suspended when a mismatch is detected in a mandatory compatibility parameter. Use the **no** form of this command to disable the feature.

Step 4	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 5	**show vpc brief** **Example:** `switch# show vpc brief`	(Optional) Displays information on the vPCs.

Example

This example shows how to enable the graceful consistency check feature:

```
switch# configure terminal
switch(config)# vpc domain 5 switch(config-vpc-
domain)# graceful consistency-check switch(config-
vpc-domain)# exit
switch(config)#
```

8.5.12 Checking the Configuration Compatibility on a vPC Peer Link

After you have configured the vPC peer link on both vPC peer devices, check that the configurations are consistent on all vPC interfaces. See the “Compatibility Parameters for vPC Interfaces” section for information about consistent configurations on the vPCs.

SUMMARY STEPS

1. **configure terminal**
2. **show vpc consistency-parameters** {**global** | **interface port-channel** *channel-number*}

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**show vpc consistency-parameters** {**global** | **interface** **port-channel** *channel-number*} **Example:**	(Optional) Displays the status of those parameters that must be consistent across all vPC interfaces.

	Command or Action	Purpose
	switch(config)# **show vpc consistency-parameters global** switch(config)#	

Example

This example shows how to check that the required configurations are compatible across all the vPC interfaces:

```
switch# configure terminal
switch(config)# show vpc consistency-parameters global
switch(config)#
```

8.5.13 Moving Other Port Channels into a vPC

We recommend that you attach the vPC domain downstream port channel to two devices for redundancy.

To connect to the downstream device, you create a port channel from the downstream device to the primary vPC peer device and you create another port channel from the downstream device to the secondary peer device. On each vPC peer device, you assign a vPC number to the port channel that connects to the downstream device. You will experience minimal traffic disruption when you are creating vPCs.

Before you begin

Ensure that you have enabled the vPC feature.

Ensure that you are using a Layer 2 port channel.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *channel-number*
3. **vpc** *number*
4. **exit**
5. **show vpc brief**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** switch# **configure terminal** switch(config)#	Enters global configuration mode.
Step 2	**interface port-channel** *channel-number* **Example:** switch(config)# **interface port-channel 20** switch(config-if)#	Selects the port channel that you want to put into the vPC to connect to the downstream device, and enters interface configuration mode.
Step 3	**vpc** *number*	Configures the selected port channel into the vPC to connect

	Command or Action	Purpose

	Example:
	`switch(config-if)# vpc 5`
	`switch(config-vpc-domain)#`
		to the downstream device. You can use any module in the device for these port channels. The range is from 1 and 4096.
		Note The vPC number that you assign to the port channel connecting to the downstream device from the vPC peer device must be identical on both vPC peer devices.

Step 4	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 5	**show vpc brief** **Example:** `switch# show vpc brief`	(Optional) Displays information on the vPCs.
Step 6	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure a port channel to connect to the downstream device:

```
switch# configure terminal switch(config)#
interface port-channel 20
switch(config-if)# vpc
5 switch(config-if)#
exit switch(config)#
```

8.5.14 Manually Configuring a vPC Domain MAC Address

When you create a vPC domain, the Inspur INOS-CN software automatically creates a vPC system MAC address, which is used for operations that are confined to the link-scope, such as LACP. However, you might choose to configure the vPC domain MAC address manually.

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**

2. **vpc domain** *domain-id* [**shut** | **no shut**]
3. **system-mac** *mac-address*
4. **exit**
5. **show vpc role**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch#` **`configure terminal`**
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vpc domain** *domain-id* [**shut** | **no shut**]
	Example:
	`switch(config)#` **`vpc domain 5`**
	`switch(config-vpc-domain)#`
		Enters the vPC domain number that you want to configure.
		The system enters vpc-domain configuration mode.

Step 3	**system-mac** *mac-address*
	Example:
	`switch(config-vpc-domain)#` **`system-mac 23fb.4ab5.4c4e`**
	`switch(config-vpc-domain)#`
		Enters the MAC address that you want for the specified vPC domain in the following format: aaaa.bbbb.cccc.

Step 4	**exit**
	Example:
	`switch(config-vpc-domain)#` **`exit`**
	`switch#`
		Exits vpc-domain configuration mode.

Step 5	**show vpc role** **Example:** `switch#` **`show vpc brief`**	(Optional) Displays the vPC system MAC address.
Step 6	**copy running-config startup-config** **Example:** `switch#` **`copy running-config startup-config`**	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to manually configure a vPC domain MAC address:

```
switch# configure terminal
switch(config)# vpc domain 5 switch(config-vpc-
domain)# system-mac 13gb.4ab5.4c4e switch(config-
vpc-domain)# exit
```

```
switch(config)#
```

8.5.15 Manually Configuring the System Priority

When you create a vPC domain, the system automatically creates a vPC system priority. However, you can also manually configure a system priority for the vPC domain.

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id* [**shut** | **no shut**]
3. **system-priority** *priority*
4. **exit**
5. **show vpc role**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vpc domain** *domain-id* [**shut** | **no shut**]
	Example:
	`switch(config)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Enters the vPC domain number that you want to configure.
		The system enters vpc-domain configuration mode.

Step 3	**system-priority** *priority*
	Example:
	`switch(config-vpc-domain)# system-priority 4000`
	`switch(config-vpc-domain)#`
		Enters the system priority that you want for the specified
		vPC domain. The range of values is from 1 to 65535. The
		default value is 32667.

Step 4	**exit**
	Example:
	`switch(config-vpc-domain)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 5	**show vpc role** **Example:** `switch# show vpc role`	(Optional) Displays the vPC system priority.
Step 6	**copy running-config startup-config**	(Optional) Copies the running configuration to the

	Command or Action	Purpose
	Example: `switch# copy running-config startup-config`	startup configuration.

Example

This example shows how to manually configure the vPC domain system priority:

```
switch# configure terminal
switch(config)# vpc domain 5 switch(config-vpc-
domain)# system-priority 4000 switch(config-
vpc-domain)# exit
switch(config)#
```

8.5.16 Manually Configuring the vPC Peer Device Role

By default, the Inspur INOS-CN software elects a primary and secondary vPC peer device after you configure the vPC domain and both sides of the vPC peer link. However, you might want to elect a specific vPC peer device as the primary device for the vPC. Then, you would manually configure the role value for the vPC peer device that you want as the primary device to be lower than the other vPC peer device.

vPCs do not support role preemption. If the primary vPC peer device fails, the secondary vPC peer device takes over to become operationally the vPC primary device. However, the original operational roles are not restored if the formerly primary vPC comes up again.

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id* [**shut** | **no shut**]
3. **role priority** *priority*
4. **exit**
5. **show vpc role**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vpc domain** *domain-id* [**shut** | **no shut**]
	Example:
	`switch(config)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Enters the vPC domain number that you want to configure.
		The system enters vpc-domain configuration mode.

	Command or Action	Purpose

Step 3	**role priority** *priority*
	Example:
	`switch(config-vpc-domain)# role priority 4`
	`switch(config-vpc-domain)#`
		Enters the role priority that you want for the vPC system priority. The range of values is from 1 to 65636, and the default value is 32667. A lower value means that this switch has a better chance of being the primary vPC.

Step 4	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 5	**show vpc role** **Example:** `switch# show vpc role`	(Optional) Displays the vPC system priority.
Step 6	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to manually configure the role priority of the vPC peer device:

```
switch# configure terminal
switch(config)# vpc domain 5 switch(config-
vpc-domain)# role priority 4

switch(config-vpc-domain)# exit

switch(config)#
```

8.5.17 Configuring the Tracking Feature on a Single-Module vPC

If you must configure all the vPC peer links and core-facing interfaces on a single module, you should configure a track object and a track list that is associated with the Layer 3 link to the core and on all the links on the vPC peer link on both primary vPC peer devices. Once you configure this feature and if the primary vPC peer device fails, the system automatically suspends all the vPC links on the primary vPC peer device. This action forces all the vPC traffic to the secondary vPC peer device until the system stabilizes.

You must put this configuration on both vPC peer devices. Additionally, you should put the identical configuration on both vPC peer devices because either device can become the operationally primary vPC peer device.

Before you begin

Ensure that you have enabled the vPC feature.

Ensure that you have configured the track object and the track list. Ensue that you assign all interfaces that connect to the core and to the vPC peer link to the track-list object on both vPC peer devices.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id* [**shut** | **no shut**]
3. **track** *track-object-id*
4. **exit**
5. **show vpc brief**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vpc domain** *domain-id* [**shut** | **no shut**]
	Example:
	`switch(config)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Enters the vPC domain number that you want to configure, and enters vpc-domain configuration mode.

Step 3	**track** *track-object-id*
	Example:
	`switch(config-vpc-domain)# track object 23`
	`switch(config-vpc-domain)#`
		Adds the previously configured track-list object with its associated interfaces to the vPC domain. See the Inspur CN12900 Series INOS-CN Unicast Routing Configuration Guide for information about configuring object tracking and track lists.

Step 4	**exit**
	Example:
	`switch(config-vpc-domain)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 5	**show vpc brief** **Example:** `switch# show vpc brief`	(Optional) Displays information about the tracked objects.
Step 6	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to put the previously configured track-list object into the vPC domain on the vPC peer device:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-vpc-domain)# track object 5
switch(config-vpc-domain)# exit
switch(config)#
```

8.5.18 Configuring for Recovery After an Outage

If an outage occurs, the vPC waits for a peer adjacency to form on a switch reload. This situation can result in an unacceptably long service disruption. You can configure the Inspur CN12900 Series device to restore vPC services when its peer fails to come on line.

Configuring Reload Restore

The **reload restore** command and procedure described in this section is deprecated. We recommend that you use the **auto-recovery** command and procedure described in the "Configuring an Autorecovery" section.

You can configure the Inspur CN12900 Series device to restore vPC services when its peer fails to come online by using the **reload restore** command.

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id* [**shut** | **no shut**]
3. **reload restore** [**delay** *time-out*]
4. **exit**
5. **show running-config vpc**
6. **show vpc consistency-parameters interface port-channel** *number*
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vpc domain** *domain-id* [**shut** | **no shut**]
	Example:
	`switch(config)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Enters the vPC domain number that you want to configure,
		and enters vpc-domain configuration mode.

Step 3	**reload restore** [**delay** *time-out*] **Example:** `switch(config-vpc-domain)# reload restore`	Configures the vPC to assume its peer is not functional and to bring up the vPC. The default delay is 240 seconds. You can configure a time-out delay from 240 to 3600 seconds.

	Command or Action	Purpose
		Use the **no** form of the command to reset the vPC to its default settings.

Step 4	**exit**
	Example:
	`switch(config-vpc-domain)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 5	**show running-config vpc** **Example:** `switch# show running-config vpc`	(Optional) Displays information about the vPC, specifically the reload status.
Step 6	**show vpc consistency-parameters interface port-channel** *number* **Example:** `switch# show vpc consistency-parameters interface port-channel 1`	(Optional) Displays information about the vPC consistency parameters for the specified interface.
Step 7	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration. **Note** To ensure the reload feature is enabled, you should perform this step.

Example

This example shows how to set the vPC reload restore feature and save it in the switch startup configuration:

```
switch# configure terminal
Enter  configuration  commands,  one  per  line.  End  with
CNTL/Z. switch(config)# vpc domain 5
switch(config-vpc-domain)# reload restore
Warning:
Enables restoring of vPCs in a peer-detached state after reload, will wait for
240 seconds (by default) to determine if peer is un-reachable
switch(config-vpc-domain)#
exit switch(config)# exit
switch#   copy   running-config   startup-
config switch# show running-config vpc
!Command: show running-config vpc
!Time:  Wed  Mar  24  18:43:54
2010 version 5.0(2)
feature vpc
logging  level  vpc
6 vpc domain 5
```

```
reload restore
```

This example shows how to examine the consistency parameters:

```
switch# show vpc consistency-parameters interface port-channel 1

Legend:
Type  1  :  vPC  will  be  suspended  in  case  of
mismatch Name Type Local Value Peer Value
------------- ---- ----------- ---------------
STP Port Type 1 Default -
STP Port Guard 1 None -
STP MST Simulate PVST 1 Default
- mode 1 on -
Speed 1 1000 Mb/s
- Duplex 1 full -
Port Mode 1 trunk
- Native Vlan 1 1
- MTU 1 1500 -
Allowed  VLANs  -  1-3967,4048-
4093 Local suspended VLANs
```

Configuring an Autorecovery

You can configure the Inspur CN12900 Series device to restore vPC services when its peer fails to come online by using the auto-recovery command.

Note:

The auto-recovery feature is not enabled by default on Inspur CN12900 Switches. When the object tracking is triggered, the vPC secondary peer device does not change its role to that primary device and it reinitializes the vPC legs. You must manually configure auto-recovery on the vPC secondary peer device so that it can take over the primary role and reinitialize its vPC legs.

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id* [**shut** | **no shut**]
3. **auto-recovery** [**reload-delay** *time*]
4. **exit**
5. **show running-config vpc**
6. **show vpc consistency-parameters interface port-channel** *number*
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vpc domain** *domain-id* [**shut** | **no shut**]	Enters the vPC domain number that you want to configure,

	Command or Action	Purpose

	Example:
	`switch(config)# vpc domain 5`
	`switch(config-vpc-domain)#`
		and enters vpc-domain configuration mode.

Step 3	**auto-recovery** [**reload-delay** *time*] **Example:** `switch(config-vpc-domain)# auto-recovery`	Configures the vPC to assume its peer is not functional and to bring up the vPC, and specifies the time to wait after a reload to restore the vPC. The default delay is 240 seconds. You can configure a delay from 240 to 3600 seconds. Use the **no** form of the command to reset the vPC to its default settings.

Step 4	**exit**
	Example:
	`switch(config-vpc-domain)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 5	**show running-config vpc** **Example:** `switch# show running-config vpc`	(Optional) Displays information about the vPC, specifically the reload status.
Step 6	**show vpc consistency-parameters interface port-channel** *number* **Example:** `switch# show vpc consistency-parameters interface port-channel 1`	(Optional) Displays information about the vPC consistency parameters for the specified interface.
Step 7	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration. **Note** To ensure the autorecovery feature is enabled, you should perform this step.

Example

This example shows how to set the vPC autorecovery feature and save it in the switch startup configuration:

```
switch# configure terminal
Enter configuration commands, one per line. End with
CNTL/Z. switch(config)# vpc domain 5
switch(config-vpc-domain)# auto-recovery
```

```
Warning:
Enables restoring of vPCs in a peer-detached state after reload, will wait for
240 seconds to determine if peer is un-reachable
switch(config-vpc-domain)#
exit switch(config)# exit
switch# copy running-config startup-config
```

8.5.19 Configuring the Suspension of Orphan Ports

When a device that is not vPC-capable connects to each peer, the connected ports are known as orphan ports because they are not members of a vPC. You can explicitly declare physical interfaces as orphan ports to be suspended (shut down) by the secondary peer when it suspends its vPC ports in response to a peer link or peer-keepalive failure. The orphan ports are restored when the vPC is restored.

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**
2. **show vpc orphan-ports**
3. **interface** *type slot/port*
4. **vpc orphan-port suspend**
5. **exit**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**show vpc orphan-ports** **Example:** `switch# show vpc orphan-ports`	(Optional) Displays a list of the orphan ports.

Step 3	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an interface to configure, and enters interface configuration mode.

Step 4	**vpc orphan-port suspend** **Example:** `switch(config-if)# vpc orphan-ports suspend`	Configures the selected interface as a vPC orphan port to be suspended by the secondary peer in the case of a vPC failure.

	Command or Action	Purpose

Step 5	**exit**
	Example:
	`switch(config-if)# exit`
	`switch#`
		Exits interface configuration mode.

Step 6	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure an interface as a vPC orphan port to be suspended by the secondary peer in the case of a vPC failure:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# vpc orphan-ports suspend
switch(config-if)#
exit switch(config)#
```

Beginning Inspur INOS-CN, the output of the **show vpc orphan-ports** command is slightly different from that of the earlier releases. This example shows the output of **show vpc orphan-ports**command:

```
switch# show vpc orphan-ports
        ::Going through port database. Please be patient.::--------
VLAN--------       Orphan Ports
-------         -------------------------
1               Eth1/18, Eth3/23
2               Eth3/23
3               Eth3/23
4               Eth3/23
5               Eth3/23
```

8.5.20 Configuring Delay Restore on an Orphan Port

You can configure **delay restore orphan-port** command on Inspur CN12900 Series switches to configure a restore timer that delays the bringing up of restored device's orphan port.

SUMMARY STEPS

1. **configure terminal**
2. **switch(config) # vpc domain <domain>**
3. **switch(config) # peer-switch**
4. **switch(config) # show vpc peer-keepalive**
5. **switch(config) # delay restore { time }**
6. **switch(config) # peer-gateway**

7. **switch(config) # delay restore orphan-port**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**switch(config) # vpc domain <domain>**	Configure the VPC domain number.
Step 3	**switch(config) # peer-switch**	Define the peer switch.
Step 4	**switch(config) # show vpc peer-keepalive**	Displays information about the peer keepalive messages
Step 5	**switch(config) # delay restore { time }**	Number of seconds to delay bringing up the restored vPC peer device. The range is from 1 to 3600.
Step 6	**switch(config) # peer-gateway**	To enable Layer 3 forwarding for packets destined to the gateway MAC address of the virtual Port Channel (vPC), use the peer-gateway command. To disable Layer 3 forwarding packets, use the no form of this command. .
Step 7	**switch(config) # delay restore orphan-port**	Number of seconds to delay bringing up the restored device's orphan port

8.5.21 Configuring the vPC Peer Switch

You can configure the Inspur CN12900 Series device to make a pair of vPC devices appear as a single STP root in the Layer 2 topology.

Configuring a Pure vPC Peer Switch Topology

You can configure a pure vPC peer switch topology by using the peer-switch command and then setting the best possible (lowest) spanning tree bridge priority value.

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id* [**shut** | **no shut**]
3. **peer-switch**
4. **spanning-tree vlan** *vlan-range* **priority** *value*
5. **exit**
6. **show spanning-tree summary**
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vpc domain** *domain-id* [**shut** | **no shut**]
	Example:
	`switch(config)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Enters the vPC domain number that you want to configure,
		and enters vpc-domain configuration mode.

Step 3	**peer-switch** **Example:** `switch(config-vpc-domain)# peer-switch`	Enables the vPC switch pair to appear as a single STP root in the Layer 2 topology. Use the **no** form of the command to disable the peer switch vPC topology.
Step 4	**spanning-tree vlan** *vlan-range* **priority** *value* **Example:** `switch(config)# spanning-tree vlan 1 priority 8192`	Configures the bridge priority of the VLAN. Valid values are multiples of 4096. The default value is 32768.

Step 5	**exit**
	Example:
	`switch(config-vpc-domain)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 6	**show spanning-tree summary** **Example:** `switch# show spanning-tree summary`	(Optional) Displays a summary of the spanning tree port states including the vPC peer switch.
Step 7	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure a pure vPC peer switch topology:

```
switch# configure terminal
Enter configuration commands, one per line. End with
CNTL/Z. switch(config)# vpc domain 5
```

```
switch(config-vpc-domain)# peer-switch
2010 Apr 28 14:44:44 switch %STP-2-VPC_PEERSWITCH_CONFIG_ENABLED: vPC peer-switch
configuration is enabled. Please make sure to configure spanning tree "bridge" priority
as per recommended guidelines to make vPC peer-switch operational.
switch(config-vpc-domain)# spanning-tree vlan 1 priority
8192 switch(config-vpc-domain)# exit
switch(config)#
```

Configuring a Hybrid vPC Peer Switch Topology

You can configure a hybrid vPC and non-vPC peer switch topology by using the spanning-tree pseudo-information command to change the designated bridge ID so that it meets the STP VLAN-based load-balancing criteria and then change the root bridge ID priority to a value that is better than the best bridge priority. You then enable the peer switch.

Before you begin

Ensure that you have enabled the vPC feature.

When using a non-VPC dedicated trunk link between the VPC peers, the non-VPC VLANs should have a different pseudo root priority on the peers to prevent STP from blocking the VLANs.

SUMMARY STEPS

1. **configure terminal**
2. **spanning-tree pseudo-information**
3. **vlan** *vlan-id* **designated priority** *priority*
4. **vlan** *vlan-id* **root priority** *priority*
5. **vpc domain** *domain-id* [**shut** | **no shut**]
6. **peer-switch**
7. **exit**
8. **show spanning-tree summary**
9. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**spanning-tree pseudo-information**
	Example:
	`switch(config)# spanning-tree pseudo-information`
	`switch(config-pseudo)#`
		Configures the spanning tree pseudo information.

Step 3	**vlan** *vlan-id* **designated priority** *priority* **Example:** `switch(config-pseudo)# vlan 1 designated`	Configures the designated bridge priority of the VLAN. Valid values are multiples of 4096 from 0 to 61440.

