

LAN Switching Configuration Guide

First Published: 2026-06-03

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED “AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

All printed copies and duplicate soft copies of this document are considered uncontrolled. See the current online version for the latest version.

Cisco has more than 200 offices worldwide. Addresses and phone numbers are listed on the Cisco website at www.cisco.com/go/offices.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: https://www.cisco.com/c/en/us/about/legal/trademarks.html. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1721R)

© 2026 Cisco Systems, Inc. All rights reserved.



CONTENTS

CHAPTER 1 **Read Me First** 1

Short Description 2

CHAPTER 2 **Configuring ERSPAN** 3

Restrictions for Configuring ERSPAN 3

Information About Configuring ERSPAN 4

ERSPAN Overview 4

ERSPAN Sources 5

ERSPAN Destination Ports 6

Using ERSPAN as Local SPAN 6

ERSPAN Support on WAN Interface 7

ERSPAN Dummy MAC Address Rewrite 7

ERSPAN IP Access Control Lists 7

How to Configure ERSPAN 7

Configuring an ERSPAN Source Session 7

Configuring an ERSPAN Destination Session 11

Configuring ERSPAN Dummy MAC Address Rewrite 13

Configuration Examples for ERSPAN 14

Example: Configuring an ERSPAN Source Session 14

Example: Configuring an ERSPAN Source Session on a WAN Interface 15

Example: Configuring an ERSPAN Destination Session 15

Example: Configuring an ERSPAN as a Local SPAN 15

Example: Configuring ERSPAN Dummy MAC Address Rewrite 15

Example: Configuring UDF-Based ERSPAN 16

Additional References for Configuring ERSPAN 16

Feature Information for Configuring ERSPAN 17

CHAPTER 3 **Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation** 19

Restrictions for Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation 19

Information About Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation 19

Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation 19

How to Configure Routing Between VLANs with IEEE 802.1Q Encapsulation 20

Configuring IP Routing over IEEE 802.1Q 20

Enabling IP Routing 20

Defining the VLAN Encapsulation Format 21

Assigning an IP Address to Network Interface 22

Monitoring and Maintaining VLAN Subinterfaces 23

Configuration Examples for Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation 24

Configuring IP Routing over IEEE 802.1Q Example 24

Additional References 24

Feature Information for Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation 25

CHAPTER 4 **IEEE 802.1Q-in-Q VLAN Tag Termination** 27

Information About IEEE 802.1Q-in-Q VLAN Tag Termination 27

IEEE 802.1Q-in-Q VLAN Tag Termination on Subinterfaces 27

Unambiguous and Ambiguous Subinterfaces 28

IEEE802.1ad Support in Port-channels and Subinterfaces 29

Restrictions for IEEE802.1ad Support in Port-channels and Subinterfaces 29

How to Configure IEEE 802.1Q-in-Q VLAN Tag Termination 29

Configuring the Interfaces for IEEE 802.1Q-in-Q VLAN Tag Termination 29

Verifying the IEEE 802.1Q-in-Q VLAN Tag Termination 31

Configuring IEEE 802.1ad in Port-channels and Subinterfaces 33

Configuration Examples for IEEE 802.1Q-in-Q VLAN Tag Termination 34

Configuring any Keyword on Subinterfaces for IEEE 802.1Q-in-Q VLAN Tag Termination Example 34

Additional References 36

Feature Information for IEEE 802.1Q-in-Q VLAN Tag Termination 37

CHAPTER 5 **VLAN Mapping to Gigabit EtherChannel Member Links** 39

Prerequisites for VLAN Mapping to GEC Member Links 39

Restrictions for VLAN Mapping to GEC Member Links 39
Information About VLAN Mapping of GEC Member Links 40
VLAN-Manual Load Balancing 40
VLAN-to-Port Channel Member Link Mapping 41
VLAN Primary and Secondary Link Association 41
Adding Channel Member Links 43
Deleting Member Links 43
Port Channel Link Down Notification 43
Port Channel Link Up Notification 43
Disabling Load Balancing on the EtherChannel 44
Removing a Member Link from the EtherChannel 44
How to Configure VLAN Mapping to GEC Links 44
Configuring VLAN-Based Manual Load Balancing 44
Troubleshooting Tips 46
Configuration Examples for VLAN Mapping to GEC Member Links 46
Example: Configuring VLAN Manual Load Balancing 46
Example: Troubleshooting 48
Additional References 49
Feature Information for VLAN Mapping to GEC Member Links 49

CHAPTER 6 **Configuring Routing Between VLANs** 51
Information About Routing Between VLANs 51
Virtual Local Area Network Definition 51
LAN Segmentation 52
Security 52
Broadcast Control 52
VLAN Performance 53
Network Management 53
Network Monitoring Using SNMP 53
Communication Between VLANs 53
Relaying Function 53
Native VLAN 55
PVST+ 56
Ingress and Egress Rules 57

Integrated Routing and Bridging 57
VLAN Colors 57
Implementing VLANS 58
Communication Between VLANs 58
Inter-Switch Link Protocol 58
IEEE 802.10 Protocol 58
IEEE 802.1Q Protocol 59
ATM LANE Protocol 59
ATM LANE Fast Simple Server Replication Protocol 59
VLAN Interoperability 60
Inter-VLAN Communications 60
VLAN Translation 60
Designing Switched VLANs 61
Frame Tagging in ISL 61
IEEE 802.1Q-in-Q VLAN Tag Termination on Subinterfaces 62
Cisco 10000 Series Internet Router Application 63
Security ACL Application on the Cisco 10000 Series Internet Router 64
Unambiguous and Ambiguous Subinterfaces 64
How to Configure Routing Between VLANS 65
Configuring a VLAN Range 65
Restrictions 65
Configuring a Range of VLAN Subinterfaces 65
Configuring Routing Between VLANs with Inter-Switch Link Encapsulation 67
Configuring AppleTalk Routing over ISL 67
Configuring Banyan VINES Routing over ISL 68
Configuring DECnet Routing over ISL 70
Configuring the Hot Standby Router Protocol over ISL 71
Configuring IP Routing over TRISL 73
Configuring IPX Routing on 802.10 VLANs over ISL 75
Configuring IPX Routing over TRISL 76
Configuring VIP Distributed Switching over ISL 78
Configuring XNS Routing over ISL 80
Configuring CLNS Routing over ISL 81
Configuring IS-IS Routing over ISL 83

Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation 84
Prerequisites 84
Restrictions 84
Configuring AppleTalk Routing over IEEE 802.1Q 85
Configuring IP Routing over IEEE 802.1Q 86
Configuring IPX Routing over IEEE 802.1Q 87
Configuring a VLAN for a Bridge Group with Default VLAN1 88
Configuring a VLAN for a Bridge Group as a Native VLAN 89
Configuring IEEE 802.1Q-in-Q VLAN Tag Termination 91
Configuring EtherType Field for Outer VLAN Tag Termination 91
Configuring the Q-in-Q Subinterface 92
Verifying the IEEE 802.1Q-in-Q VLAN Tag Termination 94
Monitoring and Maintaining VLAN Subinterfaces 97
Monitoring and Maintaining VLAN Subinterfaces Example 97
Configuration Examples for Configuring Routing Between VLANs 98
Single Range Configuration Example 98
ISL Encapsulation Configuration Examples 99
AppleTalk Routing over ISL Configuration Example 99
Banyan VINES Routing over ISL Configuration Example 100
DECnet Routing over ISL Configuration Example 100
HSRP over ISL Configuration Example 100
IP Routing with RIF Between TrBRF VLANs Example 102
IP Routing Between a TRISL VLAN and an Ethernet ISL VLAN Example 103
IPX Routing over ISL Configuration Example 103
IPX Routing on FDDI Interfaces with SDE Example 105
Routing with RIF Between a TRISL VLAN and a Token Ring Interface Example 105
VIP Distributed Switching over ISL Configuration Example 106
XNS Routing over ISL Configuration Example 107
CLNS Routing over ISL Configuration Example 107
IS-IS Routing over ISL Configuration Example 107
Routing IEEE 802.10 Configuration Example 108
IEEE 802.1Q Encapsulation Configuration Examples 109
Configuring AppleTalk over IEEE 802.1Q Example 109
Configuring IP Routing over IEEE 802.1Q Example 109

Configuring IPX Routing over IEEE 802.1Q Example 109
VLAN 100 for Bridge Group 1 with Default VLAN1 Example 109
VLAN 20 for Bridge Group 1 with Native VLAN Example 109
VLAN ISL or IEEE 802.1Q Routing Example 110
VLAN IEEE 802.1Q Bridging Example 111
VLAN IEEE 802.1Q IRB Example 111
Configuring IEEE 802.1Q-in-Q VLAN Tag Termination Example 112
Additional References 114
Feature Information for Routing Between VLANs 115

CHAPTER 7 **EtherChannel Flow-Based Limited 1 1 Redundancy** 119
Restrictions for EtherChannel Flow-based Limited 1:1 Redundancy 119
Information About EtherChannel Flow-Based Limited 1 1 Redundancy 120
EtherChannel Flow-Based Limited 1 1 Redundancy 120
How to Configure EtherChannel Flow-Based Limited 1 1 Redundancy 120
Configuring EtherChannel Flow-Based Limited 1 1 Redundancy with Fast-Switchover 120
Setting the Switchover Rate with Carrier Delay 123
Verifying EtherChannel Flow-Based Limited 1 1 Redundancy 123
Configuration Examples for EtherChannel Flow-Based Limited1 1 Redundancy 125
EtherChannel 1 1 Active Standby Example 125
Setting Priority for 1 1 Redundancy Using LACP Example 125
Additional References 126
Feature Information for EtherChannel Flow-based Limited 1 1 Redundancy 127

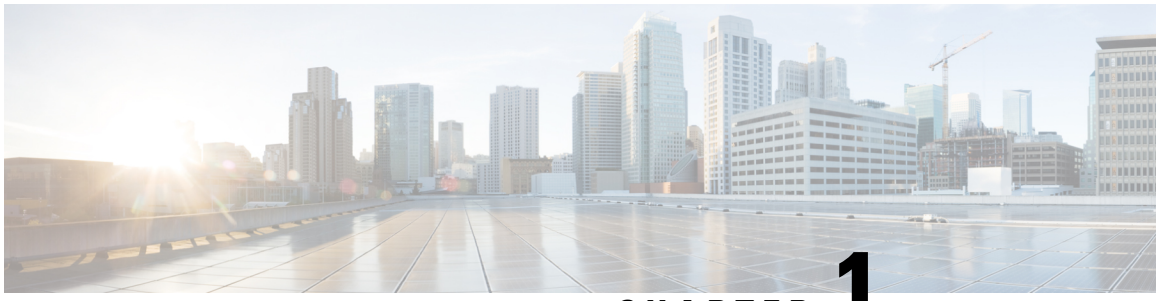
CHAPTER 8 **Flow-Based per Port-Channel Load Balancing** 129
Restrictions for Flow-Based per Port-Channel Load Balancing 129
Information About Flow-Based per Port-Channel Load Balancing 129
Flow-Based Load Balancing 129
Buckets for Flow-Based Load Balancing 130
Load Balancing on Port Channels 131
How to Enable Flow-Based per Port-Channel Load Balancing 132
Configuring Load Balancing on a Port Channel 132
Verifying Load-Balancing Configuration on a GEC Interface 133
Configuration Examples for Flow-Based per Port-Channel Load Balancing 135

Flow-Based Load Balancing Example 135
Information About Five-Tuple Hash Support for GEC Flow-based Load Balancing 136
Restrictions for Five-Tuple Hash Support for GEC Flow-based Load Balancing 136
Configuring Five-Tuple Hash Support for GEC Flow-based Load Balancing 136
Additional References 136
Feature Information for Flow-Based per Port-Channel Load Balancing 138

CHAPTER 9 **VLANs over IP Unnumbered SubInterfaces** 139
Finding Feature Information 139
Prerequisites for VLANs over IP Unnumbered Subinterfaces 139
Restrictions for VLANs over IP Unnumbered Subinterfaces 140
Information About VLANs over IP Unnumbered Subinterfaces 140
Support for VLANs over IP Unnumbered Subinterfaces 140
DHCP Option 82 141
Benefits of VLANs over IP Unnumbered Subinterfaces 142
How to Configure VLANs over IP Unnumbered Subinterfaces 142
Configuring IP Unnumbered Interface Support on an Ethernet VLAN Subinterface 142
Configuring IP Unnumbered Interface Support on a Range of Ethernet VLAN Subinterfaces 143
Configuration Examples for VLANs over IP Unnumbered Subinterfaces 145
Example: VLAN Configuration on a Single IP Unnumbered Subinterface 145
Example: VLAN Configuration on a Range of IP Unnumbered Subinterfaces 145
Additional References for VLANs over IP Unnumbered Subinterfaces 145
Feature Information for VLANs over IP Unnumbered Subinterfaces 146

CHAPTER 10 **Spanning Tree Protocol** 147
Finding Feature Information 147
Information About Spanning Tree Protocol 147
Using the Spanning Tree Protocol with the EtherSwitch Network Module 147
Spanning Tree Port States 148
Default Spanning Tree Configuration 150
Bridge Protocol Data Units 151
STP Timers 154
Spanning Tree Port Priority 154
Spanning Tree Port Cost 154

Spanning Tree Root Bridge 155
How to Configure Spanning Tree Protocol 156
Enabling Spanning Tree Protocol 156
Configuring the Bridge Priority of a VLAN 157
Configuring STP Timers 158
Configuring Hello Time 158
Configuring the Forward Delay Time for a VLAN 158
Configuring the Maximum Aging Time for a VLAN 159
Configuring Spanning Tree Port Priority 160
Configuring Spanning Tree Port Cost 161
Configuring Spanning Tree Root Bridge 162
Verifying Spanning Tree on a VLAN 163
Configuration Examples for Spanning Tree Protocol 164
Example: Enabling Spanning Tree Protocol 164
Example: Configuring the Bridge Priority of a VLAN 165
Example: Configuring STP Timers 165
Example: Configuring Hello Time 165
Example: Configuring the Forward Delay Time for a VLAN 165
Example: Configuring the Maximum Aging Time for a VLAN 165
Example: Configuring Spanning Tree Port Priority 165
Example: Configuring Spanning Tree Port Cost 166
Example: Configuring Spanning Tree Root Bridge 167
Additional References 167
Feature Information for Spanning Tree Protocol 167



CHAPTER 1

Read Me First

Important Information

Note For CUBE feature support information in Cisco IOS XE Bengaluru 17.6.1a and later releases, see Cisco Unified Border Element IOS-XE Configuration Guide.

Note The documentation set for this product strives to use bias-free language. For purposes of this documentation set, bias-free is defined as language that does not imply discrimination based on age, disability, gender, racial identity, ethnic identity, sexual orientation, socioeconomic status, and intersectionality. Exceptions may be present in the documentation due to language that is hardcoded in the user interfaces of the product software, language used based on RFP documentation, or language that is used by a referenced third-party product.

Feature Information

Use Cisco Feature Navigator to find information about feature support, platform support, and Cisco software image support. An account on Cisco.com is not required.

Related References

- Cisco IOS Command References, All Releases

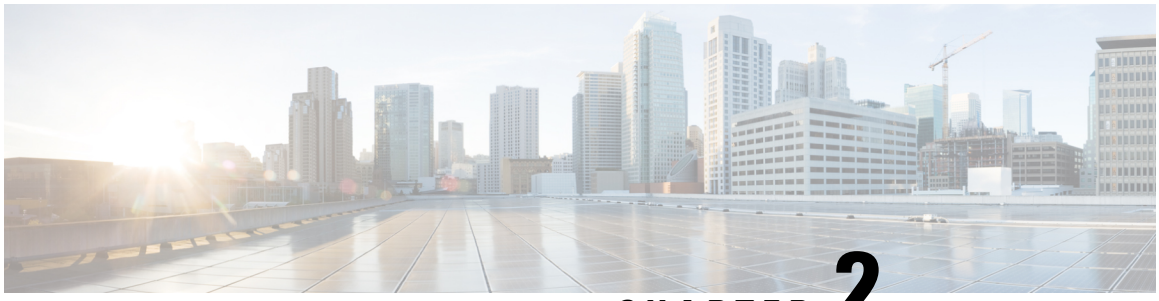
Obtaining Documentation and Submitting a Service Request

- To receive timely, relevant information from Cisco, sign up at Cisco Profile Manager.
- To get the business results you're looking for with the technologies that matter, visit Cisco Services.
- To submit a service request, visit Cisco Support.
- To discover and browse secure, validated enterprise-class apps, products, solutions and services, visit Cisco Marketplace.
- To obtain general networking, training, and certification titles, visit Cisco Press.
- To find warranty information for a specific product or product family, access Cisco Warranty Finder.

• Short Description, on page 2

Short Description

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: https://www.cisco.com/c/en/us/about/legal/trademarks.html. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1721R)



CHAPTER 2

Configuring ERSPAN

This module describes how to configure Encapsulated Remote Switched Port Analyzer (ERSPAN). The Cisco ERSPAN feature allows you to monitor traffic on one or more ports or VLANs and send the monitored traffic to one or more destination ports.

Note The ERSPAN feature is not supported on Layer 2 switching interfaces.

- Restrictions for Configuring ERSPAN, on page 3
- Information About Configuring ERSPAN, on page 4
- How to Configure ERSPAN, on page 7
- Configuration Examples for ERSPAN, on page 14
- Additional References for Configuring ERSPAN, on page 16
- Feature Information for Configuring ERSPAN , on page 17

Restrictions for Configuring ERSPAN

- The maximum number of allowed ERSPAN sessions on a Cisco ASR 1000 Series Router is 1024. A Cisco ASR 1000 Series Router can be used as an ERSPAN source device on which only source sessions are configured, an ERSPAN destination device on which only destination sessions are configured, or an ERSPAN source and destination device on which both source and destination sessions are configured. However, total number of sessions must not exceed 1024.
- The maximum number of available ports for each ERSPAN session is 128.
- ERSPAN on Cisco ASR 1000 Series Routers supports only Fast Ethernet, Gigabit Ethernet, TenGigabit Ethernet, and port-channel interfaces as source ports for a source session.
- ERSPAN on Cisco ASR 1000 Series Routers supports only Layer 3 interfaces. Ethernet interfaces are not supported on ERSPAN when configured as Layer 2 interfaces.
- ERSPAN users on Cisco ASR 1000 Series Routers can configure a list of ports as a source or a list of VLANs as a source, but cannot configure both for a given session.
- When a session is configured through the ERSPAN configuration CLI, the session ID and the session type cannot be changed. To change them, you must first use the **no** form of the configuration command to remove the session and then reconfigure the session.

- The **monitor session** *span-session-number* **type local** command is not supported on Cisco ASR 1000 Series Routers.
- The filter VLAN option is not functional in an ERSPAN monitoring session on WAN interfaces.

Information About Configuring ERSPAN

ERSPAN Overview

The Cisco ERSPAN feature allows you to monitor traffic on one or more ports or more VLANs, and send the monitored traffic to one or more destination ports. ERSPAN sends traffic to a network analyzer such as a Switch Probe device or other Remote Monitoring (RMON) probe. ERSPAN supports source ports, source VLANs, and destination ports on different routers, which provides remote monitoring of multiple routers across a network (see the figure below).

On a Cisco ASR 1000 Series Router, ERSPAN supports encapsulated packets of up to 9180 bytes. The default ERSPAN maximum transmission unit (MTU) size is 1500 bytes. If the ERSPAN payload length, which comprises the encapsulated IPv4 header, generic routing encapsulation (GRE) header, ERSPAN header, and the original packet, exceeds the ERSPAN MTU size, the replicated packet is truncated to the default ERSPAN MTU size.

ERSPAN consists of an ERSPAN source session, routable ERSPAN GRE encapsulated traffic, and an ERSPAN destination session.

You can configure an ERSPAN source session, an ERSPAN destination session, or both on a Cisco ASR 1000 Series Router. A device that has only an ERSPAN source session configured is called an ERSPAN source device, and a device that has only an ERSPAN destination session configured is called an ERSPAN termination device. A Cisco ASR 1000 Series Router can act as both an ERSPAN source device and an ERSPAN termination device. You can terminate an ERSPAN session with a destination session on the same Cisco ASR 1000 Series Router.

An ERSPAN source session is defined by the following parameters:

- A session ID
- List of source ports or source VLANs to be monitored by the session
- The destination and origin IP addresses, which are used as the destination and source IP addresses of the GRE envelope for the captured traffic, respectively
- ERSPAN flow ID
- Optional attributes, such as, IP type of service (TOS) and IP Time to Live (TTL), related to the GRE envelope

An ERSPAN destination session is defined by the following:

- Session ID
- Destination ports
- Source IP address, which is the same as the destination IP address of the corresponding source session
- ERSPAN flow ID, which is used to match the destination session with the source session

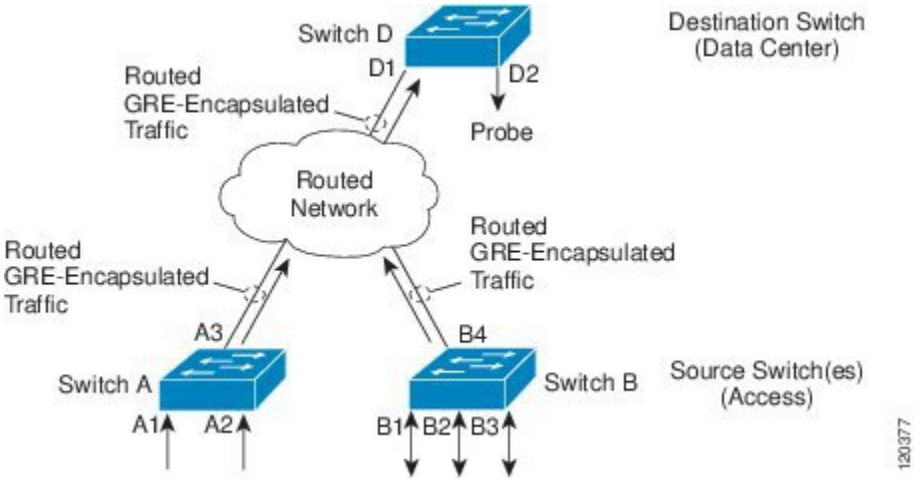
ERSPAN source sessions do not copy ERSPAN GRE-encapsulated traffic from source ports. Each ERSPAN source session can have either ports or VLANs as sources, but not both.

The ERSPAN source sessions copy traffic from the source ports or source VLANs and forwards the traffic using routable GRE-encapsulated packets to the ERSPAN destination session. The ERSPAN destination session switches the traffic to the destination ports.

Note When there is a change in the routing topology, the routing path for the ERSPAN destination could also change. If the egress bandwidth is not sufficient for ERSPAN traffic, the excess traffic is dropped.

If the specific route for the ERSPAN destination is not available in the routing table and there is a default route set, the ERSPAN traffic is sent via the default route.

Figure 1: ERSPAN Configuration



Monitored Traffic

For a source port or a source VLAN, the ERSPAN can monitor the ingress, egress, or both ingress and egress traffic. By default, ERSPAN monitors all traffic, including multicast and Bridge Protocol Data Unit (BPDU) frames.

ERSPAN Sources

The Cisco ERSPAN feature supports the following sources:

- Source ports—A source port that is monitored for traffic analysis. Source ports in any VLAN can be configured and trunk ports can be configured as source ports along with nontrunk source ports.
- Source VLANs—A VLAN that is monitored for traffic analysis.

The following tunnel interfaces are supported as source ports for a ERSPAN source session:

- GRE
- IPinIP
- IPv6
- IPv6 over IP tunnel

- Multipoint GRE (mGRE)
- Secure Virtual Tunnel Interfaces (SVTI)



Note SVTI and IPinIP tunnel interfaces support the monitoring of both IPsec-protected and non-IPsec-protected tunnel packets. Monitoring of tunnel packets allows you to see the clear-text tunnel packet after IPsec decryption if that tunnel is IPsec protected.

The following limitations apply to the enhancements introduced in Cisco IOS XE Release 3.4S:

- Monitoring of non-IPsec-protected tunnel packets is supported on IPv6 and IPv6 over IP tunnel interfaces.
- The enhancements apply only to ERSPAN source sessions, not to ERSPAN destination sessions.

ERSPAN has the following behavior in Cisco IOS XE Release 3.4S:

- The tunnel interface is removed from the ERSPAN database at all levels when the tunnel interface is deleted. If you want to create the same tunnel again, you must manually configure it in source monitor sessions to keep monitoring the tunnel traffic.
- The Layer 2 Ethernet header is generated with both source and destination MAC addresses set to zero.

In Cisco IOS XE Release 3.5S, support was added for the following types of WAN interfaces as source ports for a source session:

- Serial (T1/E1, T3/E3, DS0)
- Packet over SONET (POS) (OC3, OC12)
- Multilink PPP
- The **multilink**, **pos**, and **serial** keywords were added to the **source interface** command.

ERSPAN Destination Ports

A destination port is a Layer 2 or Layer 3 LAN port to which ERSPAN sends traffic for analysis.

When you configure a port as a destination port, it can no longer receive any traffic and, the port is dedicated for use only by the ERSPAN feature. An ERSPAN destination port does not forward any traffic except that required for the ERSPAN session. You can configure trunk ports as destination ports, which allows destination trunk ports to transmit encapsulated traffic.

Using ERSPAN as Local SPAN

To use ERSPAN to monitor traffic through one or more ports or VLANs, you must create an ERSPAN source and ERSPAN destination sessions.

You can create the two sessions either on the same router or on different routers. If the two sessions are created on two different routers, the monitoring traffic will be forwarded from the source to the destination by ERSPAN. However, if the two sessions are created on the same router, data flow takes place inside the router, which is similar to that in local SPAN.

The following factors are applicable while using ERSPAN as a local SPAN:

- Both sessions have the same ERSPAN ID.

- Both sessions have the same IP address. This IP address is the router's own IP address; that is, the loopback IP address or the IP address configured on any port.

ERSPAN Support on WAN Interface

In Cisco IOS Release 3.5S an ERSPAN source on WAN is added to allow monitoring of traffic on WAN interfaces. ERSPAN replicates the original frame and encapsulates the replicated frame inside an IP or GRE packet by adding Fabric Interface ASIC (FIA) entries on the WAN interface. The frame header of the replicated packet is modified for capturing. After encapsulation, ERSPAN sends the IP or GRE packet through an IP network to a device on the network. This device sends the original frame to an analyzing device that is directly connected to the network device.

ERSPAN Dummy MAC Address Rewrite

ERSPAN dummy MAC address rewrite supports customized MAC value for WAN interface and tunnel interface. It also allows you to monitor the traffic going through WAN interface.

ERSPAN IP Access Control Lists

From Cisco IOS XE Everest 16.4.1 release, ERSPAN has been enhanced to better monitor packets and reduce network traffic. This enhancement supports ACL on ERSPAN source session to filter only specific IP traffic according to the ACL, and is supported on the IOS XE platform. Both IPv4 and IPv6 traffic can be monitored by associating an ACL with the ERSPAN session. The ERSPAN session can associate only one IP ACL entry with its name.

How to Configure ERSPAN

ERSPAN uses separate source and destination sessions. You configure the source and destination sessions on either the same router or on different routers.

Configuring an ERSPAN Source Session

The ERSPAN source session defines the session configuration parameters and the ports or VLANs to be monitored.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-type interface-number*
4. **plim ethernet vlan filter disable**
5. **monitor session** *span-session-number* **type erspan-source**
6. **description** *string*
7. [**no**] **header-type 3**
8. **source interface** *interface-name interface-number*
9. **source vlan** {*id-single* | *id-list* | *id-range* | *id-mixed*} [**rx** | **tx** | **both**]

10. **filter vlan** {*id-single* | *id-list* | *id-range* | *id-mixed*}
11. **filter access-group** *acl-filter*
12. **destination**
13. **erspan-id** *erspan-flow-id*
14. **ip address** *ip-address*
15. **ip prec** *prec-value*
16. **ip dscp** *dscp-value*
17. **ip ttl** *ttl-value*
18. **mtu** *mtu-size*
19. **origin ip address** *ip-address* [**force**]
20. **vrf** *vrf-id*
21. **no shutdown**
22. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-type interface-number* **Example:** `Device(config)# interface GigabitEthernet1/0/1`	Specifies the interface on which ERSPAN source session is configured.
Step 4	**plim ethernet vlan filter disable** **Example:** `Device(config-if)# plim ethernet vlan filter disable`	(Optional) Disables the VLAN filtering option for Ethernet interfaces. Use this command if you are using the **vlan filter** command or if the source interface is using dot1q encapsulation.
Step 5	**monitor session** *span-session-number* **type erspan-source** **Example:** `Device(config)# monitor session 1 type erspan-source`	Defines an ERSPAN source session using the session ID and the session type, and enters ERSPAN monitor source session configuration mode. • The *span-session-number* argument range is from 1 to 1024. The same session number cannot be used more than once. • The session IDs for source sessions or destination sessions are in the same global ID space, so each session ID is globally unique for both session types.

	Command or Action	Purpose
		• The session ID (configured by the *span-session-number* argument) and the session type (configured by the **erspan-source** keyword) cannot be changed once entered. Use the **no** form of this command to remove the session and then re-create the session, with a new session ID or a new session type.
Step 6	**description** *string* **Example:** `Device(config-mon-erspan-src)# description source1`	(Optional) Describes the ERSPAN source session. • The *string* argument can be up to 240 characters and cannot contain special characters or spaces.
Step 7	[**no**] **header-type 3** **Example:** `Device(config-mon-erspan-src)# header-type 3`	Configures a switch to ERSPAN header type III.
Step 8	**source interface** *interface-name interface-number* **Example:** `Device(config-mon-erspan-src)# source interface GigabitEthernet1/0/1 rx`	Configures more than one WAN interface in a single ERSPAN session.
Step 9	**source vlan** {*id-single* | *id-list* | *id-range* | *id-mixed*} [**rx** | **tx** | **both**] **Example:** `Device(config-mon-erspan-src)# source vlan 1`	(Optional) Associates the ERSPAN source session number with the VLANs, and selects the traffic direction to be monitored. • You cannot include source VLANs and filter VLANs in the same session. You can either include source VLANs or filter VLANs, but not both at the same time.
Step 10	**filter vlan** {*id-single* | *id-list* | *id-range* | *id-mixed*} **Example:** `Device(config-mon-erspan-src)# filter vlan 1`	(Optional) Configures source VLAN filtering when the ERSPAN source is a trunk port. • You cannot include source VLANs and filter VLANs in the same session. You can have source VLANs or filter VLANs, but not both at the same time.
Step 11	**filter access-group** *acl-filter* **Example:** `Device(config-mon-erspan-src)# filter access-group ACL1`	(Optional) Associates an ACL with the ERSPAN session. • Use the **no filter access-group** *acl-filter* command to detach the ACL from the ERSPAN session. • Only ACL name is supported to associate to the ERSPAN source session. If the ACL does not exist or if there is no entry defined in the access control list, the ACL name is not attached to the ERSPAN source session. • When the ERSPAN source session is active, you cannot detach the ACL from the ERSPAN source

	Command or Action	Purpose
		session. The source session must be shut down before detaching the ACL. After the session shutdown, you must exit the session for the **shutdown** command to execute, and then re-enter the session to detach the ACL.
Step 12	**destination** **Example:** `Device(config-mon-erspan-src)# destination`	Enters ERSPAN source session destination configuration mode.
Step 13	**erspan-id** *erspan-flow-id* **Example:** `Device(config-mon-erspan-src-dst)# erspan-id 100`	Configures the ID used by the source and destination sessions to identify the ERSPAN traffic, which must also be entered in the ERSPAN destination session configuration.
Step 14	**ip address** *ip-address* **Example:** `Device(config-mon-erspan-src-dst)# ip address 10.10.0.1`	Configures the IP address that is used as the destination of the ERSPAN traffic.
Step 15	**ip prec** *prec-value* **Example:** `Device(config-mon-erspan-src-dst)# ip prec 5`	(Optional) Configures the IP precedence value of the packets in the ERSPAN traffic. • You can optionally use either the **ip prec** command or the **ip dscp** command, but not both.
Step 16	**ip dscp** *dscp-value* **Example:** `Device(config-mon-erspan-src-dst)# ip dscp 10`	(Optional) Enables the use of IP differentiated services code point (DSCP) for packets that originate from a circuit emulation (CEM) channel. • You can optionally use either the **ip prec** command or the **ip dscp** command, but not both.
Step 17	**ip ttl** *ttl-value* **Example:** `Device(config-mon-erspan-src-dst)# ip ttl 32`	(Optional) Configures the IP TTL value of the packets in the ERSPAN traffic.
Step 18	**mtu** *mtu-size* **Example:** `Device(config-mon-erspan-src-dst)# mtu 1500`	Configures the maximum transmission unit (MTU) size, in bytes, for ERSPAN encapsulation. • Valid values are from 64 to 9180. The default value is 1500.
Step 19	**origin ip address** *ip-address* [**force**] **Example:** `Device(config-mon-erspan-src-dst)# origin ip address 10.10.0.1`	Configures the IP address used as the source of the ERSPAN traffic.

	Command or Action	Purpose
Step 20	**vrf** *vrf-id* **Example:** `Device(config-mon-erspan-src-dst)# vrf 1`	(Optional) Configures the VRF name to use instead of the global routing table.
Step 21	**no shutdown** **Example:** `Device(config-mon-erspan-src-dst)# no shutdown`	Enables the configured sessions on an interface.
Step 22	**end** **Example:** `Device(config-mon-erspan-src-dst)# end`	Exits ERSPAN source session destination configuration mode, and returns to privileged EXEC mode.

Configuring an ERSPAN Destination Session

Perform this task to configure an Encapsulated Remote Switched Port Analyzer (ERSPAN) destination session. The ERSPAN destination session defines the session configuration parameters and the ports that will receive the monitored traffic.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **monitor session** *session-number* **type erspan-destination**
4. **description** *string*
5. **destination interface** {**gigabitethernet** | **port-channel**} [*interface-number*]
6. **source**
7. **erspan-id** *erspan-flow-id*
8. **ip address** *ip-address* [**force**]
9. **vrf** *vrf-id*
10. **no shutdown**
11. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**monitor session** *session-number* **type erspan-destination** **Example:** `Device(config)# monitor session 1 type erspan-destination`	Defines an ERSPAN destination session using the session ID and the session type, and enters in ERSPAN monitor destination session configuration mode. • The *session-number* argument range is from 1 to 1024. The session number must be unique and cannot be used more than once. • The session IDs for source sessions or destination sessions are in the same global ID space, so each session ID is globally unique for both session types. • The session ID (configured by the *session-number* argument) and the session type (configured by the **erspan-destination**) cannot be changed once entered. Use the **no** form of this command to remove the session, and then recreate the session with a new session ID or a new session type.
Step 4	**description** *string* **Example:** `Device(config-mon-erspan-dst)# description source1`	(Optional) Describes the ERSPAN destination session. • The *string* argument can be up to 240 characters in length and cannot contain special characters or spaces.
Step 5	**destination interface** {**gigabitethernet** | **port-channel**} [*interface-number*] **Example:** `Device(config-mon-erspan-dst)# destination interface GigabitEthernet1/0/1`	Associates the ERSPAN destination session number with the source ports, and selects the traffic direction to be monitored.
Step 6	**source** **Example:** `Device(config-mon-erspan-dst)# source`	Enters ERSPAN destination session source configuration mode.
Step 7	**erspan-id** *erspan-flow-id* **Example:** `Device(config-mon-erspan-dst-src)# erspan-id 100`	Configures the ID used by the source and destination sessions to identify the ERSPAN traffic, which must also be entered in the ERSPAN source session configuration.
Step 8	**ip address** *ip-address* [**force**] **Example:** `Device(config-mon-erspan-dst-src)# ip address 10.10.0.1`	Configures the IP address that is used as the source of the ERSPAN traffic. • The **ip address** *ip-address* **force** command changes the source IP address for all ERSPAN destination sessions.
Step 9	**vrf** *vrf-id* **Example:**	(Optional) Configures the VRF name to use instead of the global routing table.

	Command or Action	Purpose
	`Device(config-mon-erspan-dst-src)# vrf 1`	
Step 10	**no shutdown** **Example:** `Device(config-mon-erspan-dst-src)# no shutdown`	Enables the configured sessions on an interface.
Step 11	**end** **Example:** `Device(config-mon-erspan-dst-src)# end`	Exits ERSPAN destination session source configuration mode, and returns to privileged EXEC mode.

Configuring ERSPAN Dummy MAC Address Rewrite

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **monitor session** *span-session-number* **type erspan-source**
4. **source interface** *interface-name interface-number*
5. **s-mac** *address*
6. **d-mac** *address*
7. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**monitor session** *span-session-number* **type erspan-source** **Example:** `Device(config)# monitor session 100 type erspan-source`	Defines an ERSPAN source session using the session ID and the session type, and enters ERSPAN monitor source session configuration mode. • The *span-session-number* argument range is from 1 to 1024. The same session number cannot be used more than once.

	Command or Action	Purpose
		• The session IDs for source sessions or destination sessions are in the same global ID space, so each session ID is globally unique for both session types. • The session ID (configured by the *span-session-number* argument) and the session type (configured by the **erspan-source** keyword) cannot be changed once entered. Use the **no** form of this command to remove the session and then re-create the session, with a new session ID or a new session type.
Step 4	**source interface** *interface-name interface-number* **Example:** `Device(config-mon-erspan-src)# source interface GigabitEthernet1/0/1 rx`	Configures more than one WAN interface in a single ERSPAN session.
Step 5	**s-mac** *address* **Example:** `Device(config-mon-erspan-src)# s-mac 1111.1111.1111`	Defines source pseudo mac for wan interface.
Step 6	**d-mac** *address* **Example:** `Device(config-mon-erspan-src)# d-mac 2222.2222.2222`	Defines destination pseudo mac for wan interface.
Step 7	**end** **Example:** `Device(config-mon-erspan-src)# end`	Exits ERSPAN source session destination configuration mode, and returns to privileged EXEC mode.

