Cisco 7600 Series Ethernet Services 20G Line Card Configuration Guide

March 30, 2012

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

Text Part Number: OL-11907-17

The following information is for FCC compliance of Class A devices: This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio-frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case users will be required to correct the interference at their own expense.

The following information is for FCC compliance of Class B devices: The equipment described in this manual generates and may radiate radio-frequency energy. If it is not installed in accordance with Cisco's installation instructions, it may cause interference with radio and television reception. This equipment has been tested and found to comply with the limits for a Class B digital device in accordance with the specifications in part 15 of the FCC rules. These specifications are designed to provide reasonable protection against such interference in a residential installation. However, there is no guarantee that interference will not occur in a particular installation.

Modifying the equipment without Cisco's written authorization may result in the equipment no longer complying with FCC requirements for Class A or Class B digital devices. In that event, your right to use the equipment may be limited by FCC regulations, and you may be required to correct any interference to radio or television communications at your own expense.

You can determine whether your equipment is causing interference by turning it off. If the interference stops, it was probably caused by the Cisco equipment or one of its peripheral devices. If the equipment causes interference to radio or television reception, try to correct the interference by using one or more of the following measures:

- Turn the television or radio antenna until the interference stops.

- Move the equipment to one side or the other of the television or radio.

- Move the equipment farther away from the television or radio.

- Plug the equipment into an outlet that is on a different circuit from the television or radio. (That is, make certain the equipment and the television or radio are on circuits controlled by different circuit breakers or fuses.)

Modifications to this product not authorized by Cisco Systems, Inc. could void the FCC approval and negate your authority to operate the product.

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

Cisco 7600 Series Ethernet Services 20G Line Card Configuration Guide

CONTENTS

Preface xiii

Objectives xiii

Finding Platform-Independent Feature Information xiii

Document Revision History xiii

Organization xvii

Related Documentation xviii

Cisco 7600 Series Router Documentation xviii

Other Cisco IOS Software Publications xix

Document Conventions xix

Obtaining Documentation and Submitting a Service Request xx

CHAPTER 1 Overview of the Cisco 7600 Series Ethernet Services 20G Line Card 1-1

Release History 1-1

Supported Cisco 7600 Series Ethernet Services 20G Line Card Features 1-3

Layer 2 Features 1-3

Layer 3 and Layer 4 Features 1-4

Multicast Features 1-4

High Availability Features 1-4

MPLS Features 1-5

Layer 2 Protocols and Encapsulation 1-6

QoS Features 1-6

Accounting and Management Features 1-7

Cisco 7600 ES20 Ethernet Line Card Restrictions 1-7

Switch Port Analyzer (SPAN) Restrictions 1-8

Supported MIBs 1-8

Displaying the Cisco 7600 Series Ethernet Services 20G Line Card Type 1-9

CHAPTER 2 Configuring the Cisco 7600 Series Ethernet Services 20G Line Card 2-1

Required Configuration Tasks 2-2

Identifying Slots and Subslots for the Cisco 7600 Series Ethernet Services 20G Line Card 2-3

Configuring High-Availability Features 2-3

Configuring UDE on ES20 Line Cards 2-3

Restrictions and Usage Guidelines 2-3

Configuring Unidirectional Link Detection (UDLD) on Ports with EVCs 2-8
Restrictions and Usage Guidelines 2-8
Configuring UDLD Aggressive Mode 2-9
Enabling UDLD on Ports With EVC Configured 2-9
Disabling Individual UDLD on Ports With EVC Configured 2-11
Resetting Disabled UDLD on Ports With EVC Configured 2-11
Verification 2-12
ISSU Support for ES20 Line Card 2-13
Configuring IEEE 802.1ag-2007 Compliant CFM 2-13
Supported Line Cards 2-14
Scalable Limits 2-21
Restrictions and Usage Guidelines 2-21
Support for IEEE 802.1ad 2-30
Prerequisites for IEEE 802.1ad 2-30
Restrictions for IEEE 802.1ad 2-30
Information About IEEE 802.1ad 2-31
How to Configure IEEE 802.1ad 2-38
Troubleshooting Dot1ad 2-65
Configuring Layer 2 Features 2-68
Cross-Bundling 2-69
Configuring EVC EtherChannel and LACP over EVC Port Channel 2-69
Restrictions and Usage Guidelines 2-69
Multichassis Support for Link Aggregation Control Protocol 2-74
Requirements and Restrictions 2-76
Troubleshooting 2-92
Pseudo MLACP Support on Cisco 7600 2-93
Failover Operations 2-94
Failure Recovery 2-95
Restrictions for PMLACP 2-95
Configuring PMLACP on Cisco 7600 2-96
Configuration Examples 2-101
Verification 2-103
Troubleshooting Tips 2-104
Configuring Custom Ethertype for EVC Interfaces 2-104
Supported Rewrite Rules for a Custom Ethertype Configuration 2-105
Supported Rewrites for Non Range on a C-Tag on an NNI 2-105
Supported Rewrites for Range on C-Tag with an NNI 2-106
Restrictions and Usage Guidelines 2-106
Configuring Flexible QinQ Mapping and Service Awareness on 7600-ESM-2X10GE and 7600-ESM-20X1GE 2-110