	Command or Action	Purpose
	`priority 8192`	
Step 4	**vlan** *vlan-id* **root priority** *priority* **Example:** `switch(config-pseudo)# vlan 1 root priority 4096`	Configures the root bridge priority of the VLAN. Valid values are multiples of 4096 from 0 to 61440.

Step 5	**vpc domain** *domain-id* [**shut** | **no shut**]
	Example:
	`switch(config)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Enters the vPC domain number that you want to configure, and enters vpc-domain configuration mode.

Step 6	**peer-switch** **Example:** `switch(config-vpc-domain)# peer-switch`	Enables the vPC switch pair to appear as a single STP root in the Layer 2 topology. Use the **no** form of the command to disable the peer switch vPC topology.

Step 7	**exit**
	Example:
	`switch(config-vpc-domain)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 8	**show spanning-tree summary** **Example:** `switch# show spanning-tree summary`	(Optional) Displays a summary of the spanning tree port states including the vPC peer switch.
Step 9	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure a hybrid vPC peer switch topology:

```
switch# configure terminal
Enter configuration commands, one per line. End with
CNTL/Z. switch(config)# spanning-tree pseudo-information
switch(config-pseudo)# vlan 1 designated priority 8192
switch(config-pseudo)# vlan 1 root priority 4096
switch(config-pseudo)# vpc domain 5 switch(config-vpc-
domain)# peer-switch switch(config-vpc-domain)# exit

switch(config)#
```

8.5.22 Configuring Hitless vPC Role Change

Complete these steps to enable hitless vPC role change.

Before you begin

- Ensure that the vPC feature is enabled.
- Ensure that the vPC peer link is up
- Verify the role priority of devices

SUMMARY STEPS

1. **vpc role preempt**
2. **show vpc role**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**vpc role preempt**
	Example:
	`switch# vpc role preempt`
	`switch(config)#`
		Enable hitless vPC role change.

Step 2	**show vpc role** **Example:** `switch(config)# show vpc role`	(Optional) Verify hitless vPC role change feature.

Example

This example on how to configure hitless vPC role change:

```
vPC system-priority              : 32667
vPC local system-mac             : 8c:60:4f:03:84:41
vPC local role-priority          : 32668
vPC peer system-mac              : 8c:60:4f:03:84:43
vPC peer role-priority           : 32667
! Configure vPC hitless role change on the device!
switch(config)# vpc role preempt
! The following is an output from the show vpc role command
after the vPC hitless feature is configured
switch(config)#    show
vpc   role   vPC   Role
status
----------------------------------------------------
vPC role                         : primary
vPC system-mac                   : 00:00:00:00:00:00
vPC system-priority              : 32667
vPC local system-mac             : 8c:60:4f:03:84:41
vPC local role-priority          : 32666
vPC peer system-mac              : 8c:60:4f:03:84:43
vPC peer role-priority           : 32667
```

```
switch(config)#
```

Use Case Scenario for vPC Role Change

The hitless vPC role change feature can be used in the following scenarios:

- Role change request—When you want to change the roles of the peer devices in a vPC domain.
- Primary switch reload—When the devices comes up after a reload and roles are defined, you can use the hitless vPC role change feature to restore the roles. For example, after a reload if the primary device takes the role of operational secondary and the secondary device takes the role of primary operational, you can change the vPC peer roles to their original defined roles using the **vpc role preempt** command.
- Dual-active recovery—In a dual-active recovery scenario, the vPC primary switch continues to be (operational) primary, but the vPC secondary switch becomes the targeted primary switch and keeps its vPC member ports up. You can use the vPC hitless feature and restore the device roles. After the Dual-active recovery, if one side is operational primary and the other side operational secondary, then you can use the **vpc role preempt** command to restore the device roles to be primary and secondary

8.5.23 Enabling STP to Use the Inspur MAC Address

This procedure enables STP to use the Inspur MAC address (04:31:10:xx:xx:xx).

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id*
3. **[no] mac-address bpdu source version 2**
4. **exit**
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**vpc domain** *domain-id* **Example:** `switch(config)# vpc domain 5`	Creates a vPC domain if it does not already exist, and enters vpc-domain configuration mode.

Step 3	**[no] mac-address bpdu source version 2**
	Example:
	`switch(config-vpc-domain)# mac-address bpdu source`
	` version 2`
		Enables STP to use the Inspur MAC address (04:31:10:xx:xx:xx) as the source address of BPDUs
		generated on vPC ports.

Step 4	**exit**	Exits vpc-domain configuration mode.

	Command or Action	Purpose
	Example: `switch(config-vpc-domain)# exit`	
Step 5	(Optional) **copy running-config startup-config** **Example:** `switch(confg)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

8.6 Verifying the vPC Configuration

To display vPC configuration information, perform one of the following tasks:

Command	Purpose
show feature	Displays whether the vPC is enabled or not.
show vpc brief	Displays brief information about the vPCs.
show vpc consistency-parameters	Displays the status of those parameters that must be consistent across all vPC interfaces.
show running-config vpc	Displays running configuration information for vPCs.
show port-channel capacity	Displays how many port channels are configured and how many are still available on the device.
show vpc statistics	Displays statistics about the vPCs.
show vpc peer-keepalive	Displays information about the peer-keepalive messages.
show vpc role	Displays the peer status, the role of the local device, the vPC system MAC address and system priority, and the MAC address and priority for the local vPC device.

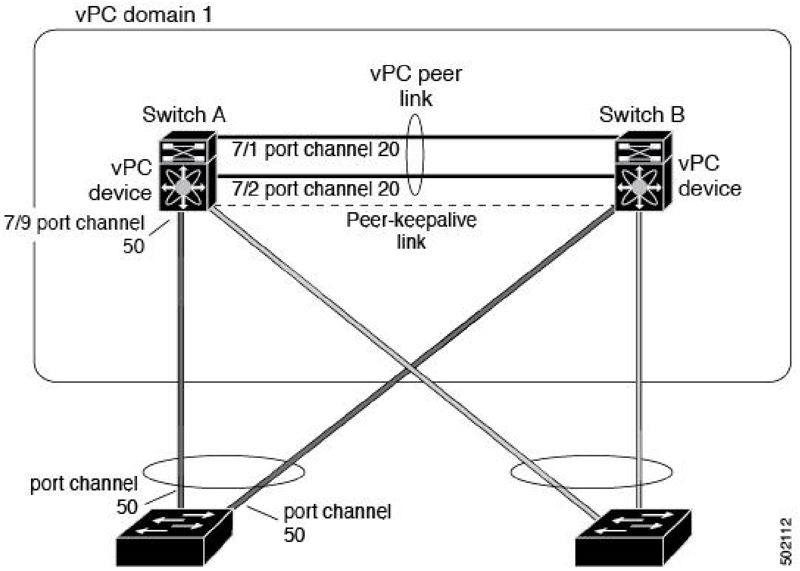
8.7 Monitoring vPCs

Use the **show vpc statistics**command to display vPC statistics.

8.8 Configuration Examples for vPCs

The following example shows how to configure vPC on device A as shown in the figure:

Figure 30 : vPC Configuration Example



1. Enable vPC and LACP.

```
switch# configure terminal
switch(config)# feature vPC
switch(config)# feature lacp
```

2. (Optional) Configure one of the interfaces that you want to be a peer link in the dedicated port mode.

```
switch(config)# interface ethernet
7/1, ethernet 7/3, ethernet 7/5.
ethernet 7/7 switch(config-if)#
shutdown switch(config-if)# exit
switch(config)# interface ethernet
7/1 switch(config-if)# rate-mode
dedicated switch(config-if)# no
shutdown switch(config-if)# exit
switch(config)#
```

3. (Optional) Configure the second, redundant interface that you want to be a peer link in the dedicated port mode.

```
switch(config)# interface ethernet 7/2, ethernet
7/4, ethernet 7/6. ethernet 7/8
switch(config-if)# shutdown
switch(config-if)# exit
switch(config)# interface ethernet 7/2
switch(config-if)# rate-mode
dedicated switch(config-if)# no
shutdown switch(config-if)# exit
switch(config)#
```

4. Configure the two interfaces (for redundancy) that you want to be in the peer link to be an active Layer 2

LACP port channel.

```
switch(config)# interface ethernet 7/1-2
switch(config-if)# switchport
switch(config-if)# switchport mode trunk
switch(config-if)# switchport trunk allowed vlan 1-
50 switch(config-if)# switchport trunk native vlan
20 switch(config-if)# channel-group 20 mode active
switch(config-if)# exit
```

5. Create and enable the VLANs.

```
switch(config)# vlan 1-50
switch(config-vlan)# no shutdown
switch(config-vlan)# exit
```

6. Create a separate VRF for the vPC peer-keepalive link and add a Layer 3 interface to that VRF.

```
switch(config)# vrf context
pkal switch(config-vrf)# exit
switch(config)# interface ethernet 8/1
switch(config-if)# vrf member pkal
switch(config-if)# ip address 172.23.145.218/24
switch(config-if)# no shutdown
switch(config-if)# exit
```

7. Create the vPC domain and add the vPC peer-keepalive link.

```
switch(config)# vpc domain 1 switch(config-
vpc-domain)# peer-keepalive
 destination 172.23.145.217 source 172.23.145.218 vrf pkal
switch(config-vpc-domain)# exit
```

8. Configure the vPC peer link.

```
switch(config)# interface port-channel 20 switch(config-
if)# switchport mode trunk switch(config-if)# switchport
trunk allowed vlan 1-50
switch(config-if)# vpc peer-link
switch(config-if)# exit
switch(config)#
```

9. Configure the interface for the port channel to the downstream device of the vPC.

```
switch(config)# interface ethernet 7/9
switch(config-if)# switchport mode trunk
switch(config-if)# allowed vlan 1-50
switch(config-if)# native vlan 20
switch(config-if)# channel-group 50 mode active
switch(config-if)# exit
switch(config)# interface port-channel 50
switch(config-if)# vpc 50 switch(config-
if)# exit
switch(config)#
```

10. Save the configuration.

```
switch(config)# copy running-config startup-config
```

8.9 Related Documents

Related Topic	Related Topic
System management	System management
High availability	High availability
Release Notes	Release Notes

CHAPTER 9 Configuring IP Tunnels

9.1 Information About IP Tunnels

IP tunnels can encapsulate a same-layer or higher layer protocol and transport the result over IP through a tunnel created between two devices.

9.1.1 IP Tunnel Overview

IP tunnels consists of the following three main components:

- Passenger protocol—The protocol that needs to be encapsulated. IPv4 is an example of a passenger protocol.
- Carrier protocol—The protocol that is used to encapsulate the passenger protocol. Inspur INOS-CN supports GRE as a carrier protocol.
- Transport protocol—The protocol that is used to carry the encapsulated protocol. IPv4 is an example of a transport protocol. An IP tunnel takes a passenger protocol, such as IPv4, and encapsulates that protocol within a carrier protocol, such as GRE. The device then transmits this carrier protocol over a transport protocol, such as IPv4.

You configure a tunnel interface with matching characteristics on each end of the tunnel.

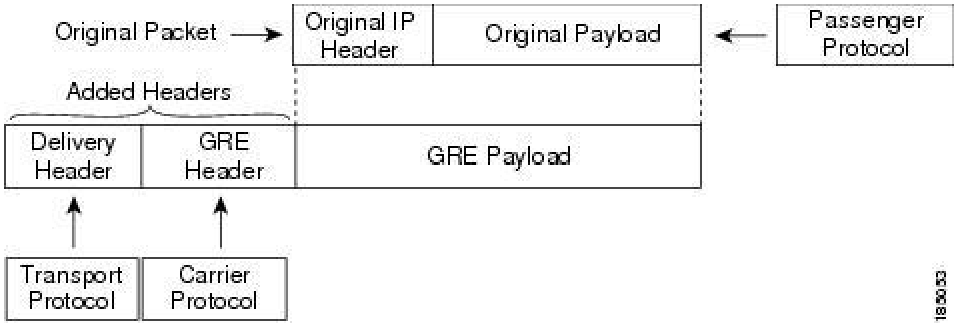
You must enable the tunnel feature before you can configure it. The system automatically takes a checkpoint prior to disabling the feature, and you can roll back to this checkpoint. See Inspur CN12900 Series INOS-CN System Management Configuration Guide for information about rollbacks and checkpoints.

9.1.2 GRE Tunnels

You can use generic routing encapsulation (GRE) as the carrier protocol for a variety of passenger protocols.

The following figure shows the IP tunnel components for a GRE tunnel. The original passenger protocol packet becomes the GRE payload and the device adds a GRE header to the packet. The device then adds the transport protocol header to the packet and transmits it.

Figure 31*：*GRE PDU



9.1.3 Point-to-Point IP-in-IP Tunnel Encapsulation and Decapsulation

The point-to-point IP-in-IP encapsulation and decapsulation is a type of tunnel that you can create to send encapsulated packets from a source tunnel interface to a destination tunnel interface. This type of tunnel will carry both inbound and outbound traffic.

9.1.4 Multi-Point IP-in-IP Tunnel Decapsulation

The multi-point IP-in-IP decapsulate-any is a type of tunnel that you can create to decapsulate packets from any number of IP-in-IP tunnels to one tunnel interface. This tunnel will not carry any outbound traffic. However, any number of remote tunnel endpoints can use a tunnel configured this way as their destination.

9.1.5 Path MTU Discovery

Path maximum transmission unit (MTU) discovery (PMTUD) prevents fragmentation in the path between two endpoints by dynamically determining the lowest MTU along the path from the packet's source to its destination. PMTUD reduces the send MTU value for the connection if the interface receives information that the packet would require fragmentation.

When you enable PMTUD, the interface sets the Don't Fragment (DF) bit on all packets that traverse the tunnel. If a packet that enters the tunnel encounters a link with a smaller MTU than the MTU value for the packet, the remote link drops the packet and sends an ICMP message back to the sender of the packet. This message indicates that fragmentation was required (but not permitted) and provides the MTU of the link that dropped the packet.

9.1.6 High Availability

IP tunnels support stateful restarts. A stateful restart occurs on a supervisor switchover. After the switchover, Inspur INOS-CN applies the runtime configuration after the switchover.

9.2 Licensing Requirements for IP Tunnels

The following table shows the licensing requirements for this feature:

Product	**License Requirement**
Inspur INOS-CN	IP tunnels require an Enterprise Services license.

9.3 Prerequisites for IP Tunnels

IP tunnels have the following prerequisites:

- You must be familiar with TCP/IP fundamentals to configure IP tunnels.
- You are logged on to the switch.
- You must enable the tunneling feature in a device before you can configure and enable any IP tunnels.

9.4 Guidelines and Limitations

IP tunnels have the following configuration guidelines and limitations:

- Guidelines for **source-direct** and **ipv6ipv6-decapsulate-any** options for tunnels:
- The **source-direct** command is supported on the Inspur CN12900 platform switches with the Application Spine Engine (ASE) and the Leaf Spine Engine (LSE).

Inspur CN12900 platform switches with the Network Forwarding Engine (NFE) do not support the **tunnel source direct** command.

The **tunnel source direct** command with the **tunnel mode ipv6ipv6 decapsulate-any** command on the Inspur CN12900 platform switches is only supported in the MPLS heavy routing template.

- The IP tunnel supports the **tunnel source** CLI command with interface, IPv4 address, IPv6 address, or IPv4 prefix. You can configure IP-in-IP tunnel decapsulation on directly connected IP addresses (for example, physical interface, port-channel, loopback, and SVI) using the new **tunnel source direct** CLI command. You can select the IP ECMP links when there are multiple IP links between the two switches. A single tunnel interface can decapsulate the tunneled packets whose outer destination IP is any of the IPv4 or IPv6 address that is locally configured and it is operationally *Up* in the switch.
- Currently, **tunnel mode ipip decapsulate-any** is supported for decapsulating IPv4 payload over IPv4 transport (IPv4inIPv4 packets). **tunnel mode ipv6ipv6 decapsulate-any** command is introduced to support IPv6 payload over IPv6 transport (IPv6inIPv6 packets).
- The tunnel **source direct** and **tunnel mode ipv6ipv6 decapsulate-any** CLI commands are not supported on Inspur CN12900 platform switches with the Network Formation Engine (NFE).
- The **tunnel source direct** CLI command is supported only when an administrator uses the IP-in-IP decapsulation to source route the packets through the network. The source-direct tunnel is always operationally *Up* unless it is administratively shut down. The directly connected interfaces are identified using the **show ip route direct** CLI command.
- The **tunnel source direct** CLI command is supported only on decapsulate-any tunnel modes, for example, **tunnel mode ipip decapsulate-any** and **tunnel mode ipv6ipv6 decapsulate-any**.
- Auto-recovery for source-direct is not supported.
- To enable IPv6 on ipv6ipv6 decap-any tunnel interface, configure a valid IPv6 address or configure **ipv6 address use-link-local-only** under the tunnel interface.
- See the following hardware limitations on the maximum sources that can be accommodated on a source direct tunnel and the related behavior:
- Source direct tunnel is now supported for Inspur CN12900 Series switches with Network Forwarding Engine (NFE), Application Spine Engine (ASE), and Leaf Spine Engine (LSE). Most of the limitations are only in case of scaled SIP (number of total IP/IPv6 addresses on the interfaces (L3, sub-interface, PC, PC-sub interfaces, loopback, SVI, and any secondary IP/IPv6 addresses.)

See the following sample use cases.

- Use Case 1: Non-deterministic behavior of which SIP gets installed if the number of IP/IPv6 interface scale is more.

Both the switches have 512 entries for tunnel SIP. With tunnel source, direct any IP or IPv6 address w.r.t **ipip or ipv6ipv6 decap any** with tunnel source gets installed in the above table.

The insertion of these entries is on a first come first serve basis without any CLI command to control which interface IP addresses get installed. If the system has more number of IP/IPv6 interfaces to be installed, the behavior is non-deterministic (The behavior can change across reload with interface flaps.)

- Use Case 2: The scale numbers are different in both switches and each has its own advantages and disadvantages.

IPv4 individual scale can be more (up to 512) in case of switches with NFE but it is shared with IPv6. In the switches with ASE and LSE, the IPv4 individual scale can be 256 but it is not shared with IPv6.

Whenever the tunnel decap table gets filled, the TABLE_FULL error is displayed. If some entry gets deleted after the table gets full, the table full error is cleared.

Table 13：Scale Numbers

Commands	Switches with NFE: Table size 512, v4 takes 1 entry, v6 takes 4 entries	Switches with ASE and LSE: Table size 512, v4 takes 1 entry, v6 takes 2 entries (paired index)
IPIP decap any with tunnel source direct	Shared between v4 and v6, v6 takes 4 entries	Dedicated 256

	v4 + 4 *v6 =512 Maximum entries can be 512 with no v6 entries	
IPv6IPv6 decap any with tunnel source direct	Shared between v4 and v6, v6 takes 4 entries v4 + 4 *v6 =512 Maximum entries can be 128 with no v4 entries	Dedicated 128

- Use Case 3: Auto-recovery is not supported.

If any entry does not get installed in the hardware due to exhaustion of above table, removal of an already installed IP/IPv6 from interfaces does not automatically trigger the addition of the failed SIP in the table though the table has space now. You need to flap the tunnel interface or IP interface to get them installed.

However, if an entry does not get installed in the hardware due to a duplicate entry (if there was already a **decap-any** with one source present and now the **source direct tunnel** CLI command is configured, there is a duplicate entry for the prior source configured) that was taken care of by removing the entry only when both the tunnels get deleted.

- For Inspur CN12900 Series switches with Network Forwarding Engine (NFE) and Application Spine Engine (ASE), the syslog is different as the dedicated IPv4 and IPv6 decap antries are carved in the syslog. If the **tunnel-decap-table** is full, the user gets a syslog as follows:

```
2017 Apr 6 12:18:04 switch %$ VDC-1 %$ %IPFIB-2-
 FIB_HW_IPV4_TUNNEL_DECAP_TABLE_FULL: IPv4 tunnel decap hardware table full.
IP tunnel decapsulation may not work for some GRE/IPinIP traffic
2017 Apr 6 12:18:11 switch %$ VDC-1 %$ %IPFIB-2-
 FIB_HW_IPV6_TUNNEL_DECAP_TABLE_FULL: IPv6 tunnel decap hardware table full.
IP tunnel decapsulation may not work for some GRE/IPinIP traffic
```

If the table is full and once some entry gets deleted from the table (due to an interface being operationally down or removal of IP address), the clear syslog for the table is displayed. Note that the deletion of the tunnel removes all the entries that are added as part of that tunnel.

```
2017 Apr 5 13:29:25 switch %$ VDC-1 %$ %IPFIB-2-
FIB_HW_IPV4_TUNNEL_DECAP_TABLE_FULL_CLRD: IPv4 tunnel decap hardware table
 full exception cleared
2017 Apr 4 19:41:22 switch %$ VDC-1 %$ %IPFIB-2-
FIB_HW_IPV6_TUNNEL_DECAP_TABLE_FULL_CLRD: IPv6 tunnel decap hardware table
 full exception cleared
```

- IP-in-IP tunnel decapsulation is supported on IPv6 enabled networks.

```
!
interface tunnel 1
  ipv6 address use-link-local-only
  tunnel mode ipv6ipv6 decapsulate-any
  tunnel source direct
  description IPinIP Decapsulation Interface
  mtu 1476
  no shutdown
```

- The **show** commands with the **internal** keyword are not supported.