Configuration Examples for ERSPAN

Example: Configuring an ERSPAN Source Session

The following example shows how to configure an ERSPAN source session:

```
Device> enable
Device# configure terminal
Device(config)# monitor session 1 type erspan-source
Device(config-mon-erspan-src)# description source1
Device(config-mon-erspan-src)# source interface GigabitEthernet1/0/1 rx
Device(config-mon-erspan-src)# source interface GigabitEthernet1/0/4 - 8 tx
Device(config-mon-erspan-src)# source interface GigabitEthernet1/0/3
Device(config-mon-erspan-src)# destination
Device(config-mon-erspan-src-dst)# erspan-id 100
Device(config-mon-erspan-src-dst)# origin ip address 10.1.0.1
Device(config-mon-erspan-src-dst)# ip prec 5
Device(config-mon-erspan-src-dst)# ip ttl 32
Device(config-mon-erspan-src-dst)# mtu 1700
Device(config-mon-erspan-src-dst)# origin ip address 10.10.0.1
```

```
Device(config-mon-erspan-src-dst)# vrf 1
Device(config-mon-erspan-src-dst)# no shutdown
Device(config-mon-erspan-src-dst)# end
```

Example: Configuring an ERSPAN Source Session on a WAN Interface

The following example shows how to configure more than one WAN interface in a single ERSPAN source monitor session. Multiple interfaces have been separated by a commas.

```
monitor session 100 type erspan-source
    source interface Serial 0/1/0:0, Serial 0/1/0:6
```

Example: Configuring an ERSPAN Destination Session

The following example shows how to configure an ERSPAN destination session:

```
monitor session 2 type erspan-destination
 destination interface GigabitEthernet1/3/2
 destination interface GigabitEthernet2/2/0
 source
  erspan-id 100
  ip address 10.10.0.1
```

Example: Configuring an ERSPAN as a Local SPAN

The following example shows how to configure an ERSPAN as a local SPAN.

```
monitor session 10 type erspan-source
 source interface GigabitEthernet0/0/0
 destination
 erspan-id 10
 ip address 10.10.10.1
 origin ip address 10.10.10.1
monitor session 20 type erspan-destination
destination interface GigabitEthernet0/0/1
 source
  erspan-id 10
  ip address 10.10.0.1
```

Example: Configuring ERSPAN Dummy MAC Address Rewrite

```
monitor session 1 type erspan-source
 s-mac 1111.1111.1111
 d-mac 2222.2222.2222
 source interface Gi2/2/0
 destination
  erspan-id 100
  mtu 1464
  ip address 200.0.0.1
  origin ip address 100.0.0.1
```

Example: Configuring UDF-Based ERSPAN

This example shows how to configure UDF-based ERSPAN to match on the inner TCP flags of an encapsulated IP-in-IP packet using the following match criteria:

- Outer source IP address: 10.0.0.2
- Inner TCP flags: Urgent TCP flag is set
- Bytes: Eth Hdr (14) + Outer IP (20) + Inner IP (20) + Inner TCP (20, but TCP flags at 13th byte)
- Offset from packet-start: 14 + 20 + 20 + 13 = 67
- UDF match value: 0x20 • UDF mask: 0xFF

```
udf udf_tcpflags packet-start 67 1
ip access-list acl-udf
permit ip 10.0.0.2/32 any udf udf_tcpflags 0x20 0xff
monitor session 1 type erspan-source
source interface Ethernet 1/1
filter access-group acl-udf
```

This example shows how to configure UDF-based ERSPAN to match regular IP packets with a packet signature (DEADBEEF) at 6 bytes after a Layer 4 header start using the following match criteria:

- Outer source IP address: 10.0.0.2
- Inner TCP flags: Urgent TCP flag is set
- Bytes: Eth Hdr (14) + IP (20) + TCP (20) + Payload: 112233445566DEADBEEF7788
- Offset from Layer 4 header start: 20 + 6 = 26
- UDF match value: 0xDEADBEEF (split into two-byte chunks and two UDFs)
- UDF mask: 0xFFFFFFFF

```
udf udf_pktsig_msb header outer l3 26 2
udf udf_pktsig_lsb header outer l3 28 2
ip access-list acl-udf-pktsig
permit udf udf_pktsig_msb 0xDEAD 0xFFFF udf udf_pktsig_lsb 0xBEEF 0xFFFF
monitor session 1 type erspan-source
source interface Ethernet 1/1
filter access-group acl-udf-pktsig
```

Additional References for Configuring ERSPAN

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
LAN Switching commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	LAN Switching Command Reference

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

Feature Information for Configuring ERSPAN

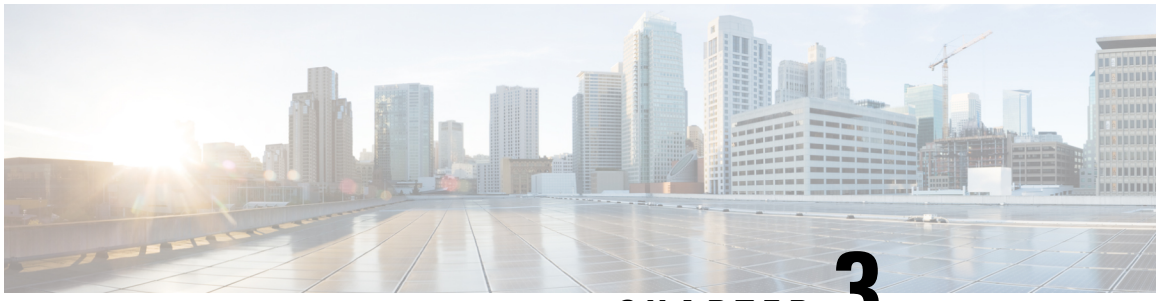
The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 1: Feature Information for Configuring ERSPAN

Feature Name	Releases	Feature Information
ERSPAN	Cisco IOS XE Release 2.1 Cisco IOS XE Release 3.8S	The Cisco ERSPAN feature allows you to monitor traffic on one or more ports or VLANs, and send the monitored traffic to one or more destination ports. The following commands were introduced or modified by this feature: **description**, **destination**, **erspan-id**, **filter**, **ip dscp**, **ip prec**, **ip ttl, monitor permit-list**, **monitor session**, **origin ip address**, **show monitor permit-list**, **source**, **switchport**, **switchport mode trunk**, **switchport nonegotiate**, **switchport trunk encapsulation**, **vrf**. In Cisco IOS XE 3.8S release, ERSPAN was enhanced to support MTU data size up to 9180 bytes. The following command was added by this feature: **mtu**.
ERSPAN Support on WAN Interface	Cisco IOS XE Release 3.5S	ERSPAN has been enhanced to support WAN interface as an ERSPAN source. The following command was modified by this feature: **source interface**.
ERSPAN Type III Header	Cisco IOS XE Denali 16.2	ERSPAN has been enhanced to configure a switch to ERSPAN type III header. The following command was introduced by this feature: **header-type 3**.

Feature Name	Releases	Feature Information
ERSPAN IP ACL	Cisco IOS XE Everest 16.4.1	ERSPAN has been enhanced to better monitor packets and reduce network traffic. This enhancement supports ACL on ERSPAN source session to filter only specific IP traffic according to the ACL. The following command was introduced by this feature: **filter access-group** *acl-filter*.



CHAPTER 3

Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation

This chapter describes the required and optional tasks for configuring routing between VLANs with IEEE 802.1Q encapsulation.

- Restrictions for Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation, on page 19
- Information About Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation, on page 19
- How to Configure Routing Between VLANs with IEEE 802.1Q Encapsulation, on page 20
- Configuration Examples for Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation, on page 24
- Additional References, on page 24
- Feature Information for Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation, on page 25

Restrictions for Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation

Shared port adapters (SPAs) on Cisco ASR 1000 Series Aggregation Services Router have a limit of 8,000 TCAM entries, which limits the number of VLANs you can create on a single SPA.

Information About Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation

Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation

The IEEE 802.1Q protocol is used to interconnect multiple switches and routers, and for defining VLAN topologies. The IEEE 802.1Q standard is extremely restrictive to untagged frames. The standard provides only a per-port VLANs solution for untagged frames. For example, assigning untagged frames to VLANs takes into consideration only the port from which they have been received. Each port has a parameter called

a *permanent virtual identification* (Native VLAN) that specifies the VLAN assigned to receive untagged frames.

The main characteristics of IEEE 802.1Q are as follows:

- Assigns frames to VLANs by filtering.
- The standard assumes the presence of a single spanning tree and of an explicit tagging scheme with one-level tagging.

How to Configure Routing Between VLANs with IEEE 802.1Q Encapsulation

Configuring IP Routing over IEEE 802.1Q

IP routing over IEEE 802.1Q extends IP routing capabilities to include support for routing IP frame types in VLAN configurations using the IEEE 802.1Q encapsulation.

To route IP over IEEE 802.1Q between VLANs, you need to customize the subinterface to create the environment in which it will be used. Perform the tasks described in the following sections in the order in which they appear:

Enabling IP Routing

IP routing is automatically enabled in the Cisco IOS XE software for routers. To reenable IP routing if it has been disabled, perform the following steps.

Once you have IP routing enabled on the router, you can customize the characteristics to suit your environment. If necessary, refer to the IP configuration chapters in the *Cisco IOS XE IP Routing Protocols Configuration Guide* , Release 2, for guidelines on configuring IP.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip routing**
4. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip routing** **Example:** `Router(config)# ip routing`	Enables IP routing on the router.
Step 4	**end** **Example:** `Router(config)# exit`	Exits privileged EXEC mode.

Defining the VLAN Encapsulation Format

To define the encapsulation format as IEEE 802.1Q, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *card* / *spaslot* / *port* **.** *subinterface-number*
4. **encapsulation dot1q** *vlanid*
5. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet** *card* / *spaslot* / *port* **.** *subinterface-number* **Example:**	Specifies the subinterface on which IEEE 802.1Q will be used, and enters interface configuration mode.

	Command or Action	Purpose
	`Router(config)# interface gigabitethernet 0/0/0.101`	
Step 4	**encapsulation dot1q** *vlanid* **Example:** `Router(config-subif)# encapsulation dot1q 101`	Defines the encapsulation format as IEEE 802.1Q (**dot1q**), and specifies the VLAN identifier
Step 5	**end** **Example:** `Router(config-subif)# end`	Exits subinterface configuration mode.

Assigning an IP Address to Network Interface

An interface can have one primary IP address. To assign a primary IP address and a network mask to a network interface, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *card* / *spaslot* / *port* **.** *subinterface-number*
4. **ip address** *ip-address mask*
5. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet** *card* / *spaslot* / *port* **.** *subinterface-number* **Example:** `Router(config)# interface gigabitethernet 0/0/0.101`	Specifies the subinterface on which IEEE 802.1Q will be used, and enters interface configuration mode.

	Command or Action	Purpose
Step 4	**ip address** *ip-address mask* **Example:** `Router(config-subif)# ip address 10.0.0.0 255.0.0.0`	Sets a primary IP address for an interface. • Enter the primary IP address for an interface. **Note** A mask identifies the bits that denote the network number in an IP address. When you use the mask to subnet a network, the mask is then referred to as a subnet mask.
Step 5	**end** **Example:** `Router(config-subif)# end`	Exits subinterface configuration mode.

Monitoring and Maintaining VLAN Subinterfaces

To indicate whether a VLAN is a native VLAN, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **show vlans**
3. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show vlans** **Example:** `Router# show vlans`	Displays VLAN information.
Step 3	**end** **Example:** `Router# end`	Exits privileged EXEC mode.

Configuration Examples for Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation

Configuring IP Routing over IEEE 802.1Q Example

This configuration example shows IP being routed on VLAN 101:

```
!
ip routing
!
interface gigabitethernet 4/1/1.101
  encapsulation dot1q 101
  ip addr 10.0.0.0 255.0.0.0
!
```

Additional References

Related Documents

Related Topic	Document Title
IP LAN switching commands: complete command syntax, command mode, defaults, usage guidelines, and examples	Cisco IOS LAN Switching Services Command Reference

Standards

Standard	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported by this feature, and support for existing standards has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

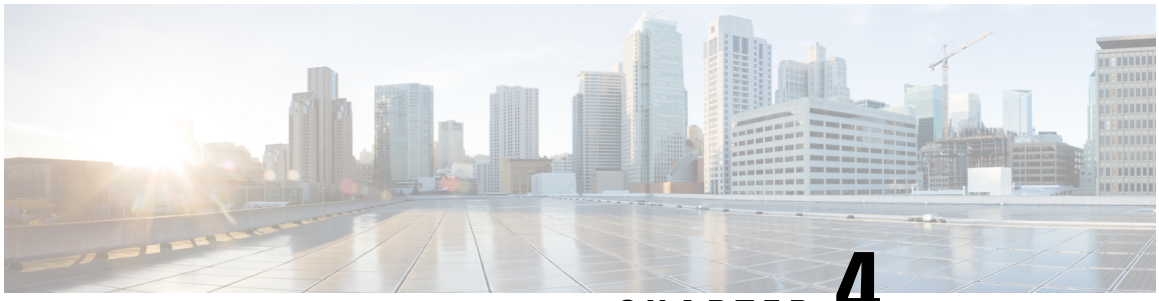
Feature Information for Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 2: Feature Information for Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation

Feature Name	Releases	Feature Information
Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation	Cisco IOS XE Release 2.1	This feature was introduced on the Cisco ASR 1000 Series Aggregation Services Routers.



CHAPTER 4

IEEE 802.1Q-in-Q VLAN Tag Termination

Encapsulating IEEE 802.1Q VLAN tags within 802.1Q enables service providers to use a single VLAN to support customers who have multiple VLANs. The IEEE 802.1Q-in-Q VLAN Tag Termination feature on the subinterface level preserves VLAN IDs and keeps traffic in different customer VLANs segregated.

- Information About IEEE 802.1Q-in-Q VLAN Tag Termination, on page 27
- How to Configure IEEE 802.1Q-in-Q VLAN Tag Termination, on page 29
- Configuration Examples for IEEE 802.1Q-in-Q VLAN Tag Termination, on page 34
- Additional References, on page 36
- Feature Information for IEEE 802.1Q-in-Q VLAN Tag Termination, on page 37

Information About IEEE 802.1Q-in-Q VLAN Tag Termination

IEEE 802.1Q-in-Q VLAN Tag Termination on Subinterfaces

IEEE 802.1Q-in-Q VLAN Tag Termination simply adds another layer of IEEE 802.1Q tag (called "metro tag" or "PE-VLAN") to the 802.1Q tagged packets that enter the network. The purpose is to expand the VLAN space by tagging the tagged packets, thus producing a "double-tagged" frame. The expanded VLAN space allows the service provider to provide certain services, such as Internet access on specific VLANs for specific customers, and yet still allows the service provider to provide other types of services for their other customers on other VLANs.

Generally the service provider's customers require a range of VLANs to handle multiple applications. Service providers can allow their customers to use this feature to safely assign their own VLAN IDs on subinterfaces because these subinterface VLAN IDs are encapsulated within a service-provider designated VLAN ID for that customer. Therefore there is no overlap of VLAN IDs among customers, nor does traffic from different customers become mixed. The double-tagged frame is "terminated" or assigned on a subinterface with an expanded **encapsulation dot1q** command that specifies the two VLAN ID tags (outer VLAN ID and inner VLAN ID) terminated on the subinterface (see the figure below).

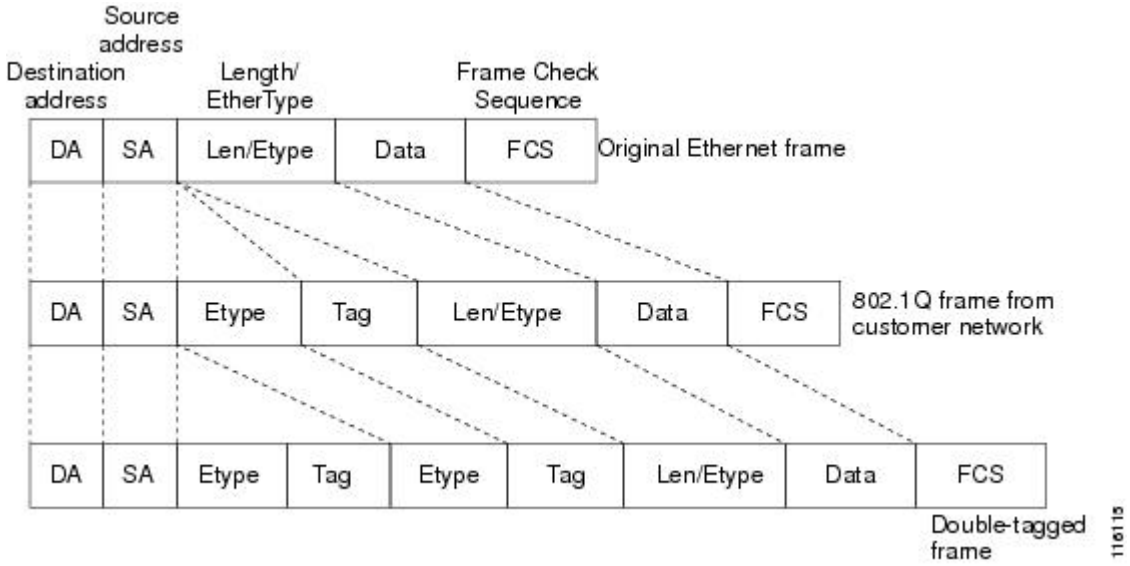
IEEE 802.1Q-in-Q VLAN Tag Termination is generally supported on whichever Cisco IOS XE features or protocols are supported on the subinterface. The only restriction is whether you assign ambiguous or unambiguous subinterfaces for the inner VLAN ID. See the Unambiguous and Ambiguous Subinterfaces section.

The primary benefit for the service provider is reduced number of VLANs supported for the same number of customers. Other benefits of this feature include:

- PPPoE scalability. By expanding the available VLAN space from 4096 to approximately 16.8 million (4096 times 4096), the number of PPPoE sessions that can be terminated on a given interface is multiplied.
- When deploying Gigabyte Ethernet DSL Access Multiplexer (DSLAM) in wholesale model, you can assign the inner VLAN ID to represent the end-customer virtual circuit (VC) and assign the outer VLAN ID to represent the service provider ID.

Whereas switches require IEEE 802.1Q tunnels on interfaces to carry double-tagged traffic, routers need only encapsulate Q-in-Q VLAN tags within another level of 802.1Q tags in order for the packets to arrive at the correct destination.

Figure 2: Untagged, 802.1Q-Tagged, and Double-Tagged Ethernet Frames



Unambiguous and Ambiguous Subinterfaces

The **encapsulation dot1q** command is used to configure Q-in-Q termination on a subinterface. The command accepts an Outer VLAN ID and one or more Inner VLAN IDs. The outer VLAN ID always has a specific value, while inner VLAN ID can either be a specific value or a range of values.

A subinterface that is configured with a single Inner VLAN ID is called an unambiguous Q-in-Q subinterface. In the following example, Q-in-Q traffic with an Outer VLAN ID of 101 and an Inner VLAN ID of 1001 is mapped to the Gigabit Ethernet 1/1/0.100 subinterface:

```
Device(config)# interface gigabitEehernet1/1/0.100
Device(config-subif)# encapsulation dot1q 101 second-dot1q 1001
```

A subinterface that is configured with multiple Inner VLAN IDs is called an ambiguous Q-in-Q subinterface. By allowing multiple Inner VLAN IDs to be grouped together, ambiguous Q-in-Q subinterfaces allow for a smaller configuration, improved memory usage and better scalability.

In the following example, Q-in-Q traffic with an Outer VLAN ID of 101 and Inner VLAN IDs anywhere in the 2001-2100 and 3001-3100 range is mapped to the Gigabit Ethernet 1/1/0.101 subinterface:

```
Device(config)# interface gigabitethernet1/1/0.101
Device(config-subif)# encapsulation dot1q 101 second-dot1q 2001-2100,3001-3100
```

Ambiguous subinterfaces can also use the **any**keyword to specify the inner VLAN ID.

See the Configuration Examples for IEEE 802.1Q-in-Q VLAN Tag Termination section for an example of how VLAN IDs are assigned to subinterfaces, and for a detailed example of how the **any** keyword is used on ambiguous subinterfaces.

Only PPPoE is supported on ambiguous subinterfaces. Standard IP routing is not supported on ambiguous subinterfaces.

IEEE802.1ad Support in Port-channels and Subinterfaces

This enhancement introduces IEEE802.1ad support on port-channel, port-channel subinterfaces, and port-channel member links, with EtherType 0x88a8, 0x9100, and 0x9200, in addition to the existing EtherType 0x8100.

Run the **dot1q tunneling ethertype xxxx** command to configure IEEE802.1ad on port-channel, port-channel subinterface, and port-channel member links.

This configuration is supported on the following platforms:

- Cisco ASR1006-X
- Cisco ASR1009-X: dual RP3, dual ESP 200, dual ESP 200-X, MIP100, and EPA (1X100GE, 10X10GE)
- Cisco ASR 1000 fixed routers

High availability is configured on ASR1006-X, ASR1009-X, and SSO/ISSU.

Restrictions for IEEE802.1ad Support in Port-channels and Subinterfaces

- This feature is not supported with SIP and SPA cards.
- Dot1q tunnel ethertype configuration is not supported in subinterfaces with high-availability configuration.

How to Configure IEEE 802.1Q-in-Q VLAN Tag Termination

Configuring the Interfaces for IEEE 802.1Q-in-Q VLAN Tag Termination

Perform this task to configure the main interface used for the Q-in-Q double tagging and to configure the subinterfaces. An optional step in this task shows you how to configure the EtherType field to be 0x9100 for the outer VLAN tag, if that is required. After the subinterface is defined, the 802.1Q encapsulation is configured to use the double tagging.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **dot1q tunneling ethertype** *ethertype*
5. **interface** *type number* **.** *subinterface-number*
6. **encapsulation dot1q** *vlan-id* **second-dot1q** {**any** | *vlan-id* | *vlan-id* - *vlan-id* [*vlan-id* - *vlan-id*]}

7. **pppoe enable** [**group** *group-name*] [**max-sessions** *max-sessions-number*]
8. **exit**
9. Repeat Step 5 to configure another subinterface.
10. Repeat Step 6 and Step 7 to specify the VLAN tags to be terminated on the subinterface.
11. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/0/0`	Configures an interface and enters interface configuration mode.
Step 4	**dot1q tunneling ethertype** *ethertype* **Example:** `Device(config-if)# dot1q tunneling ethertype 0x9100`	(Optional) Defines the Ethertype field type used by peer devices when implementing Q-in-Q VLAN tagging.
Step 5	**interface** *type number* **.** *subinterface-number* **Example:** `Device(config-if)# interface gigabitethernet 1/0/0.1`	Configures a subinterface and enters subinterface configuration mode.
Step 6	**encapsulation dot1q** *vlan-id* **second-dot1q** {**any** | *vlan-id* | *vlan-id* - *vlan-id* [*vlan-id* - *vlan-id*]} **Example:** `Device(config-subif)# encapsulation dot1q 100 second-dot1q 200`	(Required) Enables the 802.1Q encapsulation of traffic on a specified subinterface in a VLAN. • Use the **second-dot1q** keyword and the *vlan-id*argument to specify the VLAN tags to be terminated on the subinterface. • In this example, an unambiguous Q-in-Q subinterface is configured because only one inner VLAN ID is specified.

	Command or Action	Purpose
		• Q-in-Q frames with an outer VLAN ID of 100 and an inner VLAN ID of 200 will be terminated.
Step 7	**pppoe enable** [**group** *group-name*] [**max-sessions** *max-sessions-number*] **Example:** `Device(config-subif)# pppoe enable group vpn1`	Enables PPPoE sessions on a subinterface. The example specifies that the PPPoE profile, vpn1, will be used by PPPoE sessions on the subinterface.
Step 8	**exit** **Example:** `Device(config-subif)# exit`	Exits subinterface configuration mode and returns to interface configuration mode. • Repeat this step one more time to exit interface configuration mode.
Step 9	Repeat Step 5 to configure another subinterface. **Example:** `Device(config-if)# interface gigabitethernet 1/0/0.2`	(Optional) Configures a subinterface and enters subinterface configuration mode.

Step 10	Repeat Step 6 and Step 7 to specify the VLAN tags to be terminated on the subinterface.
	Example:
	`Device(config-subif)# encapsulation dot1q 100 second-dot1q 100-199,201-600`
	Example:
	`Device(config-subif)# pppoe enable group vpn1`
		Step 6 enables the 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.
		• Use the **second-dot1q** keyword and the *vlan-id*argument to specify the VLAN tags to be terminated on the subinterface.
		• In the example, an ambiguous Q-in-Q subinterface is configured because a range of inner VLAN IDs is specified.
		• Q-in-Q frames with an outer VLAN ID of 100 and an inner VLAN ID in the range of 100 to 199 or 201 to 600 will be terminated.
		Step 7 enables PPPoE sessions on the subinterface. The example specifies that the PPPoE profile, vpn1, will be used by PPPoE sessions on the subinterface.

Step 11	**end** **Example:** `Device(config-subif)# end`	Exits subinterface configuration mode and returns to privileged EXEC mode.

Verifying the IEEE 802.1Q-in-Q VLAN Tag Termination

Perform this optional task to verify the configuration of the IEEE 802.1Q-in-Q VLAN Tag Termination feature.

SUMMARY STEPS

1. **enable**
2. **show running-config**
3. **show vlans dot1q** [**internal** *interface-type interface-number* **.***subinterface-number*[**detail**] | **second-dot1q** *inner-id* **any**]] [**detail]**

DETAILED STEPS

Procedure

Step 1 **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Device> enable
```

Step 2 **show running-config**

Use this command to show the currently running configuration on the device. You can use delimiting characters to display only the relevant parts of the configuration.

Example:

```
Device# show running-config
```

Step 3 **show vlans dot1q** [**internal** *interface-type interface-number* **.***subinterface-number*[**detail**] | **second-dot1q** *inner-id* **any**]] [**detail]**

Use this command to show the statistics for all the 802.1Q VLAN IDs. In this example, only the outer VLAN ID is displayed.

Example:

```
Router# show vlans dot1q
Total statistics for 802.1Q VLAN 1:
   441 packets, 85825 bytes input
   1028 packets, 69082 bytes output
Total statistics for 802.1Q VLAN 101:
   5173 packets, 510384 bytes input
   3042 packets, 369567 bytes output
Total statistics for 802.1Q VLAN 201:
   1012 packets, 119254 bytes input
   1018 packets, 120393 bytes output
Total statistics for 802.1Q VLAN 301:
   3163 packets, 265272 bytes input
   1011 packets, 120750 bytes output
Total statistics for 802.1Q VLAN 401:
   1012 packets, 119254 bytes input
   1010 packets, 119108 bytes output
```

Configuring IEEE 802.1ad in Port-channels and Subinterfaces

Perform this task to configure IEEE802.1ad in port-channels, port-channel subinterfaces, and port-channel member links.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **dot1q tunneling ethertype** {*0x88A8* | *0x9100* | *0x9200*}
5. **interface** *type number* **.** *subinterface-number*
6. **encapsulation dot1q** *vlan-id* **second-dot1q** {**any** | *vlan-id* | *vlan-id* **-** *vlan-id* [*vlan-id* **-** *vlan-id*]}
7. **ip address** *ip-address*
8. **pppoe enable** [**group** *group name*]
9. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface port-channel 12`	Configures an interface and enters interface configuration mode.
Step 4	**dot1q tunneling ethertype** {*0x88A8* | *0x9100* | *0x9200*} **Example:** `Device(config-if)# dot1q tunneling ethertype 0x9100`	Defines the EtherType field used by peer devices when implementing Q-in-Q VLAN tagging.
Step 5	**interface** *type number* **.** *subinterface-number* **Example:** `Device(config-if)# interface port-channel 12.1`	Configures a subinterface and enters subinterface configuration mode.
Step 6	**encapsulation dot1q** *vlan-id* **second-dot1q** {**any** | *vlan-id* | *vlan-id* **-** *vlan-id* [*vlan-id* **-** *vlan-id*]}	Enables the 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.

	Command or Action	Purpose
	Example: `Device(config-subif)# encapsulation dot1q 100 second-dot1q 200`	**Note** • Use the **second-dot1q** keyword and the *vlan-id* argument to specify the VLAN tags to be terminated on the subinterface. • In this example, an unambiguous Q-in-Q subinterface is configured because only one inner VLAN ID is specified. • Q-in-Q frames with an outer VLAN ID of 100 and an inner VLAN ID of 200 will be terminated.
Step 7	**ip address** *ip-address* **Example:** `Device(config-subif)# ip address 192.0.2.1.255.255.255.0`	Defines an IP address on the interface.
Step 8	**pppoe enable** [**group** *group name*] **Example:** `Device(config-subif)# pppoe enable group 2`	Enables PPPoE sessions on the subinterface.
Step 9	**end** **Example:** `Device(config-subif)# end`	Exits subinterface configuration mode and returns to privileged EXEC mode.

Configuration Examples for IEEE 802.1Q-in-Q VLAN Tag Termination

Configuring any Keyword on Subinterfaces for IEEE 802.1Q-in-Q VLAN Tag Termination Example

Some ambiguous subinterfaces can use the **any** keyword for the inner VLAN ID specification. The **any** keyword represents any inner VLAN ID that is not explicitly configured on any other interface. In the following example, seven subinterfaces are configured with various outer and inner VLAN IDs.

Note The **any** keyword can be configured on only one subinterface of a physical interface, outer VLAN ID, or a specified port-channel subinterface.

```
interface GigabitEthernet1/0/0.1
 encapsulation dot1q 100 second-dot1q 100
interface GigabitEthernet1/0/0.2
 encapsulation dot1q 100 second-dot1q 200
```

```
interface GigabitEthernet1/0/0.3
 encapsulation dot1q 100 second-dot1q 300-400,500-600
interface GigabitEthernet1/0/0.4
 encapsulation dot1q 100 second-dot1q any
interface GigabitEthernet1/0/0.5
 encapsulation dot1q 200 second-dot1q 50
interface GigabitEthernet1/0/0.6
 encapsulation dot1q 200 second-dot1q 1000-2000,3000-4000
interface GigabitEthernet1/0/0.7
 encapsulation dot1q 200 second-dot1q any
```

The table below shows which subinterfaces are mapped to different values of the outer and inner VLAN ID on Q-in-Q frames that come in on Gigabit Ethernet interface 1/0/0.

Table 3: Subinterfaces Mapped to Outer and Inner VLAN IDs for GE Interface 1/0/0

Outer VLAN ID	Inner VLAN ID	Subinterface mapped to
100	1 through 99	GigabitEthernet1/0/0.4
100	100	GigabitEthernet1/0/0.1
100	101 through 199	GigabitEthernet1/0/0.4
100	200	GigabitEthernet1/0/0.2
100	201 through 299	GigabitEthernet1/0/0.4
100	300 through 400	GigabitEthernet1/0/0.3
100	401 through 499	GigabitEthernet1/0/0.4
100	500 through 600	GigabitEthernet1/0/0.3
100	601 through 4095	GigabitEthernet1/0/0.4
200	1 through 49	GigabitEthernet1/0/0.7
200	50	GigabitEthernet1/0/0.5
200	51 through 999	GigabitEthernet1/0/0.7
200	1000 through 2000	GigabitEthernet1/0/0.6
200	2001 through 2999	GigabitEthernet1/0/0.7
200	3000 through 4000	GigabitEthernet1/0/0.6
200	4001 through 4095	GigabitEthernet1/0/0.7

A new subinterface is now configured:

```
interface GigabitEthernet1/0/0.8
 encapsulation dot1q 200 second-dot1q 200-600,900-999
```

The table below shows the changes made to the table for the outer VLAN ID of 200. Notice that subinterface 1/0/0.7 configured with the **any** keyword now has new inner VLAN ID mappings.

Table 4: Subinterfaces Mapped to Outer and Inner VLAN IDs for GE Interface 1/0/0--Changes Resulting from Configuring GE Subinterface 1/0/0.8

Outer VLAN ID	Inner VLAN ID	Subinterface mapped to
200	1 through 49	GigabitEthernet1/0/0.7
200	50	GigabitEthernet1/0/0.5
200	51 through 199	GigabitEthernet1/0/0.7
200	200 through 600	GigabitEthernet1/0/0.8
200	601 through 899	GigabitEthernet1/0/0.7
200	900 through 999	GigabitEthernet1/0/0.8
200	1000 through 2000	GigabitEthernet1/0/0.6
200	2001 through 2999	GigabitEthernet1/0/0.7
200	3000 through 4000	GigabitEthernet1/0/0.6
200	4001 through 4095	GigabitEthernet1/0/0.7

Additional References

The following sections provide references related to the IEEE 802.1Q-in-Q VLAN Tag Termination feature.

Related Documents

Related Topic	Document Title
Related commands	*Cisco IOS LAN Switching Command Reference*

Standards

Standards	Title
IEEE 802.1Q	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

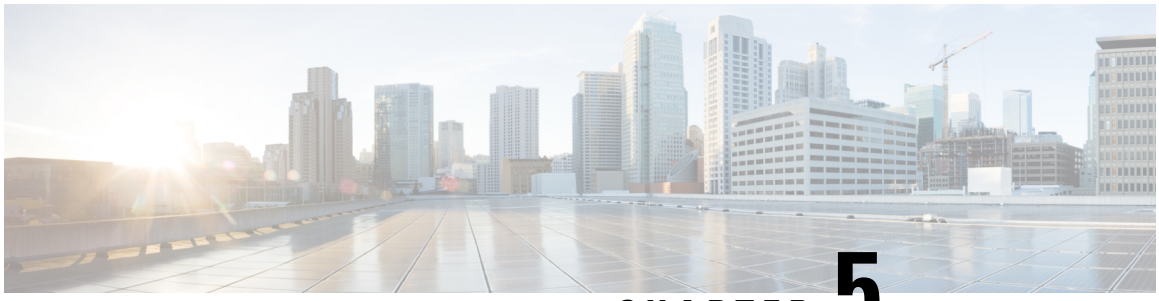
Feature Information for IEEE 802.1Q-in-Q VLAN Tag Termination

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 5: Feature Information for IEEE 802.1Q-in-Q VLAN Tag Termination

Feature Name	Releases	Feature Information
IEEE 802.1Q-in-Q VLAN Tag Termination	Cisco IOS XE Release 2.1	This feature is introduced in the Cisco ASR 1000 Series Aggregation Services Routers. The following commands have been modified for this feature: **dot1q tunneling ethertype**, **encapsulation dot1q**, and **show vlans dot1q**
IEEE802.1ad Support in Port-channels and Subinterfaces	Cisco IOS XE Bengaluru 17.6	This feature is introduced in the Cisco ASR1006-X, Cisco ASR1009-X: dual RP3, dual ESP 200, dual ESP 200-X, MIP100 and EPA (1X100GE, 10X10GE), and Cisco ASR 1000 fixed routers. This feature allows IEEE802.1ad configuration on port-channels, port-channel subinterfaces, and member links with EtherTypes 0x88a8, 0x9100, and 0x9200.



CHAPTER 5

VLAN Mapping to Gigabit EtherChannel Member Links

The VLAN Mapping to Gigabit EtherChannel (GEC) Member Links feature allows you to configure static assignment of user traffic, as identified by a VLAN ID, to a given member link of a GEC bundle. You can manually assign virtual LAN (VLAN) subinterfaces to a primary and secondary link. This feature allows load balancing to downstream equipment regardless of vendor equipment capabilities, and provides failover protection by redirecting traffic to the secondary member link if the primary link fails. Member links are supported with up to 64 GEC interfaces and 14 member links per GEC interface.

- Prerequisites for VLAN Mapping to GEC Member Links, on page 39
- Restrictions for VLAN Mapping to GEC Member Links, on page 39
- Information About VLAN Mapping of GEC Member Links, on page 40
- How to Configure VLAN Mapping to GEC Links, on page 44
- Configuration Examples for VLAN Mapping to GEC Member Links, on page 46
- Additional References, on page 49
- Feature Information for VLAN Mapping to GEC Member Links, on page 49

Prerequisites for VLAN Mapping to GEC Member Links

- Each VLAN must have IEEE 802.1Q encapsulation configured.
- One primary and one secondary link must be associated with each VLAN.
- Configure per VLAN load balancing either on the main port-channel interface or enable it globally.

Restrictions for VLAN Mapping to GEC Member Links

The following restrictions are applicable for IPv6 load balancing on Gigabit EtherChannel (GEC) links:

- IPv6 traffic distribution is enabled only on port channels with flow load balancing.
- Multiprotocol Label Switching (MPLS) Traffic Engineering (TE) is not supported on port channels.
- For Cisco ASR 1000 Series Aggregation Services Routers, the minimum number of member links per GEC interface is 1 and the maximum number is 14.

- 10 Gigabit, 40 Gigabit, 100 Gigabit Ethernet supported as a member link in VLAN mapping.
- The port-channel QinQ subinterface is not supported.
- The quality of service (QoS) policy can be applied to a port-channel subinterface when the following conditions are met:
  - Manual virtual LAN (VLAN) load balancing is supported.
  - A policy map has the appropriate service-fragment policy configured on a physical member link.

Information About VLAN Mapping of GEC Member Links

VLAN-Manual Load Balancing

When load balancing is configured for GEC links, traffic flows are mapped to different buckets as dictated by the load balancing algorithm. For each EtherChannel, a set of 16 buckets are created. The EtherChannel module decides how the buckets are distributed across member links. Each bucket has an active link associated with it that represents the interface to be used for all flows that are mapped to the same bucket.

All packets to be forwarded over the same VLAN subinterface are considered to be part of the same flow that is mapped to one bucket. Each bucket is associated with a primary and secondary pair, and the buckets point to the active interface in the pair. Only one pair is active at a time. Multiple VLAN flows can be mapped to the same bucket if their (primary and secondary) mapping is the same.

The buckets are created when VLAN manual load balancing is enabled. When VLAN load balancing is removed, the buckets are deleted. All port channels use either VLAN manual load balancing or dynamic flow-based load balancing. For information about flow-based load balancing, see the “Flow-Based Per Port-Channel Load Balancing” module.

One primary and one secondary link must be associated with a given VLAN. The primary and secondary options are available only if VLAN manual load balancing is enabled. If the following conditions are met, the load balancing information is downloaded in the forwarding plane. If any of these conditions are not met , the load balancing information is removed from the forwarding plane.

- VLAN load balancing must be enabled globally.
- IEEE 802.1Q encapsulation must be configured on each VLAN.
- One primary and one secondary member link must be enabled to manually map the VLAN traffic to the EtherChannel links.
- The primary and secondary links must be part of the port channel for traffic to use these links.

If only a primary link is specified, a secondary link is selected as the default. If neither a primary nor a secondary link is explicitly configured, the primary and secondary links are selected by default. There is no attempt to perform equal VLAN distribution across links when default links are chosen.

If the interfaces specified as primary or secondary links are not configured as part of the port channel, or if the global VLAN load balancing is not enabled, warning messages are displayed.

Warning

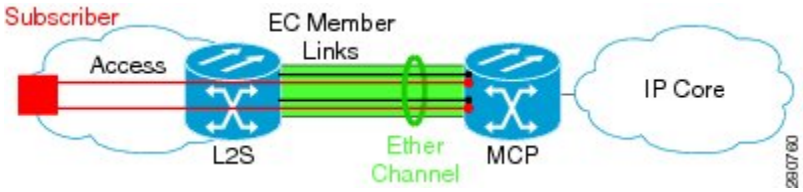
VLAN 500's main interface is not the channel group of primary=GigabitEthernet 4/0/1 Per-VLAN manual load-balancing will not take effect until channel-group is configured under the primary interface.

VLAN 500's main interface is not the channel group of secondary=GigabitEthernet 1/0/0 Per-VLAN manual load-balancing will not take effect until channel-group is configured under the primary interface.

VLAN-to-Port Channel Member Link Mapping

The figure below illustrates the traffic flow for the VLAN-to-port channel mapping.

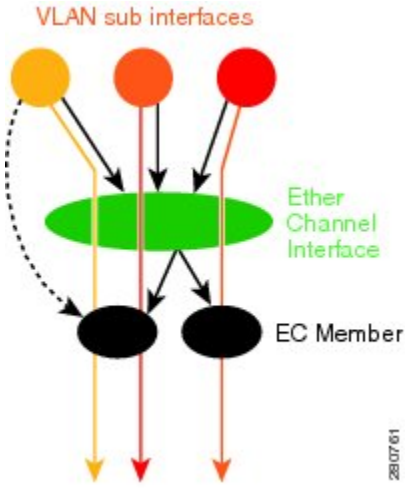
Figure 3: VLAN-to-Port Channel Member Link Mapping



The black lines represent the physical 1 Gigabit Ethernet interfaces connecting the MCP router with the Layer 2 (L2) switch. These interfaces are bundled together in port-channels, shown in green.

In the figure below, subscriber VLAN subinterfaces, shown in shades of orange and red, are configured as Layer 3 (L3) interfaces on top of EtherChannel interfaces. Mapping of the VLAN to the member link (shown with the dotted black arrow) is done through configuration and downloaded in the dataplane so that the outgoing VLA traffic (shown with orange and red arrows) is sent over the associated active primary or secondary member link. The QoS configuration in this model is applied at the VLAN subinterface and member link interface level, implying that QoS queues are created at both levels.

Figure 4: Mapping of VLAN to Member Links



VLAN Primary and Secondary Link Association

In a port-channel traffic distribution, a member link can have either a configured primary state or a secondary state, and an operational active or standby state. When the interface is up, the primary link is active. If the

primary link is down, the interface is in primary standby state while the secondary interface is in secondary active state. If the primary link is up, the secondary link is in secondary standby even if the interface is operationally up.

The primary and secondary member links are each associated with a routed VLAN configured on a port-channel main interface. When forwarding traffic for this VLAN, the primary interface is used as the outgoing interface when this interface is up; the secondary interface, if operational, is used when the primary interface is down.

If all the conditions for per-VLAN traffic distribution are not met, the mapping is not downloaded in the forwarding plane. If all the conditions are met, the dataplane is updated with this mapping.

The table below describes the primary and secondary link configuration status and the resulting function of each configuration.

Table 6: VLAN Primary and Secondary Link Mapping Status

Primary Status	Secondary Status	Description
Configured	Configured	Both primary and secondary links are specified with the **encapsulation dot1q** command. `encapsulation dot1Q vlan-id primary`
Defaulted	Defaulted	Neither a primary nor a secondary link is specified. `encapsulation dot1Q vlan-id` In a stable system, defaults for both primary and secondary links are selected in the same manner for all VLANs. The first link up that is added to the EC is selected as primary, and the second link up as secondary. If there are no links up, the primary and secondary links are selected from the down links.
Configured	Defaulted	Only the primary link is specified. `encapsulation dot1Q vlan-id primary` A secondary link that is different than the primary link is internally selected.
Configured	–	Only a primary link is specified and only one link is defined. `encapsulation dot1Q vlan-id primary` No secondary link can be selected as default when only one link is defined in the EC.
Defaulted	–	Neither a primary nor secondary link is specified, and only one link is defined. `encapsulation dot1Q vlan-id` A default for a primary link is selected. However, no default link can be selected for a secondary link if only one link is defined in the EC.

Primary Status	Secondary Status	Description
–	–	Neither a primary nor secondary link is specified, and no links are defined. `encapsulation dot1Q vlan-id` Defaults cannot be selected and no links are defined in the EC.



Note Default mappings do not override user-configured mappings even if the user-configured mappings are defined incorrectly. Once the (VLAN, primary, and secondary) association is performed (either through the CLI, default or a combination of both), the system validates the mapping and downloads it to the dataplane. If there are no VLANs configured, all traffic forwarded over the port channel is dropped.

Adding Channel Member Links

When a new member link is added, new buckets are created and downloaded in the dataplane. For all VLANs that have the interface as either primary or secondary, new VLAN-to-bucket mappings are downloaded in the dataplane. For all VLANs that need a default for primary and secondary, the default selection algorithm is triggered, and if QoS validation passes, the VLAN-to-bucket mappings are downloaded. QoS policies create VLAN queues on the newly added link.

Deleting Member Links

When a member link is removed, a warning message is displayed. All VLAN queues from the member link, VLAN-to-bucket mappings, and all affected buckets are removed.

Port Channel Link Down Notification

When a link goes down, all traffic for VLANs that have the Port Channel link assigned as primary link must be switched to secondary link if the secondary is up. The traffic for the VLANs with the Port Channel link assigned as secondary, is not affected. The Port Channel Link Down notification causes all buckets associated with a primary-secondary pair (where the primary link is down and the secondary link is up) to be updated with the secondary link. This change is communicated to the dataplane.

All buckets associated with a primary-secondary pair (and the secondary link is the down link and where primary link is up) are updated so that the primary link is now the active link. This change is communicated to the dataplane.

Port Channel Link Up Notification

When a link goes up, all traffic for VLANs that have this link assigned as primary is switched to this link. The traffic for VLANs that have this link assigned as secondary is not affected. The Port Channel Link Up notification causes all buckets associated with a primary-secondary pair, where the primary link is the link that came up, and the secondary link is up, to be notified that the primary link is up. The change is communicated to the dataplane.

All buckets associated with a primary-secondary pair, where the secondary link is the link that came up and the primary link is down are notified that the secondary link is now the primary link. The change is communicated to the dataplane.

Disabling Load Balancing on the EtherChannel

To disable load balancing on the EtherChannel, use the **no port-channel load-balancing vlan-manual** command. The following warning message is displayed if any VLAN subinterfaces exist:

```
Warning: Removing the Global VLAN LB command will affect traffic
c for all dot1Q VLANs
```

Removing a Member Link from the EtherChannel

To remove a member link from the EtherChannel (EC), use the **no channel-group** command

When a member link is removed from the EC is included in a VLAN mapping, the following warning message is displayed:

```
Warning: Removing GigabitEthernet 4/0/0 from the port-channel will affect traffic for the
dot1Q VLANs that include this link in their mapping.
```

How to Configure VLAN Mapping to GEC Links

Configuring VLAN-Based Manual Load Balancing

Perform this task to link VLAN port-channel and to enable VLAN load balancing on port channels.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **port-channel load-balancing vlan-manual**
4. **interface port-channel** *channel-number*
5. **ip address** *ip-address address-mask*
6. **exit**
7. **interface** *type subinterface-number*
8. **channel-group** *channel-number*
9. **exit**
10. **interface port-channel** *interface-number.subinterface-number*
11. **encapsulation dot1Q** *vlan-id* **primary** *interface-type slot/port* **secondary** *interface-type slot/port*
12. **ip address** *ip-address address-mask*
13. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**port-channel load-balancing vlan-manual** **Example:** `Router(config)# port-channel load-balancing vlan-manual`	Enables port-channel load balancing on the router.
Step 4	**interface port-channel** *channel-number* **Example:** `Router(config)# interface port-channel 1`	Enters interface configuration mode and defines the interface as a port channel.
Step 5	**ip address** *ip-address address-mask* **Example:** `Router(config-if)# ip address 172.16.2.3 255.255.0.0`	Specifies the IP address and mask.
Step 6	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 7	**interface** *type subinterface-number* **Example:** `Router(config)# interface gigbabitethernet 1/1/0`	Enters interface configuration mode on the Gigabit Ethernet interface.
Step 8	**channel-group** *channel-number* **Example:** `Router(config-if)# channel-group 1`	Assigns the Gigabit Ethernet interface to the specified channel group. • The channel number is the same channel number that you specified when you created the port-channel interface.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.

	Command or Action	Purpose
Step 10	**interface port-channel** *interface-number.subinterface-number* **Example:** `Device(config)# interface port-channel 1.100`	Specifies the interface type, interface number, and subinterface number.
Step 11	**encapsulation dot1Q** *vlan-id* **primary** *interface-type slot/port* **secondary** *interface-type slot/port* **Example:** `Device(config-if)# encapsulation dot1Q 100 primary GigabitEthernet 1/1/1 secondary GigabitEthernet 1/2/1`	Enables IEEE 802.1Q encapsulation on the interface.
Step 12	**ip address** *ip-address address-mask* **Example:** `Device(config-if)# ip address 172.16.2.100 255.255.255.0`	Specifies the port channel IP address and mask.
Step 13	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode, and returns to privileged EXEC mode.

Troubleshooting Tips

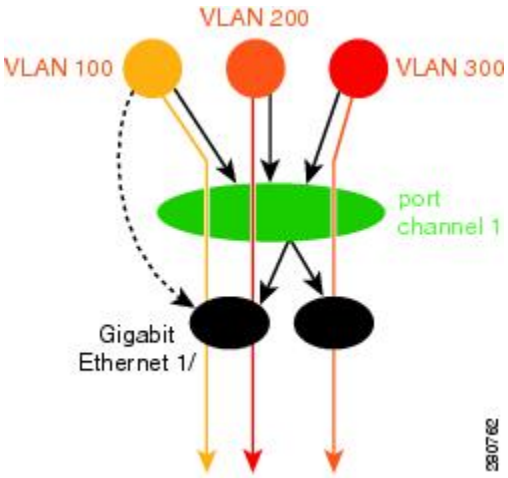
- Use the **show etherchannel load-balancing** command to display the current port channel load balancing method.
- Use the **show interfaces port-channel etherchannel** command to display the current traffic distribution.

Configuration Examples for VLAN Mapping to GEC Member Links

Example: Configuring VLAN Manual Load Balancing

This example shows how the load balancing configuration can be globally applied to define policies for handling traffic by using the **port-channel load-balancing** command. Note that IEEE 802.1Q encapsulation is configured on each port-channel interface. The figure below illustrates the port channel bundle with three VLANs used in the following configuration example:

Figure 5: Port Channel Bundle




```
port-channel load-balancing vlan-manual
!
class-map match-all BestEffort
!
class-map match-all video
!
class-map match-all voice
!
policy-map subscriber
  class voice
    priority level 1
  class video
    priority level 2
  class class-default service-fragment BE
    shape average 10000
    bandwidth remaining percent 80
policy-map aggregate-member-link
    class BestEffort service-fragment BE
    shape average 100000
!
interface Port-channel1
 ip address 172.16.2.3 255.255.0.0
!
interface Port-channel1.100
 encapsulation dot1Q 100 primary GigabitEthernet 1/1/1
              secondary GigabitEthernet 1/2/1
 ip address 172.16.2.100 255.255.255.0
 service-policy output subscriber
!
interface Port-channel1.200
 encapsulation dot1Q 200 primary GigabitEthernet 1/2/1
 ip address 172.16.2.200 255.255.255.0
 service-policy output subscriber
!
interface Port-channel1.300
 encapsulation dot1Q 300
 ip address 172.16.2.300 255.255.255.0
 service-policy output subscriber
!
interface GigabitEthernet 1/1/1
 no ip address
 channel-group 1 mode on
```

```
 service-policy output aggregate-member-link
!
interface GigabitEthernet 1/2/1
 no ip address
 channel-group 1 mode on
 service-policy output aggregate-member-link
```

Example: Troubleshooting

Example 1:

```
Device# show etherchannel load-balancing

EtherChannel Load-Balancing Configuration: vlan-manual
```

Example 2:

```
Device# show etherchannel load-balancing

EtherChannel Load-Balancing Configuration: not configured
```

Use the **show interfaces port-channel** command to display the traffic distribution currently in use.

```
Device# show interfaces port-channel 1 etherchannel

Active Member List contains 0 interfaces
 Passive Member List contains 2 interfaces
  Port: GigabitEthernet 4/0/0
    VLAN 1 (Pri, Ac, D, P)    VLAN 100 (Pri, Ac, C, P)    VLAN 200 (Sec, St, C, P)
  Port: GigabitEthernet 1/0/0
    VLAN 1 (Sec, St, D, P)    VLAN 100 (Sec, St, C, P)    VLAN 200 (Pri, Ac, C, P)
 Bucket Information for VLAN Manual LB:
    Bucket 0   (p=GigabitEthernet 4/0/0, s=GigabitEthernet 4/0/0) active GigabitEthernet
4/0/0
    Bucket 1   (p=Gigabitthernet 4/0/0, s=GigabitEthernet 1/0/0) active GigabitEthernet
4/0/0
    Bucket 4   (p=GigabitEthernet 1/0/0, s=GigabitEthernet 4/0/0) active GigabitEthernet
1/0/0
    Bucket 5   (p=GigabitEthernet 1/0/0, s=GigabitEthernet 1/0/0) active GigabitEthernet
1/0/0
```

To see the mapping of a VLAN to primary and secondary links, use the **show vlans** command.

```
Device# show vlans 100
VLAN ID: 100 (IEEE 802.1Q Encapsulation)
   Protocols Configured:          Received:        Transmitted:
VLAN trunk interfaces for VLAN ID 100:
Port-channel1.1 (100)
      Mapping for traffic load-balancing using bucket 1:
          primary   = GigabitEthernet 4/0/0 (active, C, P)
          secondary = GigabitEthernet 1/0/0 (standby, C, P)
      Total 0 packets, 0 bytes input
      Total 0 packets, 0 bytes output
No subinterface configured with ISL VLAN ID 100
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
LAN Switching commands	*Cisco IOS LAN Switching Command Reference*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

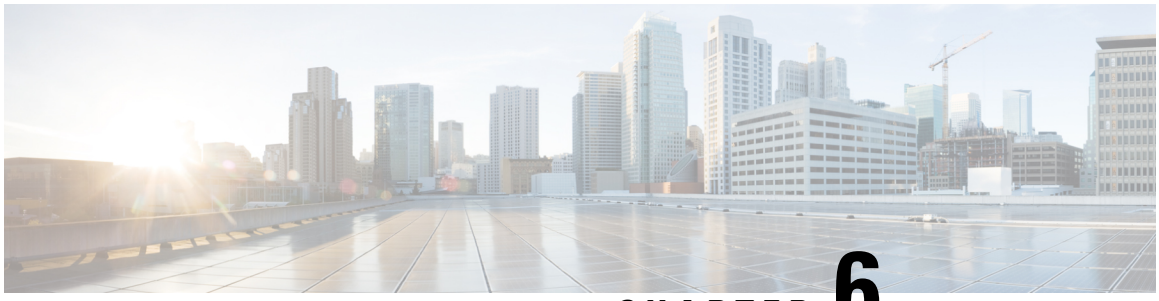
Feature Information for VLAN Mapping to GEC Member Links

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 7: Feature Information for VLAN Mapping to Gigabit EtherChannel Member Links

Feature Name	Releases	Feature Information
VLAN Mapping to Gigabit EtherChannel Member Links	Cisco IOS XE Release 2.1	The VLAN Mapping to Gigabit EtherChannel Member Links feature allows you to configure static assignment of user traffic as identified by a VLAN ID to a given member link of a GEC bundle. You can manually assign VLAN subinterfaces to a primary and secondary link. This feature allows load balancing to downstream equipment, regardless of vendor equipment capabilities, and provides failover protection by redirecting traffic to the secondary member link if the primary link fails. Member links are supported with up to 16 bundles per chassis. The following commands were modified by this feature: **encapsulation dot1q**, **port-channel load-balancing vlan-manual**, **show etherchannel load-balancing**, and **show interfaces port-channel vlan mapping**.



CHAPTER 6

Configuring Routing Between VLANs

This module provides an overview of VLANs. It describes the encapsulation protocols used for routing between VLANs and provides some basic information about designing VLANs. This module contains tasks for configuring routing between VLANS.

- Information About Routing Between VLANs, on page 51
- How to Configure Routing Between VLANS, on page 65
- Configuration Examples for Configuring Routing Between VLANs, on page 98
- Additional References, on page 114
- Feature Information for Routing Between VLANs, on page 115

Information About Routing Between VLANs

Virtual Local Area Network Definition

A virtual local area network (VLAN) is a switched network that is logically segmented on an organizational basis, by functions, project teams, or applications rather than on a physical or geographical basis. For example, all workstations and servers used by a particular workgroup team can be connected to the same VLAN, regardless of their physical connections to the network or the fact that they might be intermingled with other teams. Reconfiguration of the network can be done through software rather than by physically unplugging and moving devices or wires.

A VLAN can be thought of as a broadcast domain that exists within a defined set of switches. A VLAN consists of a number of end systems, either hosts or network equipment (such as bridges and routers), connected by a single bridging domain. The bridging domain is supported on various pieces of network equipment; for example, LAN switches that operate bridging protocols between them with a separate bridge group for each VLAN.

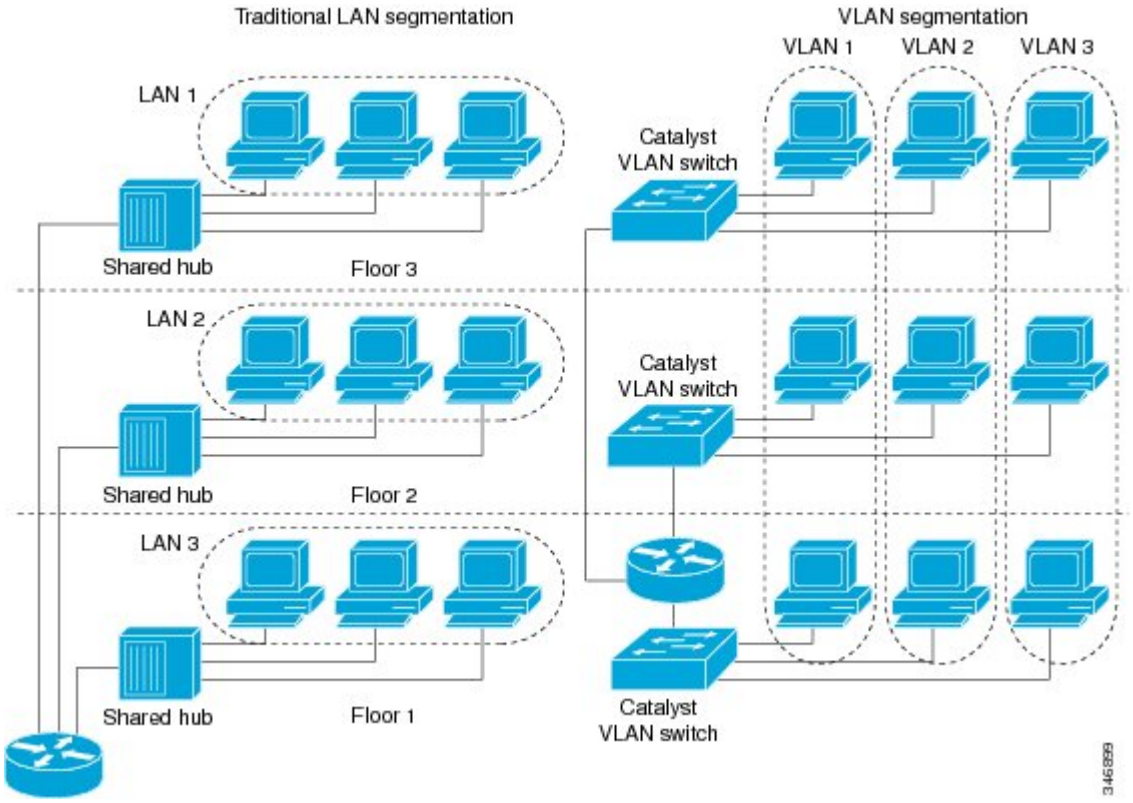
VLANs are created to provide the segmentation services traditionally provided by routers in LAN configurations. VLANs address scalability, security, and network management. Routers in VLAN topologies provide broadcast filtering, security, address summarization, and traffic flow management. None of the switches within the defined group will bridge any frames, not even broadcast frames, between two VLANs. Several key issues described in the following sections need to be considered when designing and building switched LAN internetworks:

LAN Segmentation

VLANs allow logical network topologies to overlay the physical switched infrastructure such that any arbitrary collection of LAN ports can be combined into an autonomous user group or community of interest. The technology logically segments the network into separate Layer 2 broadcast domains whereby packets are switched between ports designated to be within the same VLAN. By containing traffic originating on a particular LAN only to other LANs in the same VLAN, switched virtual networks avoid wasting bandwidth, a drawback inherent to traditional bridged and switched networks in which packets are often forwarded to LANs with no need for them. Implementation of VLANs also improves scalability, particularly in LAN environments that support broadcast- or multicast-intensive protocols and applications that flood packets throughout the network.

The figure below illustrates the difference between traditional physical LAN segmentation and logical VLAN segmentation.

Figure 6: LAN Segmentation and VLAN Segmentation



Security

VLANs improve security by isolating groups. High-security users can be grouped into a VLAN, possibly on the same physical segment, and no users outside that VLAN can communicate with them.

Broadcast Control

Just as switches isolate collision domains for attached hosts and only forward appropriate traffic out a particular port, VLANs provide complete isolation between VLANs. A VLAN is a bridging domain, and all broadcast and multicast traffic is contained within it.

VLAN Performance

The logical grouping of users allows an accounting group to make intensive use of a networked accounting system assigned to a VLAN that contains just that accounting group and its servers. That group's work will not affect other users. The VLAN configuration improves general network performance by not slowing down other users sharing the network.

Network Management

The logical grouping of users allows easier network management. It is not necessary to pull cables to move a user from one network to another. Adds, moves, and changes are achieved by configuring a port into the appropriate VLAN.

Network Monitoring Using SNMP

SNMP support has been added to provide mib-2 interfaces sparse table support for Fast Ethernet subinterfaces. Monitor your VLAN subinterface using the **show vlans** EXEC command. For more information on configuring SNMP on your Cisco network device or enabling an SNMP agent for remote access, see the "Configuring SNMP Support" module in the *Cisco IOS Network Management Configuration Guide* .

Communication Between VLANs

Communication between VLANs is accomplished through routing, and the traditional security and filtering functions of the router can be used. Cisco IOS software provides network services such as security filtering, quality of service (QoS), and accounting on a per-VLAN basis. As switched networks evolve to distributed VLANs, Cisco IOS software provides key inter-VLAN communications and allows the network to scale.

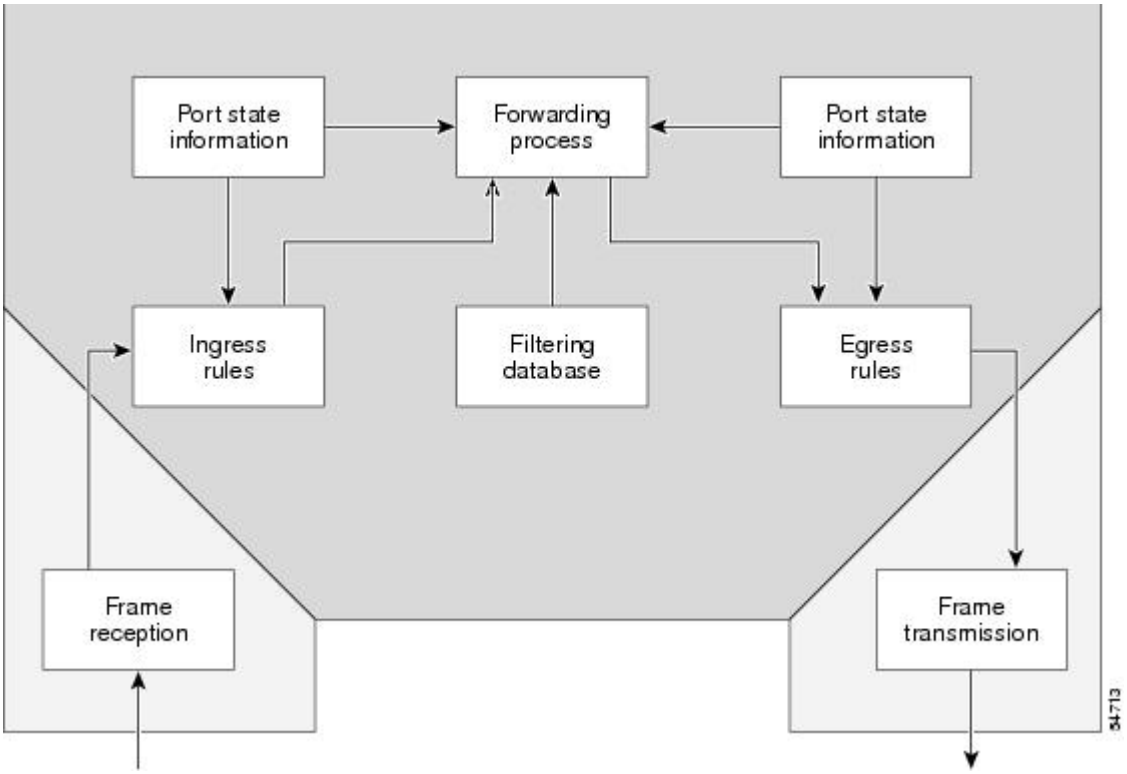
Before Cisco IOS Release 12.2, Cisco IOS support for interfaces that have 802.1Q encapsulation configured is IP, IP multicast, and IPX routing between respective VLANs represented as subinterfaces on a link. New functionality has been added in IEEE 802.1Q support for bridging on those interfaces and the capability to configure and use integrated routing and bridging (IRB).

Relaying Function

The relaying function level, as displayed in the figure below, is the lowest level in the architectural model described in the IEEE 802.1Q standard and presents three types of rules:

- Ingress rules--Rules relevant to the classification of received frames belonging to a VLAN.
- Forwarding rules between ports--Rules decide whether to filter or forward the frame.
- Egress rules (output of frames from the switch)--Rules decide if the frame must be sent tagged or untagged.

Figure 7: Relaying Function

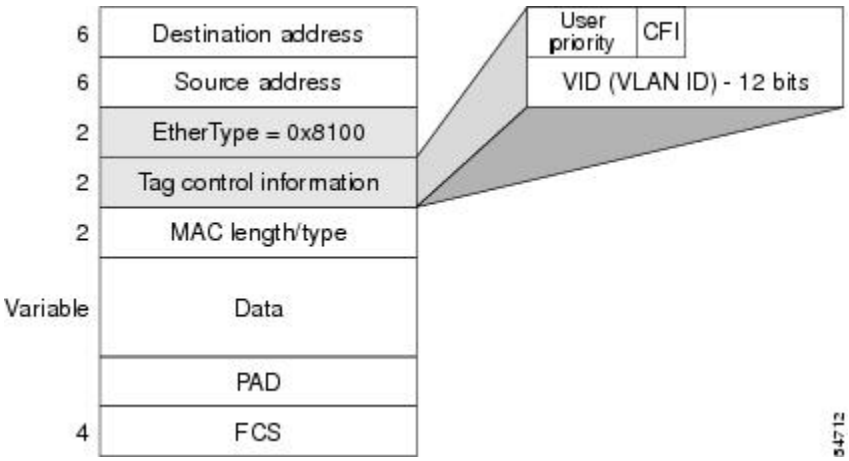


The Tagging Scheme

The figure below shows the tagging scheme proposed by the 802.3ac standard, that is, the addition of the four octets after the source MAC address. Their presence is indicated by a particular value of the EtherType field (called TPID), which has been fixed to be equal to 0x8100. When a frame has the EtherType equal to 0x8100, this frame carries the tag IEEE 802.1Q/802.1p. The tag is stored in the following two octets and it contains 3 bits of user priority, 1 bit of Canonical Format Identifier (CFI), and 12 bits of VLAN ID (VID). The 3 bits of user priority are used by the 802.1p standard; the CFI is used for compatibility reasons between Ethernet-type networks and Token Ring-type networks. The VID is the identification of the VLAN, which is basically used by the 802.1Q standard; being on 12 bits, it allows the identification of 4096 VLANs.

After the two octets of TPID and the two octets of the Tag Control Information field there are two octets that originally would have been located after the Source Address field where there is the TPID. They contain either the MAC length in the case of IEEE 802.3 or the EtherType in the case of Ethernet version 2.

Figure 8: Tagging Scheme

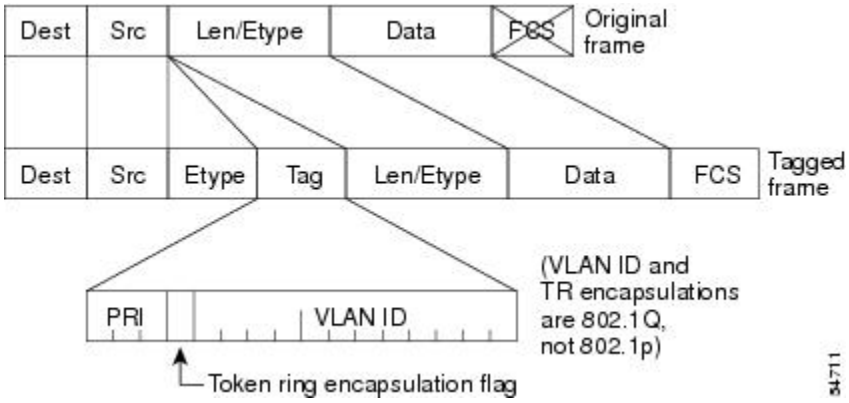


The EtherType and VLAN ID are inserted after the MAC source address, but before the original Ethertype/Length or Logical Link Control (LLC). The 1-bit CFI included a T-R Encapsulation bit so that Token Ring frames can be carried across Ethernet backbones without using 802.1H translation.

Frame Control Sequence Recomputation

The figure below shows how adding a tag in a frame recomputes the Frame Control Sequence. 802.1p and 802.1Q share the same tag.

Figure 9: Adding a Tag Recomputes the Frame Control Sequence

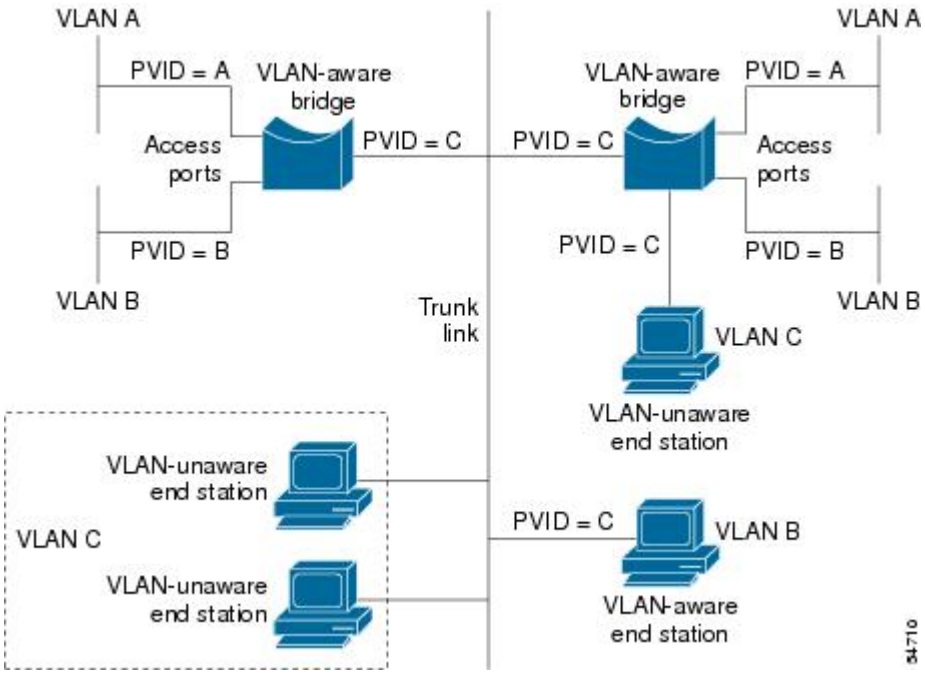


Native VLAN

Each physical port has a parameter called PVID. Every 802.1Q port is assigned a PVID value that is of its native VLAN ID (default is VLAN 1). All untagged frames are assigned to the LAN specified in the PVID parameter. When a tagged frame is received by a port, the tag is respected. If the frame is untagged, the value contained in the PVID is considered as a tag. Because the frame is untagged and the PVID is tagged to allow the coexistence, as shown in the figure below, on the same pieces of cable of VLAN-aware bridge/stations and of VLAN-unaware bridges/stations. Consider, for example, the two stations connected to the central trunk link in the lower part of the figure below. They are VLAN-unaware and they will be associated to the VLAN C, because the PVIDs of the VLAN-aware bridges are equal to VLAN C. Because the VLAN-unaware stations

will send only untagged frames, when the VLAN-aware bridge devices receive these untagged frames they will assign them to VLAN C.

Figure 10: Native VLAN



PVST+

PVST+ provides support for 802.1Q trunks and the mapping of multiple spanning trees to the single spanning tree of 802.1Q switches.

The PVST+ architecture distinguishes three types of regions:

- A PVST region
- A PVST+ region
- A MST region

Each region consists of a homogenous type of switch. A PVST region can be connected to a PVST+ region by connecting two ISL ports. Similarly, a PVST+ region can be connected to an MST region by connecting two 802.1Q ports.

At the boundary between a PVST region and a PVST+ region the mapping of spanning trees is one-to-one. At the boundary between a MST region and a PVST+ region, the ST in the MST region maps to one PVST in the PVST+ region. The one it maps to is called the common spanning tree (CST). The default CST is the PVST of VLAN 1 (Native VLAN).

All PVSTs, except for the CST, are tunneled through the MST region. Tunneling means that bridge protocol data units (BPDUs) are flooded through the MST region along the single spanning tree present in the MST region.

Ingress and Egress Rules

The BPDU transmission on the 802.1Q port of a PVST+ router will be implemented in compliance with the following rules:

- The CST BPDU (of VLAN 1, by default) is sent to the IEEE address.
- All the other BPDUs are sent to Shared Spanning Tree Protocol (SSTP)-Address and encapsulated with Logical Link Control-Subnetwork Access Protocol (LLC-SNAP) header.
- The BPDU of the CST and BPDU of the VLAN equal to the PVID of the 802.1Q trunk are sent untagged.
- All other BPDUs are sent tagged with the VLAN ID.
- The CST BPDU is also sent to the SSTP address.
- Each SSTP-addressed BPDU is also tailed by a Tag-Length-Value for the PVID checking.

The BPDU reception on the 802.1Q port of a PVST+ router will follow these rules:

- All untagged IEEE addressed BPDUs must be received on the PVID of the 802.1Q port.
- The IEEE addressed BPDUs whose VLAN ID matches the Native VLAN are processed by CST.
- All the other IEEE addressed BPDUs whose VLAN ID does not match the Native VLAN and whose port type is not of 802.1Q are processed by the spanning tree of that particular VLAN ID.
- The SSTP addressed BPDU whose VLAN ID is not equal to the TLV are dropped and the ports are blocked for inconsistency.
- All the other SSTP addressed BPDUs whose VLAN ID is not equal to the Native VLAN are processed by the spanning tree of that particular VLAN ID.
- The SSTP addressed BPDUs whose VLAN ID is equal to the Native VLAN are dropped. It is used for consistency checking.