Restrictions and Usage Guidelines 2-112
Troubleshooting 2-118
Configuring Flexible Service Mapping Based on CoS and Ethertype 2-119
Restrictions and Usage Guidelines 2-120
Configuring MultiPoint Bridging over Ethernet on 7600-ESM-2X10GE and 7600-ESM-20X1GE 2-128
Restrictions and Usage Guidelines 2-129
Configuring Gigabit Ethernet Link Aggregation with Advanced Load Balancing 2-134
Restriction and Usage Guidelines 2-134
Troubleshooting Load Balancing Features 2-140
Configuring Virtual Private LAN Service (VPLS) with Port-Channel as a Core Interface 2-141
TE-FRR Support on VPLS LAG NNI 2-142
Load Balancing 2-142
Fat Pseudo-Wire Load Balancing 2-142
Provider Router Load Balancing 2-143
BPDU PW Over LAG NNI 2-143
Restrictions and Usage Guidelines 2-143
Configuring BPDU PW on a Port Channel 2-145
Configuring the Backup Interface for Flexible UNI 2-150
Restriction and Usage Guidelines 2-150
Troubleshooting 2-156
Configuring Layer2 Access Control Lists (ACLs) on an EVC 2-157
Restrictions and Usage Guidelines 2-158
Configuring Broadcast Storm Control on Switchports and Ports with Ethernet Virtual Connections 2-160
Detecting a Broadcast Storm 2-161
Traffic Storm Control on ES20 Switchports 2-161
Restrictions and Usage Guidelines 2-161
Traffic Storm Control on ES20 with EVCs 2-161
Restrictions and Usage Guidelines 2-162
Troubleshooting 2-165
Configuring Asymmetric Carrier-Delay 2-165
Restrictions and Usage Guidelines 2-166
Configuring MST on EVC Bridge Domain 2-167
Overview of MST and STP 2-167
Overview of MST on EVC Bridge Domain 2-168
Restrictions and Usage Guidelines 2-168
Examples 2-171
Verification 2-171
DHCP Snooping with Option-82 on EVC 2-174
Restrictions and Usage Guidelines 2-174

Verification 2-177

Configuring MAC Address Security for EVC Bridge-Domain 2-178

Restrictions and Usage Guidelines 2-178

Configuring MAC Address Security for EVC Bridge-Domain 2-179

Enabling MAC Address Security for EVC Bridge-Domain 2-179

Disabling MAC Address Security for EVC Bridge-Domains on an EFP 2-180

Configuring Whitelisted MAC Address on an EFP 2-181

Configuring Sticky MAC Addresses on an EFP 2-183

Configuring Secure MAC Address Aging on an EFP 2-185

Configuring MAC Address Limiting on an EFP 2-188

Configuring MAC Address Limiting on a Bridge-Domain 2-189

Configuring Violation Response on an EFP 2-190

Troubleshooting 2-192

Configuring Static MAC on Ethernet Flow Point and Pseudowire 2-193

Restrictions and Usage Guidelines 2-193

Configuring Static MAC over EFP for the Cisco 7600 Router 2-194

Configuring MPLS on Core-Facing Interface 2-195

Configuring Static MAC over Pseudowire for the Cisco 7600 Router 2-198

Troubleshooting 2-201

Configuring Resilient Ethernet Protocol 2-201

REP Edge No-Neighbor 2-202

Configuring REP over Ethernet Virtual Circuit 2-203

Restrictions and Usage Guidelines 2-203

Configuring REP over EVC for the Cisco 7600 Router 2-205

Configuring REP over EVC using Cross connect for the Cisco 7600 Router 2-205

Configuring REP over EVC using connect for the Cisco 7600 Router 2-208

Configuring REP over EVC using bridge domain for the Cisco 7600 Router 2-213

Configuring Resilient Ethernet Protocol Configurable Timers 2-220

Restrictions and Usage Guidelines 2-220

Configuring REP Configurable Timers for the Cisco 7600 Router 2-220

Configuring the REP Link Status Layer Retries 2-221

Configuring the REP Link Status Layer Age Out Timer 2-223

Configuring the REP Link Status Layer Age Out Timer 2-225

Troubleshooting 2-227

Configuring CFM over EFP Interface with cross connect 2-228

Restrictions and Usage Guidelines 2-228

Configuring CFM over EFP with xconnect for the Cisco 7600 Router 2-228

Configuring CFM over EFP Interface with Cross Connect—Basic Configuration 2-229

Configuring CFM over EFP Interface with Cross Connect—Single Tag VLAN Cross Connect 2-231