- Inspur INOS-CN supports only the following protocols:
- IPv4 passenger protocol.
- GRE carrier protocol.
- Inspur INOS-CN supports the following maximum number of tunnels:
- IP tunnels - 8 tunnels.
- GRE and IP-in-IP regular tunnels - 8 tunnels.
- IP tunnels do not support access control lists (ACLs) or QoS policies.
- Inspur INOS-CN supports the GRE header defined in IETF RFC 2784. Inspur INOS-CN does not support tunnel keys and other options from IETF RFC 1701.
- Inspur INOS-CN does not support GRE tunnel keepalives.
- All unicast routing protocols are supported by IP tunnels.
- The IP tunnel interface cannot be configured to be a span source or destination.
- IP tunnels do not support PIM or other Multicast features and protocols.
- The selection of IP-in-IP tunnel based on the PBR policy is not supported.
- IP tunnels are supported only in the default **system routing** mode and not in other modes.
- When configuring a tunnel interface to **ipip mode**, the maximum mtu value is 9196 .

When downgrading from INOS-CN to an earlier release, with a tunnel interface in **ipip mode** that has an mtu value of 9196, the mtu configuration is lost as a result of the downgrade operation. As a best practice, adjust the mtu value to 9192 before commencing the downgrade to avoid losing the mtu configuration.

- When configuring a tunnel interface to **ipip mode**, the default mtu value is 1480 .

When downgrading from INOS-CN or later release to an earlier release, with a tunnel interface in **ipip mode** with no explicit mtu configuration, the mtu value changes as a result of the downgrade operation from 1480 to 1476. As a best practice, adjust the mtu value to 1476 before commencing the downgrade to avoid any changes to the mtu value.

When upgrading to INOS-CN or later release, , with a tunnel interface in **ipip mode** with no explicit mtu configuration, the mtu value changes as a result of the upgrade operation from 1476 to 1480. As a best practice, adjust the mtu value to 1480 before commencing the upgrade to avoid any changes to the mtu value.

- Tx packects originating from the switch, such as control pkts, are not included in Tx statistics.
- Tunnel destinations that are reachable over another tunnel are not supported.
- The consistency checker is not supported for routes over a tunnel.
- Non-IP routing protocols, such as isis, are not supported over IP-in-IP tunnels.
- RFC5549 is not be supported over tunnels.
- BGP adjacency over tunnel is not supported in a scenario where the tunnel interface and tunnel source are in same VRF (example: VRF-A) and tunnel destination is reachable with route-leak from opposite end (example: via VRF-B)

9.5 Default Settings

The following table lists the default settings for IP tunnel parameters.

Table 14： Default IP Tunnel Parameters

Parameters	**Default**
Path MTU discovery age timer	10 minutes
Path MTU discovery minimum MTU	64
Tunnel feature	Disabled

9.6 Configuring IP Tunnels

9.6.1 Enabling Tunneling

You must enable the tunneling feature before you can configure any IP tunnels.

SUMMARY STEPS

1. **configure terminal**
2. **feature tunnel**
3. **exit**
4. **show feature**
5. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature tunnel**
	Example:
	`switch(config)# feature tunnel`
	`switch(config-if)#`
		Allows the creation of a new tunnel interface.
		To disable the tunnel interface feature, use the **no** form of this command.

Step 3	**exit**
	Example:
	`switch(config-if)# exit`
	`switch#`
		Exits the interface mode and returns to the configuration mode.

Step 4	**show feature** **Example:** `switch(config-if)# show feature`	(Optional) Displays information about the features enabled on the device.
Step 5	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves this configuration change.

9.6.2 Creating a Tunnel Interface

You can create a tunnel interface and then configure this logical interface for your IP tunnel.

Command	Purpose
no interface tunnel *number* **Example:** `switch(config)# no interface tunnel 1`	Deletes the tunnel interface and the associated configuration.
description *string* **Example:** `switch(config-if)# description GRE tunnel`	Configures a description for the tunnel.
mtu *value* **Example:** `switch(config-if)# mtu 1400`	Sets the MTU of IP packets sent on an interface.
tunnel ttl *value* **Example:** `switch(config-if)# tunnel ttl 100`	Sets the tunnel time-to-live value. The range is from 1 to 255.

Before you begin

You can configure the tunnel source and the tunnel destination in different VRFs. Ensure that you have enabled the tunneling feature.

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel** *number*
3. **tunnel mode** {**gre ip** | **ipip** {**ip** | **decapsulate-any**}}
4. **tunnel source** {*ip-address* |*interface-name*}
5. **tunnel destination** {*ip-address* |*host-name*}
6. **tunnel use-vrf** *vrf-name*
7. **show interfaces tunnel** *number*
8. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface tunnel** *number*
	Example:
	`switch(config)# interface tunnel 1`
	`switch(config-if)#`
		Creates a new tunnel interface.

	Command or Action	Purpose
Step 3	**tunnel mode** {**gre ip** | **ipip** {**ip** | **decapsulate-any**}}	Sets this tunnel mode to GRE, ipip, or ipip decapsulate-only. The **gre** and **ip** keywords specify that GRE encapsulation over IP will be used. The **ipip** keyword specifies that IP-in-IP encapsulation will be used. The optional **decapsulate-any** keyword terminates IP-in-IP tunnels at one tunnel interface. This keyword creates a tunnel that will not carry any outbound traffic. However, remote tunnel endpoints can use a tunnel configured as their destination.

Step 4	**tunnel source** {*ip-address* |*interface-name*}
	Example:
	`switch(config-if)# tunnel source ethernet 1/2`
		Configures the source address for this IP tunnel. The source can be specified by IP address or logical interface name.

Step 5	**tunnel destination** {*ip-address* |*host-name*}
	Example:
	`switch(config-if)# tunnel destination 192.0.2.1`
		Configures the destination address for this IP tunnel. The destination can be specified by IP address or logical host name.

Step 6	**tunnel use-vrf** *vrf-name* **Example:** `switch(config-if)# tunnel use-vrf blue`	(Optional) Uses the configured VRF to look up the tunnel IP destination address.
Step 7	**show interfaces tunnel** *number* **Example:** `switch# show interfaces tunnel 1`	(Optional) Displays the tunnel interface statistics.
Step 8	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves this configuration change.

Example

This example shows how to create a tunnel interface

```
switch# configure terminal
switch(config)# interface tunnel 1
switch(config-if)# tunnel source ethenet 1/2
switch(config-if)# tunnel destination 192.0.2.1
switch(config-if)# copy running-config startup-config
```

Creating an IP-in-IP Tunnel with a Netmask

Creating an IP-in-IP tunnel with a netmask allows you to specify a tunnel source subnet and a tunnel destination subnet, and decap the packet if it matches.

- The IP-in-IP decap-any tunnel receives encapsulated packets from any number of IP-in-IP tunnels.
- With the netmask feature, the switch receives packets from IP addresses which comply with the netmasks.

Notes for the netmask feature:

- Routing protocols are not supported on an IP-in-IP tunnel created with a netmask.
- Encap is not supported with the netmask feature; only decap from a set of sources in the same subnet is supported.

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel** *number*
3. **tunnel mode ipip** [**ip**]
4. **tunnel source** *ip-address* / *mask_length*
5. **tunnel destination** *ip-address* / *mask_length*
6. (Optional) **no shut**
7. **ip address** *ip-prefix*/*length*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface tunnel** *number*
	Example:
	`switch(config)# interface tunnel 5`
	`switch(config-if)#`
		Creates a new tunnel interface.

Step 3	**tunnel mode ipip** [**ip**]	Sets this tunnel mode to ipip. The **ipip** keyword specifies that IP-in-IP encapsulation will be used.
Step 4	**tunnel source** *ip-address* / *mask_length* **Example:** `switch(config-if)# tunnel source`	Configures the source address for this IP tunnel. The source is specified by IP address and length of mask.

	Command or Action	Purpose
	`33.1.1.1 255.255.255.0`	

Step 5	**tunnel destination** *ip-address* / *mask_length*
	Example:
	`switch(config-if)#` **`tunnel destination`**
	`33.1.1.2 255.255.255.0`
		Configures the destination address for this IP tunnel. The
		destination is specified by IP address and length of mask.

Step 6	(Optional) **no shut**	Clears the interface.
Step 7	**ip address** *ip-prefix*/*length* **Example:** `switch(config-if)#` **`ip address 50.1.1.1/24`**	Configures an IP address for this interface.

Example

The following example shows how to create an IP-in-IP tunnel with a netmask.

```
switch(config)# interface tunnel 10
switch(config-if)# tunnel mode ipip
switch(config-if)# tunnel source 33.1.1.2/24
switch(config-if)# tunnel destination
33.1.1.1/24 switch(config-if)# no shut
switch(config-if)# ip address
10.10.10.10/24 switch(config-if)# end
switch# show interface tunnel 10
Tunnel10 is up Admin
    State: up
    Internet address is 10.10.10.10/24
    MTU 1476 bytes, BW 9 Kbit
    Tunnel protocol/transport IPIP/IP
    Tunnel source 33.1.1.2, destination 33.1.1.1
    Transport protocol is in VRF "default"
    Last clearing of "show interface" counters
    never Tx
    0 packets output, 0
    bytes Rx
    0 packets input, 0 bytes
switch# show run interface tunnel 10
!Command: show running-config interface Tunnel10
!Time: Wed Aug 26 13:50:01 2015
version 9.2(1i)
interface Tunnel10
  ip address 10.10.10.10/24
  tunnel mode ipip ip
  tunnel source 33.1.1.2 255.255.255.0
  tunnel destination 33.1.1.1
  255.255.255.0 no shutdown
```

9.6.3 Configuring a Tunnel Interface

You can set a tunnel interface to GRE tunnel mode, ipip mode, or ipip decapsulate-only mode. GRE mode is the default tunnel mode. .

Beginning with Inspur INOS-CN, the **tunnel source direct** and **tunnel mode ipv6ipv6 decapsulate-any** CLI commands are supported on Inspur CN12900 Series switches.

The **tunnel source direct** and **tunnel mode ipv6ipv6 decapsulate-any** CLI commands are supported on Inspur CN12900 Series switches.

The new CLI **tunnel mode ipv6ipv6 decapsulate-any** command is introduced to support IPv6 payload over IPv6 transport (IPv6inIPv6 packets). You can configure IP-in-IP tunnel decapsulation on directly connected IP addresses (for example, physical interface, port-channel, loopback, and SVI) using the new **tunnel source direct** CLI command.

Before you begin

Ensure that you have enabled the tunneling feature.

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel** *number*
3. **tunnel mode** {**gre ip** | **ipip** | {**ip** | **decapsulate-any**}}
4. (Optional) **tunnel mode ipv6ipv6 decapsulate-any**
5. **tunnel source direct**
6. **show interfaces tunnel** *number*
7. **mtu** *value*
8. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface tunnel** *number*
	Example:
	`switch(config)# interface tunnel 1`
	`switch(config-if)#`
		Creates a new tunnel interface.

Step 3	**tunnel mode** {**gre ip** | **ipip** | {**ip** | **decapsulate-any**}}	Sets this tunnel mode to GRE, ipip, or ipip decapsulate-only. The **gre** and **ip** keywords specify that GRE encapsulation over IP will be used. The ipip keyword specifies that IP-in-IP encapsulation will be used. The optional **decapsulate-any** keyword terminates IP-in-IP tunnels at one tunnel interface. This keyword creates a tunnel that will not carry any outbound traffic. However, remote tunnel endpoints can use a tunnel

	Command or Action	Purpose
		configured as their destination.
Step 4	(Optional) **tunnel mode ipv6ipv6 decapsulate-any**	Supports IPv6 payload over IPv6 transport (IPv6inIPv6 packets) . This step is applicable for IPv6 networks only. **Note** This command is not supported on Inspur CN12900 platform switches.
Step 5	**tunnel source direct**	Configures IP-in-IP tunnel decapsulation on any directly connected IP addresses. This option is now supported only when the IP-in-IP decapsulation is used to source route the packets through the network. **Note** This command is not supported on Inspur CN12900 platform switches with the Network Forwarding Engine (NFE).
Step 6	**show interfaces tunnel** *number* **Example:** `switch(config-if)# show interfaces tunnel 1`	(Optional) Displays the tunnel interface statistics.
Step 7	**mtu** *value*	Sets the maximum transmission unit (MTU) of IP packets sent on an interface. The range is from 64 to 9192 units. **Note** When configuring **tunnel mode ipip**, the range is dependent on INOS-CN release: • 64 to 9192 units • 64 to 9196 units
Step 8	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves this configuration change.

Example

This example shows how to create the tunnel interface to GRE:

```
switch# configure terminal
switch(config)# interface tunnel 1
switch(config-if)# tunnel mode gre ip
switch(config-if)# copy running-config startup-config
```

This example shows how to create an ipip tunnel:

```
switch# configure terminal
switch(config)# interface tunnel 1
switch(config-if)# tunnel mode
ipip switch(config-if)# mtu 1400
switch(config-if)# copy running-config startup-config
switch(config-if)# no shut
```

This example shows how to configure IP-in-IP tunnel decapsulation on directly connected IP addresses:

```
switch# configure terminal
switch(config)# interface tunnel 0
switch(config-if)# tunnel mode ipip ip
switch(config-if)# tunnel source direct
switch(config-if)# description IPinIP Decapsulation Interface
switch(config-if)# no shut
```

This example shows how to configure IP-in-IP tunnel decapsulation on IPv6 enabled networks:

```
!
interface tunnel 1
  ipv6 address use-link-local-only             <<< enable IPv6
  tunnel mode ipv6ipv6 decapsulate-any
  tunnel source direct
  description IPinIP Decapsulation Interface
  mtu 1476
  no shutdown
show running-config interface tunnel 1
interface Tunnel1
  tunnel mode ipv6ipv6 decapsulate-any
  tunnel source direct
  no shutdown
show interface tunnel 1
Tunnel1 is up    Admin State: up
MTU 1460 bytes, BW 9 Kbit
Tunnel protocol/transport IPv6/DECAPANY/IPv6
Tunnel source - direct
Transport protocol is in VRF "default"
Tunnel interface is in VRF "default"
Last clearing of "show interface" counters never
Tx    0 packets output, 0 bytes    Rx    0 packets input, 0 bytes
```

9.6.4 Configuring a GRE Tunnel

You can set a tunnel interface to GRE tunnel mode.

Before you begin

Ensure that you have enabled the tunneling feature.

SUMMARY STEPS

1. **configure terminal**

2. **interface tunnel** *number*
3. **tunnel mode gre ip**
4. **show interfaces tunnel** *number*
5. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface tunnel** *number*
	Example:
	`switch(config)# interface tunnel 1`
	`switch(config-if)#`
		Creates a new tunnel interface.

Step 3	**tunnel mode gre ip** **Example:** `switch(config-if)# tunnel mode gre ip`	Sets this tunnel mode to GRE.
Step 4	**show interfaces tunnel** *number* **Example:** `switch(config-if)# show interfaces tunnel 1`	(Optional) Displays the tunnel interface statistics.
Step 5	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves this configuration change.

9.6.5 Configuring a GRE Tunnel

GRE v6 tunnel is used to carry different types of packets over IPv6 transport. GREv6 tunnel carries only IPv4 payload. The tunnel CLIs are enhanced to select IPv6 tunnel and configure v6 tunnel source and destination.

You can set a tunnel interface to GRE tunnel mode, ipip mode, or ipip decapsulate-only mode. GRE mode is the default tunnel mode.

Before you begin

Ensure that you have enabled the tunneling feature.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **interface tunnel** *number*
3. switch(config-if)# **tunnel mode** {**gre ip** | **ipip** {**ip** | **decapsulate-any**}}
4. switch(config-if)# **tunnel use-vrf** *vrf-name*
5. switch(config-if)# **ipv6 address** *IPv6 address*
6. (Optional) switch(config-if)# **show interface tunnel** *number*

7. switch(config-if)# **mtu** *value*
8. (Optional) switch(config-if)# **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface tunnel** *number*	Enters a tunnel interface configuration mode.
Step 3	switch(config-if)# **tunnel mode** {**gre ip** | **ipip** {**ip** | **decapsulate-any**}}	Sets this tunnel mode to GRE, ipip, or ipip decapsulate-only. The **gre** and **ip** keywords specify that GRE encapsulation over IP will be used. The ipip keyword specifies that IP-in-IP encapsulation will be used. The optional **decapsulate-any** keyword terminates IP-in-IP tunnels at one tunnel interface. This keyword creates a tunnel that will not carry any outbound traffic. However, remote tunnel endpoints can use a tunnel configured as their destination.
Step 4	Required: switch(config-if)# **tunnel use-vrf** *vrf-name*	Configures tunnel VRF name.
Step 5	Required: switch(config-if)# **ipv6 address** *IPv6 address*	Configures the IPv6 address. **Note** The tunnel source and the destination addresses are still the same (IPv4 address.)
Step 6	(Optional) switch(config-if)# **show interface tunnel** *number*	Displays the tunnel interface statistics.
Step 7	switch(config-if)# **mtu** *value*	Sets the maximum transmission unit (MTU) of IP packets sent on an interface.
Step 8	(Optional) switch(config-if)# **copy running-config startup-config**	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Example

This example displays how to configure IPv6 Payload over GRE v4 tunnel. Configure the tunnel source, destination, IPv4 address, IPv6 address, and perform the **no shut** command. Once the GREv4 tunnel is created, you can configure v4 or v6 route via the tunnel:

```
switch# configure terminal switch(config)#
interface tunnel 10 switch(config)# tunnel
source 11.1.1.1 switch(config)# tunnel
destination 11.1.1.2 switch(config-if)#
tunnel mode gre ip switch(config-if)#
tunnel use-vrf red switch(config-if)# ip
address 10.1.1.1/24 switch(config-if)#
ipv6 address 2:2::2/64 switch(config-if)#
no shut
switch(config)# ip route 50.1.1.0/24 tunnel 10

switch(config)# ipv6 route 2000:100::/64 tunnel 10
```

This example shows how to view the GRE v4 tunnel interface 10 and display IPv4 and IPv6 routes:

```
switch(config)# show int tunnel
10 Tunnel 10 is up
    Admin State: up
    Internet
    address(es):
    10.1.1.1/24
    1010::1/64
    MTU 1476 bytes, BW 9 Kbit
    Tunnel protocol/transport GRE/IP
    Tunnel source 11.1.1.1, destination 11.1.1.2
    Transport protocol is in VRF "default"
switch#show ipv6 route
…
2000:100::/64, ubest/mbest: 1/0, attached
    *via Tunnel10, [1/0], 00:00:16, static
#show ip route
…
50.1.1.0/24, ubest/mbest: 1/0
    *via Tunnel10, [1/0], 00:03:33, static
```

This example displays how to configure IPv4 payload over GRE v6 tunnel. Configure the tunnel mode as GRE IPv6, tunnel v6 source and destination, IPv4 address, and perform the **no shut** command. Once the GREv6 tunnel is created, you can configure v4 route via the tunnel:

```
switch# configure terminal
switch(config)# interface tunnel 20
switch(config-if)# tunnel mode gre ipv6


switch(config)# tunnel source 1313::1
switch(config)# tunnel destination 1313::2
switch(config-if)# tunnel use-vrf red
switch(config-if)# ip address 20.1.1.1/24
switch(config-if)# no shut
switch(config)# ip route 100.1.1.0/24 tunnel 20
```

This example displays how to view the GREv6 tunnel interface 20:

```
show interface tunnel 20
Tunnel 20 is up
    Admin State: up
    Internet address is 20.1.1.1/24
    MTU 1456 bytes, BW 9 Kbit
    Tunnel protocol/transport GRE/IPv6
    Tunnel source 1313::1, destination 1313::2
    Transport protocol is in VRF "default"
#show ip route
…
100.1.1.0/24, ubest/mbest: 1/0
    *via Tunnel20, [1/0], 00:01:00, static

red10# show interface brief | grep Tunnel
Tunnel10                 up          10.1.1.1/24      GRE/IP            1476
Tunnel20                 up          20.1.1.1/24      GRE/IPv6          1456
```

This example shows how to create an ipip tunnel:

```
switch#      configure      terminal
switch(config)# interface tunnel 1
switch(config-if)#    tunnel    mode
ipip switch(config-if)# mtu 1400
switch(config-if)# copy running-config startup-config
switch(config-if)# no shut
```

9.6.6 Enabling Path MTU Discovery

Use the **tunnel path-mtu discovery** command to enable path MTU discovery on a tunnel.