Integrated Routing and Bridging

IRB enables a user to route a given protocol between routed interfaces and bridge groups or route a given protocol between the bridge groups. Integrated routing and bridging is supported on the following protocols:

- IP
- IPX
- AppleTalk

VLAN Colors

VLAN switching is accomplished through *frame tagging* where traffic originating and contained within a particular virtual topology carries a unique VLAN ID as it traverses a common backbone or trunk link. The VLAN ID enables VLAN switching devices to make intelligent forwarding decisions based on the embedded VLAN ID. Each VLAN is differentiated by a *color* , or VLAN identifier. The unique VLAN ID determines the *frame coloring* for the VLAN. Packets originating and contained within a particular VLAN carry the identifier that uniquely defines that VLAN (by the VLAN ID).

The VLAN ID allows VLAN switches and routers to selectively forward packets to ports with the same VLAN ID. The switch that receives the frame from the source station inserts the VLAN ID and the packet is switched onto the shared backbone network. When the frame exits the switched LAN, a switch strips the header and forwards the frame to interfaces that match the VLAN color. If you are using a Cisco network management product such as VlanDirector, you can actually color code the VLANs and monitor VLAN graphically.

Implementing VLANS

Network managers can logically group networks that span all major topologies, including high-speed technologies such as, ATM, FDDI, and Fast Ethernet. By creating virtual LANs, system and network administrators can control traffic patterns and react quickly to relocations and keep up with constant changes in the network due to moving requirements and node relocation just by changing the VLAN member list in the router configuration. They can add, remove, or move devices or make other changes to network configuration using software to make the changes.

Issues regarding creating VLANs should have been addressed when you developed your network design. Issues to consider include the following:

- Scalability
- Performance improvements
- Security
- Network additions, moves, and changes

Communication Between VLANs

Cisco IOS software provides full-feature routing at Layer 3 and translation at Layer 2 between VLANs. Five different protocols are available for routing between VLANs:

All five of these technologies are based on OSI Layer 2 bridge multiplexing mechanisms.

Inter-Switch Link Protocol

The Inter-Switch Link (ISL) protocol is used to interconnect two VLAN-capable Ethernet, Fast Ethernet, or Gigabit Ethernet devices, such as the Catalyst 3000 or 5000 switches and Cisco 7500 routers. The ISL protocol is a packet-tagging protocol that contains a standard Ethernet frame and the VLAN information associated with that frame. The packets on the ISL link contain a standard Ethernet, FDDI, or Token Ring frame and the VLAN information associated with that frame. ISL is currently supported only over Fast Ethernet links, but a single ISL link, or trunk, can carry different protocols from multiple VLANs.

Procedures for configuring ISL and Token Ring ISL (TRISL) features are provided in the Configuring Routing Between VLANs with Inter-Switch Link Encapsulation section.

IEEE 802.10 Protocol

The IEEE 802.10 protocol provides connectivity between VLANs. Originally developed to address the growing need for security within shared LAN/MAN environments, it incorporates authentication and encryption techniques to ensure data confidentiality and integrity throughout the network. Additionally, by functioning at Layer 2, it is well suited to high-throughput, low-latency switching environments. The IEEE 802.10 protocol can run over any LAN or HDLC serial interface.

Procedures for configuring routing between VLANs with IEEE 802.10 encapsulation are provided in the Configuring Routing Between VLANs with IEEE 802.10 section.

IEEE 802.1Q Protocol

The IEEE 802.1Q protocol is used to interconnect multiple switches and routers, and for defining VLAN topologies. Cisco currently supports IEEE 802.1Q for Fast Ethernet and Gigabit Ethernet interfaces.



Note Cisco does not support IEEE 802.1Q encapsulation for Ethernet interfaces.

Procedures for configuring routing between VLANs with IEEE 802.1Q encapsulation are provided in the Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation.

ATM LANE Protocol

The ATM LAN Emulation (LANE) protocol provides a way for legacy LAN users to take advantage of ATM benefits without requiring modifications to end-station hardware or software. LANE emulates a broadcast environment like IEEE 802.3 Ethernet on top of an ATM network that is a point-to-point environment.

LANE makes ATM function like a LAN. LANE allows standard LAN drivers like NDIS and ODI to be used. The virtual LAN is transparent to applications. Applications can use normal LAN functions without the underlying complexities of the ATM implementation. For example, a station can send broadcasts and multicasts, even though ATM is defined as a point-to-point technology and does not support any-to-any services.

To accomplish this, special low-level software is implemented on an ATM client workstation, called the LAN Emulation Client (LEC). The client software communicates with a central control point called a LAN Emulation Server (LES). A broadcast and unknown server (BUS) acts as a central point to distribute broadcasts and multicasts. The LAN Emulation Configuration Server (LECS) holds a database of LECs and the ELANs they belong to. The database is maintained by a network administrator.

These protocols are described in detail in the *Cisco Internetwork Design Guide* .

ATM LANE Fast Simple Server Replication Protocol

To improve the ATM LANE Simple Server Replication Protocol (SSRP), Cisco introduced the ATM LANE Fast Simple Server Replication Protocol (FSSRP). FSSRP differs from LANE SSRP in that all configured LANE servers of an ELAN are always active. FSSRP-enabled LANE clients have virtual circuits (VCs) established to a maximum of four LANE servers and BUSs at one time. If a single LANE server goes down, the LANE client quickly switches over to the next LANE server and BUS, resulting in no data or LE ARP table entry loss and no extraneous signalling.

The FSSRP feature improves upon SSRP such that LANE server and BUS switchover for LANE clients is immediate. With SSRP, a LANE server would go down, and depending on the network load, it may have taken considerable time for the LANE client to come back up joined to the correct LANE server and BUS. In addition to going down with SSRP, the LANE client would do the following:

- Clear out its data direct VCs
- Clear out its LE ARP entries
- Cause substantial signalling activity and data loss

FSSRP was designed to alleviate these problems with the LANE client. With FSSRP, each LANE client is simultaneously joined to up to four LANE servers and BUSs. The concept of the master LANE server and BUS is maintained; the LANE client uses the master LANE server when it needs LANE server BUS services. However, the difference between SSRP and FSSRP is that if and when the master LANE server goes down, the LANE client is already connected to multiple backup LANE servers and BUSs. The LANE client simply uses the next backup LANE server and BUS as the master LANE server and BUS.

VLAN Interoperability

Cisco IOS features bring added benefits to the VLAN technology. Enhancements to ISL, IEEE 802.10, and ATM LANE implementations enable routing of all major protocols between VLANs. These enhancements allow users to create more robust networks incorporating VLAN configurations by providing communications capabilities between VLANs.

Inter-VLAN Communications

The Cisco IOS supports full routing of several protocols over ISL and ATM LANE VLANs. IP, Novell IPX, and AppleTalk routing are supported over IEEE 802.10 VLANs. Standard routing attributes such as network advertisements, secondaries, and help addresses are applicable, and VLAN routing is fast switched. The table below shows protocols supported for each VLAN encapsulation format and corresponding Cisco IOS software releases in which support was introduced.

Table 8: Inter-VLAN Routing Protocol Support

Protocol	ISL	ATM LANE	IEEE 802.10
IP	Release 11.1	Release 10.3	Release 11.1
Novell IPX (default encapsulation)	Release 11.1	Release 10.3	Release 11.1
Novell IPX (configurable encapsulation)	Release 11.3	Release 10.3	Release 11.3
AppleTalk Phase II	Release 11.3	Release 10.3	--
DECnet	Release 11.3	Release 11.0	--
Banyan VINES	Release 11.3	Release 11.2	--
XNS	Release 11.3	Release 11.2	--
CLNS	Release 12.1	--	--
IS-IS	Release 12.1	--	--

VLAN Translation

VLAN translation refers to the ability of the Cisco IOS software to translate between different VLANs or between VLAN and non-VLAN encapsulating interfaces at Layer 2. Translation is typically used for selective inter-VLAN switching of nonroutable protocols and to extend a single VLAN topology across hybrid switching environments. It is also possible to bridge VLANs on the main interface; the VLAN encapsulating header is preserved. Topology changes in one VLAN domain do not affect a different VLAN.

Designing Switched VLANs

By the time you are ready to configure routing between VLANs, you will have already defined them through the switches in your network. Issues related to network design and VLAN definition should be addressed during your network design. See the *Cisco Internetwork Design Guide* and the appropriate switch documentation for information on these topics:

- Sharing resources between VLANs
- Load balancing
- Redundant links
- Addressing
- Segmenting networks with VLANs--Segmenting the network into broadcast groups improves network security. Use router access lists based on station addresses, application types, and protocol types.
- Routers and their role in switched networks--In switched networks, routers perform broadcast management, route processing, and distribution, and provide communication between VLANs. Routers provide VLAN access to shared resources and connect to other parts of the network that are either logically segmented with the more traditional subnet approach or require access to remote sites across wide-area links.

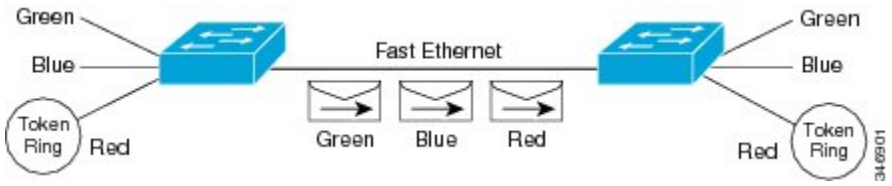
Frame Tagging in ISL

ISL is a Cisco protocol for interconnecting multiple switches and maintaining VLAN information as traffic goes between switches. ISL provides VLAN capabilities while maintaining full wire speed performance on Fast Ethernet links in full- or half-duplex mode. ISL operates in a point-to-point environment and will support up to 1000 VLANs. You can define virtually as many logical networks as are necessary for your environment.

With ISL, an Ethernet frame is encapsulated with a header that transports VLAN IDs between switches and routers. A 26-byte header that contains a 10-bit VLAN ID is propounded to the Ethernet frame.

A VLAN ID is added to the frame only when the frame is prepended for a nonlocal network. The figure below shows VLAN packets traversing the shared backbone. Each VLAN packet carries the VLAN ID within the packet header.

Figure 11: VLAN Packets Traversing the Shared Backbone



You can configure routing between any number of VLANs in your network. This section documents the configuration tasks for each protocol supported with ISL encapsulation. The basic process is the same, regardless of the protocol being routed. It involves the following tasks:

- Enabling the protocol on the router
- Enabling the protocol on the interface
- Defining the encapsulation format as ISL or TRISL

- Customizing the protocol according to the requirements for your environment

IEEE 802.1Q-in-Q VLAN Tag Termination on Subinterfaces

IEEE 802.1Q-in-Q VLAN Tag Termination simply adds another layer of IEEE 802.1Q tag (called "metro tag" or "PE-VLAN") to the 802.1Q tagged packets that enter the network. The purpose is to expand the VLAN space by tagging the tagged packets, thus producing a "double-tagged" frame. The expanded VLAN space allows the service provider to provide certain services, such as Internet access on specific VLANs for specific customers, and yet still allows the service provider to provide other types of services for their other customers on other VLANs.

Generally the service provider's customers require a range of VLANs to handle multiple applications. Service providers can allow their customers to use this feature to safely assign their own VLAN IDs on subinterfaces because these subinterface VLAN IDs are encapsulated within a service-provider designated VLAN ID for that customer. Therefore there is no overlap of VLAN IDs among customers, nor does traffic from different customers become mixed. The double-tagged frame is "terminated" or assigned on a subinterface with an expanded **encapsulation dot1q** command that specifies the two VLAN ID tags (outer VLAN ID and inner VLAN ID) terminated on the subinterface. See the figure below.

IEEE 802.1Q-in-Q VLAN Tag Termination is generally supported on whichever Cisco IOS features or protocols are supported on the subinterface; the exception is that Cisco 10000 series Internet router only supports PPPoE. For example if you can run PPPoE on the subinterface, you can configure a double-tagged frame for PPPoE. The only restriction is whether you assign ambiguous or unambiguous subinterfaces for the inner VLAN ID. See the figure below.



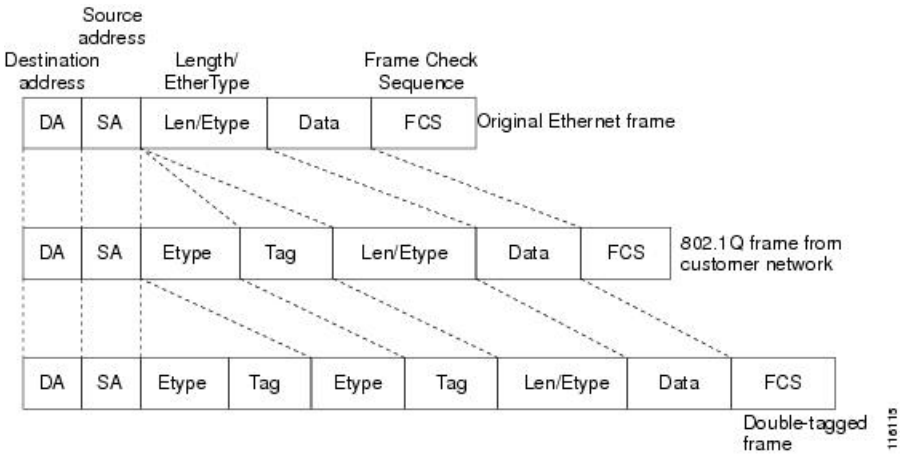
Note The Cisco 10000 series Internet router only supports Point-to-Point Protocol over Ethernet (PPPoE) and IP packets that are double-tagged for Q-in-Q VLAN tag termination. Specifically PPPoEoQ-in-Q and IPoQ-in-Q are supported.

The primary benefit for the service provider is reduced number of VLANs supported for the same number of customers. Other benefits of this feature include:

- PPPoE scalability. By expanding the available VLAN space from 4096 to approximately 16.8 million (4096 times 4096), the number of PPPoE sessions that can be terminated on a given interface is multiplied.
- When deploying Gigabyte Ethernet DSL Access Multiplexer (DSLAM) in wholesale model, you can assign the inner VLAN ID to represent the end-customer virtual circuit (VC) and assign the outer VLAN ID to represent the service provider ID.

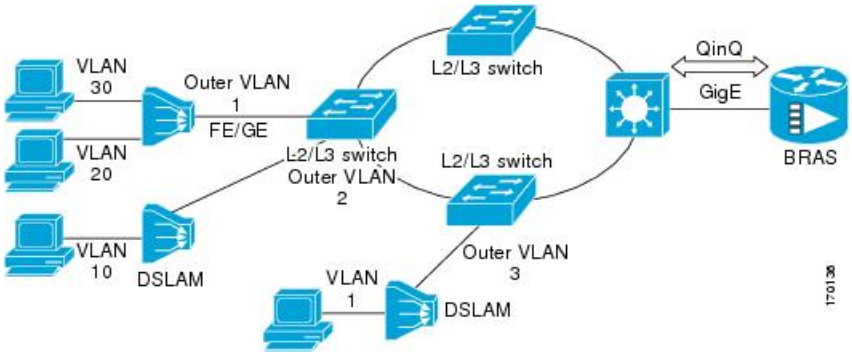
The Q-in-Q VLAN tag termination feature is simpler than the IEEE 802.1Q tunneling feature deployed for the Catalyst 6500 series switches or the Catalyst 3550 and Catalyst 3750 switches. Whereas switches require IEEE 802.1Q tunnels on interfaces to carry double-tagged traffic, routers need only encapsulate Q-in-Q VLAN tags within another level of 802.1Q tags in order for the packets to arrive at the correct destination as shown in figure below.

Figure 12: Untagged, 802.1Q-Tagged, and Double-Tagged Ethernet Frames



Cisco 10000 Series Internet Router Application

For the emerging broadband Ethernet-based DSLAM market, the Cisco 10000 series Internet router supports Q-in-Q encapsulation. With the Ethernet-based DSLAM model shown in the figure below, customers typically get their own VLAN and all these VLANs are aggregated on a DSLAM.

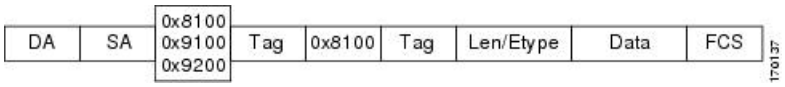


VLAN aggregation on a DSLAM will result in a lot of aggregate VLANs that at some point need to be terminated on the broadband remote access servers (BRAS). Although the model could connect the DSLAMs directly to the BRAS, a more common model uses the existing Ethernet-switched network where each DSLAM VLAN ID is tagged with a second tag (Q-in-Q) as it connects into the Ethernet-switched network.

The only model that is supported is PPPoE over Q-in-Q (PPPoEoQinQ). This can either be a PPP terminated session or as a L2TP LAC session.

The Cisco 10000 series Internet router already supports plain PPPoE and PPP over 802.1Q encapsulation. Supporting PPP over Q-in-Q encapsulation is new. PPP over Q-in-Q encapsulation processing is an extension to 802.1q encapsulation processing. A Q-in-Q frame looks like a VLAN 802.1Q frame, only it has two 802.1Q tags instead of one.

PPP over Q-in-Q encapsulation supports configurable outer tag Ethertype. The configurable Ethertype field values are 0x8100 (default), 0x9100, and 0x9200. See the figure below.



Security ACL Application on the Cisco 10000 Series Internet Router

The IEEE 802.1Q-in-Q VLAN Tag Termination feature provides limited security access control list (ACL) support for the Cisco 10000 series Internet router.

If you apply an ACL to PPPoE traffic on a Q-in-Q subinterface in a VLAN, apply the ACL directly on the PPPoE session, using virtual access interfaces (VAIs) or RADIUS attribute 11 or 242.

You can apply ACLs to virtual access interfaces by configuring them under virtual template interfaces. You can also configure ACLs by using RADIUS attribute 11 or 242. When you use attribute 242, a maximum of 30,000 sessions can have ACLs.

ACLs that are applied to the VLAN Q-in-Q subinterface have no effect and are silently ignored. In the following example, ACL 1 that is applied to the VLAN Q-in-Q subinterface level will be ignored:

```
Router(config)# interface FastEthernet3/0/0.100
Router(config-subif)# encapsulation dot1q 100 second-dot1q 200
Router(config-subif)# ip access-group 1
```

Unambiguous and Ambiguous Subinterfaces

The **encapsulation dot1q** command is used to configure Q-in-Q termination on a subinterface. The command accepts an Outer VLAN ID and one or more Inner VLAN IDs. The outer VLAN ID always has a specific value, while inner VLAN ID can either be a specific value or a range of values.

A subinterface that is configured with a single Inner VLAN ID is called an unambiguous Q-in-Q subinterface. In the following example, Q-in-Q traffic with an Outer VLAN ID of 101 and an Inner VLAN ID of 1001 is mapped to the Gigabit Ethernet 1/0.100 subinterface:

```
Router(config)# interface gigabitEehernet1/0.100
Router(config-subif)# encapsulation dot1q 101 second-dot1q 1001
```

A subinterface that is configured with multiple Inner VLAN IDs is called an ambiguous Q-in-Q subinterface. By allowing multiple Inner VLAN IDs to be grouped together, ambiguous Q-in-Q subinterfaces allow for a smaller configuration, improved memory usage and better scalability.

In the following example, Q-in-Q traffic with an Outer VLAN ID of 101 and Inner VLAN IDs anywhere in the 2001-2100 and 3001-3100 range is mapped to the Gigabit Ethernet 1/0.101 subinterface.:

```
Router(config)# interface gigabitethernet1/0.101
Router(config-subif)# encapsulation dot1q 101 second-dot1q 2001-2100,3001-3100
```

Ambiguous subinterfaces can also use the **any**keyword to specify the inner VLAN ID.

See the Monitoring and Maintaining VLAN Subinterfaces section for an example of how VLAN IDs are assigned to subinterfaces, and for a detailed example of how the **any** keyword is used on ambiguous subinterfaces.

Only PPPoE is supported on ambiguous subinterfaces. Standard IP routing is not supported on ambiguous subinterfaces.



Note On the Cisco 10000 series Internet router, Modular QoS services are only supported on unambiguous subinterfaces.

How to Configure Routing Between VLANS

Configuring a VLAN Range

Using the VLAN Range feature, you can group VLAN subinterfaces together so that any command entered in a group applies to every subinterface within the group. This capability simplifies configurations and reduces command parsing.

The VLAN Range feature provides the following benefits:

- Simultaneous Configurations: Identical commands can be entered once for a range of subinterfaces, rather than being entered separately for each subinterface.
- Overlapping Range Configurations: Overlapping ranges of subinterfaces can be configured.
- Customized Subinterfaces: Individual subinterfaces within a range can be customized or deleted.

Restrictions

- Each command you enter while you are in interface configuration mode with the **interface range** command is executed as it is entered. The commands are not batched together for execution after you exit interface configuration mode. If you exit interface configuration mode while the commands are being executed, some commands might not be executed on some interfaces in the range. Wait until the command prompt reappears before exiting interface configuration mode.
- The **no interface range** command is not supported. You must delete individual subinterfaces to delete a range.

Configuring a Range of VLAN Subinterfaces

Use the following commands to configure a range of VLAN subinterfaces.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface range** {{**ethernet** | **fastethernet** | **gigabitethernet** | **atm**} *slot* / *interface* **.** *subinterface* -{{**ethernet** | **fastethernet** | **gigabitethernet** | **atm**}*slot* / *interface* **.** *subinterface*}
4. **encapsulation dot1Q** *vlan-id*
5. **no shutdown**
6. **exit**
7. **show running-config**
8. **show interfaces**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface range** {{**ethernet** | **fastethernet** | **gigabitethernet** | **atm**} *slot* / *interface* **.** *subinterface* **-**{{**ethernet** | **fastethernet** | **gigabitethernet** | **atm**}*slot* / *interface* **.** *subinterface*} **Example:** `Router(config)# interface range fastethernet5/1.1 - fastethernet5/1.4`	Selects the range of subinterfaces to be configured. **Note** The spaces around the dash are required. For example, the command **interface range fastethernet 1 - 5**is valid; the command **interface range fastethernet 1-5** is not valid.
Step 4	**encapsulation dot1Q** *vlan-id* **Example:** `Router(config-if)# encapsulation dot1Q 301`	Applies a unique VLAN ID to each subinterface within the range. • *vlan-id* --Virtual LAN identifier. The allowed range is from 1 to 4095. • The VLAN ID specified by the *vlan-id* argument is applied to the first subinterface in the range. Each subsequent interface is assigned a VLAN ID, which is the specified *vlan-id* plus the subinterface number minus the first subinterface number (VLAN ID + subinterface number - first subinterface number).
Step 5	**no shutdown** **Example:** `Router(config-if)# no shutdown`	Activates the interface. • This command is required only if you shut down the interface.
Step 6	**exit** **Example:** `Router(config-if)# exit`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:**	Verifies subinterface configuration.

	Command or Action	Purpose
	`Router# show running-config`	
Step 8	**show interfaces** **Example:** `Router# show interfaces`	Verifies that subinterfaces have been created.

Configuring Routing Between VLANs with Inter-Switch Link Encapsulation

This section describes the Inter-Switch Link (ISL) protocol and provides guidelines for configuring ISL and Token Ring ISL (TRISL) features. This section contains the following:

Configuring AppleTalk Routing over ISL

AppleTalk can be routed over VLAN subinterfaces using the ISL and IEEE 802.10 VLAN encapsulation protocols. The AppleTalk Routing over ISL and IEEE 802.10 Virtual LANs feature provides full-feature Cisco IOS software AppleTalk support on a per-VLAN basis, allowing standard AppleTalk capabilities to be configured on VLANs.

To route AppleTalk over ISL or IEEE 802.10 between VLANs, you need to customize the subinterface to create the environment in which it will be used. Perform the steps in the order in which they appear.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **appletalk routing** [**eigrp** *router-number*]
4. **interface** *type slot* / *port* **.** *subinterface-number*
5. **encapsulation isl** *vlan-identifier*
6. **appletalk cable-range** *cable-range* [*network* **.** *node*]
7. **appletalk zone** *zone-name*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**appletalk routing** [**eigrp** *router-number*] **Example:** `Router(config)# appletalk routing`	Enables AppleTalk routing globally on either ISL or 802.10 interfaces.
Step 4	**interface** *type slot* / *port* **.** *subinterface-number* **Example:** `Router(config)# interface Fddi 1/0.100`	Specifies the subinterface the VLAN will use.
Step 5	**encapsulation isl** *vlan-identifier* **Example:** **Example:** or **Example:** **encapsulation sde** *said* **Example:** `Router(config-if)# encapsulation sde 100`	Defines the encapsulation format as either ISL (**isl**) or IEEE 802.10 (**sde**), and specifies the VLAN identifier or security association identifier, respectively.
Step 6	**appletalk cable-range** *cable-range* [*network* **.** *node*] **Example:** `Router(config-if)# appletalk cable-range 100-100 100.2`	Assigns the AppleTalk cable range and zone for the subinterface.
Step 7	**appletalk zone** *zone-name* **Example:** `Router(config-if)# appletalk zone 100`	Assigns the AppleTalk zone for the subinterface.

Configuring Banyan VINES Routing over ISL

Banyan VINES can be routed over VLAN subinterfaces using the ISL encapsulation protocol. The Banyan VINES Routing over ISL Virtual LANs feature provides full-feature Cisco IOS software Banyan VINES support on a per-VLAN basis, allowing standard Banyan VINES capabilities to be configured on VLANs.

To route Banyan VINES over ISL between VLANs, you need to configure ISL encapsulation on the subinterface. Perform the steps in the following task in the order in which they appear:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **vines routing** [*address*]
4. **interface** *type slot* / *port* **.** *subinterface-number*
5. **encapsulation isl** *vlan-identifier*
6. **vines metric** [*whole* [*fraction*]]

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**vines routing** [*address*] **Example:** `Router(config)# vines routing`	Enables Banyan VINES routing globally.
Step 4	**interface** *type slot* / *port* **.** *subinterface-number* **Example:** `Router(config)# interface fastethernet 1/0.1`	Specifies the subinterface on which ISL will be used.
Step 5	**encapsulation isl** *vlan-identifier* **Example:** `Router(config-if)# encapsulation isl 200`	Defines the encapsulation format as ISL (**isl**), and specifies the VLAN identifier.
Step 6	**vines metric** [*whole* [*fraction*]] **Example:** `Router(config-if)#vines metric 2`	Enables VINES routing metric on an interface.

Configuring DECnet Routing over ISL

DECnet can be routed over VLAN subinterfaces using the ISL VLAN encapsulation protocols. The DECnet Routing over ISL Virtual LANs feature provides full-feature Cisco IOS software DECnet support on a per-VLAN basis, allowing standard DECnet capabilities to be configured on VLANs.

To route DECnet over ISL VLANs, you need to configure ISL encapsulation on the subinterface. Perform the steps described in the following task in the order in which they appear.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Router(config)# **decnet**[*network-number*] **routing**[*decnet-address*]
4. **interface** *type slot* / *port* **.** *subinterface-number*
5. **encapsulation isl** *vlan-identifier*
6. **decnet cost** [*cost-value*]

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	Router(config)# **decnet**[*network-number*] **routing**[*decnet-address*] **Example:** `Router(config)# decnet routing 2.1`	Enables DECnet on the router.
Step 4	**interface** *type slot* / *port* **.** *subinterface-number* **Example:** `Router(config)# interface fastethernet 1/0.1`	Specifies the subinterface on which ISL will be used.
Step 5	**encapsulation isl** *vlan-identifier* **Example:** `Router(config-if)# encapsulation isl 200`	Defines the encapsulation format as ISL (**isl**), and specifies the VLAN identifier.

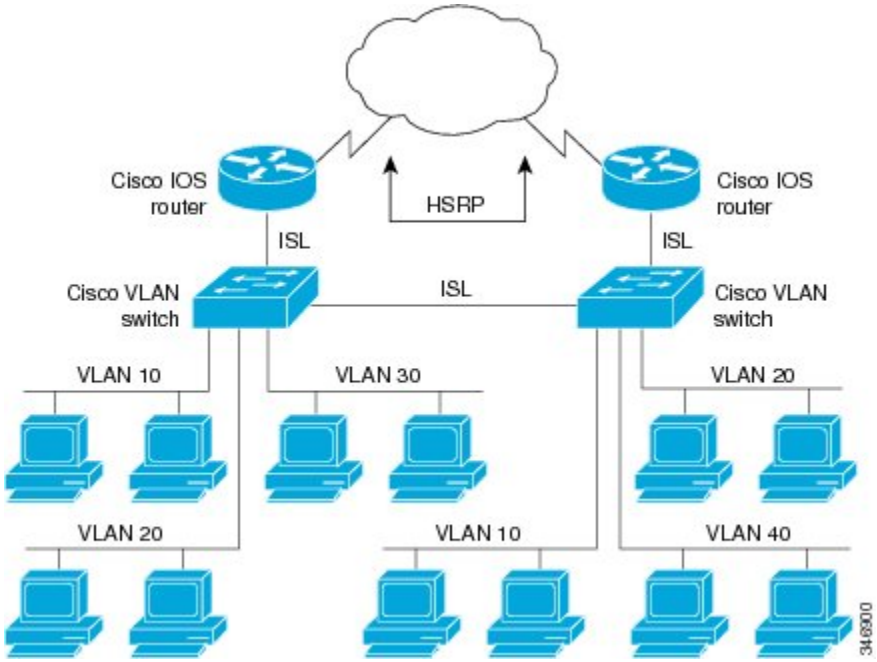
	Command or Action	Purpose
Step 6	**decnet cost** [*cost-value*] **Example:** `Router(config-if)# decnet cost 4`	Enables DECnet cost metric on an interface.

Configuring the Hot Standby Router Protocol over ISL

The Hot Standby Router Protocol (HSRP) provides fault tolerance and enhanced routing performance for IP networks. HSRP allows Cisco IOS routers to monitor each other's operational status and very quickly assume packet forwarding responsibility in the event the current forwarding device in the HSRP group fails or is taken down for maintenance. The standby mechanism remains transparent to the attached hosts and can be deployed on any LAN type. With multiple Hot Standby groups, routers can simultaneously provide redundant backup and perform loadsharing across different IP subnets.

The figure below illustrates HSRP in use with ISL providing routing between several VLANs.

Figure 13: Hot Standby Router Protocol in VLAN Configurations



A separate HSRP group is configured for each VLAN subnet so that Cisco IOS router A can be the primary and forwarding router for VLANs 10 and 20. At the same time, it acts as backup for VLANs 30 and 40. Conversely, Router B acts as the primary and forwarding router for ISL VLANs 30 and 40, as well as the secondary and backup router for distributed VLAN subnets 10 and 20.

Running HSRP over ISL allows users to configure redundancy between multiple routers that are configured as front ends for VLAN IP subnets. By configuring HSRP over ISLs, users can eliminate situations in which a single point of failure causes traffic interruptions. This feature inherently provides some improvement in overall networking resilience by providing load balancing and redundancy capabilities between subnets and VLANs.

To configure HSRP over ISLs between VLANs, you need to create the environment in which it will be used. Perform the tasks described in the following sections in the order in which they appear.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot* / *port* **.** *subinterface-number*
4. **encapsulation isl** *vlan-identifier*
5. **ip address** *ip-address mask* [**secondary**]
6. Router(config-if)# **standby** [*group-number*] **ip**[*ip-address*[**secondary**]]
7. **standby** [*group-number*] **timers** *hellotime holdtime*
8. **standby** [*group-number*] **priority** *priority*
9. **standby** [*group-number*] **preempt**
10. **standby** [*group-number*] **track** *type-number*[*interface-priority*]
11. **standby** [*group-number*] **authentication** *string*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot* / *port* **.** *subinterface-number* **Example:** `Router(config)# interface FastEthernet 1/1.110`	Specifies the subinterface on which ISL will be used and enters interface configuration mode.
Step 4	**encapsulation isl** *vlan-identifier* **Example:** `Router(config-if)#  encapsulation isl 110`	Defines the encapsulation format, and specifies the VLAN identifier.
Step 5	**ip address** *ip-address mask* [**secondary**] **Example:** `Router(config-if)# ip address 10.1.1.2 255.255.255.0`	Specifies the IP address for the subnet on which ISL will be used.

	Command or Action	Purpose
Step 6	Router(config-if)# **standby** [*group-number*] **ip**[*ip-address*[**secondary**]] **Example:** `Router(config-if)# standby 1 ip 10.1.1.101`	Enables HSRP.
Step 7	**standby** [*group-number*] **timers** *hellotime holdtime* **Example:** `Router(config-if)# standby 1 timers 10 10`	Configures the time between hello packets and the hold time before other routers declare the active router to be down.
Step 8	**standby** [*group-number*] **priority** *priority* **Example:** `Router(config-if)# standby 1 priority 105`	Sets the Hot Standby priority used to choose the active router.
Step 9	**standby** [*group-number*] **preempt** **Example:** `Router(config-if)# standby 1 priority 105`	Specifies that if the local router has priority over the current active router, the local router should attempt to take its place as the active router.
Step 10	**standby** [*group-number*] **track** *type-number*[*interface-priority*] **Example:** `Router(config-if)# standby 1 track 4 5`	Configures the interface to track other interfaces, so that if one of the other interfaces goes down, the Hot Standby priority for the device is lowered.
Step 11	**standby** [*group-number*] **authentication** *string* **Example:** `Router(config-if)# standby 1 authentication hsrpword7`	Selects an authentication string to be carried in all HSRP messages.

What to do next

Note For more information on HSRP, see the "Configuring HSRP" module in the *Cisco IOS IP Application Services Configuration Guide* .

Configuring IP Routing over TRISL

The IP routing over TRISL VLANs feature extends IP routing capabilities to include support for routing IP frame types in VLAN configurations.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip routing**
4. **interface** *type slot* / *port* **.** *subinterface-number*
5. **encapsulation tr-isl trbrf-vlan** *vlanid* **bridge-num** *bridge-number*
6. **ip address** *ip-address mask*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip routing** **Example:** `Router(config)# ip routing`	Enables IP routing on the router.
Step 4	**interface** *type slot* / *port* **.** *subinterface-number* **Example:** `Router(config)# interface FastEthernet4/0.1`	Specifies the subinterface on which TRISL will be used and enters interface configuration mode.
Step 5	**encapsulation tr-isl trbrf-vlan** *vlanid* **bridge-num** *bridge-number* **Example:** `Router(config-if# encapsulation tr-isl trbrf-vlan 999 bridge-num 14`	Defines the encapsulation for TRISL. • The DRiP database is automatically enabled when TRISL encapsulation is configured, and at least one TrBRF is defined, and the interface is configured for SRB or for routing with RIF.
Step 6	**ip address** *ip-address mask* **Example:** `Router(config-if# ip address 10.5.5.1 255.255.255.0`	Sets a primary IP address for an interface. • A mask identifies the bits that denote the network number in an IP address. When you use the mask to subnet a network, the mask is then referred to as a *subnet mask*. **Note**

	Command or Action	Purpose
		TRISL encapsulation must be specified for a subinterface before an IP address can be assigned to that subinterface.

Configuring IPX Routing on 802.10 VLANs over ISL

The IPX Encapsulation for 802.10 VLAN feature provides configurable IPX (Novell-FDDI, SAP, SNAP) encapsulation over 802.10 VLAN on router FDDI interfaces to connect the Catalyst 5000 VLAN switch. This feature extends Novell NetWare routing capabilities to include support for routing all standard IPX encapsulations for Ethernet frame types in VLAN configurations. Users with Novell NetWare environments can now configure any one of the three IPX Ethernet encapsulations to be routed using Secure Data Exchange (SDE) encapsulation across VLAN boundaries. IPX encapsulation options now supported for VLAN traffic include the following:

- Novell-FDDI (IPX FDDI RAW to 802.10 on FDDI)
- SAP (IEEE 802.2 SAP to 802.10 on FDDI)
- SNAP (IEEE 802.2 SNAP to 802.10 on FDDI)

NetWare users can now configure consolidated VLAN routing over a single VLAN trunking FDDI interface. Not all IPX encapsulations are currently supported for SDE VLAN. The IPX interior encapsulation support can be achieved by messaging the IPX header before encapsulating in the SDE format. Fast switching will also support all IPX interior encapsulations on non-MCI platforms (for example non-AGS+ and non-7000). With configurable Ethernet encapsulation protocols, users have the flexibility of using VLANs regardless of their NetWare Ethernet encapsulation. Configuring Novell IPX encapsulations on a per-VLAN basis facilitates migration between versions of Netware. NetWare traffic can now be routed across VLAN boundaries with standard encapsulation options (*arpa* , *sap* , and *snap*) previously unavailable. Encapsulation types and corresponding framing types are described in the “Configuring Novell IPX ” module of the *Cisco IOS Novell IPX Configuration Guide* .

Note Only one type of IPX encapsulation can be configured per VLAN (subinterface). The IPX encapsulation used must be the same within any particular subnet; a single encapsulation must be used by all NetWare systems that belong to the same VLAN.