Configuring CFM over EFP Interface with Cross Connect—Double Tag VLAN Cross Connect 2-233

Configuring CFM over EFP Interface with Cross Connect—Selective QinQ Cross Connect 2-235

Configuring CFM over EFP Interface with Cross Connect—Port-Based Cross Connect Tunnel 2-237

Configuring CFM over EFP Interface with Cross Connect—Port Channel-Based Cross Connect Tunnel 2-239

Troubleshooting CFM Features 2-244

Configuring Reverse Layer 2 Gateway Ports for the Cisco 7600 Router 2-245

Restrictions and Usage Guidelines 2-246

Configuring Reverse L2GP for the Cisco 7600 Router 2-246

Troubleshooting Tips 2-254

Configuring Private Host Switch Virtual Interface (VLAN and VPLS) 2-254

Port Classification 2-254

Requirements and Restrictions 2-255

Verifying the Private Hosts SVI configuration 2-256

Sample Configuration For Private Hosts VPLS Configuration 2-256

Sample configuration for Private Hosts Interface Vlan configuration 2-257

Configuring Multicast Features 2-258

Configuring IGMP/PIM Snooping for VPLS Pseudowire on 7600-ESM-2X10GE and 7600-ESM-20X1GE 2-258

Restrictions and Usage Guidelines 2-258

Configuring Link State Tracking (LST) 2-260

Restrictions and Usage Guidelines 2-260

Configuring Link-State Tracking 2-261

Verification 2-262

Troubleshooting the Link State Tracking 2-262

Configuring Multicast VLAN Registration 2-262

Using MVR in a Multicast Television Application 2-263

Configuring MVR 2-265

Configuring Layer 3 and Layer 4 Features 2-265

Configuring Layer 3 and Layer 4 Access Control List on a Service Instance 2-265

Restrictions and Usage Guidelines 2-266

VRF aware IPv6 tunnel 2-271

Restrictions for VRF aware IPv6 tunnels 2-272

Configuring VRF aware IPv6 tunnel 2-272

Configure IPv6 overlay addresses in VRF and IPv4 transport addresses in Global RT 2-273

Configure IPv6 overlay addresses in VRF and IPv4 transport addresses in VRF 2-278

Verifying the Configuration 2-285

Troubleshooting Tips 2-286

IPv6 Policy Based Routing 2-286

Policy Based Routing 2-286

Restrictions for IPv6 PBR 2-288

Configuring IPv6 PBR 2-288

Configuring MPLS Features 2-288

Configuring Any Transport over MPLS 2-289

Scalable EoMPLS on 7600-ESM-2X10GE and 7600-ESM-20X1GE 2-289

Restrictions and Usage Guidelines 2-289

Configuring MPLS Traffic Engineering Class-Based Tunnel Selection 2-293

Restrictions and Usage Guidelines 2-293

Creating Multiple MPLS Member TE or DS-TE Tunnels with the Same Headend and the Same Tailend 2-294

Creating a Master Tunnel, Attaching Member Tunnels, and Making the Master Tunnel Visible 2-296

Configuring Virtual Private LAN Service 2-302

Hierarchical Virtual Private LAN Service (H-VPLS) with MPLS to the Edge 2-303

Configuring SVI-Based IP/Routed Interworking 2-304

Restrictions and Usage Guidelines 2-305

Resetting a Cisco 7600 Series Ethernet Services 20G Line Card 2-307

SFP-GE-T Support 2-307

CHAPTER 3 **Configuring QoS on the Cisco 7600 Series Ethernet Services 20G Line Card** 3-1

QoS Functions in the Cisco 7600 Series Ethernet Services 20G Line Card 3-2

Ingress QoS Functions in a Cisco 7600 Series Ethernet Services 20G Line Card 3-2

Ingress Trust 3-2

Ingress Queue Scheduling 3-2

Ingress Classification 3-2

Ingress Policing 3-3

Ingress Marking 3-3

Ingress Shaping 3-3

Egress QoS Functions in a Cisco 7600 Series Ethernet Services 20G Line Card 3-3

Egress Classification 3-3

Egress Policing 3-3

Egress Marking 3-3

Egress Shaping 3-4

Egress Queue Scheduling 3-4

Restrictions 3-4

Configuring QoS Features Using MQC 3-4

EVCS QoS Support 3-5

- Mapping Between Bay and Ports 3-5
- Restrictions and Usage Guidelines 3-5
- Restrictions and User Guidelines for EVC Port Channel 3-6

Configuring Qos Over an EVC Group 3-6

- Restrictions and Usage Guidelines 3-7

Dual Rate Three Color (2R3C) Ingress Policer on Service Instances 3-11

- Restrictions and Usage Guidelines 3-11

Ingress Policing on EVC Port Channel 3-15

- Restrictions and Usage Guidelines 3-15
- Configuring Ingress Policing on EVC Port Channel 3-16