SUMMARY STEPS

1. **tunnel path-mtu-discovery age-timer** *min*
2. **tunnel path-mtu-discovery min-mtu** *bytes*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**tunnel path-mtu-discovery age-timer** *min* **Example:** switch(config-if)# **tunnel path-mtu-discovery age-timer 25**	Enables Path MTU Discovery (PMTUD) on a tunnel interface. • min—Number of minutes. The range is from 10 to 30. The default is 10.
Step 2	**tunnel path-mtu-discovery min-mtu** *bytes* **Example:** switch(config-if)# **tunnel path-mtu-discovery min-mtu 1500**	Enables Path MTU Discovery (PMTUD) on a tunnel interface. • bytes—Minimum MTU recognized. The range is from 64 to 9192. The default is 64.

9.6.7 Assigning VRF Membership to a Tunnel Interface

You can add a tunnel interface to a VRF.

Before you begin

Ensure that you have enabled the tunneling feature.
Assign the IP address for a tunnel interface after you have configured the interface for a VRF.

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel** *number*
3. **vrf member** *vrf-name*
4. **ip address** *ip-prefix/length*
5. **show vrf** [*vrf-name*] **interface** *interface-type number*
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface tunnel** *number*
	Example:
	`switch(config)# interface tunnel 0`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	**vrf member** *vrf-name* **Example:** `switch(config-if)# vrf member RemoteOfficeVRF`	Adds this interface to a VRF.
Step 4	**ip address** *ip-prefix/length* **Example:** `switch(config-if)# ip address 192.0.2.1/16`	Configures an IP address for this interface. You must do this step after you assign this interface to a VRF.
Step 5	**show vrf** [*vrf-name*] **interface** *interface-type number* **Example:** `switch(config-vrf)# show vrf Enterprise interface tunnel 0`	(Optional) Displays VRF information.
Step 6	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Saves this configuration change.

Example

This example shows how to add a tunnel interface to the VRF:

```
switch# configure terminal
switch(config)# interface tunnel 0
switch(config-if)# vrf member RemoteOfficeVRF
switch(config-if)# ip address 209.0.2.1/16
switch(config-if)# copy running-config startup-config
```

9.7 Verifying the IP Tunnel Configuration

To verify the IP tunnel configuration information, perform one of the following tasks:

Command	Purpose
show interface tunnel *number*	Displays the configuration for the tunnel interface (MTU, protocol, transport, and VRF). Displays input and output packets, bytes, and packet rates.
show interface tunnel *number* **brief**	Displays the operational status, IP address, encapsulation type, and MTU of the tunnel interface.
show interface tunnel *number* **counters**	Displays interface counters of input/output packets. **Note** The byte count displayed with the interface counters include the internal header size.
show interface tunnel *number* **description**	Displays the configured description of the tunnel interface.
show interface tunnel *number* **status**	Displays the operational status of the tunnel interface.
show interface tunnel *number* **status err-disabled**	Displays the error disabled status of the tunnel interface.

9.8 Configuration Examples for IP Tunneling

The following example shows a simple GRE tunnel. Ethernet 1/2 is the tunnel source for router A and the tunnel destination for router B. Ethernet interface 2/1 is the tunnel source for router B and the tunnel destination for router A.

Router A:

```
feature tunnel
interface tunnel 0
ip address 209.165.20.2/8
tunnel source ethernet 1/2
tunnel destination 192.0.2.2
tunnel mode gre ip
tunnel path-mtu-discovery 25 1500
interface ethernet 1/2
ip address 192.0.2.55/8
```

Router B:

```
feature tunnel
interface tunnel 0
ip address 209.165.20.1/8
tunnel source ethernet 2/1
tunnel destination 192.0.2.55
tunnel mode gre ip
```

```
interface ethernet 2/1
ip address 192.0.2.2/8
```

9.9 Related Documents

Related Topic	Document Title
IP Tunnel commands	*Inspur CN12900 Series INOS-CN Interfaces Command Reference*

CHAPTER 10 Configuring Q-in-Q VLAN Tunnels

10.1 Information About Q-in-Q Tunnels

This chapter describes how to configure IEEE 802.1Q-in-Q VLAN tunnels and Layer 2 protocol tunneling on Inspur INOS-CN devices.

A Q-in-Q VLAN tunnel enables a service provider to segregate the traffic of different customers in their infrastructure, while still giving the customer a full range of VLANs for their internal use by adding a second 802.1Q tag to an already tagged frame.

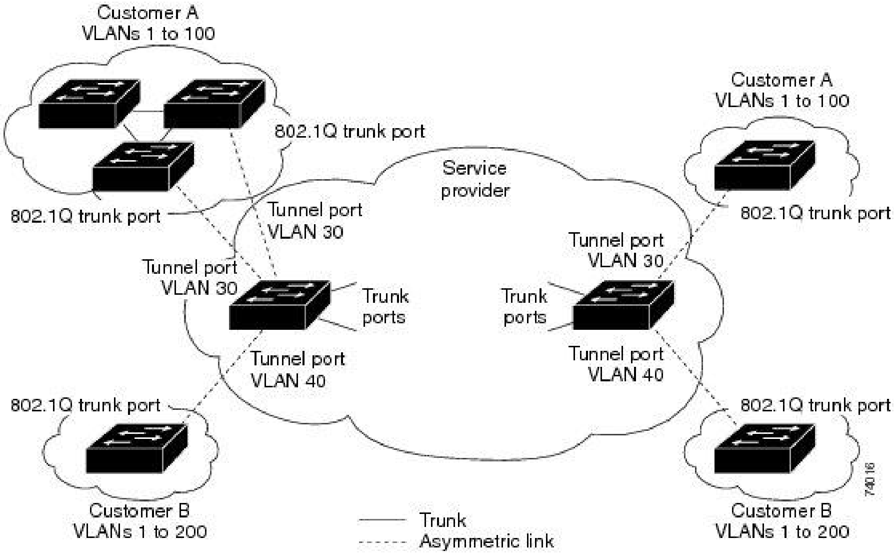
10.1.1 Q-in-Q Tunneling

Business customers of service providers often have specific requirements for VLAN IDs and the number of VLANs to be supported. The VLAN ranges required by different customers in the same service-provider network might overlap, and the traffic of customers through the infrastructure might be mixed. Assigning a unique range of VLAN IDs to each customer would restrict customer configurations and could easily exceed the VLAN limit of 4096 of the 802.1Q specification.

Using the 802.1Q tunneling feature, service providers can use a single VLAN to support customers who have multiple VLANs. Customer VLAN IDs are preserved and the traffic from different customers is segregated within the service-provider infrastructure even when they appear to be on the same VLAN. The 802.1Q tunneling expands the VLAN space by using a VLAN-in-VLAN hierarchy and tagging the tagged packets. A port configured to support 802.1Q tunneling is called a tunnel port. When you configure tunneling, you assign a tunnel port to a VLAN that is dedicated to tunneling. Each customer requires a separate VLAN, but that VLAN supports all of the customer’s VLANs.

Customer traffic that is tagged in the normal way with appropriate VLAN IDs come from an 802.1Q trunk port on the customer device and into a tunnel port on the service-provider edge switch. The link between the customer device and the edge switch is an asymmetric link because one end is configured as an 802.1Q trunk port and the other end is configured as a tunnel port. You assign the tunnel port interface to an access VLAN ID that is unique to each customer. See the figure below.

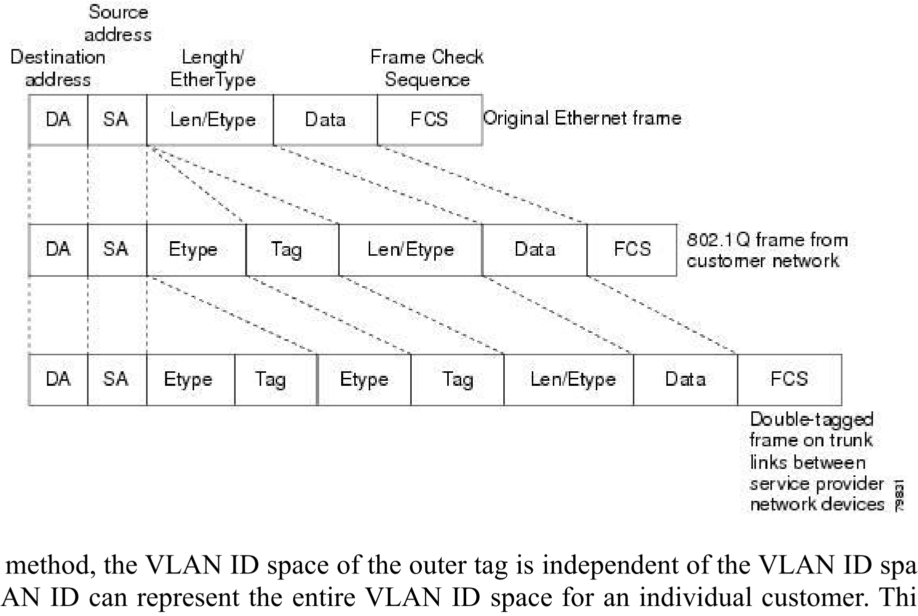
Figure 32： 802.1Q-in-Q Tunnel Ports



Packets that enter the tunnel port on the service-provider edge switch, which are already 802.1Q-tagged with the appropriate VLAN IDs, are encapsulated with another layer of an 802.1Q tag that contains a VLAN ID that is unique to the customer. The original 802.1Q tag from the customer is preserved in the encapsulated packet. Therefore, packets that enter the service-provider infrastructure are double-tagged.

The outer tag contains the customer's access VLAN ID (as assigned by the service provider), and the inner VLAN ID is the VLAN of the incoming traffic (as assigned by the customer). This double tagging is called tag stacking, Double-Q, or Q-in-Q as shown in the figure below.

Figure 33： Untagged, 802.1Q-Tagged, and Double-Tagged Ethernet Frames



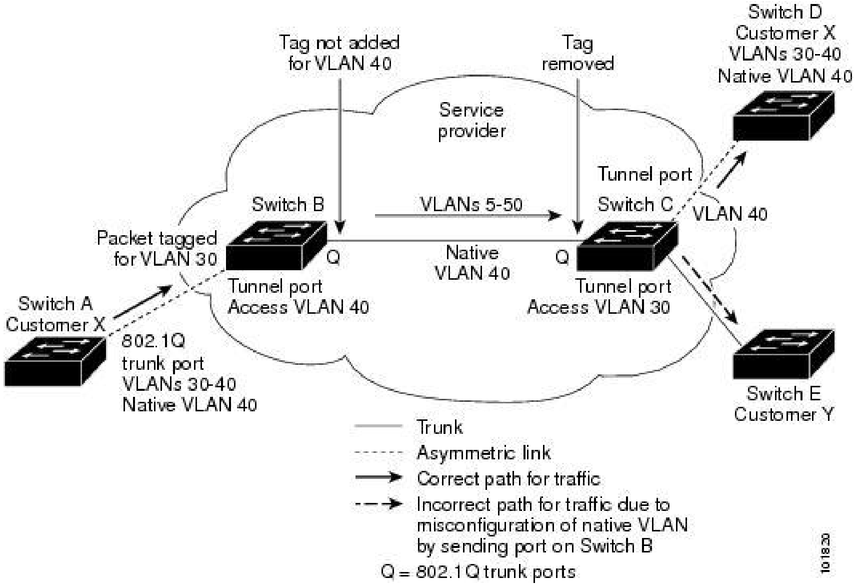
By using this method, the VLAN ID space of the outer tag is independent of the VLAN ID space of the inner tag. A single outer VLAN ID can represent the entire VLAN ID space for an individual customer. This technique allows the customer's Layer 2 network to extend across the service provider network, potentially creating a virtual LAN infrastructure over multiple sites.

10.1.2 Native VLAN Hazard

When configuring 802.1Q tunneling on an edge switch, you must use 802.1Q trunk ports for sending out packets into the service-provider network. However, packets that go through the core of the service-provider network might be carried through 802.1Q trunks, ISL trunks, or nontrunking links. When 802.1Q trunks are used in these core switches, the native VLANs of the 802.1Q trunks must not match any native VLAN of the dot1q-tunnel port on the same switch because traffic on the native VLAN is not tagged on the 802.1Q transmitting trunk port.

In the figure below, VLAN 40 is configured as the native VLAN for the 802.1Q trunk port from Customer X at the ingress edge switch in the service-provider network (Switch B). Switch A of Customer X sends a tagged packet on VLAN 30 to the ingress tunnel port of Switch B in the service-provider network that belongs to access VLAN 40. Because the access VLAN of the tunnel port (VLAN 40) is the same as the native VLAN of the edge-switch trunk port (VLAN 40), the 802.1Q tag is not added to tagged packets that are received from the tunnel port. The packet carries only the VLAN 30 tag through the service-provider network to the trunk port of the egress-edge switch (Switch C) and is misdirected through the egress switch tunnel port to Customer Y.

Figure 34： Native VLAN Hazard


Tag not added for VLAN 40
Tag removed
Switch D Customer X VLANs 30-40 Native VLAN 40
Service provider
Tunnel port
Switch B
VLANs 5-50
Switch C
VLAN 40
Packet tagged for VLAN 30
Q
Native VLAN 40
Q
Tunnel port Access VLAN 40
Tunnel port Access VLAN 30
Switch A Customer X
802.1Q trunk port VLANs 30-40 Native VLAN 40
Switch E Customer Y
Trunk
Asymmetric link
Correct path for traffic
Incorrect path for traffic due to misconfiguration of native VLAN by sending port on Switch B
Q = 802.1Q trunk ports
101820

These are a couple ways to solve the native VLAN problem:

- Configure the edge switch so that all packets going out an 802.1Q trunk, including the native VLAN, are tagged by using the vlan dot1q tag native command. If the switch is configured to tag native VLAN packets on all 802.1Q trunks, the switch accepts untagged packets but sends only tagged packets.
- Ensure that the native VLAN ID on the edge switch trunk port is not within the customer VLAN range. For example, if the trunk port carries traffic of VLANs 100 to 200, assign the native VLAN a number outside that range.

10.1.3 Information About Layer 2 Protocol Tunneling

Customers at different sites connected across a service-provider network need to run various Layer 2 protocols to scale their topology to include all remote sites, as well as the local sites. The Spanning Tree Protocol (STP) must run properly, and every VLAN should build a proper spanning tree that includes the local site and all remote sites across the service-provider infrastructure.

You can configure the switch to allow multi-tagged BPDUs on a tunnel port. If you enable the **l2protocol tunnel allow-double-tag** command, when a multi-tagged customer BPDU enters the tunnel port, the original 802.1Q tags from the customer traffic is preserved and an outer VLAN tag (customer's access VLAN ID, as assigned by the service-provider) is added in the encapsulated packet. Therefore, BPDU packets that enter the service-provider infrastructure are multi tagged. When the BPDUs leave the service-provider network, the outer tag is removed and the original multi-tagged BPDU is sent to the customer network.

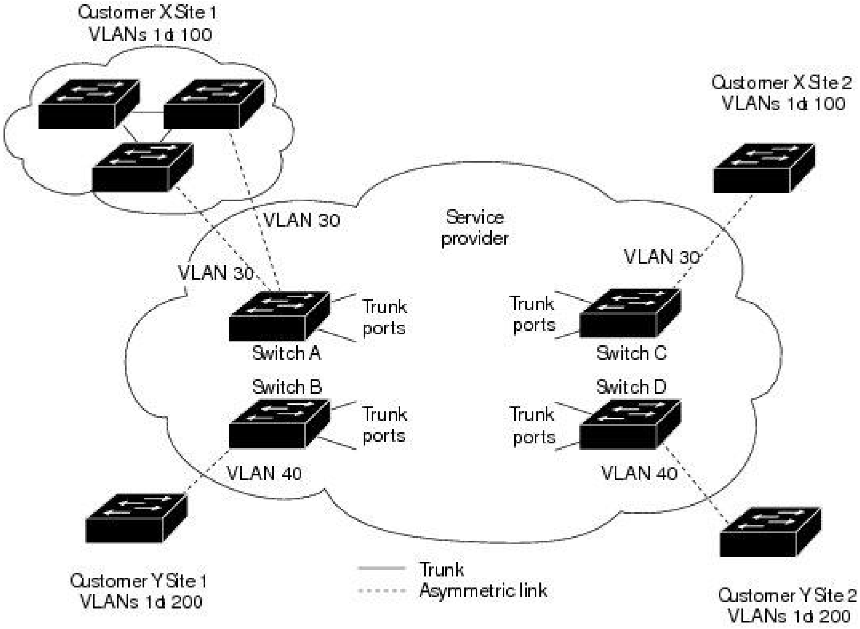
When protocol tunneling is enabled, edge switches on the inbound side of the service-provider infrastructure encapsulate Layer 2 protocol packets with a special MAC address and send them across the service-provider network. Core switches in the network do not process these packets, but forward them as normal packets.

Bridge protocol data units (BPDUs) for CDP, STP, or VTP cross the service-provider infrastructure and are delivered to customer switches on the outbound side of the service-provider network. Identical packets are received by all customer ports on the same VLANs.

If protocol tunneling is not enabled on 802.1Q tunneling ports, remote switches at the receiving end of the service-provider network do not receive the BPDUs and cannot properly run STP, CDP, 802.1X, and VTP. When protocol tunneling is enabled, Layer 2 protocols within each customer's network are totally separate from those running within the service-provider network. Customer switches on different sites that send traffic through the service- provider network with 802.1Q tunneling achieve complete knowledge of the customer's VLAN.

For example, in the figure below, Customer X has four switches in the same VLAN that are connected through the service-provider network. If the network does not tunnel BPDUs, switches on the far ends of the network cannot properly run the STP, CDP, 802.1X, and VTP protocols.

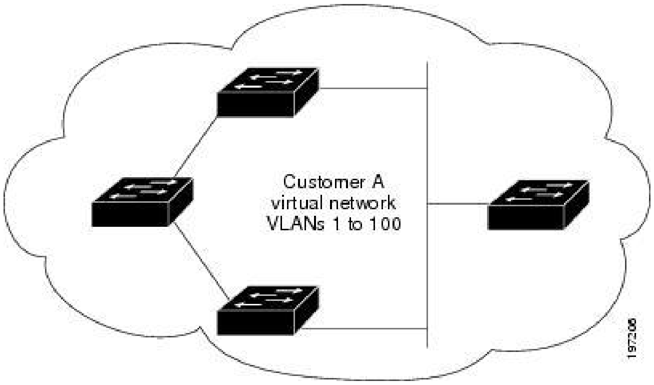
Figure 35： Layer 2 Protocol Tunneling



In the preceding example, STP for a VLAN on a switch in Customer X, Site 1 will build a spanning tree on the switches at that site without considering convergence parameters based on Customer X's switch in Site 2.

The figure below shows the resulting topology on the customer's network when BPDU tunneling is not enabled.

Figure 36 : Virtual Network Topology Without BPDU Tunneling



10.2 Licensing Requirements for Interfaces

Interfaces do not require a license.

10.3 Guidelines and Limitations

Q-in-Q tunnels and Layer 2 tunneling have the following configuration guidelines and limitations:

- Mapping multiple VLANs is supported.
- Only multi-tagged CDP and STP BPDUs are supported.
- The inner-most tag must always be 0x8100.
- Multiple selective Q-in-Q tags are not supported. That is, Q-in-Q does not support multiple SP tags on a single interface.
- **show** commands with the **internal** keyword are not supported.
- Switches in the service-provider network must be configured to handle the increase in MTU size due to Q-in-Q tagging.
- MAC address learning for Q-in-Q tagged packets is based on the outer VLAN (Service Provider VLAN) tag. Packet forwarding issues might occur in deployments where a single MAC address is used across multiple inner (customer) VLANs.
- Layer 3 and higher parameters cannot be identified in tunnel traffic (for example, Layer 3 destination and source addresses). Tunneled traffic cannot be routed.
- Inspur CN12900 Series devices can provide only MAC-layer ACL/QoS for tunnel traffic (VLAN IDs and src/dest MAC addresses).
- You should use MAC address-based frame distribution.
- Asymmetrical links do not support the Dynamic Trunking Protocol (DTP) because only one port on the link is a trunk. You must configure the 802.1Q trunk port on an asymmetrical link to trunk unconditionally.
- You cannot configure the 802.1Q tunneling feature on ports that are configured to support private VLANs. Private VLAN are not required in these deployments.
- You must disable IGMP snooping on the tunnel VLANs.
- Control Plane Policing (CoPP) is not supported.
- You should enter the vlan dot1Q tag native command to maintain the tagging on the native VLAN and drop untagged traffic. This command prevents native VLAN misconfigurations.
- You must manually configure the 802.1Q interfaces to be edge ports.
- IGMP snooping is not supported on the inner VLAN.
- Q-in-Q tunneling is not supported on Application Spine Engine 2 (ASE2) and Application Spine Engine 3 (ASE3) based Inspur switches.