To configure Cisco IOS software on a router with connected VLANs to exchange different IPX framing protocols, perform the steps described in the following task in the order in which they are appear.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipx routing** [*node*]
4. **interface** *fddi slot* / *port* **.** *subinterface-number*
5. **encapsulation sde** *vlan-identifier*
6. **ipx network** *network* **encapsulation** *encapsulation-type*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipx routing** [*node*] **Example:** `Router(config)# ipx routing`	Enables IPX routing globally.
Step 4	**interface** *fddi slot / port . subinterface-number* **Example:** `Router(config)# interface 2/0.1`	Specifies the subinterface on which SDE will be used and enters interface configuration mode.
Step 5	**encapsulation sde** *vlan-identifier* **Example:** `Router(config-if)# encapsulation isl 20`	Defines the encapsulation format and specifies the VLAN identifier.
Step 6	**ipx network** *network* **encapsulation** *encapsulation-type* **Example:** `Router(config-if)# ipx network 20 encapsulation sap`	Specifies the IPX encapsulation among Novell-FDDI, SAP, or SNAP.

Configuring IPX Routing over TRISL

The IPX Routing over ISL VLANs feature extends Novell NetWare routing capabilities to include support for routing all standard IPX encapsulations for Ethernet frame types in VLAN configurations. Users with Novell NetWare environments can configure either SAP or SNAP encapsulations to be routed using the TRISL encapsulation across VLAN boundaries. The SAP (Novell Ethernet_802.2) IPX encapsulation is supported for VLAN traffic.

NetWare users can now configure consolidated VLAN routing over a single VLAN trunking interface. With configurable Ethernet encapsulation protocols, users have the flexibility of using VLANs regardless of their NetWare Ethernet encapsulation. Configuring Novell IPX encapsulations on a per-VLAN basis facilitates migration between versions of Netware. NetWare traffic can now be routed across VLAN boundaries with standard encapsulation options (*sap* and *snap*) previously unavailable. Encapsulation types and corresponding

framing types are described in the "Configuring Novell IPX " module of the *Cisco IOS Novell IPX Configuration Guide* .



Note Only one type of IPX encapsulation can be configured per VLAN (subinterface). The IPX encapsulation used must be the same within any particular subnet: A single encapsulation must be used by all NetWare systems that belong to the same LANs.

To configure Cisco IOS software to exchange different IPX framing protocols on a router with connected VLANs, perform the steps in the following task in the order in which they are appear.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipx routing** [*node*]
4. **interface** *type slot / port* **.** *subinterface-number*
5. **encapsulation tr-isl trbrf-vlan** *trbrf-vlan* **bridge-num** *bridge-num*
6. **ipx network** *network* **encapsulation** *encapsulation-type*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipx routing** [*node*] **Example:** `Router(config)# source-bridge ring-group 100`	Enables IPX routing globally.
Step 4	**interface** *type slot / port* **.** *subinterface-number* **Example:** `Router(config)# interface TokenRing 3/1`	Specifies the subinterface on which TRISL will be used and enters interface configuration mode.
Step 5	**encapsulation tr-isl trbrf-vlan** *trbrf-vlan* **bridge-num** *bridge-num*	Defines the encapsulation for TRISL.

	Command or Action	Purpose
	Example: `Router(config-if)#encapsulation tr-isl trbrf-vlan 999 bridge-num 14`	
Step 6	**ipx network** *network* **encapsulation** *encapsulation-type* **Example:** `Router(config-if)# ipx network 100 encapsulation sap`	Specifies the IPX encapsulation on the subinterface by specifying the NetWare network number (if necessary) and the encapsulation type.

What to do next

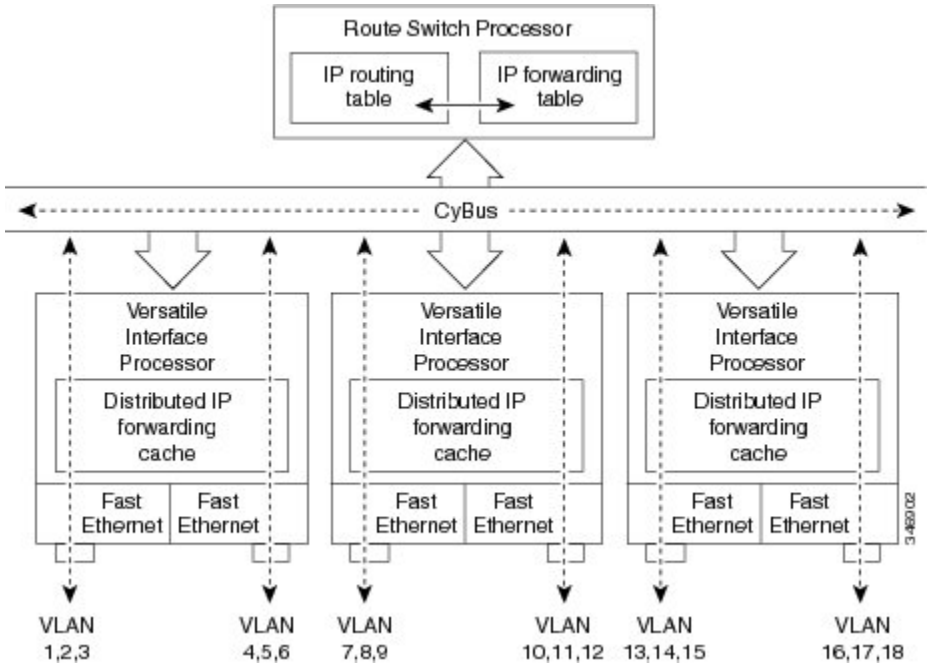
Note The default IPX encapsulation format for Cisco IOS routers is "novell-ether" (Novell Ethernet_802.3). If you are running Novell Netware 3.12 or 4.0, the new Novell default encapsulation format is Novell Ethernet_802.2 and you should configure the Cisco router with the IPX encapsulation format "sap."

Configuring VIP Distributed Switching over ISL

With the introduction of the VIP distributed ISL feature, ISL encapsulated IP packets can be switched on Versatile Interface Processor (VIP) controllers installed on Cisco 7500 series routers.

The second generation VIP2 provides distributed switching of IP encapsulated in ISL in VLAN configurations. Where an aggregation route performs inter-VLAN routing for multiple VLANs, traffic can be switched autonomously on-card or between cards rather than through the central Route Switch Processor (RSP). The figure below shows the VIP distributed architecture of the Cisco 7500 series router.

Figure 14: Cisco 7500 Distributed Architecture



This distributed architecture allows incremental capacity increases by installation of additional VIP cards. Using VIP cards for switching the majority of IP VLAN traffic in multiprotocol environments substantially increases routing performance for the other protocols because the RSP offloads IP and can then be dedicated to switching the non-IP protocols.

VIP distributed switching offloads switching of ISL VLAN IP traffic to the VIP card, removing involvement from the main CPU. Offloading ISL traffic to the VIP card substantially improves networking performance. Because you can install multiple VIP cards in a router, VLAN routing capacity is increased linearly according to the number of VIP cards installed in the router.

To configure distributed switching on the VIP, you must first configure the router for IP routing. Perform the tasks described below in the order in which they appear.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip routing**
4. **interface** *type slot / port-adapter / port*
5. **ip route-cache distributed**
6. **encapsulation isl** *vlan-identifier*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip routing** **Example:** `Router(config)# ip routing`	Enables IP routing on the router. • For more information about configuring IP routing, see the appropriate Cisco IOS *IP Routing Configuration Guide* for the version of Cisco IOS you are using.
Step 4	**interface** *type slot / port-adapter / port* **Example:** `Router(config)# interface FastEthernet1/0/0`	Specifies the interface and enters interface configuration mode.
Step 5	**ip route-cache distributed** **Example:** `Router(config-if)# ip route-cache distributed`	Enables VIP distributed switching of IP packets on the interface.
Step 6	**encapsulation isl** *vlan-identifier* **Example:** `Router(config-if)# encapsulation isl 1`	Defines the encapsulation format as ISL, and specifies the VLAN identifier.

Configuring XNS Routing over ISL

XNS can be routed over VLAN subinterfaces using the ISL VLAN encapsulation protocol. The XNS Routing over ISL Virtual LANs feature provides full-feature Cisco IOS software XNS support on a per-VLAN basis, allowing standard XNS capabilities to be configured on VLANs.

To route XNS over ISL VLANs, you need to configure ISL encapsulation on the subinterface. Perform the steps described in the following task in the order in which they appear.

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **xns routing** [*address*]
4. **interface** *type slot* / *port* **.** *subinterface-number*
5. **encapsulation isl** *vlan-identifier*
6. **xns network** [*number*]

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**xns routing** [*address*] **Example:** `Router(config)# xns routing 0123.4567.adcb`	Enables XNS routing globally.
Step 4	**interface** *type slot* / *port* **.** *subinterface-number* **Example:** `Router(config)# interface fastethernet 1/0.1`	Specifies the subinterface on which ISL will be used and enters interface configuration mode.
Step 5	**encapsulation isl** *vlan-identifier* **Example:** `Router(config-if)# encapsulation isl 100`	Defines the encapsulation format as ISL (**isl**), and specifies the VLAN identifier.
Step 6	**xns network** [*number*] **Example:** `Router(config-if)# xns network 20`	Enables XNS routing on the subinterface.

Configuring CLNS Routing over ISL

CLNS can be routed over VLAN subinterfaces using the ISL VLAN encapsulation protocol. The CLNS Routing over ISL Virtual LANs feature provides full-feature Cisco IOS software CLNS support on a per-VLAN basis, allowing standard CLNS capabilities to be configured on VLANs.

To route CLNS over ISL VLANs, you need to configure ISL encapsulation on the subinterface. Perform the steps described in the following task in the order in which they appear.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **clns routing**
4. **interface** *type slot* / *port* **.** *subinterface-number*
5. **encapsulation isl** *vlan-identifier*
6. **clns enable**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**clns routing** **Example:** `Router(config)# clns routing`	Enables CLNS routing globally.
Step 4	**interface** *type slot* / *port* **.** *subinterface-number* **Example:** `Router(config-if)# interface fastethernet 1/0.1`	Specifies the subinterface on which ISL will be used and enters interface configuration mode.
Step 5	**encapsulation isl** *vlan-identifier* **Example:** `Router(config-if)# encapsulation isl 100`	Defines the encapsulation format as ISL (**isl**), and specifies the VLAN identifier.
Step 6	**clns enable** **Example:** `Router(config-if)# clns enable`	Enables CLNS routing on the subinterface.

Configuring IS-IS Routing over ISL

IS-IS routing can be enabled over VLAN subinterfaces using the ISL VLAN encapsulation protocol. The IS-IS Routing over ISL Virtual LANs feature provides full-feature Cisco IOS software IS-IS support on a per-VLAN basis, allowing standard IS-IS capabilities to be configured on VLANs.

To enable IS-IS over ISL VLANs, you need to configure ISL encapsulation on the subinterface. Perform the steps described in the following task in the order in which they appear.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router isis** [*tag*]
4. **net** *network-entity-title*
5. **interface** *type slot* / *port* **.** *subinterface-number*
6. **encapsulation isl** *vlan-identifier*
7. **clns router isis network** [*tag*]

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router isis** [*tag*] **Example:** `Router(config)# isis routing test-proc2`	Enables IS-IS routing, and enters router configuration mode.
Step 4	**net** *network-entity-title* **Example:** `Router(config)# net 49.0001.0002.aaaa.aaaa.aaaa.00`	Configures the NET for the routing process.
Step 5	**interface** *type slot* / *port* **.** *subinterface-number* **Example:** `Router(config)# interface fastethernet 2.`	Specifies the subinterface on which ISL will be used and enters interface configuration mode.

	Command or Action	Purpose
Step 6	**encapsulation isl** *vlan-identifier* **Example:** `Router(config-if)# encapsulation isl 101`	Defines the encapsulation format as ISL (**isl**), and specifies the VLAN identifier.
Step 7	**clns router isis network** [*tag*] **Example:** `Router(config-if)# clns router is-is network test-proc2`	Specifies the interfaces that should be actively routing IS-IS.

Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation

This section describes the required and optional tasks for configuring routing between VLANs with IEEE 802.1Q encapsulation. The IEEE 802.1Q protocol is used to interconnect multiple switches and routers, and for defining VLAN topologies.

Prerequisites

Configuring routing between VLANs with IEEE 802.1Q encapsulation assumes the presence of a single spanning tree and of an explicit tagging scheme with one-level tagging.

You can configure routing between any number of VLANs in your network.

Restrictions

The IEEE 802.1Q standard is extremely restrictive to untagged frames. The standard provides only a per-port VLANs solution for untagged frames. For example, assigning untagged frames to VLANs takes into consideration only the port from which they have been received. Each port has a parameter called a *permanent virtual identification* (Native VLAN) that specifies the VLAN assigned to receive untagged frames.

The main characteristics of the IEEE 802.1Q are that it assigns frames to VLANs by filtering and that the standard assumes the presence of a single spanning tree and of an explicit tagging scheme with one-level tagging.

This section contains the configuration tasks for each protocol supported with IEEE 802.1Q encapsulation. The basic process is the same, regardless of the protocol being routed. It involves the following tasks:

- Enabling the protocol on the router
- Enabling the protocol on the interface
- Defining the encapsulation format as IEEE 802.1Q
- Customizing the protocol according to the requirements for your environment

To configure IEEE 802.1Q on your network, perform the following tasks. One of the following tasks is required depending on the protocol being used.

- Configuring AppleTalk Routing over IEEE 802.1Q, on page 85 (required)
- Configuring IP Routing over IEEE 802.1Q, on page 86 (required)

- Configuring IPX Routing over IEEE 802.1Q, on page 87 (required)

The following tasks are optional. Perform the following tasks to connect a network of hosts over a simple bridging-access device to a remote access concentrator bridge between IEEE 802.1Q VLANs. The following sections contain configuration tasks for the Integrated Routing and Bridging, Transparent Bridging, and PVST+ Between VLANs with IEEE 802.1Q Encapsulation:

- Configuring a VLAN for a Bridge Group with Default VLAN1, on page 88 (optional)
- Configuring a VLAN for a Bridge Group as a Native VLAN, on page 89 (optional)

Configuring AppleTalk Routing over IEEE 802.1Q

AppleTalk can be routed over virtual LAN (VLAN) subinterfaces using the IEEE 802.1Q VLAN encapsulation protocol. AppleTalk Routing provides full-feature Cisco IOS software AppleTalk support on a per-VLAN basis, allowing standard AppleTalk capabilities to be configured on VLANs.

To route AppleTalk over IEEE 802.1Q between VLANs, you need to customize the subinterface to create the environment in which it will be used. Perform the steps in the order in which they appear.

Use the following task to enable AppleTalk routing on IEEE 802.1Q interfaces.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **appletalk routing** [**eigrp** *router-number*]
4. **interface fastethernet** *slot* / *port* **.** subinterface-number
5. **encapsulation dot1q** *vlan-identifier*
6. **appletalk cable-range** *cable-range* [*network* **.** *node*]
7. **appletalk zone** *zone-name*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**appletalk routing** [**eigrp** *router-number*] **Example:**	Enables AppleTalk routing globally.

	Command or Action	Purpose
	`Router(config)# appletalk routing`	
Step 4	**interface fastethernet** *slot* / *port* **.** subinterface-number **Example:** `Router(config)# interface fastethernet 4/1.00`	Specifies the subinterface the VLAN will use and enters interface configuration mode.
Step 5	**encapsulation dot1q** *vlan-identifier* **Example:** `Router(config-if)# encapsulation dot1q 100`	Defines the encapsulation format as IEEE 802.1Q (**dot1q**), and specifies the VLAN identifier.
Step 6	**appletalk cable-range** *cable-range* [*network* **.** *node*] **Example:** `Router(config-if)# appletalk cable-range 100-100 100.1`	Assigns the AppleTalk cable range and zone for the subinterface.
Step 7	**appletalk zone** *zone-name* **Example:** `Router(config-if)# appletalk zone eng`	Assigns the AppleTalk zone for the subinterface.

What to do next

Note For more information on configuring AppleTalk, see the "Configuring AppleTalk" module in the *Cisco IOS AppleTalk Configuration Guide* .

Configuring IP Routing over IEEE 802.1Q

IP routing over IEEE 802.1Q extends IP routing capabilities to include support for routing IP frame types in VLAN configurations using the IEEE 802.1Q encapsulation.

To route IP over IEEE 802.1Q between VLANs, you need to customize the subinterface to create the environment in which it will be used. Perform the tasks described in the following sections in the order in which they appear.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. ip routing
4. **interface fastethernet** *slot* / *port* **.** subinterface-number
5. **encapsulation dot1q** vlanid
6. **ip address** *ip-address mask*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	ip routing **Example:** `Router(config)# ip routing`	Enables IP routing on the router.
Step 4	**interface fastethernet** *slot* / *port* **.** subinterface-number **Example:** `Router(config)# interface fastethernet 4/1.101`	Specifies the subinterface on which IEEE 802.1Q will be used and enters interface configuration mode.
Step 5	**encapsulation dot1q** vlanid **Example:** `Router(config-if)# encapsulation dot1q 101`	Defines the encapsulation format at IEEE.802.1Q (dot1q) and specifies the VLAN identifier.
Step 6	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip addr 10.0.0.11 255.0.0.0`	Sets a primary IP address and mask for the interface.

What to do next

Once you have IP routing enabled on the router, you can customize the characteristics to suit your environment. See the appropriate *Cisco IOS IP Routing Configuration Guide* for the version of Cisco IOS you are using.

Configuring IPX Routing over IEEE 802.1Q

IPX routing over IEEE 802.1Q VLANs extends Novell NetWare routing capabilities to include support for routing Novell Ethernet_802.3 encapsulation frame types in VLAN configurations. Users with Novell NetWare environments can configure Novell Ethernet_802.3 encapsulation frames to be routed using IEEE 802.1Q encapsulation across VLAN boundaries.

To configure Cisco IOS software on a router with connected VLANs to exchange IPX Novell Ethernet_802.3 encapsulated frames, perform the steps described in the following task in the order in which they appear.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipx routing** [*node*]
4. **interface fastethernet** *slot* / *port* **.** subinterface-number
5. **encapsulation dot1q** vlanid
6. **ipx network** *network*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipx routing** [*node*] **Example:** `Router(config)# ipx routing`	Enables IPX routing globally.
Step 4	**interface fastethernet** *slot* / *port* **.** subinterface-number **Example:** `Router(config)# interface fastethernet 4/1.102`	Specifies the subinterface on which IEEE 802.1Q will be used and enters interface configuration mode.
Step 5	**encapsulation dot1q** vlanid **Example:** `Router(config-if)# encapsulation dot1q 102`	Defines the encapsulation format at IEEE.802.1Q (**dot1q**) and specifies the VLAN identifier.
Step 6	**ipx network** *network* **Example:** `Router(config-if)# ipx network 100`	Specifies the IPX network number.

Configuring a VLAN for a Bridge Group with Default VLAN1

Use the following task to configure a VLAN associated with a bridge group with a default native VLAN.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface fastethernet** *slot* / *port* **.** subinterface-number
4. **encapsulation dot1q** vlanid
5. **bridge-group** *bridge-group*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface fastethernet** *slot* / *port* **.** subinterface-number **Example:** `Router(config)# interface fastethernet 4/1.100`	Selects a particular interface to configure and enters interface configuration mode.
Step 4	**encapsulation dot1q** vlanid **Example:** `Router(config-subif)# encapsulation dot1q 1`	Defines the encapsulation format at IEEE.802.1Q (dot1q) and specifies the VLAN identifier. • The specified VLAN is by default the native VLAN. **Note** If there is no explicitly defined native VLAN, the default VLAN1 becomes the native VLAN.
Step 5	**bridge-group** *bridge-group* **Example:** `Router(config-subif)# bridge-group 1`	Assigns the bridge group to the interface.

Configuring a VLAN for a Bridge Group as a Native VLAN

Use the following task to configure a VLAN associated to a bridge group as a native VLAN.

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **interface fastethernet** *slot* / *port* **.** subinterface-number
4. **encapsulation dot1q** *vlanid* **native**
5. **bridge-group** *bridge-group*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface fastethernet** *slot* / *port* **.** subinterface-number **Example:** `Router(config)# interface fastethernet 4/1.100`	Selects a particular interface to configure and enters interface configuration mode.
Step 4	**encapsulation dot1q** *vlanid* **native** **Example:** `Router(config-subif)# encapsulation dot1q 20 native`	Defines the encapsulation format at IEEE.802.1Q (**dot1q**) and specifies the VLAN identifier. VLAN 20 is specified as the native VLAN. **Note** If there is no explicitly defined native VLAN, the default VLAN1 becomes the native VLAN.
Step 5	**bridge-group** *bridge-group* **Example:** `Router(config-subif)# bridge-group 1`	Assigns the bridge group to the interface.

What to do next

Note If there is an explicitly defined native VLAN, VLAN1 will only be used to process CST.

Configuring IEEE 802.1Q-in-Q VLAN Tag Termination

Encapsulating IEEE 802.1Q VLAN tags within 802.1Q enables service providers to use a single VLAN to support customers who have multiple VLANs. The IEEE 802.1Q-in-Q VLAN Tag Termination feature on the subinterface level preserves VLAN IDs and keeps traffic in different customer VLANs segregated.

You must have checked Feature Navigator to verify that your Cisco device and software image support this feature.

You must be connected to an Ethernet device that supports double VLAN tag imposition/disposition or switching.

The following restrictions apply to the Cisco 10000 series Internet router for configuring IEEE 802.1Q-in-Q VLAN tag termination:

- Supported on Ethernet, FastEthernet, or Gigabit Ethernet interfaces.
- Supports only Point-to-Point Protocol over Ethernet (PPPoE) packets that are double-tagged for Q-in-Q VLAN tag termination.
- IP and Multiprotocol Label Switching (MPLS) packets are not supported.
- Modular QoS can be applied to unambiguous subinterfaces only.
- Limited ACL support.

Perform these tasks to configure the main interface used for the Q-in-Q double tagging and to configure the subinterfaces.

Configuring EtherType Field for Outer VLAN Tag Termination

The following restrictions are applicable for the Cisco 10000 series Internet router:

- PPPoE is already configured.
- Virtual private dial-up network (VPDN) is enabled.

The first task is optional. A step in this task shows you how to configure the EtherType field to be 0x9100 for the outer VLAN tag, if that is required.

After the subinterface is defined, the 802.1Q encapsulation is configured to use the double tagging.

To configure the EtherType field for Outer VLAN Tag Termination, use the following steps. This task is optional.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **dot1q tunneling ethertype** *ethertype*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 1/0/0`	Configures an interface and enters interface configuration mode.
Step 4	**dot1q tunneling ethertype** *ethertype* **Example:** `Router(config-if)# dot1q tunneling ethertype 0x9100`	(Optional) Defines the Ethertype field type used by peer devices when implementing Q-in-Q VLAN tagging. • Use this command if the Ethertype of peer devices is 0x9100 or 0x9200 (0x9200 is only supported on the Cisco 10000 series Internet router). • Cisco 10000 series Internet router supports both the 0x9100 and 0x9200 Ethertype field types.

Configuring the Q-in-Q Subinterface

Use the following steps to configure Q-in-Q subinterfaces. This task is required.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* **.** *subinterface-number*
4. **encapsulation dot1q** *vlan-id* **second-dot1q** {**any** | *vlan-id*| *vlan-id* **-** *vlan-id* [**,** *vlan-id* **-** *vlan-id*]}
5. **pppoe enable** [**group** *group-name*]
6. **exit**
7. Repeat Step 3 to configure another subinterface.
8. Repeat Step 4 and Step 5 to specify the VLAN tags to be terminated on the subinterface.
9. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **.** *subinterface-number* **Example:** `Router(config)# interface gigabitethernet 1/0/0.1`	Configures a subinterface and enters subinterface configuration mode.
Step 4	**encapsulation dot1q** *vlan-id* **second-dot1q** {**any** | *vlan-id*| *vlan-id* **-** *vlan-id* [**,** *vlan-id* **-** *vlan-id*]} **Example:** `Router(config-subif)# encapsulation dot1q 100 second-dot1q 200`	(Required) Enables the 802.1Q encapsulation of traffic on a specified subinterface in a VLAN. • Use the **second-dot1q** keyword and the *vlan-id*argument to specify the VLAN tags to be terminated on the subinterface. • In this example, an unambiguous Q-in-Q subinterface is configured because only one inner VLAN ID is specified. • Q-in-Q frames with an outer VLAN ID of 100 and an inner VLAN ID of 200 will be terminated.
Step 5	**pppoe enable** [**group** *group-name*] **Example:** `Router(config-subif)# pppoe enable group vpn1`	Enables PPPoE sessions on a subinterface. • The example specifies that the PPPoE profile, vpn1, will be used by PPPoE sessions on the subinterface.
Step 6	**exit** **Example:** `Router(config-subif)# exit`	Exits subinterface configuration mode and returns to interface configuration mode. • Repeat this step one more time to exit interface configuration mode.
Step 7	Repeat Step 3 to configure another subinterface. **Example:** `Router(config-if)# interface gigabitethernet 1/0/0.2`	(Optional) Configures a subinterface and enters subinterface configuration mode.

	Command or Action	Purpose

Step 8	Repeat Step 4 and Step 5 to specify the VLAN tags to be terminated on the subinterface.
	Example:
	`Router(config-subif)# encapsulation dot1q 100 second-dot1q 100-199,201-600`
	Example:
	Example:
	`Router(config-subif)# pppoe enable group vpn1`
	Example:
		Step 4 enables the 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.
		• Use the **second-dot1q** keyword and the *vlan-id*argument to specify the VLAN tags to be terminated on the subinterface.
		• In the example, an ambiguous Q-in-Q subinterface is configured because a range of inner VLAN IDs is specified.
		• Q-in-Q frames with an outer VLAN ID of 100 and an inner VLAN ID in the range of 100 to 199 or 201 to 600 will be terminated.
		Step 5 enables PPPoE sessions on the subinterface. The example specifies that the PPPoE profile, vpn1, will be used by PPPoE sessions on the subinterface.
		Note
		Step 5 is required for the Cisco 10000 series Internet router because it only supports PPPoEoQinQ traffic.

Step 9	**end** **Example:** `Router(config-subif)# end`	Exits subinterface configuration mode and returns to privileged EXEC mode.

Verifying the IEEE 802.1Q-in-Q VLAN Tag Termination

Perform this optional task to verify the configuration of the IEEE 802.1Q-in-Q VLAN Tag Termination feature.

SUMMARY STEPS

1. **enable**
2. **show running-config**
3. **show vlans dot1q** [**internal** | *interface-type interface-number* **.***subinterface-number*[**detail**] | *outer-id*[*interface-type interface-number* | **second-dot1q** [*inner-id*| **any**]] [**detail**]]

DETAILED STEPS

Procedure

Step 1 **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Router> enable
```

Step 2 **show running-config**

Use this command to show the currently running configuration on the device. You can use delimiting characters to display only the relevant parts of the configuration.

The following shows the currently running configuration on a Cisco 7300 series router:

Example:

```
Router# show running-config
.
.
.
interface FastEthernet0/0.201
 encapsulation dot1Q 201
 ip address 10.7.7.5 255.255.255.252
!
interface FastEthernet0/0.401
 encapsulation dot1Q 401
 ip address 10.7.7.13 255.255.255.252
!
interface FastEthernet0/0.201999
 encapsulation dot1Q 201 second-dot1q any
 pppoe enable
!
interface FastEthernet0/0.2012001
 encapsulation dot1Q 201 second-dot1q 2001
 ip address 10.8.8.9 255.255.255.252
!
interface FastEthernet0/0.2012002
 encapsulation dot1Q 201 second-dot1q 2002
 ip address 10.8.8.13 255.255.255.252
!
interface FastEthernet0/0.4019999
 encapsulation dot1Q 401 second-dot1q 100-900,1001-2000
 pppoe enable
!
interface GigabitEthernet5/0.101
 encapsulation dot1Q 101
 ip address 10.7.7.1 255.255.255.252
!
interface GigabitEthernet5/0.301
 encapsulation dot1Q 301
 ip address 10.7.7.9 255.255.255.252
!
interface GigabitEthernet5/0.301999
 encapsulation dot1Q 301 second-dot1q any
 pppoe enable
!
interface GigabitEthernet5/0.1011001
 encapsulation dot1Q 101 second-dot1q 1001
 ip address 10.8.8.1 255.255.255.252
!
interface GigabitEthernet5/0.1011002
 encapsulation dot1Q 101 second-dot1q 1002
 ip address 10.8.8.5 255.255.255.252
!
interface GigabitEthernet5/0.1019999
 encapsulation dot1Q 101 second-dot1q 1-1000,1003-2000
 pppoe enable
.
```

```
.
.
```

The following shows the currently running configuration on a Cisco 10000 series Internet router:

Example:

```
Router# show running-config
.
.
.
interface FastEthernet1/0/0.201
 encapsulation dot1Q 201
 ip address 10.7.7.5 255.255.255.252
!
interface FastEthernet1/0/0.401
 encapsulation dot1Q 401
 ip address 10.7.7.13 255.255.255.252
!
interface FastEthernet1/0/0.201999
 encapsulation dot1Q 201 second-dot1q any
 pppoe enable
!
interface FastEthernet1/0/0.4019999
 encapsulation dot1Q 401 second-dot1q 100-900,1001-2000
 pppoe enable
!
interface GigabitEthernet5/0/0.101
 encapsulation dot1Q 101
 ip address 10.7.7.1 255.255.255.252
!
interface GigabitEthernet5/0/0.301
 encapsulation dot1Q 301
 ip address 10.7.7.9 255.255.255.252
!
interface GigabitEthernet5/0/0.301999
 encapsulation dot1Q 301 second-dot1q any
 pppoe enable
!
interface GigabitEthernet5/0/0.1019999
 encapsulation dot1Q 101 second-dot1q 1-1000,1003-2000
 pppoe enable
.
.
.
```

Step 3 **show vlans dot1q** [**internal** | *interface-type interface-number* **.***subinterface-number*[**detail**] | *outer-id*[*interface-type interface-number* | **second-dot1q** [*inner-id*| **any**]] [**detail**]]

Use this command to show the statistics for all the 802.1Q VLAN IDs. In this example, only the outer VLAN ID is displayed.

Note
The **show vlans dot1q**command is not supported on the Cisco 10000 series Internet router.

Example:

```
Router# show vlans dot1q
Total statistics for 802.1Q VLAN 1:
   441 packets, 85825 bytes input
   1028 packets, 69082 bytes output
Total statistics for 802.1Q VLAN 101:
```

```
   5173 packets, 510384 bytes input
   3042 packets, 369567 bytes output
Total statistics for 802.1Q VLAN 201:
   1012 packets, 119254 bytes input
   1018 packets, 120393 bytes output
Total statistics for 802.1Q VLAN 301:
   3163 packets, 265272 bytes input
   1011 packets, 120750 bytes output
Total statistics for 802.1Q VLAN 401:
   1012 packets, 119254 bytes input
   1010 packets, 119108 bytes output
```

Monitoring and Maintaining VLAN Subinterfaces

Use the following task to determine whether a VLAN is a native VLAN.

SUMMARY STEPS

1. **enable**
2. **show vlans**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show vlans** **Example:** `Router# show vlans`	Displays VLAN subinterfaces.

Monitoring and Maintaining VLAN Subinterfaces Example

The following is sample output from the **show vlans**command indicating a native VLAN and a bridged group:

```
Router# show vlans
Virtual LAN ID:  1 (IEEE 802.1Q Encapsulation)
   vLAN Trunk Interface:   FastEthernet1/0/2
 This is configured as native Vlan for the following interface(s) :
FastEthernet1/0/2
   Protocols Configured:   Address: Received:        Transmitted:
Virtual LAN ID:  100 (IEEE 802.1Q Encapsulation)
   vLAN Trunk Interface:   FastEthernet1/0/2.1
   Protocols Configured:   Address: Received:        Transmitted:
        Bridging        Bridge Group 1 0                  0
```

The following is sample output from the **show vlans**command that shows the traffic count on Fast Ethernet subinterfaces:

```
Router# show vlans
Virtual LAN ID:  2 (IEEE 802.1Q Encapsulation)
    vLAN Trunk Interface:   FastEthernet5/0.1

    Protocols Configured:   Address:              Received:        Transmitted:
           IP               172.16.0.3                   16               92129

Virtual LAN ID:  3 (IEEE 802.1Q Encapsulation)

    vLAN Trunk Interface:   Ethernet6/0/1.1

    Protocols Configured:   Address:              Received:        Transmitted:
           IP               172.20.0.3                 1558               1521

Virtual LAN ID:  4 (Inter Switch Link Encapsulation)

    vLAN Trunk Interface:   FastEthernet5/0.2

    Protocols Configured:   Address:              Received:        Transmitted:
           IP               172.30.0.3                    0                  7
```

Configuration Examples for Configuring Routing Between VLANs

Single Range Configuration Example

The following example configures the Fast Ethernet subinterfaces within the range 5/1.1 and 5/1.4 and applies the following VLAN IDs to those subinterfaces:

Fast Ethernet5/1.1 = VLAN ID 301 (vlan-id)

Fast Ethernet5/1.2 = VLAN ID 302 (vlan-id = 301 + 2 - 1 = 302)

Fast Ethernet5/1.3 = VLAN ID 303 (vlan-id = 301 + 3 - 1 = 303)

Fast Ethernet5/1.4 = VLAN ID 304 (vlan-id = 301 + 4 - 1 = 304)

```
Router(config)# interface range fastethernet5/1.1 - fastethernet5/1.4

Router(config-if)# encapsulation dot1Q 301
Router(config-if)# no shutdown

Router(config-if)#
*Oct  6 08:24:35: %LINK-3-UPDOWN: Interface FastEthernet5/1.1, changed state to up
*Oct  6 08:24:35: %LINK-3-UPDOWN: Interface FastEthernet5/1.2, changed state to up
*Oct  6 08:24:35: %LINK-3-UPDOWN: Interface FastEthernet5/1.3, changed state to up
*Oct  6 08:24:35: %LINK-3-UPDOWN: Interface FastEthernet5/1.4, changed state to up
*Oct  6 08:24:36: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet5/1.1, changed
 state to up
*Oct  6 08:24:36: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet5/1.2, changed
 state to up
*Oct  6 08:24:36: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet5/1.3, changed
 state to up
*Oct  6 08:24:36: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet5/1.4, changed
 state to up
```

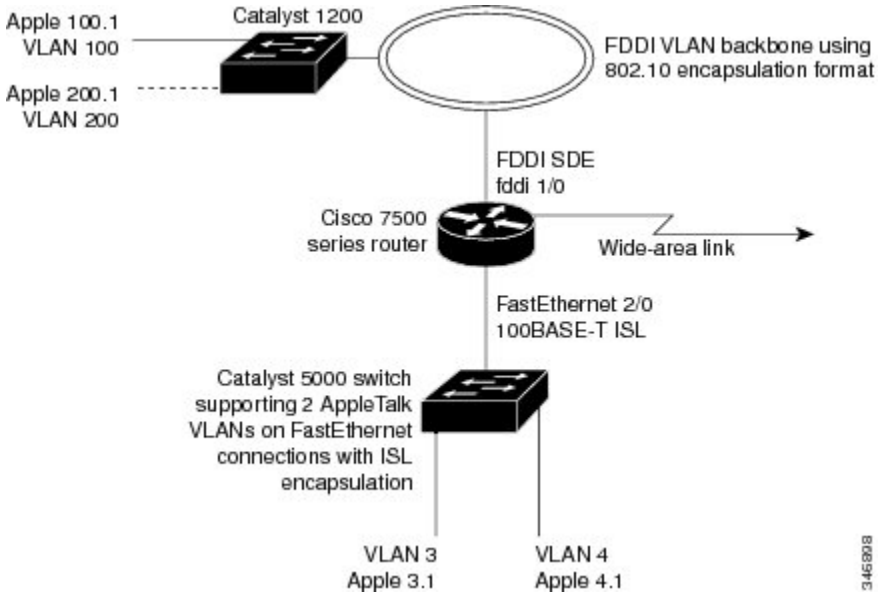
ISL Encapsulation Configuration Examples

This section provides the following configuration examples for each of the protocols described in this module:

AppleTalk Routing over ISL Configuration Example

The configuration example illustrated in the figure below shows AppleTalk being routed between different ISL and IEEE 802.10 VLAN encapsulating subinterfaces.

Figure 15: Routing AppleTalk over VLAN Encapsulations



As shown in the figure above, AppleTalk traffic is routed to and from switched VLAN domains 3, 4, 100, and 200 to any other AppleTalk routing interface. This example shows a sample configuration file for the Cisco 7500 series router with the commands entered to configure the network shown in the figure above.

Cisco 7500 Router Configuration

```
!
appletalk routing
interface Fddi 1/0.100
 encapsulation sde 100
 appletalk cable-range 100-100 100.2
 appletalk zone 100
!
interface Fddi 1/0.200
 encapsulation sde 200
 appletalk cable-range 200-200 200.2
 appletalk zone 200
!
interface FastEthernet 2/0.3
 encapsulation isl 3
 appletalk cable-range 3-3 3.2
 appletalk zone 3
!
interface FastEthernet 2/0.4
 encapsulation isl 4
```

```
 appletalk cable-range 4-4 4.2
 appletalk zone 4
!
```

Banyan VINES Routing over ISL Configuration Example

To configure routing of the Banyan VINES protocol over ISL trunks, you need to define ISL as the encapsulation type. This example shows Banyan VINES configured to be routed over an ISL trunk:

```
vines routing
interface fastethernet 0.1
 encapsulation isl 100
 vines metric 2
```

DECnet Routing over ISL Configuration Example

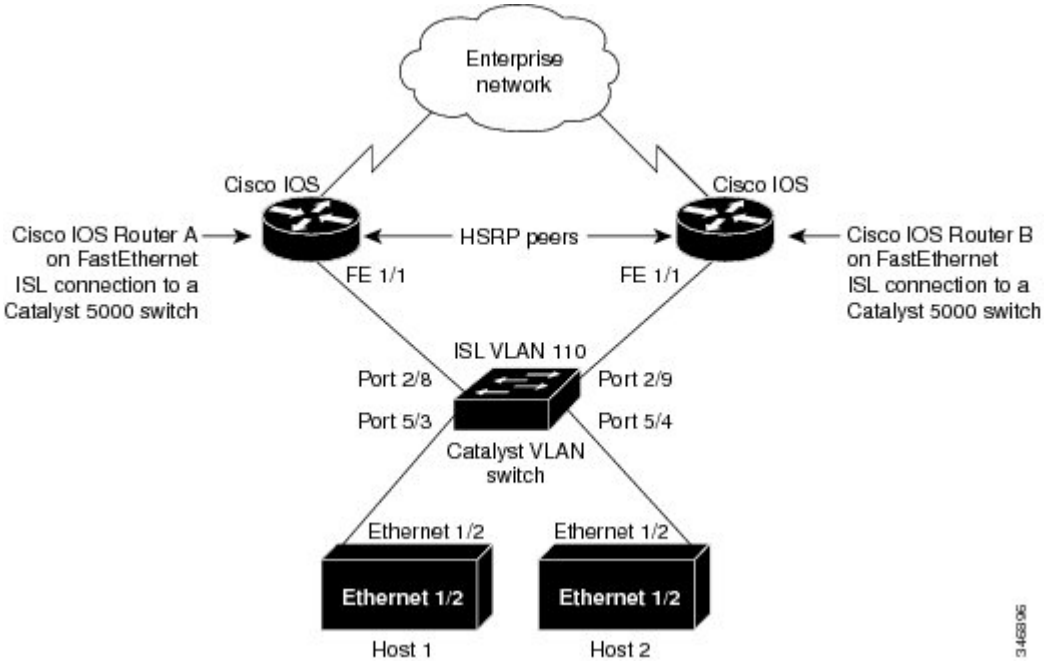
To configure routing the DECnet protocol over ISL trunks, you need to define ISL as the encapsulation type. This example shows DECnet configured to be routed over an ISL trunk:

```
decnet routing 2.1
interface fastethernet 1/0.1
 encapsulation isl 200
 decnet cost 4
```

HSRP over ISL Configuration Example

The configuration example shown in the figure below shows HSRP being used on two VLAN routers sending traffic to and from ISL VLANs through a Catalyst 5000 switch. Each router forwards its own traffic and acts as a standby for the other.

Figure 16: Hot Standby Router Protocol Sample Configuration



The topology shown in the figure above shows a Catalyst VLAN switch supporting Fast Ethernet connections to two routers running HSRP. Both routers are configured to route HSRP over ISLs.

The standby conditions are determined by the standby commands used in the configuration. Traffic from Host 1 is forwarded through Router A. Because the priority for the group is higher, Router A is the active router for Host 1. Because the priority for the group serviced by Host 2 is higher in Router B, traffic from Host 2 is forwarded through Router B, making Router B its active router.

In the configuration shown in the figure above, if the active router becomes unavailable, the standby router assumes active status for the additional traffic and automatically routes the traffic normally handled by the router that has become unavailable.

Host 1 Configuration

```
interface Ethernet 1/2
 ip address 10.1.1.25 255.255.255.0
 ip route 0.0.0.0 0.0.0.0 10.1.1.101
```

Host 2 Configuration

```
interface Ethernet 1/2
 ip address 10.1.1.27 255.255.255.0
 ip route 0.0.0.0 0.0.0.0 10.1.1.102
!
```

Router A Configuration

```
interface FastEthernet 1/1.110
 encapsulation isl 110
 ip address 10.1.1.2 255.255.255.0
 standby 1 ip 10.1.1.101
 standby 1 preempt
 standby 1 priority 105
 standby 2 ip 10.1.1.102
 standby 2 preempt
!
end
!
```

Router B Configuration

```
interface FastEthernet 1/1.110
 encapsulation isl 110
 ip address 10.1.1.3 255.255.255.0
 standby 1 ip 10.1.1.101
 standby 1 preempt
 standby 2 ip 10.1.1.102
 standby 2 preempt
 standby 2 priority 105
router igrp 1
!
network 10.1.0.0
network 10.2.0.0
!
```

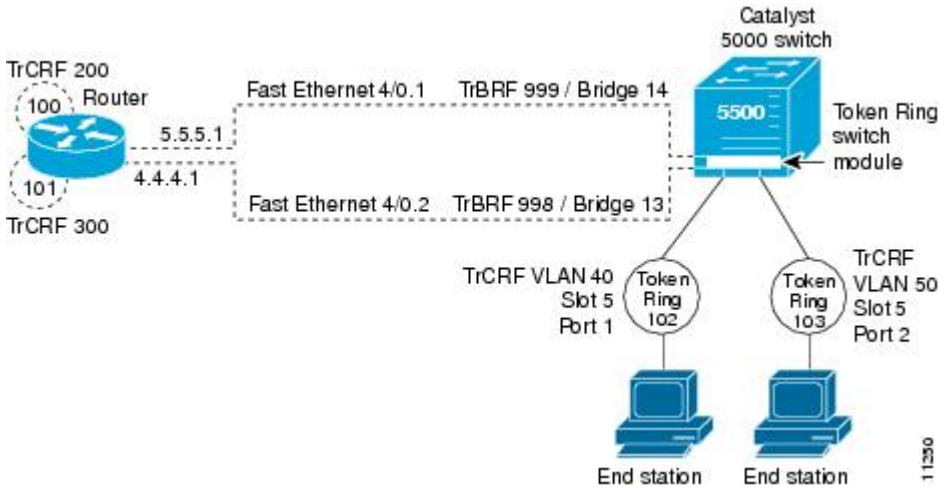
VLAN Switch Configuration

```
set vlan 110 5/4
set vlan 110 5/3
set trunk 2/8 110
set trunk 2/9 110
```

IP Routing with RIF Between TrBRF VLANs Example

The figure below shows IP routing with RIF between two TrBRF VLANs.

Figure 17: IP Routing with RIF Between TrBRF VLANs



The following is the configuration for the router:

```
interface FastEthernet4/0.1
 ip address 10.5.5.1 255.255.255.0
 encapsulation tr-isl trbrf-vlan 999 bridge-num 14
 multiring trcrf-vlan 200 ring 100
 multiring all
!
 interface FastEthernet4/0.2
 ip address 10.4.4.1 255.255.255.0
 encapsulation tr-isl trbrf-vlan 998 bridge-num 13
 multiring trcrf-vlan 300 ring 101
 multiring all
```

The following is the configuration for the Catalyst 5000 switch with the Token Ring switch module in slot 5. In this configuration, the Token Ring port 102 is assigned with TrCRF VLAN 40 and the Token Ring port 103 is assigned with TrCRF VLAN 50:

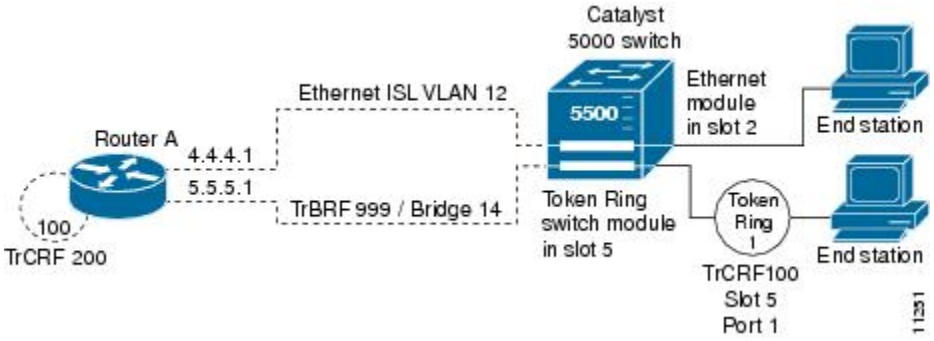
```
#vtp
set vtp domain trisl
set vtp mode server
set vtp v2 enable
#drip
set set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 999 name trbrf type trbrf bridge 0xe stp ieee
set vlan 200 name trcrf200 type trcrf parent 999 ring 0x64 mode srb
```

```
set vlan 40 name trcrf40 type trcrf parent 999 ring 0x66 mode srb
set vlan 998 name trbrf type trbrf bridge 0xd stp ieee
set vlan 300 name trcrf300 type trcrf parent 998 ring 0x65 mode srb
set vlan 50 name trcrf50 type trcrf parent 998 ring 0x67 mode srb
#add token port to trcrf 40
set vlan 40   5/1
#add token port to trcrf 50
set vlan 50   5/2
set trunk 1/2 on
```

IP Routing Between a TRISL VLAN and an Ethernet ISL VLAN Example

The figure below shows IP routing between a TRISL VLAN and an Ethernet ISL VLAN.

Figure 18: IP Routing Between a TRISL VLAN and an Ethernet ISL VLAN



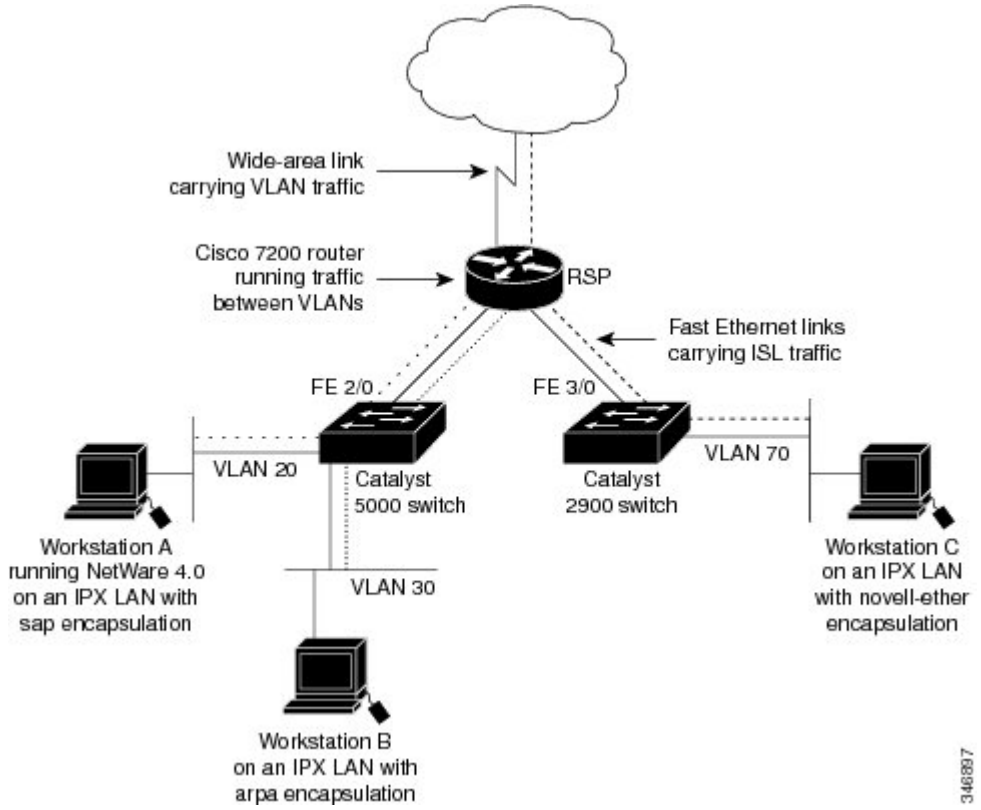
The following is the configuration for the router:

```
interface FastEthernet4/0.1
 ip address 10.5.5.1 255.255.255.0
 encapsulation tr-isl trbrf-vlan 999 bridge-num 14
 multiring trcrf-vlan 20 ring 100
 multiring all
!
interface FastEthernet4/0.2
 ip address 10.4.4.1 255.255.255.0
 encapsulation isl 12
```

IPX Routing over ISL Configuration Example

The figure below shows IPX interior encapsulations configured over ISL encapsulation in VLAN configurations. Note that three different IPX encapsulation formats are used. VLAN 20 uses SAP encapsulation, VLAN 30 uses ARPA, and VLAN 70 uses novell-ether encapsulation. Prior to the introduction of this feature, only the default encapsulation format, “novell-ether,” was available for routing IPX over ISL links in VLANs.

Figure 19: Configurable IPX Encapsulations Routed over ISL in VLAN Configurations



VLAN 20 Configuration

```
ipx routing
interface FastEthernet 2/0
 no shutdown
interface FastEthernet 2/0.20
 encapsulation isl 20
 ipx network 20 encapsulation sap
```

VLAN 30 Configuration

```
ipx routing
interface FastEthernet 2/0
 no shutdown
interface FastEthernet 2/0.30
 encapsulation isl 30
 ipx network 30 encapsulation arpa
```

VLAN 70 Configuration

```
ipx routing
interface FastEthernet 3/0
 no shutdown
interface Fast3/0.70
```

```
 encapsulation isl 70
 ipx network 70 encapsulation novell-ether
```

IPX Routing on FDDI Interfaces with SDE Example

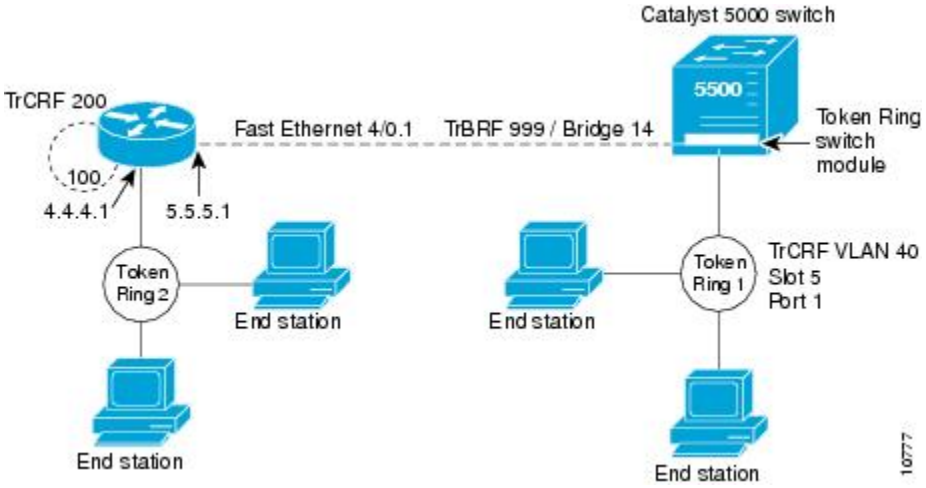
The following example enables IPX routing on FDDI interfaces 0.2 and 0.3 with SDE. On FDDI interface 0.2, the encapsulation type is SNAP. On FDDI interface 0.3, the encapsulation type is Novell's FDDI_RAW.

```
ipx routing
interface fddi 0.2 enc sde 2
 ipx network f02 encapsulation snap
interface fddi 0.3 enc sde 3
 ipx network f03 encapsulation novell-fddi
```

Routing with RIF Between a TRISL VLAN and a Token Ring Interface Example

The figure below shows routing with RIF between a TRISL VLAN and a Token Ring interface.

Figure 20: Routing with RIF Between a TRISL VLAN and a Token Ring Interface



The following is the configuration for the router:

```
source-bridge ring-group 100
!
interface TokenRing 3/1
 ip address 10.4.4.1 255.255.255.0
!
interface FastEthernet4/0.1
 ip address 10.5.5.1 255.255.255.0
 encapsulation tr-isl trbrf 999 bridge-num 14
 multiring trcrf-vlan 200 ring-group 100
 multiring all
```

The following is the configuration for the Catalyst 5000 switch with the Token Ring switch module in slot 5. In this configuration, the Token Ring port 1 is assigned to the TrCRF VLAN 40:

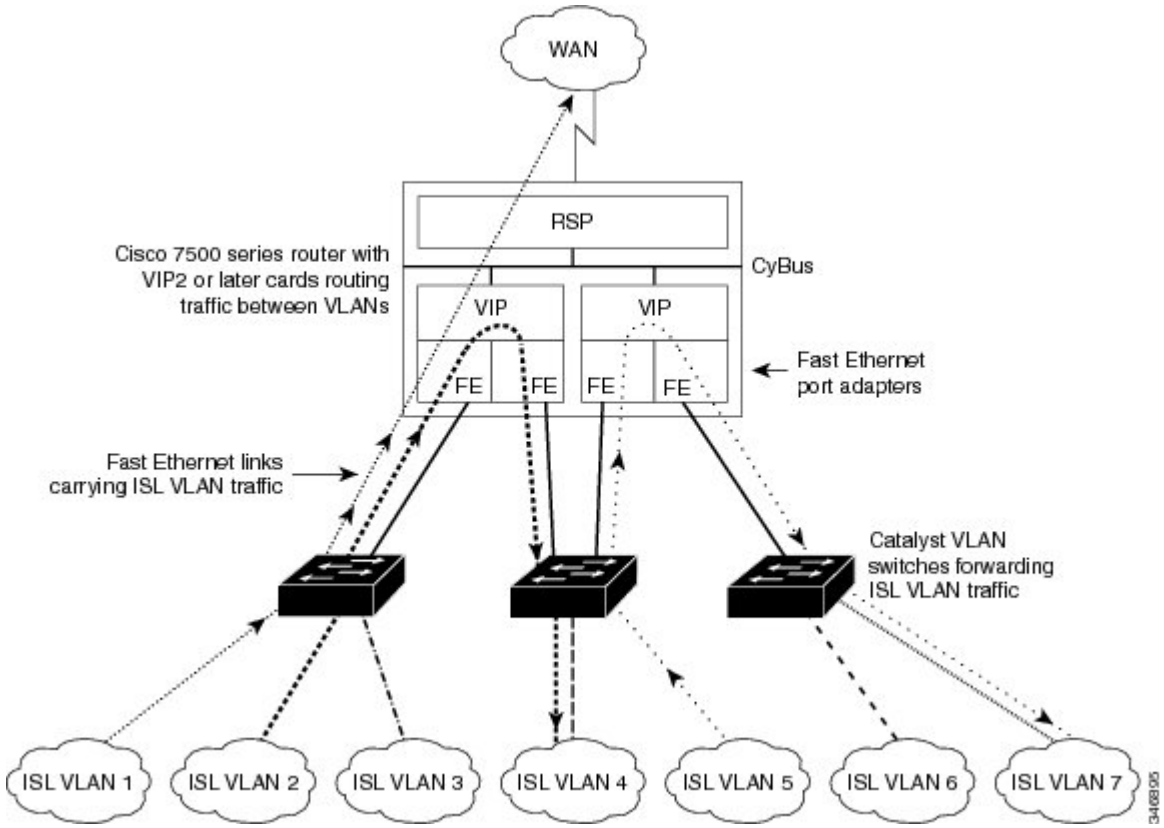
```
#vtp
set vtp domain trisl
set vtp mode server
set vtp v2 enable
#drip
```

```
set set tokenring reduction enable
set tokenring distrib-crf disable
#vlans
set vlan 999 name trbrf type trbrf bridge 0xe stp ieee
set vlan 200 name trcrf200 type trcrf parent 999 ring 0x64 mode srt
set vlan 40 name trcrf40 type trcrf parent 999 ring 0x1 mode srt
#add token port to trcrf 40
set vlan 40   5/1
set trunk 1/2 on
```

VIP Distributed Switching over ISL Configuration Example

The figure below shows a topology in which Catalyst VLAN switches are connected to routers forwarding traffic from a number of ISL VLANs. With the VIP distributed ISL capability in the Cisco 7500 series router, each VIP card can route ISL-encapsulated VLAN IP traffic. The inter-VLAN routing capacity is increased linearly by the packet-forwarding capability of each VIP card.

Figure 21: VIP Distributed ISL VLAN Traffic



In the figure above, the VIP cards forward the traffic between ISL VLANs or any other routing interface. Traffic from any VLAN can be routed to any of the other VLANs, regardless of which VIP card receives the traffic.

These commands show the configuration for each of the VLANs shown in the figure above:

```
interface FastEthernet1/0/0
 ip address 10.1.1.1 255.255.255.0
 ip route-cache distributed
 full-duplex
```

```
interface FastEthernet1/0/0.1
 ip address 10.1.1.1 255.255.255.0
 encapsulation isl 1
interface FastEthernet1/0/0.2
 ip address 10.1.2.1 255.255.255.0
 encapsulation isl 2
interface FastEthernet1/0/0.3
 ip address 10.1.3.1 255.255.255.0
 encapsulation isl 3
interface FastEthernet1/1/0
 ip route-cache distributed
 full-duplex
interface FastEthernet1/1/0.1
 ip address 172.16.1.1 255.255.255.0
 encapsulation isl 4
interface Fast Ethernet 2/0/0
 ip address 10.1.1.1 255.255.255.0
 ip route-cache distributed
 full-duplex
interface FastEthernet2/0/0.5
 ip address 10.2.1.1 255.255.255.0
 encapsulation isl 5
interface FastEthernet2/1/0
 ip address 10.3.1.1 255.255.255.0
 ip route-cache distributed
 full-duplex
interface FastEthernet2/1/0.6
 ip address 10.4.6.1 255.255.255.0
 encapsulation isl 6
interface FastEthernet2/1/0.7
 ip address 10.4.7.1 255.255.255.0
 encapsulation isl 7
```

XNS Routing over ISL Configuration Example

To configure routing of the XNS protocol over ISL trunks, you need to define ISL as the encapsulation type. This example shows XNS configured to be routed over an ISL trunk:

```
xns routing 0123.4567.adcb
interface fastethernet 1/0.1
 encapsulation isl 100
 xns network 20
```

CLNS Routing over ISL Configuration Example

To configure routing of the CLNS protocol over ISL trunks, you need to define ISL as the encapsulation type. This example shows CLNS configured to be routed over an ISL trunk:

```
clns routing
interface fastethernet 1/0.1
 encapsulation isl 100
 clns enable
```

IS-IS Routing over ISL Configuration Example

To configure IS-IS routing over ISL trunks, you need to define ISL as the encapsulation type. This example shows IS-IS configured over an ISL trunk:

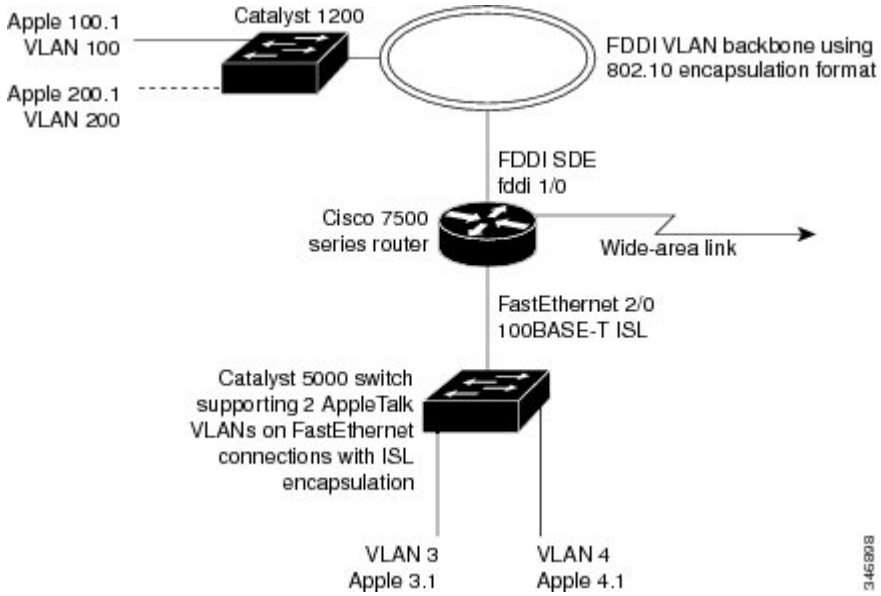
```
isis routing test-proc2
```

```
net 49.0001.0002.aaaa.aaaa.aaaa.00
interface fastethernet 2.0
 encapsulation isl 101
 clns router is-is test-proc2
```

Routing IEEE 802.10 Configuration Example

The figure below shows AppleTalk being routed between different ISL and IEEE 802.10 VLAN encapsulating subinterfaces.

Figure 22: Routing AppleTalk over VLAN encapsulations



As shown in the figure above, AppleTalk traffic is routed to and from switched VLAN domains 3, 4, 100, and 200 to any other AppleTalk routing interface. This example shows a sample configuration file for the Cisco 7500 series router with the commands entered to configure the network shown in the figure above.

Cisco 7500 Router Configuration

```
!
interface Fddi 1/0.100
 encapsulation sde 100
 appletalk cable-range 100-100 100.2
 appletalk zone 100
!
interface Fddi 1/0.200
 encapsulation sde 200
 appletalk cable-range 200-200 200.2
 appletalk zone 200
!
interface FastEthernet 2/0.3
 encapsulation isl 3
 appletalk cable-range 3-3 3.2
 appletalk zone 3
!
interface FastEthernet 2/0.4
 encapsulation isl 4
```

```
 appletalk cable-range 4-4 4.2
 appletalk zone 4
!
```

IEEE 802.1Q Encapsulation Configuration Examples

Configuration examples for each protocols are provided in the following sections:

Configuring AppleTalk over IEEE 802.1Q Example

This configuration example shows AppleTalk being routed on VLAN 100:

```
!
appletalk routing
!
interface fastethernet 4/1.100
  encapsulation dot1q 100
  appletalk cable-range 100-100 100.1
  appletalk zone eng
!
```

Configuring IP Routing over IEEE 802.1Q Example

This configuration example shows IP being routed on VLAN 101:

```
!
ip routing
!
interface fastethernet 4/1.101
  encapsulation dot1q 101
  ip addr 10.0.0.11 255.0.0.0
!
```

Configuring IPX Routing over IEEE 802.1Q Example

This configuration example shows IPX being routed on VLAN 102:

```
!
ipx routing
!
interface fastethernet 4/1.102
  encapsulation dot1q 102
  ipx network 100
!
```

VLAN 100 for Bridge Group 1 with Default VLAN1 Example

The following example configures VLAN 100 for bridge group 1 with a default VLAN1:

```
interface FastEthernet 4/1.100
encapsulation dot1q 1
bridge-group 1
```

VLAN 20 for Bridge Group 1 with Native VLAN Example

The following example configures VLAN 20 for bridge group 1 as a native VLAN:

```
interface FastEthernet 4/1.100
encapsulation dot1q 20 native
bridge-group 1
```

VLAN ISL or IEEE 802.1Q Routing Example

The following example configures VLAN ISL or IEEE 802.10 routing:

```
ipx routing
appletalk routing
!
interface Ethernet 1
ip address 10.1.1.1 255.255.255.0
appletalk cable-range 1-1 1.1
appletalk zone 1
ipx network 10 encapsulation snap
!
router igrp 1
network 10.1.0.0
!
end
!
#Catalyst5000
!
set VLAN 110 2/1
set VLAN 120 2/2
!
set trunk 1/1 110,120
# if 802.1Q, set trunk 1/1 nonegotiate 110, 120
!
end
!
ipx routing
appletalk routing
!
interface FastEthernet 1/1.110
encapsulation isl 110
!if 802.1Q, encapsulation dot1Q 110
ip address 10.1.1.2 255.255.255.0
appletalk cable-range 1.1 1.2
appletalk zone 1
ipx network 110 encapsulation snap
!
interface FastEthernet 1/1.120
encapsulation isl 120
!if 802.1Q, encapsulation dot1Q 120
ip address 10.2.1.2 255.255.255.0
appletalk cable-range 2-2 2.2
appletalk zone 2
ipx network 120 encapsulation snap
!
router igrp 1
network 10.1.0.0
network 10.2.1.0.0
!
end
!
ipx routing
appletalk routing
!
interface Ethernet 1
ip address 10.2.1.3 255.255.255.0
```

```
appletalk cable-range 2-2 2.3
appletalk zone 2
ipx network 120 encapsulation snap
!
router igrp 1
network 10.2.0.0
!
end
```

VLAN IEEE 802.1Q Bridging Example

The following examples configures IEEE 802.1Q bridging:

```
interface FastEthernet4/0
 no ip address
 no ip route-cache
 half-duplex
!
interface FastEthernet4/0.100
 encapsulation dot1Q 100
 no ip route-cache
 bridge-group 1
!
interface FastEthernet4/0.200
 encapsulation dot1Q 200 native
 no ip route-cache
 bridge-group 2
!
interface FastEthernet4/0.300
 encapsulation dot1Q 1
 no ip route-cache
 bridge-group 3
!
interface FastEthernet10/0
 no ip address
 no ip route-cache
 half-duplex
!
interface FastEthernet10/0.100
 encapsulation dot1Q 100
 no ip route-cache
 bridge-group 1
!
interface Ethernet11/3
 no ip address
 no ip route-cache
 bridge-group 2
!
interface Ethernet11/4
 no ip address
 no ip route-cache
 bridge-group 3
!
bridge 1 protocol ieee
bridge 2 protocol ieee
bridge 3 protocol ieee
```

VLAN IEEE 802.1Q IRB Example

The following examples configures IEEE 802.1Q integrated routing and bridging:

```
ip cef
appletalk routing
ipx routing 0060.2f27.5980
!
bridge irb
!
interface TokenRing3/1
 no ip address
 ring-speed 16
 bridge-group 2
!
interface FastEthernet4/0
 no ip address
 half-duplex
!
interface FastEthernet4/0.100
 encapsulation dot1Q 100
 bridge-group 1
!
interface FastEthernet4/0.200
 encapsulation dot1Q 200
 bridge-group 2
!
interface FastEthernet10/0
ip address 10.3.1.10 255.255.255.0
 half-duplex
 appletalk cable-range 200-200 200.10
 appletalk zone irb
 ipx network 200
!
interface Ethernet11/3
 no ip address
 bridge-group 1
!
interface BVI 1
 ip address 10.1.1.11 255.255.255.0
 appletalk cable-range 100-100 100.11
 appletalk zone bridging
 ipx network 100
!
router rip
 network 10.0.0.0
 network 10.3.0.0
!
bridge 1 protocol ieee
 bridge 1 route appletalk
 bridge 1 route ip
 bridge 1 route ipx
bridge 2 protocol ieee
!
```

Configuring IEEE 802.1Q-in-Q VLAN Tag Termination Example

Some ambiguous subinterfaces can use the **any** keyword for the inner VLAN ID specification. The **any** keyword represents any inner VLAN ID that is not explicitly configured on any other interface. In the following example, seven subinterfaces are configured with various outer and inner VLAN IDs.

Note The **any** keyword can be configured on only one subinterface of a specified physical interface and outer VLAN ID.

```
interface GigabitEthernet1/0/0.1
 encapsulation dot1q 100 second-dot1q 100
interface GigabitEthernet1/0/0.2
 encapsulation dot1q 100 second-dot1q 200
interface GigabitEthernet1/0/0.3
 encapsulation dot1q 100 second-dot1q 300-400,500-600
interface GigabitEthernet1/0/0.4
 encapsulation dot1q 100 second-dot1q any
interface GigabitEthernet1/0/0.5
 encapsulation dot1q 200 second-dot1q 50
interface GigabitEthernet1/0/0.6
 encapsulation dot1q 200 second-dot1q 1000-2000,3000-4000
interface GigabitEthernet1/0/0.7
 encapsulation dot1q 200 second-dot1q any
```

The table below shows which subinterfaces are mapped to different values of the outer and inner VLAN ID on Q-in-Q frames that come in on Gigabit Ethernet interface 1/0/0.

Table 9: Subinterfaces Mapped to Outer and Inner VLAN IDs for GE Interface 1/0/0

Outer VLAN ID	Inner VLAN ID	Subinterface mapped to
100	1 through 99	GigabitEthernet1/0/0.4
100	100	GigabitEthernet1/0/0.1
100	101 through 199	GigabitEthernet1/0/0.4
100	200	GigabitEthernet1/0/0.2
100	201 through 299	GigabitEthernet1/0/0.4
100	300 through 400	GigabitEthernet1/0/0.3
100	401 through 499	GigabitEthernet1/0/0.4
100	500 through 600	GigabitEthernet1/0/0.3
100	601 through 4095	GigabitEthernet1/0/0.4
200	1 through 49	GigabitEthernet1/0/0.7
200	50	GigabitEthernet1/0/0.5
200	51 through 999	GigabitEthernet1/0/0.7
200	1000 through 2000	GigabitEthernet1/0/0.6
200	2001 through 2999	GigabitEthernet1/0/0.7
200	3000 through 4000	GigabitEthernet1/0/0.6

Outer VLAN ID	Inner VLAN ID	Subinterface mapped to
200	4001 through 4095	GigabitEthernet1/0/0.7

A new subinterface is now configured:

```
interface GigabitEthernet1/0/0.8
 encapsulation dot1q 200 second-dot1q 200-600,900-999
```

The table below shows the changes made to the table for the outer VLAN ID of 200. Notice that subinterface 1/0/0.7 configured with the **any** keyword now has new inner VLAN ID mappings.

Table 10: Subinterfaces Mapped to Outer and Inner VLAN IDs for GE Interface 1/0/0--Changes Resulting from Configuring GE Subinterface 1/0/0.8

Outer VLAN ID	Inner VLAN ID	Subinterface mapped to
200	1 through 49	GigabitEthernet1/0/0.7
200	50	GigabitEthernet1/0/0.5
200	51 through 199	GigabitEthernet1/0/0.7
200	200 through 600	GigabitEthernet1/0/0.8
200	601 through 899	GigabitEthernet1/0/0.7
200	900 through 999	GigabitEthernet1/0/0.8
200	1000 through 2000	GigabitEthernet1/0/0.6
200	2001 through 2999	GigabitEthernet1/0/0.7
200	3000 through 4000	GigabitEthernet1/0/0.6
200	4001 through 4095	GigabitEthernet1/0/0.7

Additional References

The following sections provide references related to the Managed LAN Switch feature.

Related Documents

Related Topic	Document Title
IP LAN switching commands: complete command syntax, command mode, defaults, usage guidelines, and examples	Cisco IOS LAN Switching Services Command Reference
LAN switching	“LAN Switching” module of the *Internetworking Technology Handbook*

Standards

Standards	Title
No new or modified RFCs are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIBs	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature, and support for existing standards has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Routing Between VLANs

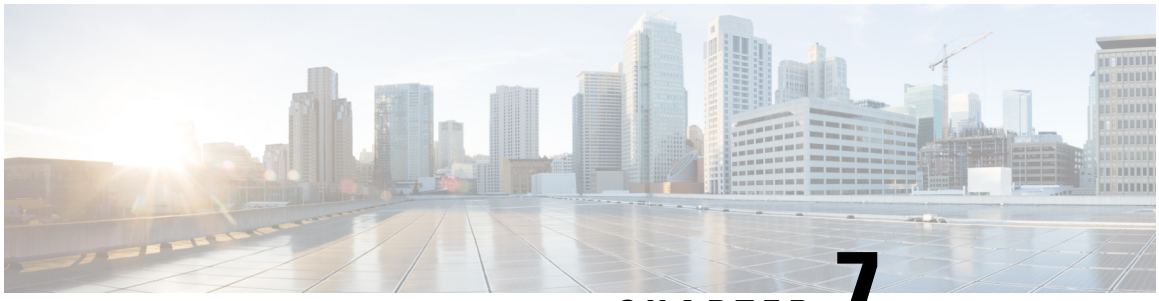
The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 11: Feature Information for Routing Between VLANs

Feature Name	Releases	Feature Information
IEEE 802.1Q-in-Q VLAN Tag Termination	12.0(28)S, 12.3(7)(X17) 12.0(32)S1, 12.2(31)SB 12.3(7)T 12.3((7)XI1	Encapsulating IEEE 802.1Q VLAN tags within 802.1Q enables service providers to use a single VLAN to support customers who have multiple VLANs. The IEEE 802.1Q-in-Q VLAN Tag Termination feature on the subinterface level preserves VLAN IDs and keeps traffic in different customer VLANs segregated.
Configuring Routing Between VLANs with IEEE 802.1Q Encapsulation	12.0(7)XE 12.1(5)T 12.2(2)DD 12.2(4)B 12.2(8)T 12.2(13)T Cisco IOS XE 3.8(S) Cisco IOS XE 3.9(S)	The IEEE 802.1Q protocol is used to interconnect multiple switches and routers, and for defining VLAN topologies. The IEEE 802.1Q standard is extremely restrictive to untagged frames. The standard provides only a per-port VLANs solution for untagged frames. For example, assigning untagged frames to VLANs takes into consideration only the port from which they have been received. Each port has a parameter called a *permanent virtual identification* (Native VLAN) that specifies the VLAN assigned to receive untagged frames. In Cisco IOS XE Release 3.8(S), support was added for the Cisco ISR 4400 Series Routers. In Cisco IOS XE Release 3.9(S), support was added for the Cisco CSR 1000V Series Routers.
Configuring Routing Between VLANs with Inter-Switch Link Encapsulation	12.0(7)XE 12.1(5)T 12.2(2)DD 12.2(4)B 12.2(8)T 12.2(13)T	ISL is a Cisco protocol for interconnecting multiple switches and maintaining VLAN information as traffic goes between switches. ISL provides VLAN capabilities while maintaining full wire speed performance on Fast Ethernet links in full- or half-duplex mode. ISL operates in a point-to-point environment and will support up to 1000 VLANs. You can define virtually as many logical networks as are necessary for your environment.
Configuring Routing Between VLANs with IEEE 802.10 Encapsulation	12.0(7)XE 12.1(5)T 12.2(2)DD 12.2(4)B 12.2(8)T 12.2(13)T	AppleTalk can be routed over VLAN subinterfaces using the ISL or IEEE 802.10 VLANs feature that provides full-feature Cisco IOS software AppleTalk support on a per-VLAN basis, allowing standard AppleTalk capabilities to be configured on VLANs.

Feature Name	Releases	Feature Information
VLAN Range	12.0(7)XE 12.1(5)T 12.2(2)DD 12.2(4)B 12.2(8)T 12.2(13)T	Using the VLAN Range feature, you can group VLAN subinterfaces together so that any command entered in a group applies to every subinterface within the group. This capability simplifies configurations and reduces command parsing. In Cisco IOS Release 12.0(7)XE, the **interface range** command was introduced. The **interface range** command was integrated into Cisco IOS Release 12.1(5)T. In Cisco IOS Release 12.2(2)DD, the **interface range** command was expanded to enable configuration of subinterfaces. The **interface range** command was integrated into Cisco IOS Release 12.2(4)B. The VLAN Range feature was integrated into Cisco IOS Release 12.2(8)T. This VLAN Range feature was integrated into Cisco IOS Release 12.2(13)T.
256+ VLANS	12.1(2)E, 12.2(8)T Cisco IOS XE 3.8(S) Cisco IOS XE 3.9(S)	The 256+ VLAN feature enables a device to route more than 256 VLAN interfaces. This feature requires the MSFC2. The routed VLAN interfaces can be chosen from any of the VLANs supported on the device. Catalyst switches can support up to 4096 VLANs. If MSFC is used, up to 256 VLANs can be routed, but this can be selected from any VLANs supported on the device. In Cisco IOS XE Release 3.8(S), support was added for the Cisco ISR 4400 Series Routers. In Cisco IOS XE Release 3.9(S), support was added for the Cisco CSR 1000V Series Routers.



CHAPTER 7

EtherChannel Flow-Based Limited 1 1 Redundancy

EtherChannel flow-based limited 1:1 redundancy provides MAC, or layer 2, traffic protection to avoid higher layer protocols from reacting to single link failures and re-converging. To use EtherChannel flow-based limited 1:1 redundancy, you configure an EtherChannel with two ports (one active and one standby). If the active link goes down, the EtherChannel stays up and the system performs fast switchover to the hot-standby link. Depending on how you have the priorities set, when the failed link becomes operational again, the EtherChannel performs another fast switchover to revert to the original active link. if all port-priorities are the same, it will not revert, but remain on the current active link.

With 1:1 redundancy configured, only one link is active at any given time so all flows are directed over the active link.

- Restrictions for EtherChannel Flow-based Limited 1:1 Redundancy, on page 119
- Information About EtherChannel Flow-Based Limited 1 1 Redundancy, on page 120
- How to Configure EtherChannel Flow-Based Limited 1 1 Redundancy, on page 120
- Configuration Examples for EtherChannel Flow-Based Limited1 1 Redundancy, on page 125
- Additional References, on page 126
- Feature Information for EtherChannel Flow-based Limited 1 1 Redundancy, on page 127

Restrictions for EtherChannel Flow-based Limited 1:1 Redundancy

When you are using the Cisco ASR 1001-X, the following restrictions apply for collecting traffic statistics for VLAN egress on sub-interfaces. Obtaining input/output counters using SNMP is unsupported. This is because the Cisco ASR 1001-X has a built-in SPA.

Restrictions that apply when obtaining traffic statistics for two types of interfaces are shown below:

- **Physical sub-interfaces**

  For the Cisco ASR 1001-X, statistics for the VLAN egress are available for physical sub-interfaces. The output counter is used from cpp, not from the built-in SPA hardware. To show VLAN egress statistics, use the **show vlans** *vlan id* command.

  Example