Configuring Classification 3-16

- Restrictions and Usage Guidelines 3-16

Configuring Policing 3-21

- Restrictions and Usage Guidelines 3-21
- Configuration Tasks 3-23

Attaching a QoS Traffic Policy to an Interface 3-28

- Attaching a QoS Traffic Policy for an Input Interface 3-28
- Attaching a QoS Traffic Policy to an Output Interface 3-29

Configuring Marking 3-29

- Restrictions and Usage Guidelines 3-29

Configuring Shaping 3-34

- Restrictions and Usage Guidelines 3-35

Configuring Shaping on the ES20 Main Interface 3-37

- About Traffic Shaping 3-37
- Restrictions and Usage Guidelines 3-38
- Non example 3-38
- Configuring the Child Policy 3-39

Configuring QoS Queue Scheduling 3-42

- Restrictions and Usage Guidelines 3-42
- Configuring CBWFQ 3-44

Configuring LLQ 3-46

- Examples 3-47

Configuring PFC QoS on a Cisco 7600 Series Ethernet Services 20G Line Card 3-47

PFC QoS on a Cisco 7600 Series Ethernet Services 20G Line Card Configuration Guidelines 3-48

Configuring Hierarchical QoS 3-48

Hybrid Policy Support 3-48

Configuring Hierarchical QoS with Tiered Policy-Maps 3-50

- Configuration Guidelines 3-50

Configuring Hierarchical QoS for EoMPLS VCs 3-53

Hierarchical QoS with Service Instances 3-53

Configuration Guidelines 3-53
EVC Configuration Examples 3-54
Configuring Bandwidth Remaining Ratio (BRR) - Utilizing Unused Bandwidth Support 3-57
Hierarchical Policy-Maps 3-58
HQoS Policies with Bandwidth and Shape in Child Classes 3-59
Restrictions and Usage Guidelines 3-60
HQoS Polices with Shape in Child Classes 3-60
Restrictions and Usage Guideline 3-61
HQoS Policies with Police+Priority (LLQ) and Shape in Child Classes 3-61
Restrictions and Usage Guideline 3-62
Flat Policy- Maps 3-62
BRR Propagation 3-63
Restrictions and Usage Guidelines 3-63
Example of a Hybrid Policy 3-63
Hardware Restrictions 3-64
Troubleshooting QoS Features in a ES20 Line Card 3-65

CHAPTER 4 Troubleshooting the Cisco 7600 Series Ethernet Services 20G Line Card 4-1
General Troubleshooting Information 4-1
Interpreting Console Error Messages 4-1
Using debug Commands 4-2
Using show Commands 4-2
Using the Cisco IOS Event Tracer to Troubleshoot Problems 4-2
Troubleshooting SFP Issues 4-3
Preparing for Online Insertion and Removal of Cisco 7600 Series Ethernet Services 20G Line Card 4-3
Preparing for Online Removal of a Cisco 7600 Series Ethernet Services 20G Line Card 4-4
Deactivating a Cisco 7600 Series Ethernet Services 20G Line Card 4-4
Reactivating a Cisco 7600 Series Ethernet Services 20G Line Card 4-4
Verifying Deactivation and Activation of a Cisco 7600 Series Ethernet Services 20G Line Card 4-5
Deactivation and Activation Configuration Examples 4-6
Deactivation of a Cisco 7600 Series Ethernet Services 20G Line Card Configuration Example 4-6
Activation of a Cisco 7600 Series Ethernet Services 20G Line Card Configuration Example 4-6
Line Card Online Diagnostics 4-7
Onboard Failure Logging 4-7

CHAPTER 5 Upgrading Field-Programmable Devices 5-1
FPD Quick Upgrade 5-1
FPD Quick Upgrade Before Upgrading your Cisco IOS Release (Recommended) 5-2

FPD Quick Upgrade After Upgrading your Cisco IOS Release 5-2

Overview of FPD Images and Packages 5-2

Upgrading FPD Images 5-3

Migrating to a Newer Cisco IOS Release 5-3

Upgrading FPD Images Before Upgrading Cisco IOS Release (Recommended) 5-3

Upgrade FPD Images after Upgrading the New Cisco IOS Release 5-4

Upgrading FPD Images in a Production System 5-4

Using a NonProduction System to Upgrade the Cisco 7600 Series ES20 Line Card FPD Image 5-5