- Layer 2 protocol tunnelling with Link Aggregation Control Protocol (LACP) is not supported.
- Q-in-Q tagging is not supported.
- Layer 2 protocol tunneling is not supported on Inspur CN12900 Series switches with CN129-X636C-R, CN129-X636Q-R, CN129-X6136YC-R line cards.
- Inspur CN12900 Series switches with CN129-X636C-R, CN129-X636Q-R, CN129-X6136YC-R line cards, Q-in-Q is supported only on port or port-channel Layer 2 Access VLAN Edge devices.
- If the command **l2potocol tunnel stp** is configured on a tunnel interface, the VLAN that you configure on the service provider must be different from that of the customer network.

10.4 Configuring Q-in-Q Tunnels and Layer 2 Protocol Tunneling

10.4.1 Creating a 802.1Q Tunnel Port

You create the dot1q-tunnel port using the **switchport mode** command.

The following CLI is mandatory only on LSE based Inspur CN12900 Series switches. For seamless packet forwarding and preservation of all VLAN tags on pure transit boxes in the SP cloud that have no Q-in-Q encapsulation or decapsulation requirement, configure the CLI command **system dot1q-tunnel transit**. To remove the CLI, use **no system dot1q-tunnel transit** CLI command.

The caveats with the CLI that is executed on the switches are:

- L2 frames that egress out of the trunk ports are tagged even on the native VLAN on the port.
- Any other tunneling mechanism, for example, VXLAN and MPLS does not work with the CLI configured.

Before you begin

You must first configure the interface as a switchport.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **interface ethernet** *slot/port*
3. switch(config-if)# **switchport**
4. switch(config-if)# **switchport mode dot1q-tunnel**
5. (Optional) switch(config-if)# **no switchport mode dot1q-tunnel**
6. switch(config-if)# **exit**
7. (Optional) switch(config)# **show dot1q-tunnel** [**interface** *if-range*]
8. (Optional) switch(config)# **no shutdown**
9. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface ethernet** *slot/port*	Specifies an interface to configure, and enters interface configuration mode.
Step 3	switch(config-if)# **switchport**	Sets the interface as a Layer 2 switching port.
Step 4	switch(config-if)# **switchport mode dot1q-tunnel**	Creates a 802.1Q tunnel on the port. The port will go down

	Command or Action	Purpose
		and reinitialize (port flap) when the interface mode is changed. BPDU filtering is enabled and CDP is disabled on tunnel interfaces.
Step 5	(Optional) switch(config-if)# **no switchport mode dot1q-tunnel**	Disables the 802.1Q tunnel on the port.
Step 6	switch(config-if)# **exit**	Exits configuration mode.
Step 7	(Optional) switch(config)# **show dot1q-tunnel** [**interface** *if-range*]	Displays all ports that are in dot1q-tunnel mode. Optionally, you can specify an interface or range of interfaces to display.
Step 8	(Optional) switch(config)# **no shutdown**	Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.
Step 9	(Optional) switch(config)# **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to create an 802.1Q tunnel port:

```
switch# configure terminal
switch(config)# interface ethernet 7/1
switch(config-if)# switchport
switch(config-if)# switchport mode dot1q-tunnel
switch(config-if)# exit
switch(config)# exit
switch# show dot1q-tunnel
```

10.4.2 Configuring VLAN Mapping for Selective Q-in-Q on a 802.1Q Tunnel Port

To configure VLAN mapping for selective Q-in-Q on a 802.1Q tunnel port, complete the following steps.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **interface** *interface-id*
3. switch(config-if)# **switchport mode dot1q-tunnel**
4. switch(config-if)# **switchport vlan mapping** *vlan-id-range* **dot1q-tunnel** *outer vlan-id*

5. switch(config-if)# **exit**
6. switch# **show interfaces** *interface-id* **vlan mapping**
7. switch# **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface** *interface-id*	Enters interface configuration mode for the interface connected to the service provider network. You can enter a physical interface or an EtherChannel port channel.
Step 3	switch(config-if)# **switchport mode dot1q-tunnel**	Configure the interface as a tunnel port.
Step 4	switch(config-if)# **switchport vlan mapping** *vlan-id-range* **dot1q-tunnel** *outer vlan-id*	Enters the VLAN IDs to be mapped: • vlan-id-range—The customer VLAN ID range (C-VLAN) entering the switch from the customer network. The range is from 1 to 4094. You can enter a string of VLAN-IDs. • outer vlan-id—Enter the outer VLAN ID (S-VLAN) of the service provider network. The range is from 1 to 4094.
Step 5	switch(config-if)# **exit**	Exits the configuration mode.
Step 6	switch# **show interfaces** *interface-id* **vlan mapping**	Verifies the configuration.
Step 7	switch# **copy running-config startup-config**	(Optional) Saves your entries in the configuration file.

Use the **no switchport vlan mapping** *vlan-id-range* **dot1q-tunnel** *outer vlan-id* command to remove the VLAN mapping configuration.

The following example shows how to configure selective Q-in-Q mapping on the port so that traffic with a C-VL

```
switch(config)# interface gigabiethernet0/1
switch(config-if)# switchport vlan mapping 1-5 dot1q-tunnel 100
 Switch(config-if)# exit
```

10.4.3 Changing the EtherType for Q-in-Q

You can change the 802.1Q EtherType value to be used for Q-in-Q encapsulation.

Caution	The EtherType value you set affects all the tagged packets that go out on the interface (not just Q-in-Q packets).

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **interface ethernet** *slot/port*
3. switch(config-if)# **switchport**
4. switch(config-if)# **switchport dot1q ethertype** *value*
5. (Optional) switch(config-if)# **no switchport dot1q ethertype**
6. switch(config-if)# **exit**
7. (Optional) switch(config)# **no shutdown**
8. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface ethernet** *slot/port*	Specifies an interface to configure, and enters interface configuration mode.
Step 3	switch(config-if)# **switchport**	Sets the interface as a Layer 2 switching port.
Step 4	switch(config-if)# **switchport dot1q ethertype** *value*	Sets the EtherType for the Q-in-Q tunnel on the port.
Step 5	(Optional) switch(config-if)# **no switchport dot1q ethertype**	Resets the EtherType on the port to the default value of 0x8100.
Step 6	switch(config-if)# **exit**	Exits configuration mode.
Step 7	(Optional) switch(config)# **no shutdown**	Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.
Step 8	(Optional) switch(config)# **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to create an 802.1Q tunnel port:

```
switch# configure terminal
switch(config)# interface ethernet 7/1
           switch(config-if)# switchport
           switch(config-if)# switchport dot1q ethertype 0x9100
           switch(config-if)# exit
           switch(config)# exit
           switch# show dot1q-tunnel
```

10.4.4 Configuring Custom EtherType or Tag Protocol Identifier (TPID)

The switch uses a default ethertype of 0x8100 for 802.1Q and Q-in-Q encapsulations. You can configure EtherTypes 0x9100, 0x9200 and 0x88a8 on a per port basis by enabling the **dot1q ethertype** command on the switchport interface. You can configure a custom EtherType field value on a port to support network devices that do not use the standard 0x8100 EtherType field value on 802.1Q-tagged or 802.1p-tagged frames.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **interface ethernet** *slot/port*
3. switch(config-if)# **switchport**
4. switch(config-if)# **switchport mode**
5. switch(config-if)# **switchport dot1q ethertype** *value*
6. (Optional) switch(config-if)# **switchport access vlan** *value*
7. switch(config-if)# **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface ethernet** *slot/port*	Specifies an interface to configure, and enters interface configuration mode.
Step 3	switch(config-if)# **switchport**	Sets the interface as a Layer 2 switching port.
Step 4	switch(config-if)# **switchport mode**	Sets the interface as a Layer 2 switching port mode.
Step 5	switch(config-if)# **switchport dot1q ethertype** *value*	Sets the EtherType for the Q-in-Q tunnel on the port.
Step 6	(Optional) switch(config-if)# **switchport access vlan** *value*	Sets the interface access VLAN.
Step 7	switch(config-if)# **exit**	Exits configuration mode.

Example

This example shows how to configure custom ethertype on an 802.1Q tunnel port:

```
switch# configure terminal
switch(config)# interface ethernet
1/1 switch(config-if)# switchport
switch(config-if)# switchport mode dot1q-tunnel
switch(config-if)# switchport dot1q ethertype
0x9100 switch(config-if)# switchport access vlan 30
switch(config-if)# exit
switch(config)# exit
```

10.4.5 Enabling the Layer 2 Protocol Tunnel

You can enable protocol tunneling on the 802.1Q tunnel port.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **interface ethernet** *slot/port*
3. switch(config-if)# **switchport**
4. switch(config-if)# **switchport mode dot1q-tunnel**
5. switch(config-if)# **l2protocol tunnel** [**cdp** | **stp** | **lacp** | **lldp** |**vtp**]
6. (Optional) switch(config-if)# **no l2protocol tunnel** [**cdp** | **stp** | **lacp** | **lldp** |**vtp**]
7. switch(config-if)# **exit**
8. (Optional) switch(config)# **no shutdown**
9. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface ethernet** *slot/port*	Specifies an interface to configure, and enters interface configuration mode.
Step 3	switch(config-if)# **switchport**	Sets the interface as a Layer 2 switching port.
Step 4	switch(config-if)# **switchport mode dot1q-tunnel**	Creates a 802.1Q tunnel on the port. The port will go down and reinitialize (port flap) when the interface mode is changed. BPDU filtering is enabled and CDP is disabled on tunnel interfaces.
Step 5	switch(config-if)# **l2protocol tunnel** [**cdp** | **stp** | **lacp** | **lldp** |**vtp**]	Enables Layer 2 protocol tunneling. Optionally, you can enable CDP, STP, LACP, LLDP, or VTP tunneling.
Step 6	(Optional) switch(config-if)# **no l2protocol tunnel** [**cdp** | **stp** | **lacp** | **lldp** |**vtp**]	Disables protocol tunneling.
Step 7	switch(config-if)# **exit**	Exits configuration mode.
Step 8	(Optional) switch(config)# **no shutdown**	Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command

	Command or Action	Purpose
		allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.
Step 9	(Optional) switch(config)# **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to enable protocol tunneling on an 802.1Q tunnel port:

```
switch# configure terminal
switch(config)# interface ethernet 7/1
switch(config-if)# switchport
switch(config-if)# switchport mode dot1q-tunnel
switch(config-if)# l2protocol tunnel stp
switch(config-if)# exit
switch(config)# exit
```

10.4.6 Configuring Global CoS for L2 Protocol Tunnel Ports

You can specify a Class of Service (CoS) value globally so that ingress BPDUs on the tunnel ports are encapsulated with the specified class.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **l2protocol tunnel cos** *value*
3. (Optional) switch(config)# **no l2protocol tunnel cos**
4. switch(config)# **exit**
5. (Optional) switch# **no shutdown**
6. (Optional) switch# **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **l2protocol tunnel cos** *value*	Specifies a global CoS value on all Layer 2 protocol tunneling ports. The default cos-value is 5.
Step 3	(Optional) switch(config)# **no l2protocol tunnel cos**	Sets the global CoS value to default.
Step 4	switch(config)# **exit**	Exits configuration mode.
Step 5	(Optional) switch# **no shutdown**	Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command

		allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.
Step 6	(Optional) switch# **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to specify a global CoS value for the purpose of Layer 2 protocol tunneling:

```
switch# configure terminal
switch(config)# l2protocol tunnel cos 6
switch(config)# exit
```

10.4.7 Configuring Thresholds for Layer 2 Protocol Tunnel Ports

You can specify the port drop and shutdown value for a Layer 2 protocol tunneling port.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **interface ethernet** *slot/port*
3. switch(config-if)# **switchport**
4. switch(config-if)# **switchport mode dot1q-tunnel**
5. switch(config-if)# **l2protocol tunnel drop-threshold** [**cdp** | **stp** | **vtp**] *packets-per-sec*
6. (Optional) switch(config-if)# **no l2protocol tunnel drop-threshold** [**cdp** | **stp** | **vtp**]
7. switch(config-if)# **l2protocol tunnel shutdown-threshold** [**cdp** | **stp** | **vtp**] *packets-per-sec*
8. (Optional) switch(config-if)# **no l2protocol tunnel shutdown-threshold** [**cdp** | **stp** | **vtp**]
9. switch(config-if)# **exit**
10. (Optional) switch(config)# **no shutdown**
11. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface ethernet** *slot/port*	Specifies an interface to configure, and enters interface configuration mode.
Step 3	switch(config-if)# **switchport**	Sets the interface as a Layer 2 switching port.
Step 4	switch(config-if)# **switchport mode dot1q-tunnel**	Creates a 802.1Q tunnel on the port.
Step 5	switch(config-if)# **l2protocol tunnel drop-threshold** [**cdp** | **stp** | **vtp**] *packets-per-sec*	Specifies the maximum number of packets that can be processed on an interface before being dropped. Optionally,

	Command or Action	Purpose
		you can specify CDP, STP, or VTP. Valid values for the packets are from 1 to 4096.
Step 6	(Optional) switch(config-if)# **no l2protocol tunnel drop-threshold** [**cdp** | **stp** | **vtp**]	Resets the threshold values to 0 and disables the drop threshold.
Step 7	switch(config-if)# **l2protocol tunnel shutdown-threshold** [**cdp** | **stp** | **vtp**] *packets-per-sec*	Specifies the maximum number of packets that can be processed on an interface. When the number of packets is exceeded, the port is put in error-disabled state. Optionally, you can specify CDP, STP, or VTP. Valid values for the packets is from 1 to 4096.
Step 8	(Optional) switch(config-if)# **no l2protocol tunnel shutdown-threshold** [**cdp** | **stp** | **vtp**]	Resets the threshold values to 0 and disables the shutdown threshold.
Step 9	switch(config-if)# **exit**	Exits configuration mode.
Step 10	(Optional) switch(config)# **no shutdown**	Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.
Step 11	(Optional) switch(config)# **copy running-config startup-config**	Copies the running configuration to the startup configuration.

10.5 Configuring Q-in-Q Double Tagging

Enable multi-tagging for STP and CDP BPDUs.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface*
3. **switchport**
4. **switchport mode dot1q-tunnel**
5. **l2protocol tunnel [cdp | stp]**
6. (Optional) **no l2protocol tunnel [cdp | stp]**
7. **l2protocol tunnel allow-double-tag**

8. (Optional) **no l2protocol tunnel allow-double-tag**
9. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**interface** *interface* **Example:** `switch(config)# interface ethernet 7/1`	Specifies the interface that you are configuring.
Step 3	**switchport** **Example:** `switch(config-if)# switchport`	Sets the interface as a Layer 2 switching port.
Step 4	**switchport mode dot1q-tunnel** **Example:** `switch(config-if)# switchport mode dot1q-tunnel`	Creates an 802.1Q tunnel on the port. The port goes down and reinitializes (port flap) when the interface mode is changed. BPDU filtering is enabled and CDP is disabled on tunnel interfaces.
Step 5	**l2protocol tunnel [cdp | stp]** **Example:** `switch(config-if)# l2protocol tunnel cdp`	Enables Layer 2 protocol tunneling. Optionally, you can enable CDP or STP.
Step 6	(Optional) **no l2protocol tunnel [cdp | stp]** **Example:** `switch(config-if)# no l2protocol tunnel stp`	Disables protocol tunneling.
Step 7	**l2protocol tunnel allow-double-tag** **Example:** `switch(config-if)# l2protocol tunnel allow-double-tag`	Enables multi-tagging for STP and CDP BPDUs on the interface.
Step 8	(Optional) **no l2protocol tunnel allow-double-tag** **Example:** `switch(config-if)# no l2protocol tunnel allow-double-tag`	Disables multi-tagging for STP and CDP BPDUs on the interface.
Step 9	**exit**	Exits configuration mode.

Example: `switch(config-if)# exit`	

Example

This example shows how to enable multi-tagging for STP and CDP BPDUs:

```
switch# configure terminal
switch(config)# interface ethernet
7/1 switch(config-if)# switchport
switch(config-if)# switchport mode dot1q-tunnel
switch(config-if)# l2protocol tunnel cdp
switch(config-if)# l2protocol tunnel stp
switch(config-if)# l2protocol tunnel allow-double-tag
switch(config-if)# exit
switch(config)#
exit switch#
```

10.6 Verifying the Q-in-Q Configuration

Command	Purpose
clear l2protocol tunnel counters [**interface** *if-range*]	Clears all the statistics counters. If no interfaces are specified, the Layer 2 protocol tunnel statistics are cleared for all interfaces.
show dot1q-tunnel [**interface** *if-range*]	Displays a range of interfaces or all interfaces that are in dot1q-tunnel mode.
show l2protocol tunnel [**interface** *if-range* | **vlan** *vlan-id*]	Displays Layer 2 protocol tunnel information for a range of interfaces, for all dot1q-tunnel interfaces that are part of a specified VLAN or all interfaces.
show l2protocol tunnel summary	Displays a summary of all ports that have Layer 2 protocol tunnel configurations.
show running-config l2pt	Displays the current Layer 2 protocol tunnel running configuration.

10.7 Configuration Examples for Q-in-Q and Layer 2 Protocol Tunneling

This example shows a service provider switch that is configured to process Q-in-Q for traffic coming in on Ethernet 7/1. A Layer 2 protocol tunnel is enabled for STP BPDUs. The customer is allocated VLAN 10 (outer VLAN tag).

```
switch# configure terminal
Enter configuration commands, one per line. End with
CNTL/Z. switch(config)# vlan 10
switch(config-vlan)# no shutdown
switch(config-vlan)# no ip igmp
snooping switch(config-vlan)# exit
switch(config)# interface ethernet 7/1
switch(config-if)# switchport
switch(config-if)# switchport mode dot1q-
tunnel switch(config-if)# switchport access
vlan 10 switch(config-if)# spanning-tree port
type edge switch(config-if)# l2protocol tunnel
stp switch(config-if)# no shutdown
switch(config-if)# exit
switch(config)#
exit switch#
```

CHAPTER 11 Configuring Static and Dynamic NAT Translation

11.1 Network Address Translation Overview

Network Address Translation (NAT) enables private IP internetworks that use nonregistered IP addresses to connect to the Internet. NAT operates on a device, usually connecting two networks, and translates private (not globally unique) IP addresses in the internal network into legal IP addresses before packets are forwarded to another network. You can configure NAT to advertise only one IP address for the entire network to the outside world. This ability provides additional security, effectively hiding the entire internal network behind one IP address.

A device configured with NAT has at least one interface to the inside network and one to the outside network. In a typical environment, NAT is configured at the exit router between a stub domain and a backbone. When a packet leaves the domain, NAT translates the locally significant source IP address into a globally unique IP address. When a packet enters the domain, NAT translates the globally unique destination IP address into a local IP address. If more than one exit point exists, NAT configured at each point must have the same translation table.

NAT is described in RFC 1631.

11.2 Information About Static NAT

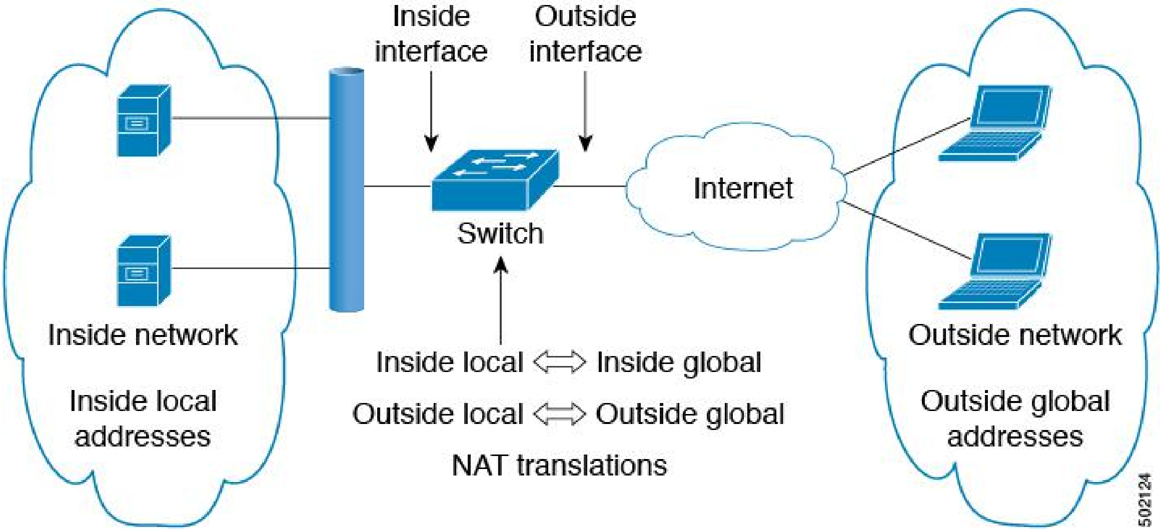
Static Network Address Translation (NAT) allows the user to configure one-to-one translations of the inside local addresses to the outside global addresses. It allows both IP addresses and port number translations from the inside to the outside traffic and the outside to the inside traffic. The Inspur device supports Hitless NAT, which means that you can add or remove a NAT translation in the NAT configuration without affecting the existing NAT traffic flows.