```
# show vlans 10
VLAN ID: 10 (IEEE 802.1Q Encapsulation)
>
>    Protocols Configured:          Received:        Transmitted:
>                     IP             133                104
```

- **Port Channel sub-interfaces**

  For the Cisco ASR 1001-X, showing traffic statistics for the VLAN egress is not supported for port channel sub-interfaces.

  cpp or the built-in SPA can not be used to give an output counter value for port channel sub-interfaces.

Information About EtherChannel Flow-Based Limited 1 1 Redundancy

EtherChannel Flow-Based Limited 1 1 Redundancy

EtherChannel flow-based limited 1:1 redundancy provides an EtherChannel configuration with one active link and fast switchover to a hot standby link. To use EtherChannel flow-based limited 1:1 redundancy, you configure a Link Aggregation Control Protocol (LACP) EtherChannel with two ports (one active and one standby). If the active link goes down, the EtherChannel stays up and the system performs fast switchover to the hot standby link. Depending on how the priorities of the links are set, when the failed link becomes operational again, the EtherChannel performs another fast switchover to revert to the original active link, or to the link with the higher priority.

For EtherChannel flow-based limited 1:1 redundancy to work correctly (especially the fast switchover capability) the feature must be enabled at both ends of the link.

How to Configure EtherChannel Flow-Based Limited 1 1 Redundancy

Configuring EtherChannel Flow-Based Limited 1 1 Redundancy with Fast-Switchover

To configure an LACP EtherChannel with two ports (one active and one standby), perform the following steps. This feature must be enabled at both ends of the link.

You can control which link is the primary active link by setting the port priority on the links used for the redundancy. To configure a primary link and enable the EtherChannel to revert to the original link, one link must have a higher port priority than the other and the LACP max-bundle must be set to 1. This configuration results in link 1 being active and link 2 being in hot standby state.

To prevent the switchover to revert, you can assign both links the same priority.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *channel -number*
4. **lacp fast-switchover**
5. **lacp max-bundle 1**
6. **exit**
7. **interface tengigabitethernet** *slot / port / number*
8. **channel-group 1 mode** *mode*
9. **lacp port-priority** *priority*
10. **exit**
11. **interface tengigabitethernet** *slot / port / number*
12. **channel-group 1 mode** *mode*
13. **lacp port-priority** *priority*
14. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *channel -number* **Example:** `Router(config)# interface port-channel 1`	Selects an LACP port channel interface.
Step 4	**lacp fast-switchover** **Example:** `Router(config-if)# lacp fast-switchover`	Enables the fast switchover feature for this EtherChannel.
Step 5	**lacp max-bundle 1** **Example:** `Router(config-if)# lacp max-bundle 14`	Sets the maximum number of active member ports to 14. **Note** For Cisco ASR 1000 Series Aggregation Services Routers, the minimum number of active member ports is 1 and the maximum number is 14.

	Command or Action	Purpose
Step 6	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 7	**interface tengigabitethernet** *slot / port / number* **Example:** `Router(config)# interface tengigabitethernet 0/0/0`	Selects the first interface to add to the port channel.
Step 8	**channel-group 1 mode** *mode* **Example:** `Router(config-if)# channel-group 1 mode active`	Adds the member link to the port-channel and actively participates in LACP negotiation.
Step 9	**lacp port-priority** *priority* **Example:** `Router(config-if)# lacp port-priority 32768`	Sets the priority on the port-channel. This priority is set to the default value.
Step 10	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 11	**interface tengigabitethernet** *slot / port / number* **Example:** `Router(config)# interface tengigabitethernet 1/0/0`	Selects the interface to add to the port channel.
Step 12	**channel-group 1 mode** *mode* **Example:** `Router(config-if)# channel-group 1 mode active`	Adds the member link to the port-channel and actively participates in LACP negotiation.
Step 13	**lacp port-priority** *priority* **Example:** `Router(config-if)# lacp port-priority 32767`	Sets the port priority higher than the other link by using a value lower than the default value of 32768. This forces this link to be the active link whenever it is capable of carrying traffic.
Step 14	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode.

Setting the Switchover Rate with Carrier Delay

Optionally, you can control the speed of the switchover between the active and standby links by setting the carrier delay on each link. The **carrier-delay** command controls how long it takes for Cisco IOS to propagate the information about the links status to other modules.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tengigabitethernet** *slot* / *port* / *number*
4. **carrier-delay msec** *msec*
5. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface tengigabitethernet** *slot* / *port* / *number* **Example:** `Router(config)# interface tengigabitethernet 0/1/0`	Enters interface configuration mode and opens the configuration for the specified interface.
Step 4	**carrier-delay msec** *msec* **Example:** `Router(config-if)# carrier-delay msec 11`	Sets how long it takes to propagate the link status to other modules.
Step 5	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode.

Verifying EtherChannel Flow-Based Limited 1 1 Redundancy

Use these show commands to verify the configuration and to display information about the port channel.

SUMMARY STEPS

1. **enable**
2. **show running-config interface** *type slot / port / number*
3. **show interfaces port-channel** *channel-number* **etherchannel**
4. **show etherchannel** *channel-number* **port-channel**
5. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show running-config interface** *type slot / port / number* **Example:** `Router# show running-config interface tengigabitethernet 0/0/0`	Verifies the configuration. • *type* --**gigabitethernet** or **tengigabitethernet**.
Step 3	**show interfaces port-channel** *channel-number* **etherchannel** **Example:** `Router# show interfaces port-channel 1 etherchannel`	Displays the bucket distribution currently in use.
Step 4	**show etherchannel** *channel-number* **port-channel** **Example:** `Router# show etherchannel 1 port-channel`	Displays the port channel fast-switchover feature capability.
Step 5	**end** **Example:** `Router# end`	Exits privileged EXEC mode.

Configuration Examples for EtherChannel Flow-Based Limited1 1 Redundancy

EtherChannel 1 1 Active Standby Example

This example shows how to configure a port channel for 1:1 link redundancy for equal priority ports so there is no preference which port is active.

```
Router# enable
Router# configure terminal
Router(config)# interface port-channel 2
Router(config-if)# ip address 10.1.1.1 255.255.0.0
Router(config-if)# negotiation auto
Router(config-if)# lacp max-bundle 1
Router(config-if)# lacp fast-switchover
Router(config)# interface Tengigabitethernet0/1/0
Router(config-if)# channel-group 2 mode active
Router(config-if)# negotiation auto
Router(config)# interface Tengigabitethernet 2/1/0
Router(config-if)# channel-group 2 mode active
Router(config-if)# negotiation auto
Router(config)# interface GigabitEthernet0/1/6
Router(config-if)# negotiation auto
Router(config-if)# channel-group 19 mode active
Router(config)# interface GigabitEthernet0/1/7
Router(config-if)# negotiation auto
Router(config-if)# channel-group 19 mode active
Router(config-if)# interface Port-channel19
Router(config-if)# ip address 10.19.1.1 255.255.255.0
Router(config-if)# no negotiation auto
Router(config-if)# lacp fast-switchover
Router(config-if)# lacp max-bundle 1
Router(config-if)# end
```

Notice in the **show** command display the priorities are the same value.

```
Router# show lacp internal
Flags: S - Device is requesting Slow LACPDUs
 F - Device is requesting Fast LACPDUs
 A - Device is in Active mode P - Device is in Passive mode
Channel group 19
LACP port Admin Oper Port Port
Port Flags State Priority Key Key Number State
Gi0/1/6 SA bndl 32768 0x13 0x13 0x47 0x3D
Gi0/1/7 FA hot-sby 32768 0x13 0x13 0x48 0x7
```

Setting Priority for 1 1 Redundancy Using LACP Example

This example shows how to configure an LACP EtherChannel with 1:1 redundancy. GigabitEthernet 0/1/7 is the active link, because it is configured with a lower number which give it a higher port priority.

```
Router# configure terminal
Router(config)# interface GigabitEthernet0/1/6
```

```
Router(config-if)# lacp port-priority 32767
Router(config-if)# exit
Router(config)# interface GigabitEthernet0/1/7
Router(config-if)# lacp fast-switchover
Router(config-if)# lacp max-bundle 1
Router(config-if)# negotiation auto
Router(config-if)# channel-group 19 mode active
```

In this show display, notice that the bundled link is set at a higher priority. This will ensure that the bundled link is used as the first active link in the standby configuration.

```
Router# show lacp internal

Flags: S - Device is requesting Slow LACPDUs
 F - Device is requesting Fast LACPDUs
 A - Device is in Active mode P - Device is in Passive mode
Channel group 19
LACP port Admin Oper Port Port
Port Flags State Priority Key Key Number State
Gi0/1/6 FA hot-sby 32768 0x13 0x13 0x47 0x7
Gi0/1/7 SA bndl 32767 0x13 0x13 0x48 0x3D
```

Additional References

The following sections provide references related to the EtherChannel Flow-based Limited1:1 Redundancy feature.

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
LAN Switching commands	*Cisco IOS LAN Switching Command Reference*

Standards

Standard	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported by this feature, and support for existing standards has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

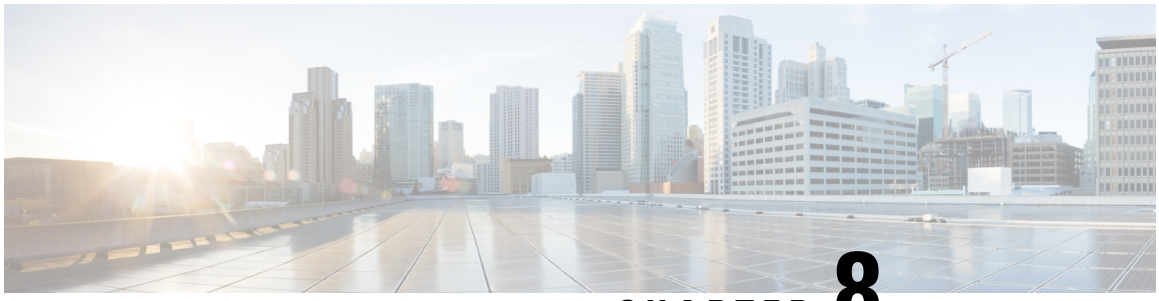
Feature Information for EtherChannel Flow-based Limited 1 1 Redundancy

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 12: Feature Information for EtherChannel Flow-based Limited 1:1 Redundancy

Feature Name	Releases	Feature Information
EtherChannel Flow-Based Limited 1:1 Redundancy	Cisco IOS XE Release 2.4	EtherChannel flow-based limited 1:1 redundancy provides MAC, or layer 2, traffic protection to avoid higher layer protocols from reacting to single link failures and re-converging. To use EtherChannel flow-based limited 1:1 redundancy, you configure an EtherChannel with two ports (one active and one standby). If the active link goes down, the EtherChannel stays up and the system performs fast switchover to the hot-standby link. Depending on how you have the priorities set, when the failed link becomes operational again, the EtherChannel performs another fast switchover to revert to the original active link. if all port-priorities are the same, it will not revert, but remain on the current active link. No commands were modified or created to support this feature.



CHAPTER 8

Flow-Based per Port-Channel Load Balancing

The Flow-Based per Port-Channel Load Balancing feature allows different flows of traffic over a Gigabit EtherChannel (GEC) interface to be identified based on the packet header and then mapped to the different member links of the port channel. This feature enables you to apply flow-based load balancing and VLAN-manual load balancing to specific port channels.

- Restrictions for Flow-Based per Port-Channel Load Balancing, on page 129
- Information About Flow-Based per Port-Channel Load Balancing, on page 129
- How to Enable Flow-Based per Port-Channel Load Balancing, on page 132
- Configuration Examples for Flow-Based per Port-Channel Load Balancing, on page 135
- Information About Five-Tuple Hash Support for GEC Flow-based Load Balancing, on page 136
- Additional References, on page 136
- Feature Information for Flow-Based per Port-Channel Load Balancing, on page 138

Restrictions for Flow-Based per Port-Channel Load Balancing

- Supports up to 64 GEC interfaces.
- Supports up to 14 member links per GEC interface.

Note This feature achieves load balancing of MPLS traffic only by using source IP address and destination IP address. The MPLS label is not considered for load balancing.

Information About Flow-Based per Port-Channel Load Balancing

Flow-Based Load Balancing

Flow-based load balancing identifies different flows of traffic based on the key fields in the data packet. For example, IPv4 source and destination IP addressees can be used to identify a flow. The various data traffic flows are then mapped to the different member links of a port channel. After the mapping is done, the data traffic for a flow is transmitted through the assigned member link. The flow mapping is dynamic and changes when there is any change in the state of a member link to which a flow is assigned. The flow mappings can

also change if member links are added to or removed from the GEC interface. Multiple flows can be mapped to each member link.

Buckets for Flow-Based Load Balancing

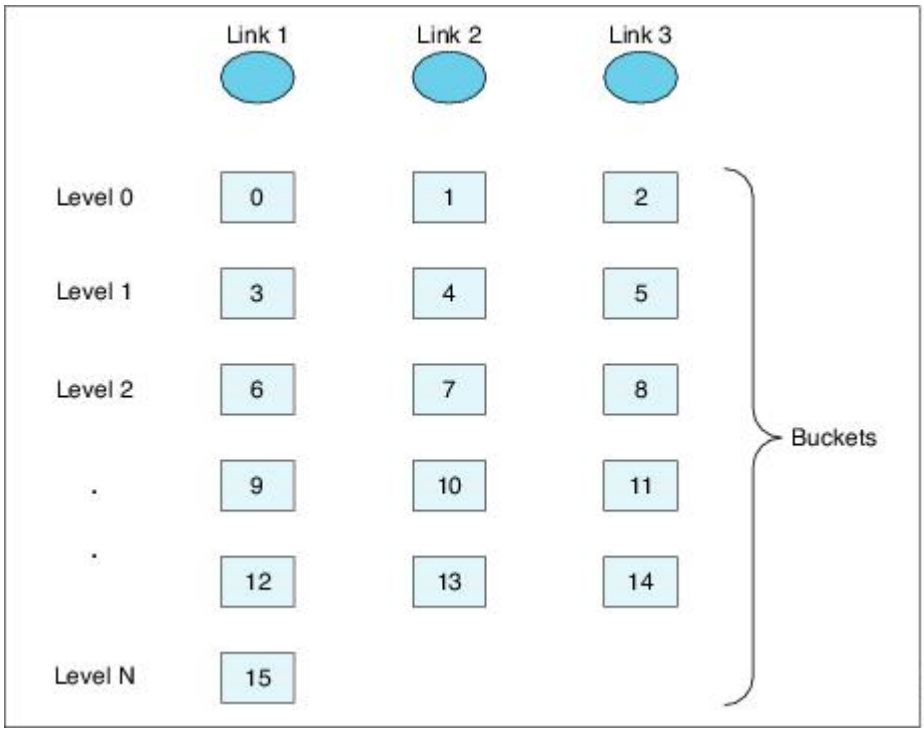
Load balancing dynamically maps traffic flows to the member links of a GEC interface through the concept of buckets. The various defined traffic flows are mapped to the buckets and the buckets are evenly distributed among the member links. Each port channel maintains 16 buckets, with one active member link associated with each bucket. All traffic flows mapped to a bucket use the member link to which the bucket is assigned.

The router creates the buckets-to-member links mappings when you apply flow-based load balancing to a port channel and the port channel has at least one active member link. The mappings are also created when the first member link is added, or comes up, and the load-balancing method is set to flow-based.

When a member link goes down or is removed from a port channel, the buckets associated with that member link are redistributed among the other active member links in a round-robin fashion. When a member link comes up or is added to a port channel, some of the buckets associated with other links are assigned to this link.

The figure below illustrates an example of 16 buckets distributed among three member links. The numbers shown in the buckets are the bucket IDs. Note that the first member link has an extra bucket.

Figure 23: Example of 16 Buckets Mapped to Three Member Links



If you change the load-balancing method, the bucket-to-member link mappings for flow-based load balancing are deleted. The mappings are also deleted if the port channel is deleted or the last member link in the port channel is deleted or goes down.

Load Balancing on Port Channels

GEC interfaces can use either dynamic flow-based load balancing or VLAN-manual load balancing. You can configure the load-balancing method globally for all port channels or directly on specific port channels. The global configuration applies only to those port channels for which you have not explicitly configured load balancing. The port-channel configuration overrides the global configuration.

Flow-based load balancing is enabled by default at the global level. You must explicitly configure VLAN load balancing or the load-balancing method is flow-based.

For more information about configuring VLAN load balancing, see the module VLAN Mapping to Gigabit EtherChannel (GEC) Member Links.

The table below lists the load-balancing method that is applied to port channels based on the configuration:

Table 13: Flow-Based Load Balancing Configuration Options

Global Configuration	Port-Channel Configuration	Load Balancing Applied
Not configured	Not configured	Flow-based
	Flow-based	Flow-based
	VLAN-manual	VLAN-manual
VLAN-manual	Not configured	VLAN-manual
	Flow-based	Flow-based
	VLAN-manual	VLAN-manual

The table below lists the configuration that results if you change the global load-balancing method.

Table 14: Results When Global Configuration Changes

Port-Channel Configuration	Global Configuration	Action Taken at Port Channel	
–	From	To	–
Not configured	Not configured	VLAN-manual	Changed from flow-based to VLAN-manual
	VLAN-manual	Not configured	Changed from VLAN-manual to flow-based
Configured	Any	Any	No change

The table below lists the configuration that results if you change the port-channel load-balancing method.

Table 15: Results When Port-Channel Configuration Changes

Global Configuration	Port-Channel Configuration	Action Taken at Port Channel	
–	From	To	–

Global Configuration	Port-Channel Configuration	Action Taken at Port Channel	
Not configured	Not configured	VLAN-manual	Changed from flow-based to VLAN-manual
	Not configured	Flow-based	No action taken
	VLAN-manual	Flow-based	Changed from VLAN-manual to flow-based
	VLAN-manual	Not configured	Changed from VLAN-manual to flow-based
	Flow-based	VLAN-manual	Changed from flow-based to VLAN-manual
	Flow-based	Not configured	No action taken
VLAN-manual	Not configured	VLAN-manual	No action taken
	Not configured	Flow-based	Changed from VLAN-manual to flow-based
	VLAN-manual	Flow-based	Changed from VLAN-manual to flow-based
	VLAN-manual	Not configured	No action taken
	Flow-based	VLAN-manual	Changed from flow-based to VLAN-manual
	Flow-based	Not configured	Changed from flow-based to VLAN-manual

How to Enable Flow-Based per Port-Channel Load Balancing

Configuring Load Balancing on a Port Channel

To configure load balancing on a port channel, perform the following steps. Repeat these steps for each GEC interface.

Before you begin

If you have already configured your desired load-balancing method globally and want to use that method for all port channels, you need not perform this task. To configure load balancing globally, use the **port-channel load-balancing vlan-manual** command. If you do not configure the global command, flow-based load balancing is applied to all port channels.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *channel-number*
4. **load-balancing** {**flow** | **vlan**}
5. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *channel-number* **Example:** `Router(config)# interface port-channel 1`	Enters interface configuration mode and defines the interface as a port channel.
Step 4	**load-balancing** {**flow**	**vlan**} **Example:** `Router(config-if)# load-balancing flow`	Applies a load-balancing method to the specific port channel. • If you do not configure this command, the port channel uses the global load-balancing method configured with the **port-channel load-balancing vlan-manual** command. The global default is flow-based.
Step 5	**end** **Example:** `Router(config-if)# end`	Exits configuration mode.

Verifying Load-Balancing Configuration on a GEC Interface

Use these show commands to verify the load-balancing configuration and to display information about the bucket distribution on the port channel. You can use these commands in any order.

SUMMARY STEPS

1. **show running-config interface port-channel** *channel-number*
2. **show etherchannel load-balancing**
3. **show interfaces port-channel** *channel-number* **etherchannel**

DETAILED STEPS

Procedure

Step 1 **show running-config interface port-channel** *channel-number*

Use this command to verify the configuration of the port channel.

Example:

```
Router# show running-config interface port-channel 1
Building configuration...

Current configuration : 88 bytes
!
interface Port-channel1
 ip address 10.1.1.1 255.0.0.0
 no negotiation auto
 load-balancing flow
end
```

Step 2 **show etherchannel load-balancing**

Use this command to display the load-balancing method applied to each port channel. The following example shows output for a configuration with load balancing set globally to VLAN-manual and set to flow-based on port channel 1:

Example:

```
Router# show etherchannel load-balancing


EtherChannel Load-Balancing Method:
Global LB Method: vlan-manual

  Port-Channel:                      LB Method
    Port-channel1                  :  flow-based
```

Step 3 **show interfaces port-channel** *channel-number* **etherchannel**

Use this command to display the bucket distribution currently in use. The following example shows output for an interface with load balancing set to flow-based:

Example:

```
Router(config)# show interface port-channel 2 etherchannel

 All IDBs List contains 3 configured interfaces
  Port: GigabitEthernet2/1/6 (index: 0)
  Port: GigabitEthernet2/1/7 (index: 1)
  Port: GigabitEthernet2/1/0 (index: 2)
```

```
Active Member List contains 1 interfaces
 Port: GigabitEthernet2/1/0

Passive Member List contains 2 interfaces
 Port: GigabitEthernet2/1/6

 Port: GigabitEthernet2/1/7

Load-Balancing method applied: flow-based

Bucket Information for Flow-Based LB:
Interface:                                Buckets
   GigabitEthernet2/1/0:
                          Bucket 0 , Bucket 1 , Bucket 2 , Bucket 3
                          Bucket 4 , Bucket 5 , Bucket 6 , Bucket 7
                          Bucket 8 , Bucket 9 , Bucket 10, Bucket 11
                          Bucket 12, Bucket 13, Bucket 14, Bucket 15
```

Configuration Examples for Flow-Based per Port-Channel Load Balancing

Flow-Based Load Balancing Example

The following example shows a configuration where flow-based load balancing is configured on port-channel 2 while the VLAN-manual method is configured globally:

```
!
no aaa new-model
port-channel load-balancing vlan-manual
ip source-route
.
.
.
interface Port-channel2
 ip address 10.0.0.1 255.255.255.0
 no negotiation auto
 load-balancing flow
!
interface Port-channel2.10
 ip rsvp authentication key 11223344
 ip rsvp authentication
!
interface Port-channel2.50
 encapsulation dot1Q 50
!
interface GigabitEthernet2/1/0
 no ip address
 negotiation auto
 cdp enable
 channel-group 2
!
```

Information About Five-Tuple Hash Support for GEC Flow-based Load Balancing

The five-tuple hash support for gigabit etherchannel (GEC) flow-based load balancing feature decides which member link to use for routing traffic based on the following five parameters:

- Source IP address
- Destination IP address
- Source Port
- Destination Port
- Protocol ID (type of protocol: TCP/UDP)

Earlier, the GEC flow-based load balancing feature was applicable only for layer 3 (network layer). With the five-tuple hash support, it's applicable for layer 4 (TCP/IP layer) also. But it is supported only for the TCP and UDP, layer 4 protocols.

Restrictions for Five-Tuple Hash Support for GEC Flow-based Load Balancing

The five-tuple hash support for GEC flow-based load balancing feature is not supported for MPLS traffic.

Configuring Five-Tuple Hash Support for GEC Flow-based Load Balancing

Use the **port-channel load-balance-hash-algo** command to enable the five-tuple hash support for GEC flow-based load balancing feature.

The following example shows how to configure a five-tuple hash support for GEC flow-based load balancing feature:

```
Device (config)# port-channel load-balance-hash-algo ?

src-dst-ip Source XOR Destination IP Addr
src-dst-mixed-ip-port Source XOR Destination Port, IP addr
```

The **src-dst-mixed-ip-port** option specifies load distribution based on the hash value obtained from the calculation of five parameters: source ip address, destination ip address, source port, destination port, and L4 protocol.

Example

Additional References

The following sections provide references related to the Flow-Based per Port-Channel Load Balancing feature.

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Cisco IOS LAN switching commands	*Cisco IOS LAN Switching Command Reference*

Standards

Standard	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported by this feature, and support for existing standards has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

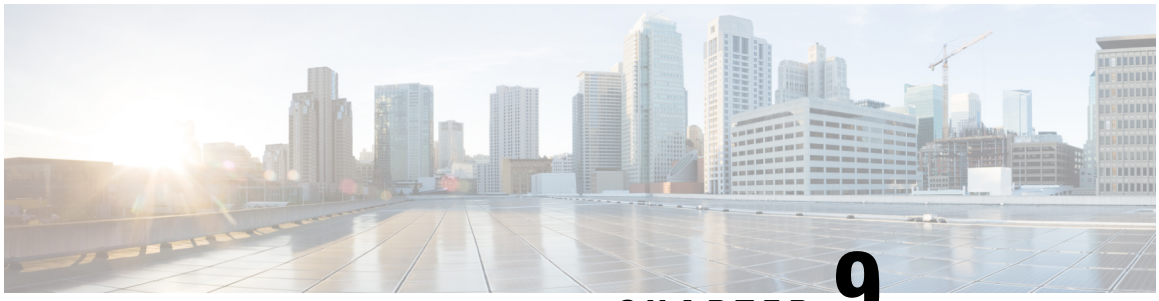
Feature Information for Flow-Based per Port-Channel Load Balancing

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 16: Feature Information for Flow-Based per Port-Channel Load Balancing

Feature Name	Releases	Feature Information
Flow-Based per Port-Channel Load Balancing	Cisco IOS XE Release 2.5	This feature allows different flows of traffic over a GEC interface to be identified and mapped to the different member links. It also enables you to apply load balancing to specific port channels. The following commands were introduced or modified: **load-balancing**, **port-channel load-balancing vlan-manual**, **show etherchannel load-balancing, show interfaces port-channel etherchannel**.
IPv6 Loadbalancing on GEC	Cisco IOS XE Release 3.4S	The IPv6 Loadbalancing on GEC feature provides load balancing for IPv6 traffic on Gigabit EtherChannel.
Five-Tuple Hash Support for GEC Flow-based Load Balancing	Cisco IOS XE Everest 16.4.1	The five-tuple hash support for gigabit etherchannel (GEC) flow-based load balancing feature decides which member link to use for routing traffic based on the hash value obtained from the calculation of 5 parameters: source ip address, destination ip address, source port, destination port, and L4 protocol.



CHAPTER 9

VLANs over IP Unnumbered SubInterfaces

The VLANs over IP Unnumbered Subinterfaces feature allows IP unnumbered interface support to be configured on Ethernet VLAN subinterfaces. This feature also provides support for DHCP on VLAN subinterfaces. Configuring Ethernet VLANs on IP unnumbered subinterfaces can save IPv4 address space and simplify configuration management, address management, and migration for DSL providers from ATM networks to IP.

- Finding Feature Information, on page 139
- Prerequisites for VLANs over IP Unnumbered Subinterfaces, on page 139
- Restrictions for VLANs over IP Unnumbered Subinterfaces, on page 140
- Information About VLANs over IP Unnumbered Subinterfaces, on page 140
- How to Configure VLANs over IP Unnumbered Subinterfaces, on page 142
- Configuration Examples for VLANs over IP Unnumbered Subinterfaces, on page 145
- Additional References for VLANs over IP Unnumbered Subinterfaces, on page 145
- Feature Information for VLANs over IP Unnumbered Subinterfaces, on page 146

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for VLANs over IP Unnumbered Subinterfaces

Configure DHCP and ensure that it is operational.

Restrictions for VLANs over IP Unnumbered Subinterfaces

- Only Ethernet VLAN subinterfaces, in addition to serial interfaces, can be configured as IP unnumbered interfaces.
- Interface ranges (the **interface range** command) are not supported in Cisco IOS Release 12.2(18)SXE.
- A physical interface cannot be used for Layer 3 services (no IP address configurations are supported on the physical interface) if one of the subinterface is configured as native.

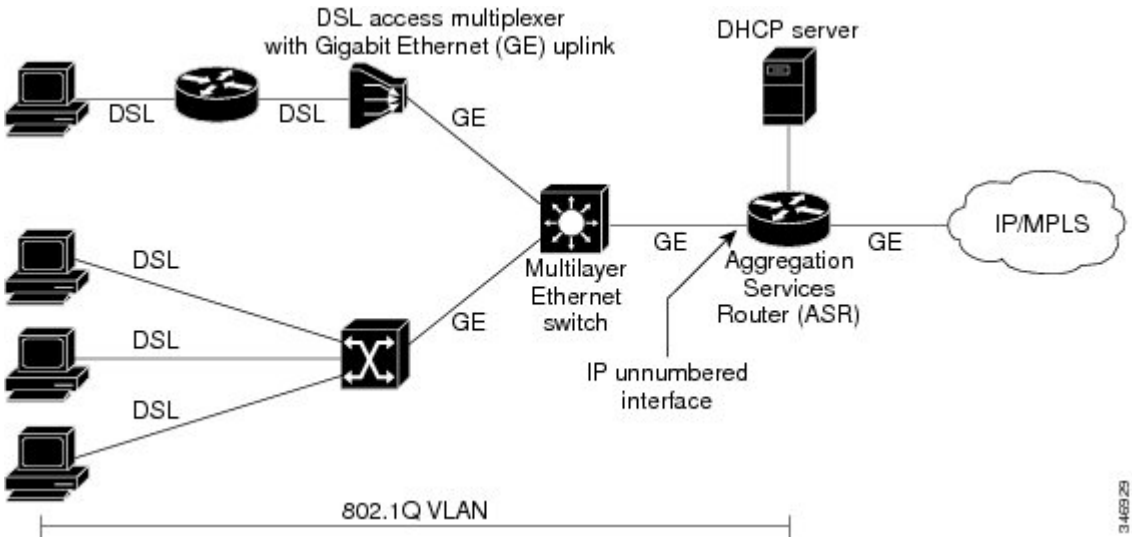
Information About VLANs over IP Unnumbered Subinterfaces

Support for VLANs over IP Unnumbered Subinterfaces

The VLANs over IP Unnumbered Subinterfaces feature enables Ethernet VLANs to be configured on IP unnumbered subinterfaces. The IP unnumbered interface configuration enables IP processing on an interface without assigning an IP address to the interface. The IP unnumbered interface borrows an IP address from another interface that is already configured on the device to conserve network and address space.

Figure 1 shows the implementation of the VLANs over IP Unnumbered Subinterfaces feature in a sample network topology. In this topology, the aggregation services routers dynamically establish IP routes when the DHCP server assigns IP addresses to hosts.

Figure 24: Sample Network Topology Using VLANs over IP Unnumbered Subinterfaces Feature



The VLANs over IP Unnumbered Subinterfaces feature supports the following functions:

- Allocating peer IP address through DHCP.
- Configuring IP unnumbered interface support for a range of VLAN subinterfaces.
- Configuring service selection gateway support for VLANs over IP unnumbered subinterfaces.

- Supporting DHCP relay agent information feature (Option 82).

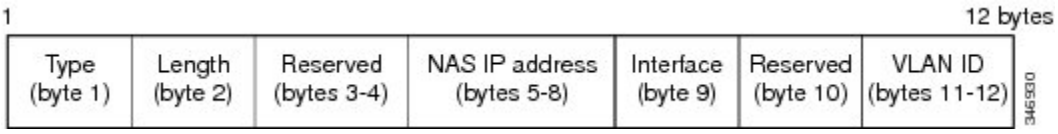
DHCP Option 82

DHCP provides a framework for passing configuration information to hosts on a TCP/IP network. Configuration parameters and other control information are carried in tagged data items (also called options) that are stored in the options field of the DHCP message. Option 82 is organized as a single DHCP option that contains information known by the relay agent.

The DHCP Relay Agent Information feature communicates information to the DHCP server using a suboption of the DHCP relay agent information option called agent remote ID. The information sent in the agent remote ID includes an IP address identifying the relay agent and information about the interface and the connection over which the DHCP request was received. The DHCP server uses this information to assign IP addresses to interfaces and to form security policies.

Figure 2 shows the agent remote ID suboption format that is used with the VLANs over IP Unnumbered Subinterfaces feature.

Figure 25: Format of the Agent Remote ID Suboption



Field	Description
Type	Format type (1 byte). Value 2 specifies the format for use with this feature.
Length	Length of the agent remote ID suboption (1 byte). The type field and the remaining bytes of the length field are not included.
Reserved	Reserved (2 bytes).
NAS IP Address	Network-attached storage (NAS) IP address (4 bytes) of the interface specified by the **ip unnumbered** command.
Interface	Physical interface (1 byte). This field has the following format: slot (4 bits) | module (1 bit) | port (3 bits). For example, if the interface is Ethernet 2/1/1, the slot is 2, the module is 1, and the port is 1.
Reserved	Reserved (1 byte).
VLAN ID	VLAN identifier (2 bytes) for the Ethernet subinterface.

Benefits of VLANs over IP Unnumbered Subinterfaces

The VLANs over IP Unnumbered Subinterfaces feature provides the following benefits:

- Migration from other interfaces to Gigabit Ethernet uplinks and IP core becomes easier for DSL providers.
- All ports share the same subnet, therefore saving the IPv4 address space.
- Each user is on a separate VLAN. DHCP communicates routing information, and there is no Address Resolution Protocol (ARP) or MAC address spoofing, which leads to enhancement in security layers.
- IP address management with DHCP becomes simpler.
- Configuring interface ranges with Ethernet VLAN subinterfaces leads to easier NVRAM configuration and saves overall memory.

How to Configure VLANs over IP Unnumbered Subinterfaces

Configuring IP Unnumbered Interface Support on an Ethernet VLAN Subinterface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* [*name-tag*]
4. **encapsulation dot1q** *vlan-id* [**native**]
5. **ip unnumbered** *type number*
6. **end**
7. **show running-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type number* [*name-tag*] **Example:** `Device(config)# interface fastethernet 1/0.1`	Configures an interface type and enters interface or subinterface configuration mode.
Step 4	**encapsulation dot1q** *vlan-id* [**native**] **Example:** `Device(config-subif)# encapsulation dot1q 10`	Enables IEEE 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.
Step 5	**ip unnumbered** *type number* **Example:** `Device(config-subif)# ip unnumbered ethernet 3/0`	Enables IP processing on an interface without assigning an explicit IP address to the interface. • The *type* and *number* arguments specify an interface with a predefined IP address on the device. Do not specify an unnumbered interface, if one already exists.
Step 6	**end** **Example:** `Device(config-subif)# end`	Exits subinterface configuration mode and returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Displays contents of the current running configuration file on the device including the configuration of the IP unnumbered support feature.

Configuring IP Unnumbered Interface Support on a Range of Ethernet VLAN Subinterfaces

Note The **interface range** command is not supported in Cisco IOS Release 12.2(18)SXE.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface range** {{**ethernet** | **fastethernet** | **gigabitethernet** | **vlan** *vlan*} *slot*/*interface.subinterface* - {**ethernet** | **fastethernet** | **gigabitethernet** | **vlan** *vlan*} *slot*/*interface.subinterface* | **macro** *macro-name*}
4. **encapsulation dot1q** *vlan-id* [**native**]
5. **ip unnumbered** *type number*
6. **end**
7. **show running-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface range** {{**ethernet** | **fastethernet** | **gigabitethernet** | **vlan** *vlan*} *slot*/*interface.subinterface* **-** {**ethernet** | **fastethernet** | **gigabitethernet** | **vlan** *vlan*} *slot*/*interface.subinterface* | **macro** *macro-name*} **Example:** `Device(config)# interface range fastethernet 1/0.1 - fastethernet 1/0.100`	Executes commands on multiple subinterfaces simultaneously. The device prompt changes to configuration interface range mode after the commands are executed. • Separate the interface range with a hyphen and space as shown in the example.
Step 4	**encapsulation dot1q** *vlan-id* [**native**] **Example:** `Device(config-if-range)# encapsulation dot1q 10`	Applies a unique VLAN ID to each subinterface within the range. • The VLAN ID specified by the *vlan-id* argument is applied to the first subinterface in the range. Each subsequent interface is assigned a VLAN ID, which is the specified *vlan-id* including the subinterface number and excluding the first subinterface number (VLAN ID + subinterface number - first subinterface number).
Step 5	**ip unnumbered** *type number* **Example:** `Device(config-if-range)# ip unnumbered ethernet 3/0`	Enables IP processing on an interface without assigning an explicit IP address to the interface. • The *type* and *number* arguments specify an interface with a predefined IP address on the device. Do not specify an unnumbered interface, if one already exists.
Step 6	**end** **Example:** `Device(config-if-range)# end`	Exits interface-range configuration mode and returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Displays contents of the current running configuration file on the device including the configuration of the IP unnumbered support feature.

Configuration Examples for VLANs over IP Unnumbered Subinterfaces

Example: VLAN Configuration on a Single IP Unnumbered Subinterface

The following example shows how to configure IP unnumbered subinterface using Ethernet VLAN subinterface 3/0.2:

```
interface ethernet 3/0.2
 encapsulation dot1q 200
 ip unnumbered ethernet 3/1
```

Example: VLAN Configuration on a Range of IP Unnumbered Subinterfaces

The following example shows how to configure IP unnumbered subinterfaces using Fast Ethernet subinterfaces in the range from 5/1.1 to 5/1.4:

```
interface range fastethernet 5/1.1 - fastethernet 5/1.4
 ip unnumbered ethernet 3/1
```

Additional References for VLANs over IP Unnumbered Subinterfaces

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
IP Addressing commands	Cisco IOS IP Addressing Services Command Reference
IP Addressing Services configuration tasks	Cisco IOS IP Addressing Services Configuration Guide
VLAN configuration tasks	Cisco IOS LAN Switching Configuration Guide
VLAN configuration commands	Cisco IOS LAN Switching Command Reference

RFCs

RFCs	Title
RFC 1812	*Requirements for IP Version 4 Routers*, June 1995

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

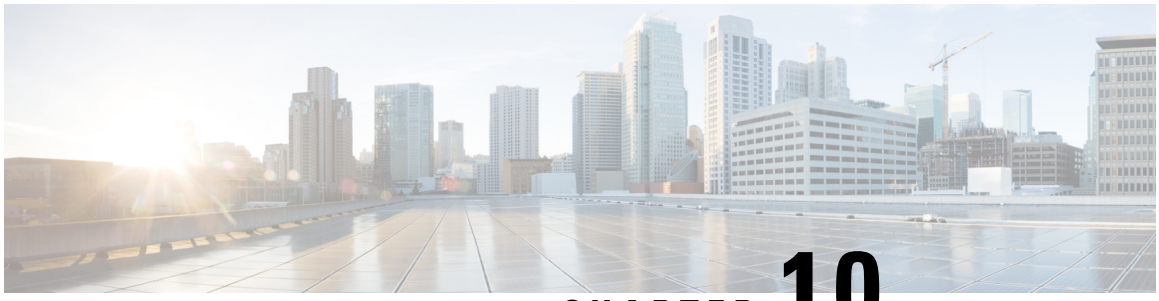
Feature Information for VLANs over IP Unnumbered Subinterfaces

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 17: Feature Information for VLANs over IP Unnumbered Subinterfaces

Feature Name	Releases	Feature Information
VLANs over IP Unnumbered Subinterfaces	Cisco IOS XE Release 3.9S	The VLANs over IP Unnumbered Subinterfaces feature allows IP unnumbered interface support to be configured on Ethernet VLAN subinterfaces. This feature also provides support for DHCP on VLAN subinterfaces. Configuring Ethernet VLANs on IP unnumbered subinterfaces can save IPv4 address space and simplify configuration management, address management, and migration for DSL providers from ATM networks to IP. The following command was modified: **ip unnumbered**



CHAPTER 10

Spanning Tree Protocol

For conceptual information about Spanning Tree Protocol, see the "Using the Spanning Tree Protocol with the EtherSwitch Network Module" section of the EtherSwitch Network feature module.

- Finding Feature Information, on page 147
- Information About Spanning Tree Protocol, on page 147
- How to Configure Spanning Tree Protocol, on page 156
- Configuration Examples for Spanning Tree Protocol, on page 164
- Additional References, on page 167
- Feature Information for Spanning Tree Protocol, on page 167

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Spanning Tree Protocol

Using the Spanning Tree Protocol with the EtherSwitch Network Module

The EtherSwitch Network Module uses Spanning Tree Protocol (STP) (the IEEE 802.1D bridge protocol) on all VLANs. By default, a single instance of STP runs on each configured VLAN (provided that you do not manually disable STP). You can enable and disable STP on a per-VLAN basis.

When you create fault-tolerant internetworks, you must have a loop-free path between all nodes in a network. The spanning tree algorithm calculates the best loop-free path throughout a switched Layer 2 network. Switches send and receive spanning tree frames at regular intervals. The switches do not forward these frames but use the frames to construct a loop-free path.

Multiple active paths between end stations cause loops in the network. If a loop exists in the network, end stations might receive duplicate messages and switches might learn endstation MAC addresses on multiple Layer 2 interfaces. These conditions result in an unstable network.

STP defines a tree with a root switch and a loop-free path from the root to all switches in the Layer 2 network. STP forces redundant data paths into a standby (blocked) state. If a network segment in the spanning tree fails and a redundant path exists, the spanning tree algorithm recalculates the spanning tree topology and activates the standby path.

When two ports on a switch are part of a loop, the spanning tree port priority and port path cost setting determine which port is put in the forwarding state and which port is put in the blocking state. The spanning tree port priority value represents the location of an interface in the network topology and how well located it is to pass traffic. The spanning tree port path cost value represents media speed.

Spanning Tree Port States

Propagation delays occur when protocol information passes through a switched LAN. As a result, topology changes take place at different times and at different places in a switched network. When a Layer 2 interface changes from nonparticipation in the spanning tree topology to the forwarding state, it creates temporary data loops. Ports must wait for new topology information to propagate through the switched LAN before starting to forward frames. They must allow the frame lifetime to expire for frames that are forwarded using the old topology.

Each Layer 2 interface on a switch using Spanning Tree Protocol (STP) exists in one of the following states:

- Blocking—The Layer 2 interface does not participate in frame forwarding.
- Disabled—The Layer 2 interface does not participate in spanning tree and is not forwarding frames.
- Forwarding—The Layer 2 interface forwards frames.
- Learning—The Layer 2 interface prepares to participate in frame forwarding.
- Listening—First transitional state after the blocking state when spanning tree determines that the Layer 2 interface must participate in frame forwarding.

A Layer 2 interface moves through the following states:

- From blocking state to listening or disabled state.
- From forwarding state to disabled state.
- From initialization to blocking state.
- From learning state to forwarding or disabled state.
- From listening state to learning or disabled state.

The figure below illustrates how a port moves through these five states.

Boot-up Initialization

When you enable Spanning Tree Protocol (STP), every port in the switch, VLAN, or network goes through the blocking state and transitory states of listening and learning at power up. If properly configured, each Layer 2 interface stabilizes to the forwarding or blocking state.

When the spanning tree algorithm places a Layer 2 interface in the forwarding state, the following process occurs:

1. The Layer 2 interface is put into the listening state while it waits for protocol information to go to the blocking state.
2. The Layer 2 interface waits for the forward delay timer to expire, moves the Layer 2 interface to the learning state, and resets the forward delay timer.
3. The Layer 2 interface continues to block frame forwarding in the learning state as it learns end station location information for the forwarding database.
4. The Layer 2 interface waits for the forward delay timer to expire and then moves the Layer 2 interface to the forwarding state, where both learning and frame forwarding are enabled.

Blocking State

A Layer 2 interface in the blocking state does not participate in frame forwarding, as shown in the figure below. After initialization, a bridge protocol data unit (BPDU) is sent out to each Layer 2 interface in the switch. The switch initially assumes it is the root until it exchanges BPDUs with other switches. This exchange establishes which switch in the network is the root or root bridge. If only one switch is in the network, no exchange occurs, the forward delay timer expires, and the ports move to the listening state. A port enters the blocking state following switch initialization.

A Layer 2 interface in the blocking state performs as follows:

- Discards frames received from the attached segment.
- Discards frames switched from another interface for forwarding.
- Does not incorporate end station location into its address database. (There is no learning on a blocking Layer 2 interface, so there is no address database update.)
- Does not transmit BPDUs received from the system module.
- Receives BPDUs and directs them to the system module.
- Receives and responds to network management messages.

Listening State

The listening state is the first transitional state a Layer 2 interface enters after the blocking state. The Layer 2 interface enters this state when STP determines that the Layer 2 interface must participate in frame forwarding. The figure below shows a Layer 2 interface in the listening state.

A Layer 2 interface in the listening state performs as follows:

- Discards frames received from the attached segment.
- Discards frames switched from another interface for forwarding.
- Does not incorporate end station location into its address database. (There is no learning on a blocking Layer 2 interface, so there is no address database update.)
- Receives and directs BPDUs to the system module.
- Receives, processes, and transmits BPDUs received from the system module.
- Receives and responds to network management messages.

Learning State

The learning state prepares a Layer 2 interface to participate in frame forwarding. The Layer 2 interface enters the learning state from the listening state. The figure below shows a Layer 2 interface in the learning state.

A Layer 2 interface in the learning state performs as follows:

- Discards frames received from the attached segment.
- Discards frames switched from another interface for forwarding.
- Incorporates end station location into its address database.
- Receives BPDUs and directs them to the system module.
- Receives, processes, and transmits BPDUs received from the system module.
- Receives and responds to network management messages.

Forwarding State

A Layer 2 interface in the forwarding state forwards frames, as shown in the figure below. The Layer 2 interface enters the forwarding state from the learning state.

A Layer 2 interface in the forwarding state performs as follows:

- Forwards frames received from the attached segment.
- Forwards frames switched from another Layer 2 interface for forwarding.
- Incorporates end station location information into its address database.
- Receives BPDUs and directs them to the system module.
- Processes BPDUs received from the system module.
- Receives and responds to network management messages.

Disabled State

A Layer 2 interface in the disabled state does not participate in frame forwarding or spanning tree, as shown in the figure below. A Layer 2 interface in the disabled state is virtually nonoperational.

A Layer 2 interface in the disabled state performs as follows:

- Discards frames received from the attached segment.
- Discards frames switched from another Layer 2 interface for forwarding.
- Does not incorporate end station location into its address database. (There is no learning on a blocking Layer 2 interface, so there is no address database update.)
- Does not receive BPDUs for transmission from the system module.

Default Spanning Tree Configuration

The table below shows the default Spanning Tree Protocol (STP) configuration values.

Table 18: SPT Default Configuration Values

Feature	Default Value
Bridge priority	32768
Enable state	Spanning tree enabled for all VLANs
Forward delay time	15 seconds
Hello time	2 seconds
Maximum aging time	20 seconds
Spanning tree port cost (configurable on a per-interface basis; used on interfaces configured as Layer 2 access ports)	Fast Ethernet: 19 Ethernet: 100 Gigabit Ethernet: 19 when operated in 100 Mb mode, and 4 when operated in 1000 Mb mode
Spanning tree port priority (configurable on a per-interface basis; used on interfaces configured as Layer 2 access ports)	128
Spanning tree VLAN port cost (configurable on a per-VLAN basis; used on interfaces configured as Layer 2 trunk ports)	Fast Ethernet: 10 Ethernet: 10
Spanning tree VLAN port priority (configurable on a per-VLAN basis; used on interfaces configured as Layer 2 trunk ports)	128

Bridge Protocol Data Units

The stable active spanning tree topology of a switched network is determined by the following:

- Port identifier (port priority and MAC address) associated with each Layer 2 interface.
- Spanning tree path cost to the root bridge.
- Unique bridge ID (bridge priority and MAC address) associated with each VLAN on each switch.

The bridge protocol data units (BPDUs) are transmitted in one direction from the root switch and each switch sends configuration BPDUs to communicate and compute the spanning tree topology. Each configuration BPDU contains the following minimal information:

- Bridge ID of the transmitting bridge
- Message age
- Port identifier of the transmitting port
- Spanning tree path cost to the root
- Unique bridge ID of the switch that the transmitting switch believes to be the root switch
- Values for the hello, forward delay, and max-age protocol timers

When a switch transmits a BPDU frame, all switches connected to the LAN on which the frame is transmitted receive the BPDU. When a switch receives a BPDU, it does not forward the frame but uses the information in the frame to calculate a BPDU, and, if the topology changes, begin a BPDU transmission.

A BPDU exchange results in the following:

- A designated bridge for each LAN segment is selected. This is the switch closest to the root bridge through which frames are forwarded to the root.
- A root port is selected. This is the port providing the best path from the bridge to the root bridge.
- One switch is elected as the root switch.
- Ports included in the spanning tree are selected.
- The shortest distance to the root switch is calculated for each switch based on the path cost.

For each VLAN, the switch with the highest bridge priority (the lowest numerical priority value) is elected as the root switch. If all switches are configured with the default priority (32768), the switch with the lowest MAC address in the VLAN becomes the root switch.

The spanning tree root switch is the logical center of the spanning tree topology in a switched network. All paths that are not needed to reach the root switch from anywhere in the switched network are placed in spanning tree blocking mode.

BPDUs contain information about the transmitting bridge and its ports, including bridge and MAC addresses, bridge priority, port priority, and path cost. Spanning tree uses this information to elect the root bridge and root port for the switched network, as well as the root port and designated port for each switched segment.

MAC Address Allocation

MAC addresses are allocated sequentially, with the first MAC address in the range assigned to VLAN 1, the second MAC address in the range assigned to VLAN 2, and so forth. For example, if the MAC address range is 00-e0-1e-9b-2e-00 to 00-e0-1e-9b-31-ff, the VLAN 1 bridge ID is 00-e0-1e-9b-2e-00, the VLAN 2 bridge ID is 00-e0-1e-9b-2e-01, the VLAN 3 bridge ID is 00-e0-1e-9b-2e-02, and so forth.

BackboneFast

BackboneFast is started when a root port or blocked port on a switch receives inferior bridge protocol data units (BPDUs) from its designated bridge. An inferior BPDU identifies one switch as both the root bridge and the designated bridge. When a switch receives an inferior BPDU, it means that a link to which the switch is not directly connected is failed. That is, the designated bridge has lost its connection to the root switch. Under Spanning Tree Protocol (STP) rules, the switch ignores inferior BPDUs for the configured maximum aging time specified by the **spanning-tree max-age** command.

The switch determines if it has an alternate path to the root switch. If the inferior BPDU arrives on a blocked port, the root port and other blocked ports on the switch become alternate paths to the root switch. If the inferior BPDU arrives on the root port, all blocked ports become alternate paths to the root switch. If the inferior BPDU arrives on the root port and there are no blocked ports, the switch assumes that it lost connectivity to the root switch, causes the maximum aging time on the root to expire, and becomes the root switch according to normal STP rules.

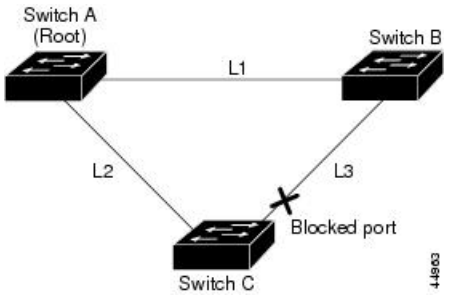


Note Self-looped ports are not considered as alternate paths to the root switch.

If the switch possesses alternate paths to the root switch, it uses these alternate paths to transmit the protocol data unit (PDU) that is called the root link query PDU. The switch sends the root link query PDU on all alternate paths to the root switch. If the switch determines that it has an alternate path to the root, it causes the maximum aging time on ports on which it received the inferior BPDU to expire. If all the alternate paths to the root switch indicate that the switch has lost connectivity to the root switch, the switch causes the maximum aging time on the ports on which it received an inferior BPDU to expire. If one or more alternate paths connect to the root switch, the switch makes all ports on which it received an inferior BPDU its designated ports and moves them out of the blocking state (if they were in the blocking state), through the listening and learning states, and into the forwarding state.

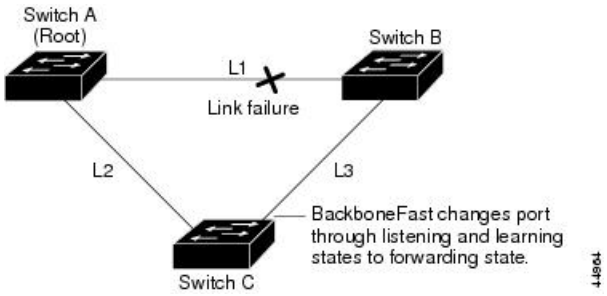
The figure below shows an example topology with no link failures. Switch A, the root switch, connects directly to Switch B over link L1 and to Switch C over link L2. The interface on Switch C that connects directly to Switch B is in the blocking state.

Figure 26: BackboneFast Example Before Indirect Link Failure



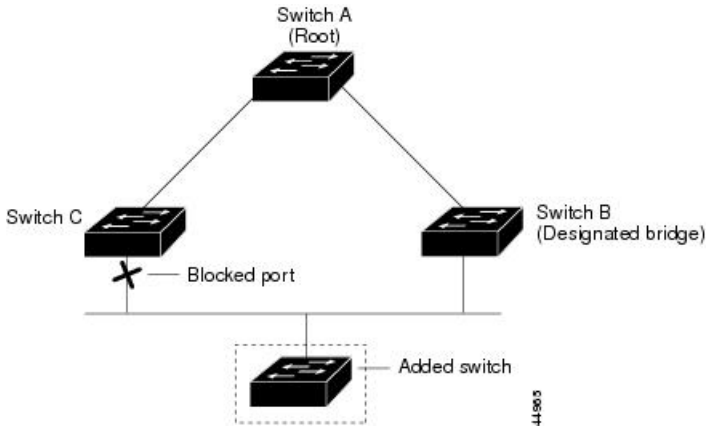
If link L1 fails, Switch C cannot detect this failure because it is not connected directly to link L1. However, Switch B is directly connected to the root switch over L1 and it detects the failure, elects itself as the root switch, and begins sending BPDUs to Switch C. When Switch C receives the inferior BPDUs from Switch B, Switch C assumes that an indirect failure has occurred. At that point, BackboneFast allows the blocked port on Switch C to move to the listening state without waiting for the maximum aging time for the port to expire. BackboneFast then changes the interface on Switch C to the forwarding state, providing a path from Switch B to Switch A. This switchover takes 30 seconds, twice the forward delay time, if the default forward delay time of 15 seconds is set. The figure below shows how BackboneFast reconfigures the topology to account for the failure of link L1.

Figure 27: BackboneFast Example After Indirect Link Failure



If a new switch is introduced into a shared-medium topology as shown in the figure below, BackboneFast is not activated because inferior BPDUs did not come from the designated bridge (Switch B). The new switch begins sending inferior BPDUs that say it is the root switch. However, the other switches ignore these inferior BPDUs, and the new switch learns that Switch B is the designated bridge to Switch A, the root switch.

Figure 28: Adding a Switch in a Shared-Medium Topology



STP Timers

The table below describes the Spanning Tree Protocol (STP) timers that affect the entire spanning tree performance.

Table 19: STP Timers

Timer	Purpose
Forward delay timer	Determines how long listening state and learning state last before the port begins forwarding.
Hello timer	Determines how often the switch broadcasts hello messages to other switches.
Maximum age timer	Determines how long a switch can store the protocol information received on a port.

Spanning Tree Port Priority

Spanning tree considers port priority when selecting an interface to put into the forwarding state if there is a loop. You can assign higher priority values to interfaces that you want spanning tree to select first, and lower priority values to interfaces that you want spanning tree to select last. If all interfaces possess the same priority value, spanning tree puts the interface with the lowest interface number in the forwarding state and blocks other interfaces. The spanning tree port priority range is from 0 to 255, configurable in increments of 4. The default value is 128.

Cisco software uses the port priority value when an interface is configured as an access port and uses VLAN port priority values when an interface is configured as a trunk port.

Spanning Tree Port Cost

The spanning tree port path cost default value is derived from the media speed of an interface. if there is a loop, spanning tree considers port cost value when moving an interface to the forwarding state. You can assign lower port cost values to interfaces that you want spanning tree to select first and higher port cost values to

interfaces that you want spanning tree to select last. If all interfaces have the same port cost value, spanning tree puts the interface with the lowest interface number to the forwarding state and blocks other interfaces.

The port cost range is from 0 to 65535. The default value is media-specific.

Spanning tree uses the port cost value when an interface is configured as an access port and uses VLAN port cost value when an interface is configured as a trunk port.

Spanning tree port cost value calculations are based on the bandwidth of the port. There are two classes of port cost values. Short (16-bit) values are specified by the IEEE 802.1D specification and the range is from 1 to 65535. Long (32-bit) values are specified by the IEEE 802.1t specification and the range is from 1 to 200,000,000.

Assigning Short Port Cost Values

You can manually assign port cost values in the range of 1 to 65535. Default port cost values are listed in Table 2.

Table 20: Default Port Cost Values

Port Speed	Default Port Cost Value
10 Mbps	100
100 Mbps	19

Assigning Long Port Cost Values

You can manually assign port cost values in the range of 1 to 200,000,000. Default port cost values are listed in Table 3.

Table 21: Default Port Cost Values

Port Speed	Recommended Value	Recommended Range
10 Mbps	2,000,000	200,000 to 20,000,000
100 Mbps	200,000	20,000 to 2,000,000

Spanning Tree Root Bridge

The EtherSwitch HWIC maintains a separate instance of spanning tree for each active VLAN configured on the device. A bridge ID, consisting of the bridge priority and the bridge MAC address, is associated with each instance. For each VLAN, the device with the lowest bridge ID will become the root bridge for that VLAN.

To configure a VLAN instance to become the root bridge, the bridge priority can be modified from the default value (32768) to a lower value so that the bridge becomes the root bridge for the specified VLAN. Use the **spanning-tree vlan root** command to alter the bridge priority.

The device checks the bridge priority of current root bridges for each VLAN. The bridge priority for specified VLANs is set to 8192, if this value is caused the device to become the root for specified VLANs.

If any root device for specified VLANs has a bridge priority lower than 8192, the device sets the bridge priority for specified VLANs to 1 less than the lowest bridge priority.

For example, if all devices in a network have the bridge priority for VLAN 100 set to the default value of 32768, entering the **spanning-tree vlan 100 root primary** command on a device sets the bridge priority for VLAN 100 to 8192, causing the device to become the root bridge for VLAN 100.

Note The root device for each instance of spanning tree must be a backbone or distribution device. Do not configure an access device as the spanning tree primary root.

Use the **diameter** keyword to specify the Layer 2 network diameter. That is, the maximum number of bridge hops between any two end stations in the Layer 2 network. When you specify the network diameter, the device automatically picks an optimal hello time, a forward delay time, and a maximum age time for a network of that diameter, which reduces the spanning tree convergence time. You can use the **hello** keyword to override the automatically calculated hello time.

Note We recommend that you do not configure the hello time, forward delay time, and maximum age time manually after you configure the device as the root bridge.

How to Configure Spanning Tree Protocol

Enabling Spanning Tree Protocol

You can enable spanning tree protocol on a per-VLAN basis. The device maintains a separate instance of spanning tree for each VLAN except for which you disable spanning tree.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree vlan** *vlan-id*
4. **end**
5. **show spanning-tree vlan** *vlan-id*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**spanning-tree vlan** *vlan-id* **Example:** `Device(config)# spanning-tree vlan 200`	Enables spanning tree on a per-VLAN basis.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.
Step 5	**show spanning-tree vlan** *vlan-id* **Example:** `Device# show spanning-tree vlan 200`	Verifies spanning tree configuration.

Configuring the Bridge Priority of a VLAN

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree vlan** *vlan-id* **priority** *bridge-priority*
4. **show spanning-tree vlan bridge**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree vlan** *vlan-id* **priority** *bridge-priority* **Example:** `Device(config)# spanning-tree vlan 200 priority 2`	Configures the bridge priority of a VLAN. The bridge priority value ranges from 0 to 65535. **Caution** Use the **spanning-tree vlan** *vlan-id* **root primary** command and the **spanning-tree vlan** *vlan-id* **root secondary** command to modify the bridge priority.

	Command or Action	Purpose
Step 4	**show spanning-tree vlan bridge** **Example:** `Device(config-if)# spanning-tree cost 200`	Verifies the bridge priority.

Configuring STP Timers

Configuring Hello Time

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree vlan** *vlan-id* **hello-time** *hello-time*
4. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree vlan** *vlan-id* **hello-time** *hello-time* **Example:** `Device(config)# spanning-tree vlan 200 hello-time 5`	Configures the hello time for a VLAN.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.

Configuring the Forward Delay Time for a VLAN

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **spanning-tree vlan** *vlan-id* **forward-time** *forward-time*
4. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree vlan** *vlan-id* **forward-time** *forward-time* **Example:** `Device(config)# spanning-tree vlan 20 forward-time 5`	Configures the forward delay time for a VLAN.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.

Configuring the Maximum Aging Time for a VLAN

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree vlan** *vlan-id* **max-age** *max-age*
4. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**spanning-tree vlan** *vlan-id* **max-age** *max-age* **Example:** `Device(config)# spanning-tree vlan 200 max-age 30`	Configures the maximum aging time for a VLAN.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.

Configuring Spanning Tree Port Priority

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **spanning-tree port-priority** *port-priority*
5. **spanning-tree vlan** *vlan-id* **port-priority** *port-priority*
6. **end**
7. **show spanning-tree interface fastethernet** *interface-id*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface fastethernet 0/1/6`	Configures an interface and enters interface configuration mode.

	Command or Action	Purpose
Step 4	**spanning-tree port-priority** *port-priority* **Example:** `Device(config-if)# spanning-tree port-priority 8`	Configures the port priority for an interface.
Step 5	**spanning-tree vlan** *vlan-id* **port-priority** *port-priority* **Example:** `Device (config-if)# spanning-tree vlan vlan1 port-priority 12`	Configures the port priority for a VLAN.
Step 6	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.
Step 7	**show spanning-tree interface fastethernet** *interface-id* **Example:** `Device# show spanning-tree interface fastethernet 0/1/6`	(Optional) Saves your entries in the configuration file.

Configuring Spanning Tree Port Cost

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **spanning-tree cost** *port-cost*
5. **spanning-tree vlan** *vlan-id* **cost** *port-cost*
6. **end**
7. **show spanning-tree interface fastethernet** *interface-id*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Device(config)# interface fastethernet 0/1/6`	Configures an interface and enters interface configuration mode.
Step 4	**spanning-tree cost** *port-cost* **Example:** `Device(config-if)# spanning-tree cost 2000`	Configures the port cost for an interface.
Step 5	**spanning-tree vlan** *vlan-id* **cost** *port-cost* **Example:** `Device(config-if)# spanning-tree vlan 200 cost 2000`	Configures the VLAN port cost for an interface.
Step 6	**end** **Example:** `Device(config)# end`	Exits interface configuration mode and enters privileged EXEC mode.
Step 7	**show spanning-tree interface fastethernet** *interface-id* **Example:** `Device# show spanning-tree interface fastethernet 0/1/6`	(Optional) Saves your entries in the configuration file.

Configuring Spanning Tree Root Bridge

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree vlan** *vlanid* **root primary** [**diameter** *hops* [**hello-time** *seconds*]]
4. **no spanning-tree vlan** *vlan-id*
5. **show spanning-tree vlan** *vlan-id*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**spanning-tree vlan** *vlanid* **root primary** [**diameter** *hops* [**hello-time** *seconds*]] **Example:** `Device(config)# spanning-tree vlan 200 root primary`	Configures a device as the root device.
Step 4	**no spanning-tree vlan** *vlan-id* **Example:** `Device(config)# no spanning-tree vlan 200 root primary`	Disables spanning tree on a per-VLAN basis.
Step 5	**show spanning-tree vlan** *vlan-id* **Example:** `Device(config)# show spanning-tree vlan 200`	Verifies spanning tree on a per-VLAN basis.

Verifying Spanning Tree on a VLAN

SUMMARY STEPS

1. **enable**
2. **show spanning-tree** [*bridge-group*] [**active** | **backbonefast** | **blockedports** | **bridge** | **brief** | **inconsistentports** | **interface** *interface-type interface-number* | **pathcost method** | **root** | **summary** [**totals**] | **uplinkfast** | **vlan** *vlan-id*]

DETAILED STEPS

Procedure

Step 1 **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Device> enable
```

Step 2 **show spanning-tree** [*bridge-group*] [**active** | **backbonefast** | **blockedports** | **bridge** | **brief** | **inconsistentports** | **interface** *interface-type interface-number* | **pathcost method** | **root** | **summary** [**totals**] | **uplinkfast** | **vlan** *vlan-id*]

Use this command with the **vlan** keyword to display the spanning tree information about a specified VLAN.

Example:

```
Device# show spanning-tree vlan 200
VLAN200 is executing the ieee compatible Spanning Tree protocol
  Bridge Identifier has priority 32768, address 0050.3e8d.6401
  Configured hello time 2, max age 20, forward delay 15
  Current root has priority 16384, address 0060.704c.7000
  Root port is 264 (FastEthernet5/8), cost of root path is 38
  Topology change flag not set, detected flag not set
```

```
  Number of topology changes 0 last change occurred 01:53:48 ago

 Times:  hold 1, topology change 24, notification 2
          hello 2, max age 14, forward delay 10
  Timers: hello 0, topology change 0, notification 0
```

Example:

```
Port 264 (FastEthernet5/8) of VLAN200 is forwarding
   Port path cost 19, Port priority 128, Port Identifier 129.9.
   Designated root has priority 16384, address 0060.704c.7000
   Designated bridge has priority 32768, address 00e0.4fac.b000
   Designated port id is 128.2, designated path cost 19
   Timers: message age 3, forward delay 0, hold 0
   Number of transitions to forwarding state: 1
   BPDU: sent 3, received 3417
```

Use this command with the **interface** keyword to display spanning tree information about a specified interface.

Example:

```
Device# show spanning-tree interface fastethernet 5/8
Port 264 (FastEthernet5/8) of VLAN200 is forwarding
   Port path cost 19, Port priority 100, Port Identifier 129.8.
   Designated root has priority 32768, address 0010.0d40.34c7
   Designated bridge has priority 32768, address 0010.0d40.34c7
   Designated port id is 128.1, designated path cost 0
   Timers: message age 2, forward delay 0, hold 0
   Number of transitions to forwarding state: 1
   BPDU: sent 0, received 13513
```

Use this command with the **bridge**, **brief**, and **vlan** keywords to display the bridge priority information.

Example:

```
Device# show spanning-tree bridge brief vlan 200
 Hello Max  Fwd
Vlan                 Bridge ID      Time  Age Delay  Protocol
---------------- -------------------- ---- ---- -----  --------
VLAN200          33792 0050.3e8d.64c8    2   20    15  ieee
```

Configuration Examples for Spanning Tree Protocol

Example: Enabling Spanning Tree Protocol

The following example shows how to enable spanning tree protocol on VLAN 20:

```
Device# configure terminal
Device(config)# spanning-tree vlan 20
Device(config)# end
Device#
```

Note Because spanning tree is enabled by default, the **show running** command will not display the command you entered to enable spanning tree protocol.

The following example shows how to disable spanning tree protocol on VLAN 20:

```
Device# configure terminal
Device(config)# no spanning-tree vlan 20
Device(config)# end
Device#
```

Example: Configuring the Bridge Priority of a VLAN

The following example shows how to configure the bridge priority of VLAN 20 to 33792:

```
Device# configure terminal
Device(config)# spanning-tree vlan 20 priority 33792
Device(config)# end
```

Example: Configuring STP Timers

Example: Configuring Hello Time

The following example shows how to configure the hello time for VLAN 20 to 7 seconds:

```
Device# configure terminal
Device(config)# spanning-tree vlan 20 hello-time 7
Device(config)# end
```

Example: Configuring the Forward Delay Time for a VLAN

The following example shows how to configure the forward delay time for VLAN 20 to 21 seconds:

```
Device#configure terminal
Device(config)#spanning-tree vlan 20 forward-time 21
Device(config)#end
```

Example: Configuring the Maximum Aging Time for a VLAN

The following example shows how to configure the maximum aging time for VLAN 20 to 36 seconds:

```
Device#configure terminal
Device(config)#spanning-tree vlan 20 max-age 36
Device(config)#end
```

Example: Configuring Spanning Tree Port Priority

The following example shows how to configure VLAN port priority on an interface:

```
Device# configure terminal
Device(config)# interface fastethernet 0/3/2
Device(config-if)# spanning-tree vlan 20 port priority 64
Device(config-if)# end
```

The following example shows how to verify the configuration of VLAN 20 on an interface when it is configured as a trunk port:

```
Device#show spanning-tree vlan 20

 VLAN20 is executing the ieee compatible Spanning Tree protocol
  Bridge Identifier has priority 32768, address 00ff.ff90.3f54
  Configured hello time 2, max age 20, forward delay 15
  Current root has priority 32768, address 00ff.ff10.37b7
  Root port is 33 (FastEthernet0/3/2), cost of root path is 19
  Topology change flag not set, detected flag not set
  Number of topology flags 0 last change occurred 00:05:50 ago
  Times: hold 1, topology change 35, notification 2
     hello 2, max age 20, forward delay 15
  Timers: hello 0, topology change 0, notification 0, aging 0
 Port 33 (FastEthernet0/3/2) of VLAN20 is forwarding
  Port path cost 18, Port priority 64, Port Identifier 64.33
  Designated root has priority 32768, address 00ff.ff10.37b7
  Designated bridge has priority 32768, address 00ff.ff10.37b7
  Designated port id is 128.13, designated path cost 0
  Timers: message age 2, forward delay 0, hold 0
  Number of transitions to forwarding state: 1
  BPDU: sent 1, received 175
```

Example: Configuring Spanning Tree Port Cost

The following example shows how to change the spanning tree port cost of a Fast Ethernet interface:

```
Device# configure terminal
Device(config)# interface fastethernet0/3/2
Device(config-if)# spanning-tree cost 18
Device(config-if)# end
Device#
Device# show run interface fastethernet0/3/2
Building configuration...
Current configuration: 140 bytes
!
interface FastEthernet0/3/2
 switchport access vlan 20
  no ip address
  spanning-tree vlan 20 port-priority 64
  spanning-tree cost 18
end
```

The following example shows how to verify the configuration of a Fast Ethernet interface when it is configured as an access port:

```
Device# show spanning-tree interface fastethernet0/3/2

 Port 33 (FastEthernet0/3/2) of VLAN20 is forwarding
  Port path cost 18, Port priority 64, Port Identifier 64.33
  Designated root has priority 32768, address 00ff.ff10.37b7
  Designated bridge has priority 32768, address 00ff.ff10.37b7
  Designated port id is 128.13, designated path cost 0
  Timers: message age 2, forward delay 0, hold 0
  Number of transitions to forwarding state: 1
  BPDU: sent 1, received 175
```

Example: Configuring Spanning Tree Root Bridge

The following example shows how to configure the spanning tree root bridge for VLAN 10, with a network diameter of 4:

```
Device# configure terminal
Device(config)# spanning-tree vlan 10 root primary diameter 4
Device(config)# exit
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
LAN switching commands	

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Spanning Tree Protocol

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 22: Feature Information for Spanning Tree Protocol

Feature Name	Releases	Feature Information
Spanning Tree Protocol	12.1(1)E	Spanning Tree Protocol (STP) is a Layer 2 link management protocol that provides path redundancy while preventing undesirable loops in the network. The following commands were introduced or modified: **spanning-tree vlan**, **spanning-tree port-priority**, and **spanning-tree cost**.