Upgrading FPD Images Using Fast Software Upgrade 5-6

Optional FPD Procedures 5-6

Manually Upgrading ES20 Line Card FPD Images 5-6

Upgrading FPD from an FTP or TFTP Server 5-7

Modifying the Default Path for the FPD Image Package File Location 5-8

Displaying Current and Minimum Required FPD Image Versions 5-8

Displaying Information About the Default FPD Image Package 5-10

FPD Image Upgrade Examples 5-11

Automatic FPD Image Upgrade Example 5-11

Manual FPD Image Upgrade Example 5-12

APPENDIX A Command Summary for the Cisco 7600 Series Ethernet Services 20G Line Card A-1

INDEX

Preface

This preface describes the objectives and organization of this document and explains how to find additional information on related products and services. This preface contains the following sections:

- Objectives, page xiii
- Finding Platform-Independent Feature Information, page xiii
- Document Revision History, page xiii
- Organization, page xvii
- Related Documentation, page xviii
- Document Conventions, page xix
- Obtaining Documentation and Submitting a Service Request, page xx

Objectives

This document describes the 7600 ES20 Ethernet Services 20G line card, hereinafter referred to as ES20 line card, that is supported on the Cisco 7600 series routers. This document also describes how to configure the Cisco 7600 ES20 Ethernet Services 20G line card and how to troubleshoot the installation. Do note that the Platform-independent feature configuration information is not covered in this document.

Finding Platform-Independent Feature Information

For the latest platform-independent feature information and caveats, see the Cross-Platform Release Notes for Cisco IOS Release 15.1(1)S at:
http://www.cisco.com/en/US/products/ps10890/prod_release_notes_list.html

Document Revision History

Table 1 records technical changes to this document. The table shows the Cisco IOS software release number and document revision number for the change, the date of the change, and a brief summary of the change.

Table 1 Document Revision History

Release No.	Revision	Date	Change Summary
15.2(2)S	OL-11907-17	March 2012	• Added VRF aware IPv6 tunnel, page 271.
15.2(1)S	OL-11907-16	November 2011	• Added IPv6 Policy Based Routing, page 286.
15.1(3)S	OL-11907-15	July 2011	• Updated Configuring Private Host Switch Virtual Interface (VLAN and VPLS), page 254. Added support for Multiple BPDU PW. • Updated Configuring Multicast Features. Added support for Multicast VLAN Registration. • Added Pseudo MLACP Support on Cisco 7600
15.1(2)S	OL-11907-14	March 2011	• Updated the Configuring REP Configurable Timers for the Cisco 7600 Router. Added SSO support. • Added BPDU PW Over LAG NNI.
15.1(1)S	OL-11907-13	November 2010	Updated the following features: • Updated Configuring Custom Ethertype for EVC Interfaces section. Added support for EVC Port Channel. • Updated Configuring MST on EVC Bridge Domain section. Added support for EVC Port Channel. • Updated Configuring Unidirectional Link Detection (UDLD) on Ports with EVCs section. Added support for EVC Port Channel. • Updated Configuring Resilient Ethernet Protocol. Added support for the REP No-Neighbor functionality. • Added Configuring Link State Tracking (LST) section.
12.2(33) SRD5	OL-11907-12	October 2010	• Added troubleshooting information for carrier ethernet features in Chapter 2, "Configuring the Cisco 7600 Series Ethernet Services 20G Line Card" and QoS features in Chapter 3, "Configuring QoS on the Cisco 7600 Series Ethernet Services 20G Line Card".

Table 1 Document Revision History

15.0(1)S	OL-11907-11	July 2010	- Added support for Ingress Policing on EVC Port Channel. - Added "Support for IEEE 802.1ad" section on page 30. - Added "TE-FRR Support on VPLS LAG NNI" section on page 142. - Updated configuration, restrictions and usage guidelines for "Traffic Storm Control on ES20 with EVCs" section on page 161. - Added support for "Configuring REP Configurable Timers for the Cisco 7600 Router" section on page 220.
12.2(33)SRE1	OL-11907-10	June 2010	- Added restrictions pertaining to established connections for QoS classification.
12.2(33)SRE1	OL-11907-10	May 2010	- Updated configurable values for cir and pir when using the bandwidth command.
12.2(33)SRE1	OL-11907-10	April 2010	- Updated IEEE 802.1ag-2007 Compliant CFM section with restrictions on EVC manual load balancing configuration.
12.2(33)SRE1	OL-11907-10	April 2010	- Extended feature support for Private Host SVI.
12.2(33)SRD4	OL-11907-09	March 2010	- Added a new QoS restriction on TCAM entry and the maximum number of unique class-maps supported.
12.2(33)SRD4	OL-11907-09	February 2010	- Added support for Private Host SVI in mainline documentation.
12.2(33)SRE	OL-11907-08	February 2010	- Updated the section Service Scalability in chapter 2.
12.2(33)SRE	OL-11907-08	January 2010	- Updated restrictions in the EVCS QoS Support section.
12.2(33)SRE	OL-11907-08	December 2009	- Added information about cross-bundling and support on ingress for **Set MPLS experimental** (EXP) bit on label imposition. - Add information about egress MPLS EXP classification on ES20 ports.