Static NAT creates a fixed translation of private addresses to public addresses. Because static NAT assigns addresses on a one-to-one basis, you need an equal number of public addresses as private addresses. Because the public address is the same for each consecutive connection with static NAT, and a persistent translation rule exists, static NAT enables hosts on the destination network to initiate traffic to a translated host if an access list exists that allows it .

With dynamic NAT and Port Address Translation (PAT), each host uses a different address or port for each subsequent translation. The main difference between dynamic NAT and static NAT is that static NAT allows a remote host to initiate a connection to a translated host if an access list exists that allows it, while dynamic NAT does not.

The figure shows a typical static NAT scenario. The translation is always active so both translated and remote hosts can originate connections, and the mapped address is statically assigned by the **static** command.

Figure 37： Static NAT



These are key terms to help you understand static NAT:

- NAT inside interface—The Layer 3 interface that faces the private network.
- NAT outside interface—The Layer 3 interface that faces the public network.
- Local address—Any address that appears on the inside (private) portion of the network.
- Global address—Any address that appears on the outside (public) portion of the network.
- Legitimate IP address—An address that is assigned by the Network Information Center (NIC) or service provider.
- Inside local address—The IP address assigned to a host on the inside network. This address does not need to be a legitimate IP address.
- Outside local address—The IP address of an outside host as it appears to the inside network. It does not have to be a legitimate address, because it is allocated from an address space that can be routed on the inside network.
- Inside global address—A legitimate IP address that represents one or more inside local IP addresses to the outside world.
- Outside global address—The IP address that the host owner assigns to a host on the outside network. The address is a legitimate address that is allocated from an address or network space that can be routed.

11.3 Dynamic NAT Overview

Dynamic Network Address Translation (NAT) translates a group of real IP addresses into mapped IP addresses that are routable on a destination network. Dynamic NAT establishes a one-to-one mapping between unregistered and registered IP addresses; however, the mapping can vary depending on the registered IP address that is available at the time of communication.

A dynamic NAT configuration automatically creates a firewall between your internal network and outside networks or the Internet. Dynamic NAT allows only connections that originate inside the stub domain—a device on an external network cannot connect to devices in your network, unless your device has initiated the contact.

Dynamic NAT translations do not exist in the NAT translation table until a device receives traffic that requires translation. Dynamic translations are cleared or timed out when not in use to make space for new entries. Usually, NAT translation entries are cleared when the ternary content addressable memory (TCAM) entries are limited. The default minimum timeout for dynamic NAT translations is 30 minutes.

Dynamic NAT supports Port Address Translation (PAT) and access control lists (ACLs). PAT, also known as overloading, is a form of dynamic NAT that maps multiple unregistered IP addresses to a single registered IP address by using different ports. Your NAT configuration can have multiple dynamic NAT translations with same or different ACLs. However, for a given ACL, only one interface can be specified.

11.4 Timeout Mechanisms

The following NAT translation timeout timers are supported on the switch:

- **timeout**—Timeout value for dynamic NAT translations.

The timeout value ranges from 60 seconds to 172800 seconds, including the sampling-timeout.

- **sampling-timeout**—Time after which the device checks for dynamic translation activity. The timeout value ranges from 900 seconds to 172800 seconds.

The **udp-timeout** and the **timeout** value timers are triggered after the timeout configured for the **ip nat translation sampling-timeout** command expires.

11.5 NAT Inside and Outside Addresses

NAT inside refers to networks owned by an organization that must be translated. When NAT is configured, hosts within this network will have addresses in one space (known as the local address space) that will appear to those outside the network as being in another space (known as the global address space).

Similarly, NAT outside refers to those networks to which the stub network connects. They are not generally under the control of the organization. Hosts in outside networks can be subject to translation and can have local and global addresses.

NAT uses the following definitions:

- Local address—A local IP address that appears on the inside of a network.
- Global address—A global IP address that appears on the outside of a network.
- Inside local address—The IP address that is assigned to a host on the inside network. The address is probably not a legitimate IP address assigned by the Internet Network Information Center (InterNIC) or a service provider.
- Inside global address—A legitimate IP address (assigned by InterNIC or a service provider) that represents one or more inside local IP addresses to the outside world.
- Outside local address—The IP address of an outside host as it appears to the inside network. The address is not necessarily legitimate; it was allocated from the address space that is routable on the inside.
- Outside global address—The IP address that is assigned to a host on the outside network by the owner of the host. The address was allocated from a globally routable address or a network space.

11.6 Pool Support for Dynamic NAT

Inspur INOS-CN provides pool support for dynamic NAT. Dynamic NAT allows the configuration of a pool of global addresses that can be used to dynamically allocate a global address from the pool for every new translation. The addresses are returned to the pool after the session ages out or is closed. This allows for a more efficient use of addresses based on requirements.

Support for PAT includes the use of the global address pool. This further optimizes IP address utilization. PAT exhausts one IP address at a time with the use of port numbers. If no port is available from the appropriate group and more than one IP address is configured, PAT moves to the next IP address and tries to allocate the original source port again. This process continues until PAT runs out of available ports and IP addresses.

With dynamic NAT and PAT, each host uses a different address or port for each subsequent translation. The main difference between dynamic NAT and static NAT is that static NAT allows a remote host to initiate a connection to a translated host if an access list exists that allows it, while dynamic NAT does not.

11.7 Static and Dynamic Twice NAT Overview

When both the source IP address and the destination IP address are translated as a single packet that goes through a Network Address Translation (NAT) device, it is referred to as twice NAT. Twice NAT is supported for static and dynamic translations.

Twice NAT allows you to configure two NAT translations (one inside and one outside) as part of a group of translations. These translations can be applied to a single packet as it flows through a NAT device. When you add two translations as part of a group, both the individual translations and the combined translation take effect.

A NAT inside translation modifies the source IP address and port number when a packet flows from inside to outside. It modifies the destination IP address and port number when the packet returns from outside to inside. NAT outside translation modifies the source IP address and port number when the packet flows from outside to inside, and it modifies the destination IP address and port number when the packet returns from inside to outside.

Without twice NAT, only one of the translation rules is applied on a packet, either the source IP address and port number or the destination IP address and port number.

Static NAT translations that belong to the same group are considered for twice NAT configuration. If a static configuration does not have a configured group ID, the twice NAT configuration will not work. All inside and outside NAT translations that belong to a single group that is identified by the group ID are paired to form twice NAT translations.

Dynamic twice NAT translations dynamically select the source IP address and port number information from pre-defined **ip nat pool** or **interface overload** configurations. Packet filtration is done by configuring ACLs, and traffic must originate from the dynamic NAT translation rule direction such that source translation is done by using dynamic NAT rules.

Dynamic twice NAT allows you to configure two NAT translations (one inside and one outside) as part of a group of translations. One translation must be dynamic and other translation must be static. When these two translations are part of a group of translations, both the translations can be applied on a single packet as it goes through the NAT device either from inside to outside or from outside to inside.

11.8 VRF Aware NAT

The VRF aware NAT feature enables a switch to understand an address space in a VRF (virtual routing and forwarding instances) and to translate the packet. This allows the NAT feature to translate traffic in an overlapping address space that is used between two VRFs.

- Traffic flowing from one non-default-vrf to another non-default-vrf is not translated. (For example, vrfA to vrfB.)
- For traffic flowing from a VRF to a global-VRF, a nat-outside configuration is not supported on a non-default VRF interface.
- VRF aware NAT is supported by static and dynamic NAT configurations.
- When traffic is configured to flow from a non-default VRF (inside) to a default VRF (outside), the **match-in-vrf** option of the **ip nat** command cannot be specified.
- When traffic is configured to flow from a non-default VRF (inside) to the same non-default VRF (outside), the **match-in-vrf** option of the **ip nat** command must be specified.

The following is an example configuration:

```
Switch(config)#ip nat inside source list <ACL_NAME>
      <[interface <INTERFACE NAME> overload] | pool <POOL NAME> [overload]]>
      [ group <1-1024> [dynamic] ] [ vrf <vrf-name> [match-in-vrf] ]
Switch(config)#ip nat inside source static [<LOCAL IP>
      <GLOBAL IP> | [tcp | udp] <LOCAL IP> <LOCAL PORT> <GLOBAL IP> <GLOBAL PORT>
 ]
```

```
      [ group <1-1024> [dynamic] ] [ vrf <vrf-name> [match-in-vrf] ]


Switch(config)#ip  nat  outside  source  list  <ACL_NAME>
      <[interface  <INTERFACE  NAME>]  |  pool  <POOL
      NAME>]>
      [ group <1-1024> [dynamic] ] [ vrf <vrf-name> [match-in-
vrf] ] Switch(config)#ip nat outside source static [<LOCAL IP>
      <GLOBAL IP> | [tcp | udp] <LOCAL IP> <LOCAL PORT> <GLOBAL IP> <GLOBAL PORT>
 ]
      [ group <1-1024> [dynamic] ] [ vrf <vrf-name> [match-in-vrf] ]
```

- VRF aware NAT does not support fragmented packets.
- VRF aware NAT does not support application layer translations.

Therefore, Layer 4 and other embedded IPs are not translated and the following will fail:

- FTP
- ICMP failures
- IPSec
- HTTPS
- VRF aware NAT supports NAT or VACL on an interface. (However, both features cannot be supported at the same time on an interface.)
- VRF aware NAT supports egress ACLs that are applied to the original packet, not on the NAT translated packet.
- VRF aware NAT supports only the default VRF.
- VRF aware NAT does not provide MIB support.
- VRF aware NAT supports only a single global VDC.
- VRF aware NAT does not support the active/standby supervisor model.

11.9 Guidelines and Limitations for Static NAT

Static NAT has the following configuration guidelines and limitations:

- For Broadcom-based Inspur CN12900 Series switches, if the route to your inside global address on the translating device is reachable via the outside interface, packets for Network Address Translated flows coming from outside to inside get software forwarded, duplicated, and looped in the network. For this situation, you must enter the **add-route** CLI argument on the end of the NAT configuration for this flow.

For example, **`ip nat inside source static 192.168.1.1 172.16.1.1 add-route`**.

- **show** commands with the **internal** keyword are not supported.
- On CN12900 platform switches with 9700-EX line cards, the **add-route** option is required for both inside and outside policies.
- NAT supports up to 1024 translations which include both static and dynamic NAT.
- If the translated IP is part of the outside interface subnet, then use the **ip proxy-arp** command on the NAT outside interface. If the **add-route** keyword is used, **ip proxy-arp** should be enabled.
- NAT and sFlow are not supported on the same port.
- The Inspur device supports NAT on the following interface types:
- Switch Virtual Interfaces (SVIs)
- Routed ports
- Layer 3 and Layer 3 subinterfaces.
- NAT is supported on the default Virtual Routing and Forwarding (VRF) table only.
- NAT is supported for IPv4 Unicast only.

- The Inspur device does not support the following:
- Software translation. All translations are done in the hardware.
- NAT and VXLAN routing.
- Application layer translation. Layer 4 and other embedded IPs are not translated, including FTP, ICMP failures, IPSec, and HTTPs.
- NAT and VLAN Access Control Lists (VACLs) that are configured on an interface at the same time.
- PAT translation of fragmented IP packets.
- NAT translation on software forwarded packets. For example, packets with IP-options are not NAT translated.
- By default no TCAM entries are allocated for the NAT feature. You allocate the TCAM size for the NAT feature by adjusting the TCAM size of other features. The TCAM can be allocated with the **hardware access-list tcam region nat** *tcam-size* command.
- HSRP and VRRP are supported only on Static NAT.
- If an IP address is used for Static NAT or PAT translations, it cannot be used for any other purpose. For example, it cannot be assigned to an interface.
- For Static NAT, the outside global IP address should be different from the outside interface IP address.
- NAT statistics are not available.
- When configuring a large number of translations (more than 100), it is faster to configure the translations before configuring the NAT interfaces.
- NAT supports (non-disruptive) In Service Software Upgrade (ISSU).
- UDF-based features may not work when NAT TCAM is carved.
- To enable TCP-aware NAT feature, you need to carve TCP-NAT TCAM using the hardware access-list tcam region tcp-nat tcam-size command.
- For each NAT inside policy with "x" aces, "x" number of entries is required.
- For each configured NAT pool, one entry is required.
- TCP-NAT TCAM size must be doubled when atomic update mode is enabled.

11.10 Restrictions for Dynamic NAT

The following restrictions apply to dynamic Network Address Translation (NAT):

- For Broadcom-based Inspur CN12900 Series switches, if the route to your inside global address on the translating device is reachable via the outside interface, packets for Network Address Translated flows coming from outside to inside get software forwarded, duplicated, and looped in the network. For this situation, you must enter the **add-route** CLI argument on the end of the NAT configuration for this flow. For example, **`ip nat inside source static 192.168.1.1 172.16.1.1 add-route`**.
- **show** commands with the **internal** keyword are not supported.
- On CN12900 platform switches with 9700-EX line cards, the **add-route** option is required for both inside and outside policies.
- VXLAN routing is not supported on Inspur devices.
- Fragmented packets are not supported.
- Application layer gateway (ALG) translations are not supported. ALG, also known as application-level gateway, is an application that translates IP address information inside the payload of an application packet.
- Egress ACLs are not applied to translated packets.
- Nondefault virtual routing and forwarding (VRF) instances are not supported.
- MIBs are not supported.
- Multiple global virtual device contexts (VDCs) are not supported on Inspur devices.
- Dynamic NAT translations are not synchronized with active and standby devices.
- Stateful NAT is not supported. However, NAT and Hot Standby Router Protocol (HSRP) can coexist.
- The timeout value for take up to the configured time-out + 119 seconds.
- Normally, ICMP NAT flows time out after the expiration of the configured sampling-timeout and translation-timeout. However, when ICMP NAT flows present in the switch become idle, they time out immediately after

the expiration of the sampling-timeout configured.

- When creating a new translation on a Inspur CN12900 Series switch, the flow is software forwarded until the translation is programmed in the hardware, which might take a few seconds. During this period, there is no translation entry for the inside global address. Therefore, returning traffic is dropped. To overcome this limitation, create a loopback interface and give it an IP address that belongs to the NAT pool.
- For dynamic NAT, pool overload and interface overload are not supported for the outside NAT.
- Because the NAT overload uses PBR (Policy-Based Routing), the maximum number of available next-hop entries in the PBR table determines NAT scale. If the number of NAT inside interfaces are within the range of available next-hops entries in the PBR table, the maximum NAT translation scale remains same.

Otherwise, the maximum number of supported translations may reduce. PBR and NAT-overload are not mutually exclusive; they are mutually limiting.

11.11 Guidelines and Limitations for Dynamic Twice NAT

For Broadcom-based Inspur CN12900 Series switches, if the route to your inside global address on the translating device is reachable via the outside interface, packets for Network Address Translated flows coming from outside to inside get software forwarded, duplicated, and looped in the network. For this situation, you must enter the **add-route** CLI argument on the end of the NAT configuration for this flow. For example, **`ip nat inside source static 192.168.1.1 172.16.1.1 add-route`**.

IP packets without TCP/UDP/ICMP headers are not translated with dynamic NAT.

In dynamic twice NAT, if dynamic NAT flows are not created before creating static NAT flows, dynamic twice NAT flows are not created correctly.

When an empty ACL is created, the default rule of **permit ip any any** is configured. The NAT-ACL does not match further ACL entries if the first ACL is blank.

11.12 Configuring Static NAT

11.12.1 Enabling Static NAT

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **feature nat**
3. switch(config)# **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **feature nat**	Enables the static NAT feature on the device.
Step 3	switch(config)# **copy running-config startup-config**	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

11.12.2 Configuring Static NAT on an Interface

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **interface** *type slot*/*port*
3. switch(config-if)# **ip nat** {**inside** | **outside**}
4. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface** *type slot*/*port*	Specifies an interface to configure, and enters interface configuration mode.
Step 3	switch(config-if)# **ip nat** {**inside** | **outside**}	Specifies the interface as inside or outside. **Note** Only packets that arrive on a marked interface can be translated.
Step 4	(Optional) switch(config)# **copy running-config** **startup-config**	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Example

This example shows how to configure an interface with static NAT from the inside:

```
switch# configure terminal
switch(config)# interface ethernet 1/4
switch(config-if)# ip nat inside
```

11.12.3 Enabling Static NAT for an Inside Source Address

For inside source translation, the traffic flows from inside interface to the outside interface. NAT translates the inside local IP address to the inside global IP address. On the return traffic, the destination inside global IP address gets translated back to the inside local IP address.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **ip nat inside source static** *local-ip-address global-ip-address* [**group** *group-id*] [**vrf** *vrf-name* [**match-in-vrf**]]
3. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **ip nat inside source static** *local-ip-*	Configures static NAT to translate the inside local

	Command or Action	Purpose
	address *global-ip-address* [**group** *group-id*] [**vrf** *vrf-name* [**match-in-vrf**]]	address to the inside global address or to translate the opposite (the inside global traffic to the inside local traffic). Specifying **group** specifies the group to which this translation belongs on the static twice NAT. **Note** While performing twice NAT configuration in Inspur CN12900 Series switches, you cannot use the same group ID across different VRFs. A unique group ID should be used for unique twice NAT rules.
Step 3	(Optional) switch(config)# **copy running-config startup-config**	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Example

This example shows how to configure static NAT for an inside source address:

```
switch# configure terminal
switch(config)# ip nat inside source static 1.1.1.1
5.5.5.5 switch(config)# copy running-config startup-config
```

11.12.4 Enabling Static NAT for an Outside Source Address

For outside source translation, the traffic flows from the outside interface to the inside interface. NAT translates the outside global IP address to the outside local IP address. On the return traffic, the destination outside local IP address gets translated back to outside global IP address.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **ip nat outside source static** *global-ip-address* [**group** *group-id*] [**add-route**] [**vrf** *vrf-name* [**match-in-vrf**]]
3. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **ip nat outside source static**	Configures static NAT to translate the outside global

	global-ip-address [**group** *group-id*] [**add-route**] [**vrf** *vrf-name* [**match-in-vrf**]]	address to the outside local address or to translate the opposite (the outside local traffic to the outside global traffic). Specifying **group** specifies the group to which this translation belongs on the static twice NAT. When an inside translation without ports is configured, an implicit add route is performed. The original add route functionality is an option while configuring an outside translation.
Step 3	(Optional) switch(config)# **copy running-config** **startup-config**	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Example

This example show how to configure static NAT for an outside source address:

```
switch# configure terminal
switch(config)# ip nat outside source static 2.2.2.2
6.6.6.6 switch(config)# copy running-config startup-config
```

11.12.5 Configuring Static PAT for an Inside Source Address

You can map services to specific inside hosts using Port Address Translation (PAT).

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **ip nat inside source static** {*inside-local-address outside-local-address* | {**tcp**| **udp**} *inside-local-address* {*local-tcp-port* | *local-udp-port*} *inside-global-address* {*global-tcp-port* | *global-udp-port*}} *global-ip-address* {**group** *group-id*} {**vrf** *vrf-name* {**match-in-vrf**}}
3. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **ip nat inside source static** {*inside-local-address outside-local-address* | {**tcp**| **udp**} *inside-local-address* {*local-tcp-port* | *local-udp-port*} *inside-global-address* {*global-tcp-port* | *global-udp-port*}} *global-ip-address* {**group** *group-id*} {**vrf** *vrf-name*	Maps static NAT to an inside local port to an inside global port.

	Command or Action	Purpose
	{**match-in-vrf**}}	
Step 3	(Optional) switch(config)# **copy running-config** **startup-config**	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Example

This example shows how to map UDP services to a specific inside source address and UDP port:

```
switch# configure terminal
switch(config)# ip nat inside source static udp 20.1.9.2 63 35.48.35.48 130
switch(config)# copy running-config startup-config
```

11.12.6 Configuring Static PAT for an Outside Source Address

You can map services to specific outside hosts using Port Address Translation (PAT).

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **ip nat outside source static** {*outside-global-address outside-local-address* | {**tcp** | **udp**} *outside-global-address* {*global-tcp-port* | *global-udp-port*} *outside-local-address* {*global-tcp-port* | *global-udp-port*}} *global-ip-address* {**group** *group-id*} {**add-route**} {**vrf** *vrf-name* {**match-in-vrf**}}
3. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.

Step 2	switch(config)# **ip nat outside source static**
	{*outside-global-address outside-local-address* | {**tcp** | **udp**}
	outside-global-address {*global-tcp-port* | *global-udp-port*}
	outside-local-address {*global-tcp-port* | *global-udp-port*}}
	global-ip-address {**group** *group-id*} {**add-route**} {**vrf** *vrf-name* {**match-in-vrf**}}
		Maps static NAT to an outside global port to an outside local port.
		Specifying **group** specifies the group to which this
		translation belongs on the static twice NAT. When an inside
		translation without ports is configured, an implicit add route
		is performed. The original add route functionality is an option while configuration an outside translation.

Step 3	(Optional) switch(config)# **copy running-config** **startup-config**	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Example

This example shows how to map TCP services to a specific outside source address and TCP port:

```
switch# configure terminal
switch(config)# ip nat outside source static tcp 20.1.9.2 63 35.48.35.48 130
switch(config)# copy running-config startup-config
```

11.12.7 Configuring Static Twice NAT

All translations within the same group are considered for creating static twice Network Address Translation (NAT) rules.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat inside source static** *inside-local-ip-address inside-global-ip-address* [**group** *group-id*]
4. **ip nat outside source static** *outside-global-ip-address outside-local-ip-address* [**group** *group-id*] [**add-route**]
5. **interface** *type number*
6. **ip address** *ip-address mask*
7. **ip nat inside**
8. **exit**
9. **interface** *type number*
10. **ip address** *ip-address mask*
11. **ip nat outside**
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**Enable** **Example:** `switch> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `switch# configure terminal`	Enters privileged EXEC mode.
Step 3	**ip nat inside source static** *inside-local-ip-address* *inside-global-ip-address* [**group** *group-id*] **Example:** `switch(config)# ip nat inside source static 10.1.1.1 192.168.34.4 group 4`	Configures static twice NAT to translate an inside local IP address to the corresponding inside global IP address. • The **group** keyword determines the group to which a translation belongs.
Step 4	**ip nat outside source static** *outside-global-ip-address outside-local-ip-address* [**group** *group-id*] [**add-route**] **Example:**	Configures static twice NAT to translate an outside global IP address to the corresponding outside local IP address. • The **group** keyword determines the group to which

	Command or Action	Purpose
	`switch(config)# ip nat outside source static 209.165.201.1 10.3.2.42 group 4 add-route`	a translation belongs.
Step 5	**interface** *type number* **Example:** `switch(config)# interface ethernet 1/2`	Configures an interface and enters interface configuration mode.
Step 6	**ip address** *ip-address mask* **Example:** `switch(config-if)# ip address 10.2.4.1 255.255.255.0`	Sets a primary IP address for an interface.
Step 7	**ip nat inside** **Example:** `switch(config-if)# ip nat inside`	Connects the interface to an inside network, which is subject to NAT.
Step 8	**Exit** **Example:** `switch(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 9	**interface** *type number* **Example:** `switch(config)# interface ethernet 1/1`	Configures an interface and enters interface configuration mode.
Step 10	**ip address** *ip-address mask* **Example:** `switch(config-if)# ip address 10.5.7.9 255.255.255.0`	Sets a primary IP address for an interface.
Step 11	**ip nat outside** **Example:** `switch(config-if)# ip nat outside`	Connects the interface to an outside network, which is subject to NAT.
Step 12	**End** **Example:** `switch(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

11.12.8 Configuration Example for Static NAT and PAT

This example shows the configuration for static NAT:

```
ip nat inside source static 103.1.1.1 11.3.1.1 ip
nat inside source static 139.1.1.1 11.39.1.1 ip nat
inside source static 141.1.1.1 11.41.1.1 ip nat
inside source static 149.1.1.1 95.1.1.1 ip nat
inside source static 149.2.1.1 96.1.1.1 ip nat
outside source static 95.3.1.1 95.4.1.1 ip nat
outside source static 96.3.1.1 96.4.1.1 ip nat
outside source static 102.1.2.1 51.1.2.1 ip nat
outside source static 104.1.1.1 51.3.1.1 ip nat
outside source static 140.1.1.1 51.40.1.1
```

This example shows the configuration for static PAT:

```
ip nat inside source static tcp 10.11.1.1 1 210.11.1.1 101

ip nat inside source static tcp 10.11.1.1 2 210.11.1.1 201 ip
nat inside source static tcp 10.11.1.1 3 210.11.1.1 301 ip nat
inside source static tcp 10.11.1.1 4 210.11.1.1 401 ip nat
inside source static tcp 10.11.1.1 5 210.11.1.1 501 ip nat
inside source static tcp 10.11.1.1 6 210.11.1.1 601 ip nat
inside source static tcp 10.11.1.1 7 210.11.1.1 701 ip nat
inside source static tcp 10.11.1.1 8 210.11.1.1 801 ip nat
inside source static tcp 10.11.1.1 9 210.11.1.1 901 ip nat
inside source static tcp 10.11.1.1 10 210.11.1.1 1001 ip nat
inside source static tcp 10.11.1.1 11 210.11.1.1 1101 ip nat
inside source static tcp 10.11.1.1 12 210.11.1.1 1201
```

11.12.9 Example: Configuring Static Twice NAT

The following example shows how to configure the inside source and outside source static twice NAT configurations:

```
Switch> enable
Switch# configure terminal
Switch(config)# ip nat inside source static 10.1.1.1 192.168.34.4 group 4
Switch(config)# ip nat outside source static 209.165.201.1 10.3.2.42 group 4
Switch(config)# interface ethernet 1/2
Switch(config-if)# ip address 10.2.4.1 255.255.255.0
Switch(config-if)# ip nat inside
switch(config-if)# exit switch(config)#
interface ethernet 1/1
switch(config-if)# ip address 10.5.7.9 255.255.255.0
switch(config-if)# ip nat outside
Switch(config-if)# end
```

11.12.10 Verifying the Static NAT Configuration

To display the static NAT configuration, perform this task:

SUMMARY STEPS

1. switch# **show ip nat translations**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **show ip nat translations**	Shows the translations for the inside global, inside

		local, outside local, and outside global IP addresses.

Example

This example shows how to display the static NAT configuration:

```
switch# sh ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
--- ---                ---                51.3.1.1           104.1.1.1
--- ---                ---                95.4.1.1           95.3.1.1



--- ---                ---                96.4.1.1           96.3.1.1
--- ---                ---                51.40.1.1          140.1.1.1
--- ---                ---                51.42.1.1          142.1.2.1
--- ---                ---                51.1.2.1           102.1.2.1
--- 11.1.1.1           101.1.1.1          ---                ---
--- 11.3.1.1           103.1.1.1          ---                ---
--- 11.39.1.1          139.1.1.1          ---                ---
--- 11.41.1.1          141.1.1.1          ---                ---
--- 95.1.1.1           149.1.1.1          ---                ---
--- 96.1.1.1           149.2.1.1          ---                ---
    130.1.1.1:590      30.1.1.100:5000    ---                ---
    130.2.1.1:590      30.2.1.100:5000    ---                ---
    130.3.1.1:590      30.3.1.100:5000    ---                ---
    130.4.1.1:590      30.4.1.100:5000    ---                ---

    130.1.1.1:591      30.1.1.101:5000    ---                ---


switch# sh ip nat translations verbose
Pro Inside global      Inside local       Outside local      Outside global
any ---                ---                22.1.1.3           22.1.1.2
    Flags:0x200009 time-left(secs):-1 id:0 state:0x0 grp_id:10
any 11.1.1.130         11.1.1.3           ---                ---
    Flags:0x1 time-left(secs):-1 id:0 state:0x0 grp_id:0
any 11.1.1.133         11.1.1.33          ---                ---
    Flags:0x1 time-left(secs):-1 id:0 state:0x0 grp_id:10
any 11.1.1.133         11.1.1.33          22.1.1.3           22.1.1.2
    Flags:0x200009 time-left(secs):-1 id:0  state:0x0 grp_id:0

tcp 10.1.1.100:64490   10.1.1.2:0         20.1.1.2:0         20.1.1.2:0
    Flags:0x82 time-left(secs):43192 id:31 state:0x3 grp_id:0 vrf: default
switch-1#
```

11.13 Configuring Dynamic NAT

11.13.1 Configuring Dynamic Translation and Translation Timeouts

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip access-list** *access-list-name*

4. **permit** *protocol source source-wildcard* **any**
5. **deny** *protocol source source-wildcard* **any**
6. **exit**
7. **ip nat inside source list** *access-list-name* **interface** *type number* **overload** [**vrf** *vrf-name* [**match-in-vrf**]]
8. **interface** *type number*
9. **ip address** *ip-address mask*
10. **ip nat inside**
11. **exit**
12. **interface** *type number*
13. **ip address** *ip-address mask*
14. **ip nat outside**
15. **exit**
16. **ip nat translation max-entries** *number-of-entries*
17. **ip nat translation timeout** *seconds*
18. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**Enable** **Example:** `Switch> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Switch# configure terminal`	Enters global configuration mode.
Step 3	**ip access-list** *access-list-name* **Example:** `Switch(config)# ip access-list acl1`	Defines an access list and enters access-list configuration mode.
Step 4	**permit** *protocol source source-wildcard* **any** **Example:** `Switch(config-acl)# permit ip 10.111.11.0/24 any`	Sets conditions in an IP access list that permit traffic matching the conditions.
Step 5	**deny** *protocol source source-wildcard* **any** **Example:** `Switch(config-acl)# deny udp 10.111.11.100/32 any`	Sets conditions in an IP access list that deny packets from entering a network.
Step 6	**Exit** **Example:** `Switch(config-acl)# exit`	Exits access-list configuration mode and returns to global configuration mode.
Step 7	**ip nat inside source list** *access-list-name*	Establishes dynamic source translation by

	Command or Action	Purpose

	interface *type*
	number **overload** [**vrf** *vrf-name* [**match-in-vrf**]]
	Example:
	`Switch(config)# ip nat inside source list`
	`acl1`
	`interface ethernet 1/1 overload`
		specifying the access list defined in Step 3.

Step 8	**interface** *type number* **Example:** `Switch(config)# interface ethernet 1/4`	Configures an interface and enters interface configuration mode.

Step 9	**ip address** *ip-address mask*
	Example:
	`Switch(config-if)# ip address 10.111.11.39`
	`255.255.255.0`
		Sets a primary IP address for the interface.

Step 10	**ip nat inside** **Example:** `Switch(config-if)# ip nat inside`	Connects the interface to an inside network, which is subject to NAT.
Step 11	**Exit** **Example:** `Switch(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 12	**interface** *type number* **Example:** `Switch(config)# interface ethernet 1/1`	Configures an interface and enters interface configuration mode.

Step 13	**ip address** *ip-address mask*
	Example:
	`Switch(config-if)# ip address 172.16.232.182`
	`255.255.255.240`
		Sets a primary IP address for an interface.

Step 14	**ip nat outside** **Example:** `Switch(config-if)# ip nat outside`	Connects the interface to an outside network.
Step 15	**Exit** **Example:** `Switch(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.

	Command or Action	Purpose
Step 16	**ip nat translation max-entries** *number-of-entries* **Example:** `Switch(config)# ip nat translation max-entries 300`	Specifies the maximum number of dynamic NAT translations. The number of entries can be between 1 and 1023.
Step 17	**ip nat translation timeout** *seconds* **Example:** `switch(config)# ip nat translation timeout 13000`	Specifies the timeout value for dynamic NAT translations.
Step 18	**End** **Example:** `Switch(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

11.13.2 Configuring Dynamic NAT Pool

You can create a NAT pool by either defining the range of IP addresses in a single **ip nat pool** command or by using the **ip nat pool** and **address** commands

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **feature nat**
3. switch(config)# **ip nat pool** *pool-name* [*startip endip*] {**prefix** *prefix-length* | **netmask** *network-mask*}
4. (Optional) switch(config-ipnat-pool)# **address** *startip endip*
5. (Optional) switch(config)# **no ip nat pool** *pool-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **feature nat**	Enables the NAT feature on the device.
Step 3	switch(config)# **ip nat pool** *pool-name* [*startip endip*] {**prefix** *prefix-length* | **netmask** *network-mask*}	Creates a NAT pool with a range of global IP addresses. The IP addresses are filtered by using either a prefix length or a network mask.
Step 4	(Optional) switch(config-ipnat-pool)# **address** *startip endip*	Specifies the range of global IP addresses if they were not specified during creation of the pool.
Step 5	(Optional) switch(config)# **no ip nat pool** *pool-name*	Deletes the specified NAT pool.

Example

This example shows how to create a NAT pool with a prefix length:

```
switch# configure terminal
switch(config)# ip nat pool pool1 30.1.1.1 30.1.1.2 prefix-length 24
switch(config)#
```

This example shows how to create a NAT pool with a network mask:

```
switch# configure terminal
switch(config)# ip nat pool pool5 20.1.1.1 20.1.1.5 netmask 255.0.255.0
switch(config)#
```

This example shows how to create a NAT pool and define the range of global IP addresses using the **ip nat pool** and **address** commands:

```
switch# configure terminal
switch(config)# ip nat pool pool7 netmask 255.255.0.0
switch(config-ipnat-pool)# address 40.1.1.1 40.1.1.5
switch(config-ipnat-pool)#
```

This example shows how to delete a NAT pool:

```
switch# configure terminal
switch(config)# no ip nat pool pool4
switch(config)#
```

11.13.3 Configuring Source Lists

You can configure a source list of IP addresses for the inside interface and the outside interface.

Before you begin

Ensure that you configure a pool before configuring the source list for the pool.

SUMMARY STEPS

1. switch# **configure terminal**
2. (Optional) switch# **ip nat inside source list** *list-name* **pool** *pool-name* [**overload**]
3. (Optional) switch# **ip nat outside source list** *list-name* **pool** *pool-name* [**add-route**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	(Optional) switch# **ip nat inside source list** *list-name* **pool** *pool-name* [**overload**]	Creates a NAT inside source list with pool with or without overloading.
Step 3	(Optional) switch# **ip nat outside source list** *list-name* **pool** *pool-name* [**add-route**]	Creates a NAT outside source list with pool without overloading.

Example

This example shows how to create a NAT inside source list with pool without overloading:

```
switch# configure terminal
switch(config)# ip nat inside source list list1 pool pool1
switch(config)#
```

This example shows how to create a NAT inside source list with pool with overloading:

```
switch# configure terminal
switch(config)# ip nat inside source list list2 pool pool2 overload
switch(config)#
```

This example shows how to create a NAT outside source list with pool without overloading:

```
switch# configure terminal
switch(config)# ip nat outside source list list3 pool pool3
switch(config)#
```

11.13.4 Configuring Dynamic Twice NAT for an Inside Source Address

For an inside source address translation, the traffic flows from the inside interface to the outside interface. You can configure dynamic twice NAT for an inside source address.

Before you begin

Ensure that you enable NAT on the switch.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **ip nat outside source static** *outside-global-ip-address outside-local-ip-address* | [**tcp** | **udp**] *outside-global-ip-address outside-global-port outside-local-ip-address outside-local-port* [**group** *group-id*] [**add-route**] [**dynamic**]
3. switch(config)# **ip nat inside source list** *access-list-name* [**interface** *type slot/port* **overload** | **pool** *pool-name*] [**group** *group-id*] [**dynamic**]]
4. switch(config)# **ip nat pool** *pool-name* [*startip endip*] {**prefix** *prefix-length* | **netmask** *network-mask*}
5. switch(config)# **interface** *type slot/port*
6. switch(config-if)# **ip nat outside**
7. switch(config-if)# **exit**
8. switch(config)# **interface** *type slot/port*
9. switch(config-if)# **ip nat inside**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **ip nat outside source static** *outside-global-ip-address outside-local-ip-address* | [**tcp** | **udp**] *outside-global-ip-address outside-global-port outside-local-ip-address outside-local-port* [**group** *group-id*] [**add-route**] [**dynamic**]	Configures static NAT to translate an outside global address to an inside local address or to translate inside local traffic to inside global traffic. The **group** keyword determines the group to which a translation belongs.
Step 3	switch(config)# **ip nat inside source list** *access-list-name* [**interface** *type slot/port* **overload** | **pool** *pool-name*]	Establishes dynamic source translation by creating a NAT inside source list with pool with or without overloading.

	Command or Action	Purpose
	[**group** *group-id*] [**dynamic**]]	The **group** keyword determines the group to which a translation belongs.
Step 4	switch(config)# **ip nat pool** *pool-name* [*startip endip*] {**prefix** *prefix-length* | **netmask** *network-mask*}	Creates a NAT pool with a range of global IP addresses. The IP addresses are filtered by using either a prefix length or a network mask.
Step 5	switch(config)# **interface** *type slot/port*	Configures an interface and enters interface configuration mode.
Step 6	switch(config-if)# **ip nat outside**	Connects the interface to an outside network.
Step 7	switch(config-if)# **exit**	Exits interface configuration mode and returns to global configuration mode.
Step 8	switch(config)# **interface** *type slot/port*	Configures an interface and enters interface configuration mode.
Step 9	switch(config-if)# **ip nat inside**	Connects the interface to an inside network, which is subject to NAT.

Example

This example shows how to configure dynamic twice NAT for an inside source address:

```
switch# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
switch(config)# ip nat outside source static 2.2.2.2 4.4.4.4 group 20 dynamic
switch(config)# ip nat inside source list acl_1 pool pool_1 overload group 20
dynamic switch(config)# ip nat pool pool_1 3.3.3.3 3.3.3.10 prefix-length 24
switch(config)# interface Ethernet1/8
switch(config-if)# ip nat outside
switch(config-if)# exit switch(config)#
interface Ethernet1/15 switch(config-
if)# ip nat inside
```

11.13.5 Configuring Dynamic Twice NAT for an Outside Source Address

For an outside source address translation, the traffic flows from the outside interface to the inside interface. You can configure dynamic twice NAT for an outside source address.

Before you begin

Ensure that you enable NAT on the switch.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **ip nat inside source static** *inside-local-ip-address inside-global-ip-address* | [**tcp** | **udp**] *inside-local-ip-address local-port inside-global-ip-address global-port* [**group** *group-id*] [**dynamic**]
3. switch(config)# **ip nat outside source list** *access-list-name* [**interface** *type slot/port* **overload** | **pool** *pool-name*] [**group** *group-id*] [**add-route**] [**dynamic**]
4. switch(config)# **ip nat pool** *pool-name* [*startip endip*] {**prefix** *prefix-length* | **netmask** *network-mask*}
5. switch(config)# **interface** *type slot/port*
6. switch(config-if)# **ip nat outside**
7. switch(config-if)# **exit**
8. switch(config)# **interface** *type slot/port*
9. switch(config-if)# **ip nat inside**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **ip nat inside source static** *inside-local-ip-address inside-global-ip-address* | [**tcp** | **udp**] *inside-local-ip-address local-port inside-global-ip-address global-port* [**group** *group-id*] [**dynamic**]	Configures static NAT to translate an inside global address to an inside local address or to translate inside local traffic to inside global traffic. The **group** keyword determines the group to which a translation belongs.
Step 3	switch(config)# **ip nat outside source list** *access-list-name* [**interface** *type slot/port* **overload** | **pool** *pool-name*] [**group** *group-id*] [**add-route**] [**dynamic**]	Establishes dynamic source translation by creating a NAT outside source list with pool with or without overloading.
Step 4	switch(config)# **ip nat pool** *pool-name* [*startip endip*] {**prefix** *prefix-length* | **netmask** *network-mask*}	Creates a NAT pool with a range of global IP addresses. The IP addresses are filtered by using either a prefix length or a network mask.
Step 5	switch(config)# **interface** *type slot/port*	Configures an interface and enters interface configuration mode.
Step 6	switch(config-if)# **ip nat outside**	Connects the interface to an outside network.
Step 7	switch(config-if)# **exit**	Exits interface configuration mode and returns to global configuration mode.
Step 8	switch(config)# **interface** *type slot/port*	Configures an interface and enters interface configuration mode.

	Command or Action	Purpose
Step 9	switch(config-if)# **ip nat inside**	Connects the interface to an inside network, which is subject to NAT.

Example

This example shows how to configure dynamic twice NAT for an outside source address:

```
switch# configure terminal
Enter configuration commands, one per line. End with CNTL/Z. switch(config)#
ip nat inside source static 7.7.7.7 5.5.5.5 group 30 dynamic switch(config)#
ip nat pool pool_2 4.4.4.4 4.4.4.10 prefix-length 24 switch(config)#
interface Ethernet1/6
switch(config-if)# ip nat outside
switch(config-if)# exit switch(config)#
interface Ethernet1/11 switch(config-
if)# ip nat inside
```

11.13.6 Configuring FINRST and SYN Timers

This section describes how to configure FINRST and SYN timer values. When you reload the switch, restoring or erasing the configured FINRST and/or SYN timer values depends on whether or not the TCP TCAM carved. If the TCAM is carved, the switch restores the currently configured values. If the timer values are not configured, it sets a default value of 60. If the TCAM is not carved, the switch removes any currently configured values and sets a default value as never. This is because the the TCP AWARE feature gets disabled when the TCP TCAM is not carved.