Table 1 Document Revision History

12.2(33)SRE	OL-11907-08	November 2009	• Added QoS support for EVC Group • Added GE LAG with LACP on UNI with Advanced Load Balancing • Added support for L3/L4 ACL on service instance • Added support for Flexible Service Mapping based on CoS, Ethertype • Added support for L3 classification and marking on EVC • Added support for H-VPLS with port-channel core interface • Added Multichassis support for LACP • Added support for IEEE 802.1ag Draft 8.1compliant Connectivity Fault Management • Added support for CFM (D8.1) over EFP with xconnect for 7600 • Added support for REP integration with EVC & VPLS inter-working • Added support for Excalibur Reverse L2GP for 7600 • Added support for Static MAC binding to EVCs and Pseudowires • Added support for Static MAC on EFP and PW • Added support for Resilient Ethernet Protocol over Ethernet Virtual Circuit • Added support for CFM over EFP Interface with xconnect • Added support for Reverse L2GP for the Cisco 7600 Router • Added support for 802.1ah
12.2(33)SRD3	OL-11907-07	September	• ******Private Host SVI feature released only to specific customers with a copy of private documentation that is not available in CCO.*******
12.2(33)SRD	OL-11907-06	June 2009	• Added SPAN restrictions

Table 1 Document Revision History

12.2(33)SRD	OL-11907-06	October 2008	• Added configuring L2 ACL on EVC • Added configuring Broadcast Storm Control on Switchports and ports with Ethernet Virtual Connections • Added configuring Asymmetric Carrier Delay • Added configuring ATM/FR to Ethernet • Added configuring Custom Ethertype interfaces • Added configuring Dual Rate Three Color (2R3C) Ingress Service Policing • Added configuring Unidirectional Link Detection (UDLD • Added configuring Bandwidth Remaining Ratio (BRR) • Added configuring L2 Classification on Default EVC • Added configuring MAC security for EVC bridge-domain • Added shaping support on an ES20 maininterface
12.2(33)SRC2	OL-11907-05	September 2008	• Added UDE support on ES20 line cards • Added Shaping H-QoS on ES20 support
12.2(33)SRC1	OL-11907-04	April 2008	• Added 802.3ad LACP over EVC support
12.2(33)SRC	OL-11907-03	January 2008	• Added SFP-GE-T support • Added 32K EVC scale
12.2(33)SRB1	OL-11907-02	June 2007	• Added 1 rate 2 color per EVC micro-flow policer • Added Backup Interface for Flexible UNI.
12.2SRB	OL-11907-01	February 2007	Initial version

Organization

This document contains the following chapters:

Section	Title	Description
Chapter 1	Overview of the Cisco 7600 Series Ethernet Services 20G Line Card	Provides an introduction to the Cisco 7600 Series Ethernet Services 20G line card.
Chapter 2	Configuring the Cisco 7600 Series Ethernet Services 20G Line Card	Provides information on configuring software features.
Chapter 3	Configuring QoS on the Cisco 7600 Series Ethernet Services 20G Line Card	Provides information on configuring QoS specific to the Cisco 7600 Series Ethernet Services 20G line card.
Chapter 4	Command Summary for the Cisco 7600 Series Ethernet Services 20G Line Card	Provides a summary of the commands for the Ethernet Services 20G line card on the Cisco 7600 series router.
Chapter 5	Troubleshooting the Cisco 7600 Series Ethernet Services 20G Line Card	Provides troubleshooting information associated with the Ethernet Services.
Chapter 6	Upgrading Field-Programmable Devices	Provides information about verify image versions and performing Cisco 7600 Series ES20 line card Field-Programmable Device upgrades.

Related Documentation

This section refers you to other documentation that also might be useful as you configure your Cisco 7600 series router. The documentation listed below is available online.

Cisco 7600 Series Router Documentation

As you configure your Cisco 7600 series router, you should also refer to the following companion publication for important hardware installation information:

- *Cisco 7600 Series Ethernet Services 20G Line Card Hardware Installation Guide*

An overview of the Cisco 7600 series router features, benefits, and applications you should also refer to the followinginformation:

- *Cisco 7600 Series Internet Router Essentials*

Some of the following other Cisco 7600 series router publications might be useful to you as you configure your Cisco 7600 series router.