Before you begin

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config-if)# **ip nat translation syn-timeout** {**seconds** | **never**}
3. switch(config-if)# **ip nat translation finrst-timeout** {**seconds** | **never**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config-if)# **ip nat translation syn-timeout** {**seconds** | **never**}	Specifies the timeout value for TCP data packets that sends the SYN request, but do not receive a SYN-ACK reply. The timeout value ranges from 1 to 172800 seconds. When the TCP TCAM is carved the default value is 60 seconds. When the TCP TCAM is not carved the default value is *never*. The *never* keyword deactivates SYN timer. **Note** You cannot configure SYN timer when TCP

	Command or Action	Purpose
		TCAM is not carved..
Step 3	switch(config-if)# **ip nat translation finrst-timeout** {**seconds** | **never**}	Specifies the timeout value for the flow entries when a connection is terminated by receiving finish (FIN) or reset (RST) packets. You must use the configure the behavior for both RST and FIN. The timeout value ranges from 1 to 172800 seconds. When the TCP TCAM is carved the default value is 60 seconds. When the TCP TCAM is not carved the default value is *never*. The *never* keyword deactivates FIN or RST timers. **Note** You cannot configure FINRST timer when TCP TCAM is not carved..

Example

The following example that shows when TCP TCAM is carved

```
switch(config)# ip nat translation syn-timeout 20
```

The following example that shows when TCP TCAM is not carved

```
switch(config)# ip nat translation syn-timeout 20 Error:

SYN TIMER CONFIG FAILED.TCP TCAM NOT CONFIGURED
```

11.13.7 Clearing Dynamic NAT Translations

To clear dynamic translations, perform the following task:

Command	Purpose
clear ip nat translation [**all** | **inside** *global-ip-address local-ip-address* [**outside** *local-ip-address global-ip-address*] | **outside** *local-ip-address global-ip-address*]	Deletes all or specific dynamic NAT translations.

Example

This example shows how to clear all dynamic translations:

```
switch# clear ip nat translation all
```

This example shows how to clear dynamic translations for inside and outside addresses:

```
switch# clear ip nat translation inside 2.2.2.2 4.4.4.4 outside 5.5.5.5 7.7.7.7
```

11.13.8 Verifying Dynamic NAT Configuration

To display dynamic NAT configuration, perform the following tasks:

Command	Purpose
show ip nat translations	Displays active Network Address Translation (NAT) translations. Displays additional information for each translation table entry, including when an entry was created and used.
show run nat	Displays NAT configuration.

Example

This example shows how to display running configuration for NAT:

```
switch# show run nat
!Command: show running-config nat
!Time: Wed Apr 23 11:17:43 2014
version 6.0(2)A3(1)

feature nat

ip nat inside source list list1 pool pool1
ip nat inside source list list2 pool pool2 overload
ip nat inside source list list7 pool pool7 overload
ip nat outside source list list3 pool pool3
ip nat pool pool1 30.1.1.1 30.1.1.2 prefix-length 24 ip nat
pool pool2 10.1.1.1 10.1.1.2 netmask 255.0.255.0 ip nat
pool pool3 30.1.1.1 30.1.1.8 prefix-length 24 ip nat pool
pool5 20.1.1.1 20.1.1.5 netmask 255.0.255.0 ip nat pool
pool7 netmask 255.255.0.0
  address 40.1.1.1 40.1.1.5
```

This example shows how to display active NAT translations:
Inside pool with overload

```
switch# show ip nat translation
Pro  Inside global      Inside local       Outside local      Outside global
icmp 20.1.1.3:64762     10.1.1.2:133       20.1.1.1:0         20.1.1.1:0

icmp 20.1.1.3:64763     10.1.1.2:134       20.1.1.1:0         20.1.1.1:0
```

Outside pool without overload

```
switch# show ip nat translation
Pro  Inside global      Inside local       Outside local      Outside global
any  ---                ---                177.7.1.1:0        77.7.1.64:0
any  ---                ---                40.146.1.1:0       40.46.1.64:0

any  ---                ---                10.4.146.1:0       10.4.46.64:0
```

11.13.9 Example: Configuring Dynamic Translation and Translation Timeouts

The following example shows how to configure dynamic overload Network Address Translation (NAT) by specifying an access list:

```
Switch> enable

Switch# configure terminal
Switch(config)# ip access-list acl1
Switch(config-acl)# permit ip 10.111.11.0/24 any
Switch(config-acl)# deny udp 10.111.11.100/32 any
Switch(config-acl)# exit
Switch(config)# ip nat inside source list acl1 interface ethernet 1/1 overload
Switch(config)# interface ethernet 1/4
Switch(config-if)# ip address 10.111.11.39 255.255.255.0
Switch(config-if)# ip nat inside
Switch(config-if)# exit
Switch(config)# interface ethernet 1/1
Switch(config-if)# ip address 172.16.232.182 255.255.255.240
Switch(config-if)# ip nat outside
Switch(config-if)# exit
Switch(config)# ip nat translation max-entries 300
Switch(config)# ip nat translation timeout 13000
Switch(config)# end
```

CHAPTER 12 Configuring IP Event Dampening

12.1 IP Event Dampening Overview

Interface state changes occur when interfaces are administratively brought up or down or if an interface changes state. When an interface changes state or flaps, routing protocols are notified of the status of the routes that are affected by the change in state. Every interface state change requires all affected devices in the network to recalculate best paths, install or remove routes from the routing tables, and then advertise valid routes to peer routers. An unstable interface that flaps excessively can cause other devices in the network to consume substantial amounts of system processing resources and cause routing protocols to lose synchronization with the state of the flapping interface.

The IP Event Dampening feature introduces a configurable exponential decay mechanism to suppress the effects of excessive interface flapping events on routing protocols and routing tables in the network. This feature allows the network operator to configure a router to automatically identify and selectively dampen a local interface that is flapping. Dampening an interface removes the interface from the network until the interface stops flapping and becomes stable. Configuring the IP Event Dampening feature improves convergence times and stability throughout the network by isolating failures so that disturbances are not propagated. This, in turn, reduces the utilization of system processing resources by other devices in the network and improves overall network stability.

12.2 Guidelines and Limitations

The IP Event Dampening feature introduces a configurable exponential decay mechanism to suppress the effects of excessive interface flapping events on routing protocols and routing tables in the network. This feature allows the network operator to configure a router to automatically identify and selectively dampen a local interface that is flapping. See the following guidelines and limitations before configuring IP Event Dampening feature:

- Beginning from Inspur INOS-CN Release, IP event dampening is supported on Inspur CN12900 platform switches.
- Due to changes in the netstack-IP component, all the IP clients observe the impact of dampening or interface.
- For each flap of the interface, a certain penalty is added. The penalty decays exponentially whose parameters are configured.
- When penalty exceeds a certain high level, the interface is dampened. It is unsuppressed when the penalty decays below a low level.
- When an interface is dampened, the IP address and the static routes are removed from the interface. All the clients of IP get an IP delete notification.
- When an interface is unsuppressed, the IP address and the relevant routes are added back. All the clients of IP get an IP address add notification for all the IP addresses of the interface.
- All Layer 3 interfaces that are configured on the Ethernet interface, port channels, and SVI support this feature.

12.3 Interface State Change Events

IP Event Dampening feature employs a configurable exponential decay mechanism that is used to suppress the effects of excessive interface flapping or state changes. When the IP Event Dampening feature is enabled, flapping interfaces are dampened from the perspective of the routing protocol by filtering excessive route updates. Flapping interfaces are identified, assigned penalties, suppressed if necessary, and made available to the network when the interface stabilizes.

12.3.1 Suppress Threshold

The suppress threshold is the value of the accumulated penalty that triggers the router to dampen a flapping interface. The flapping interface is identified by the router and assigned a penalty for each up and down state change, but the interface is not automatically dampened. The router tracks the penalties that a flapping interface accumulates. When the accumulated penalty reaches the default or preconfigured suppress threshold, the interface is placed in a dampened state.

12.3.2 Half-Life Period

The half-life period determines how fast the accumulated penalty can decay exponentially. When an interface is placed in a dampened state, the router monitors the interface for additional up and down state changes. If the interface continues to accumulate penalties and the interface remains in the suppress threshold range, the interface will remain dampened. If the interface stabilizes and stops flapping, the penalty is reduced by half after each half-life period expires. The accumulated penalty will be reduced until the penalty drops to the reuse threshold. The configurable range of the half-life period timer is from 1 to 30 seconds. The default half-life period timer is 5 seconds.

12.3.3 Reuse Threshold

When the accumulated penalty decreases until the penalty drops to the reuse threshold, the route is unsuppressed and made available to other devices in the network. The range of the reuse value is from 1 to 20000 penalties. The default value is 1000 penalties.

12.3.4 Maximum Suppress Time

The maximum suppress time represents the maximum time an interface can remain dampened when a penalty is assigned to an interface. The maximum suppress time can be configured from 1 to 255 seconds. The maximum penalty is truncated to maximum 20000 unit. The maximum value of the accumulated penalty is calculated based on the maximum suppress time, reuse threshold, and half-life period.

IP event dampening configuration command applies dampening to routing protocols for both IP and CLNS.

The first set of parameters ([half-life | reuse | suppress max-suppress]) configure the different parameters of the dampening algorithm. The second set ([restart [penalty]]) enables dampening penalty to be applied when the interface comes up the first time after reboot. The default restart penalty is applied only if you specify the restart parameter. Both parameter sets are optional

12.4 Affected Components

When an interface is not configured with dampening, or when an interface is configured with dampening but is not suppressed, the routing protocol behavior as a result of interface state transitions is not changed by the IP Event Dampening feature. However, if an interface is suppressed, the routing protocols and routing tables are immune to any further state transitions of the interface until it is unsuppressed.

12.4.1 Route Types

- Connected routes:
- The connected routes of dampened interfaces are not installed into the routing table.
- When a dampened interface is unsuppressed, the connected routes will be installed into the routing table if the interface is up.
- Static routes:
- Static routes assigned to a dampened interface are not installed into the routing table.
- When a dampened interface is unsuppressed, the static route will be installed into the routing table if the interface is up.

12.4.2 Supported Protocols

All the protocols that are used are impacted by the IP Event Dampening feature. The IP Event Dampening feature supports Border Gateway Protocol (BGP), Enhanced Interior Gateway Routing Protocol (EIGRP), Hot Standby Routing Protocol (HSRP), Open Shortest Path First (OSPF), Routing Information Protocol (RIP), and VRRP. Ping and SSH to the concerned interface IP address does not work.

12.5 How to Configure IP Event Dampening

12.5.1 Enabling IP Event Dampening

The **dampening** command is entered in interface configuration mode to enable the IP Event Dampening feature. If this command is applied to an interface that already has dampening configured, all dampening states are reset and the accumulated penalty will be set to 0. If the interface has been dampened, the accumulated penalty will fall into the reuse threshold range, and the dampened interface will be made available to the network. The flap counts, however, are retained.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type number*
3. **dampening** [*half-life-period reuse-threshold*] [*suppress-threshold max-suppress* [*restart-penalty*]]
4. **no dampening**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure Terminal**	Enters global configuration mode.
Step 2	**interface** *type number*	Enters interface configuration mode and configures the specified interface.
Step 3	**dampening** [*half-life-period reuse-threshold*] [*suppress-threshold max-suppress* [*restart-penalty*]]	Enables interface dampening. • Entering the **dampening** command without any arguments enables interface dampening with default configuration parameters. • When manually configuring the timer for the *restart-penalty* argument, the values must be manually entered for all arguments.
Step 4	**no dampening**	Disables interface dampening.
Step 5	**end**	Exits interface configuration mode.

12.5.2 Verifying IP Event Dampening

Use the **show dampening interface** or **show interface dampening** commands to verify the configuration of the IP Event Dampening feature.

SUMMARY STEPS

1. **show ip interface** [**interface**]
2. **show dampening interface**
3. **show interface dampening**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show ip interface** [**interface**]	Displays all the configured dampening parameters including penalty information. You see the output only if you have the IP enabled on the interface.
Step 2	**show dampening interface**	Displays dampened interfaces.
Step 3	**show interface dampening**	Displays dampened interfaces on the local router.

12.5.3 Default Settings for IP Dampening Parameters

Table 15：Default values for IP Dampening Parameters

Parameters	Range	Default
Half-life	1-30	5
Reuse threshold	1-20000	800
Suppress threshold	1-20000	2000
Max suppress time	1-255 seconds	20 seconds
Apply restart penalty		False
Restart penalty	true / false	false

CHAPTER 13 ConfiguringIP TCP MSS

13.1 Information About IP TCP MSS

The IP TCP Maximum Segment Size (MSS) feature enables a switch to set a maximum segment size for all TCP connections that originate or terminate at a Inspur CN12900 Series switch. The MSS in a TCP header field is the maximum data size or payload that a host can send or receive in a single segment. By default, a Inspur CN12900 Series switch sets the MSS value to 536 bytes for IPv4 TCP connections and 1240 bytes for IPv6 TCP connections. This default value is set by the switch during the initial TCP connection establishment.

The switch from which the TCP connection originates will always set the MSS to the user-configured MSS or the difference between the route interface MTU and the protocol header, whichever is lower. Thus, Host A sends a SYN packet with the proposed MSS of 1460 bytes to Host B. After receiving the SYN packet with the proposed MSS, Host B sends a SYN-ACK packet to Host A, accepting the proposed MSS value for the TCP connection. Host A sends an ACK packet to Host B, setting the MSS value to 1460 for the TCP connection.

13.2 Licensing Requirements for IP TCP MSS

P TCP MSS requires no license. Any feature that is not included in a license package is bundled with the Inspur INOS-CN system images and is provided at no extra charge to you.

13.3 Default Settings for IP TCP MSS

Table 16 : Default Settings for IP TCP MSS

Parameter	Default Setting
IP TCP MSS	536 bytes for IPv4 TCP connections 1240 bytes for IPv6 TCP connections

13.4 Guidelines and Limitations for IP TCP MSS

If the MSS has to be set to a value that is more than 1460 bytes for IPv4 TCP connections, the corresponding MTU value should be set to the required MSS value plus 40 bytes. If the MSS has to be set to a value that is more than 1440 bytes for IPv6 TCP connections, the corresponding MTU value should be set to the required MSS value plus 60 bytes.

13.5 Configuring IP TCP MSS

13.5.1 Setting the MSS for TCP Connections

Before you begin

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **ip tcp mss** <*bytes*>
3. switch# **show ip tcp mss**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enter global configuration mode
Step 2	switch(config)# **ip tcp mss** <*bytes*>	Set a maximum segment size.
Step 3	switch# **show ip tcp mss**	Display the configured IP TCP MSS.

Example: Running Configuration

Example

This example shows a running configuration, followed by a verification command that displays the configured IP TCP MSS:

```
configure terminal
 ip tcp mss 5000
 Setting TCP MSS to 5000 bytes
switch# show ip tcp mss

TCP MSS value 5000 bytes
```

13.5.2 Removing a Set IP TCP MSS

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **no ip tcp mss**
3. switch# **show ip tcp mss**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enter global configuration mode
Step 2	switch(config)# **no ip tcp mss**	Remove the configured IP TCP MSS and set the IP TCP MSS to default values.
Step 3	switch# **show ip tcp mss**	Display the configured IP TCP MSS.

Example: Running Configuration

Example

This example shows a running configuration, followed by a verification command that displays the configured IP TCP MSS:

```
configure terminal
```

```
 no ip tcp mss 5000
 Setting default MSS value is 536 bytes
switch# show ip tcp mss

TCP MSS value 536 bytes
```

13.5.3 Example: Setting the MSS for TCP Connections

This example shows a setting the MSS for TCP connections:

```
configure terminal

ip tcp mss 2000
```

13.5.4 Example: Removing a Set IP TCP MSS

This example shows how to remove the MSS:

```
configure terminal

no ip tcp mss
```

13.6 Verifying IP TCP MSS

Table 17：Verifying IP TCP MSS

Command	Purpose
show ip tcp mss	Displays the set IP TCP MSS.

APPENDIX A
Configuring Layer 2 Data Center Interconnect

This section contains an example of how to configure a Layer 2 Data Center Interconnect (DCI) with the use of a Virtual Port-Channel (vPC).

Overview

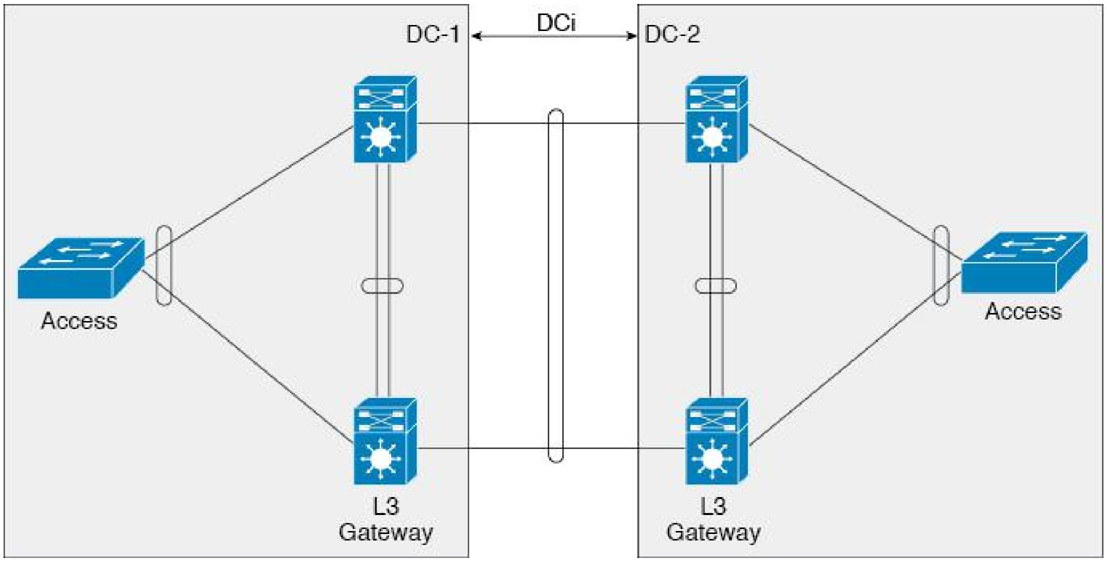
The purpose of a Data Center Interconnect (DCI) is to extend specific VLANs between different data centers, which offers Layer 2 adjacency for servers and Network Attached Storage (NAS) devices that are separated by large distances.

The vPC provides the benefit of STP isolation between the two sites (no Bridge Protocol Data Unit (BPDU) across the DCI vPC) This means that an outage in the data center is not propagated to the remote data center because redundant links are still provided between the data centers.

Example of Layer 2 Data Center Interconnect

The following is an example configuration of a Layer 2 Data Center Interconnect (DCI) with use of vPC. The example allows for First Hop Redundancy Protocol (FHRP) isolation.

Figure 38： Dual Layer 2/Layer 3 POD Interconnect



In this example, the Layer 3 (L3) gateway is configured on the same vPC pair and acts as the DCI. In order to isolate the Hot Standby Routing Protocol (HSRP), you must configure a Port Access Control List (PACL) on the DCI port-channel and disable HSRP Gratuitous Address Resolution Protocols (ARPs) (GARPs) on the Switched Virtual Interfaces (SVIs) for the VLANs that move across the DCI.

```
ip access-list DENY_HSRP_IP
```

```
  10 deny udp any 224.0.0.2/32 eq 1985
  20 deny udp any 224.0.0.102/32 eq 1985
  30 permit ip any any
interface <DCI-Port-Channel>
  ip port access-group DENY_HSRP_IP in
interface Vlan <x>
  no ip arp gratuitous hsrp duplicate
```

APPENDIX B
IETF RFCs supported by Inspur INOS-CN Interfaces

This appendix lists the IETF RFCs for interfaces supported by Inspur INOS-CN.

IPv6 RFCs

RFCs	**Title**
RFC 1981	*Path MTU Discovery for IP version 6*
RFC 2373	*IP Version 6 Addressing Architecture*
RFC 2374	*An Aggregatable Global Unicast Address Format*
RFC 2460	*Internet Protocol, Version 6 (IPv6) Specification*
RFC 2462	*IPv6 Stateless Address Autoconfiguration*
RFC 2464	*Transmission of IPv6 Packets over Ethernet Networks*
RFC 2467	*Transmission of IPv6 Packets over FDDI Networks*
RFC 2472	*IP Version 6 over PPP*
RFC 2492	*IPv6 over ATM Networks*
RFC 2590	*Transmission of IPv6 Packets over Frame Relay Networks Specification*
RFC 3021	*Using 31-Bit Prefixes on IPv4 Point-to-Point Links*
RFC 3152	*Delegation of IP6.ARPA*
RFC 3162	*RADIUS and IPv6*
RFC 3513	*Internet Protocol Version 6 (IPv6) Addressing Architecture*
RFC 3596	*DNS Extensions to Support IP version 6*
RFC 4193	*Unique Local IPv6 Unicast Addresses*

APPENDIX C
Configuration Limits for Inspur INOS-CN Interfaces

The configuration limits are documented in the *Inspur CN12900 Series INOS-CN Verified Scalability Guide*.