- *Cisco 7600 Series Cisco IOS Software Configuration Guide*

  http://www.cisco.com/en/US/products/hw/routers/ps368/products_installation_and_configuration_guides_list.html

- *Cisco 7600 Series Cisco IOS Command Reference*

  http://www.cisco.com/en/US/products/hw/routers/ps368/prod_command_reference_list.html

- *Cisco 7600 Series Cisco IOS System Message Guide*

  http://www.cisco.com/en/US/products/hw/routers/ps368/products_system_message_guides_list.html

- Cisco 7600 Series Internet Router MIB Specifications Guide http://www.cisco.com/en/US/products/hw/routers/ps368/prod_technical_reference_list.html

Several other publications are also related to the Cisco 7600 series router. For a complete reference of related documentation, refer to the *Cisco 7600 Series Routers Documentation Roadmap* located at the following URL:

http://www.cisco.com/en/US/products/hw/routers/ps368/products_documentation_roadmaps_list.html

Other Cisco IOS Software Publications

Your router and the Cisco IOS software running on it contain extensive features. You can find documentation for Cisco IOS software features at the following URL:

http://www.cisco.com/cisco/web/psa/default.html?mode=prod

Cisco IOS Release 12.2SR Software Publications

Documentation for Cisco IOS Release 12.2SR, including command reference and system error messages, can be found at the following URL:

http://www.cisco.com/en/US/products/ps6922/tsd_products_support_series_home.html

Document Conventions

Within the SIP and SPA software configuration guides, the term *router* is generally used to refer to a variety of Cisco products (for example, routers, access servers, and switches). Routers, access servers, and other networking devices that support Cisco IOS software are shown interchangeably within examples. These products are used only for illustrative purposes; that is, an example that shows one product does not necessarily indicate that other products are not supported.

This documentation uses the following conventions:

Convention	Description
^ or **Ctrl**	The **^** and **Ctrl** symbols represent the Control key. For example, the key combination **^D** or **Ctrl-D** means hold down the **Control** key while you press the **D** key. Keys are indicated in capital letters but are not case sensitive.
string	A string is a nonquoted set of characters shown in italics. For example, when setting an SNMP *community* string to *public*, do not use quotation marks around the string or the string will include the quotation marks.

Command syntax descriptions use the following conventions:

Convention	Description
bold	Bold text indicates commands and keywords that you enter exactly as shown.
italics	Italic text indicates arguments for which you supply values.
[x]	Square brackets enclose an optional element (keyword or argument).
|	A vertical line indicates a choice within an optional or required set of keywords or arguments.

Convention	Description
[x | y]	Square brackets enclosing keywords or arguments separated by a vertical line indicate an optional choice.
{x | y}	Braces enclosing keywords or arguments separated by a vertical line indicate a required choice.

Nested sets of square brackets or braces indicate optional or required choices within optional or required elements. For example:

Convention	Description
[x {y | z}]	Braces and a vertical line within square brackets indicate a required choice within an optional element.

Examples use the following conventions:

Convention	Description
`screen`	Examples of information displayed on the screen are set in Courier font.
`bold screen`	Examples of text that you must enter are set in Courier bold font.
< >	Angle brackets enclose text that is not printed to the screen, such as passwords.
!	An exclamation point at the beginning of a line indicates a comment line. (Exclamation points are also displayed by the Cisco IOS software for certain processes.)
[ ]	Square brackets enclose default responses to system prompts.

The following conventions are used to attract the attention of the reader:

⚠ **Caution** Means *reader be careful*. In this situation, you might do something that could result in equipment damage or loss of data.

Note Means *reader take note*. Notes contain helpful suggestions or references to materials that may not be contained in this manual.

Tip Means *the following information will help you solve a problem*. The tips information might not be troubleshooting or even an action, but could be useful information, similar to a Timesaver.

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, submitting a service request, and gathering additional information, see the monthly *What's New in Cisco Product Documentation*, which also lists all new and revised Cisco technical documentation, at:

http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html

Subscribe to the *What's New in Cisco Product Documentation* as a Really Simple Syndication (RSS) feed and set content to be delivered directly to your desktop using a reader application. The RSS feeds are a free service and Cisco currently supports RSS Version 2.0.

CHAPTER 1

Overview of the Cisco 7600 Series Ethernet Services 20G Line Card

This chapter provides an overview of the Cisco 7600 Series Ethernet Services 20G (ES20) line card, and feature and Management Information Base (MIB) support for the ES20 line card. This chapter includes the following sections:

- Release History, page 1-1
- Supported Cisco 7600 Series Ethernet Services 20G Line Card Features, page 1-3
- Cisco 7600 ES20 Ethernet Line Card Restrictions, page 1-7
- Supported MIBs, page 1-8
- Displaying the Cisco 7600 Series Ethernet Services 20G Line Card Type, page 1-9

Release History

Release	Modification
15.1(1)S	• Updated the feature Configuring Custom Ethertype for EVC Interfaces. Added support for EVC Port Channel. • Updated the feature Configuring MST on EVC Bridge Domain. Added support for EVC Port Channel. • Updated the feature Configuring Unidirectional Link Detection (UDLD) on Ports with EVCs. Added support for EVC Port Channel. • Updated the feature Configuring Resilient Ethernet Protocol. Added support for the REP No-Neighbor functionality. • Added a new feature Configuring Link State Tracking.
15.0(1)S	Added support for the following features: • Ingress Policing on EVC Port Channel. • IEEE 802.1ad. • TE-FRR Support on VPLS LAG NNI. • Broadcast Storm Control on ES20 Line Cards with EVC Port Channel.
12.2(33)SRE1	Extended support for configuring Private Host in VPLS.
12.2(33)SRD4	Support for configuring Private Host in VPLS.

12.2(33)SRE	Support for configuring: • QoS in a EVC Group • GE LAG with LACP on UNI with Advanced Load Balancing • L3/L4 ACL on service instance • Flexible Service Mapping based on CoS, Ethertype • L3 classification and marking on EVC • H-VPLS with port-channel core interface • Multichassis support for LACP • IEEE 802.1ag Draft 8.1compliant Connectivity Fault Management • CFM (D8.1) over EFP with xconnect for 7600 • REP integration with EVC & VPLS interworking • Excalibur Reverse L2GP for the Cisco 7600 router • Static MAC binding to EVCs and Pseudowires • Support for 802.1ah
12.2(33)SRD	Support for configuring: • L2 ACL on EVC. • Broadcast Storm Control on Switchports and ports with Ethernet Virtual Connections. • Asymmetric Carrier Delay. • ATM/FR to Ethernet. • Custom Ethertype interfaces. • Dual Rate Three Color (2R3C) Ingress Service Policing. • Unidirectional Link Detection (UDLD. • Bandwidth Remaining Ratio (BRR). • L2 Classification on Default EVC. • MAC security for EVC bridge-domain. • Shaping support on an ES20 main interface.
Cisco IOS Release12.2(33)SRC2	• Added UDE support on ES20 line cards • Added Shaping H-QoS on ES20 support
Cisco IOS Release12.2(33)SRC1	• Support for 802.3ad LACP over Ethernet Virtual Connection (EVC)
Cisco IOS Release 12.2(33)SRC	Support for the following features was introduced on the ES20 line card: • SFP-GE-T support • 32K EVC scale

Cisco IOS Release 12.2(33)SRB1	Support for the following features was introduced on the ES20 line card: • 1 rate 2 color per EVC policer • Backup interface for flexible UNI
Cisco IOS Release 12.2SRB	Support for the following features was introduced on the ES20 line card: • Hierarchical Quality of Service (HQoS) with Multipoint Bridging (MPB) on 7600 ES20 • Flexible QinQ Mapping and Service Awareness on 2-port 10GE ES20 and 20-port GE ES20 • MultiPoint Bridging over Ethernet on 2-port 10GE ES20 and 20-port GE ES20 • IGMP/PIM Snooping for VPLS pseudowire on 2-port 10GE ES20 and 20-port GE ES20 • Scalable EoMPLS on 2-port 10GE ES20 and 20-port GE ES20 • QoS Enhancement for Dual Priority Queues on Service Instances

Supported Cisco 7600 Series Ethernet Services 20G Line Card Features

The following software features are supported on the ES20 line card:

- Layer 2 Features, page 1-3
- Layer 3 and Layer 4 Features, page 1-4
- Multicast Features, page 1-4
- High Availability Features, page 1-4
- MPLS Features, page 1-5
- Layer 2 Protocols and Encapsulation, page 1-6
- QoS Features, page 1-6
- Accounting and Management Features, page 1-7

Layer 2 Features

- Layer 2 switch port (EtherChannel only)
- EtherChannel and Link Aggregate Control Protocol (IEEE 802.3ad)
- Multiple Registration Protocol (IEEE 802.1ak)
- Subinterfaces
- Switch virtual interface (SVI)
- Subinterface Switchport / Subinterfaces MultiPoint Bridging (MPB) with Spanning Tree
- Jumbo frames
- Ethernet encapsulation
- VLAN scaling Layer 2 switching /VPLS

- 16K VLAN scaling
- Ethernet Interface flow control, rate and transmission
- Pause frames
- Address Resolution Protocol (ARP)/Reverse Address Resolution Protocol (RARP)
- Source MAC filtering
- Layer 2 switching
- Ethernet Multipoint Bridging with Local VLAN significance per port
- VLAN termination and grouping policy (VLAN technology)
- Double-tag IP termination
- LACP over EVC Port Channel
- L2 ACL (Access Control List) on EVC
- Broadcast Storm Control on Switchports and Ports with EVC (Ethernet Virtual Connections)
- Asymmetric Carrier Delay
- ATM/FR to Ethernet
- Custom Ethertype for EVC Interfaces
- Multichassis support for LACP
- H-Virtual Private LAN Service (VPLS) Within a Port-Channel Core Interface
- Flexible Service Mapping Based on CoS and Ethertype
- Gigabit Ethernet Link Aggregation with Link Aggregation Control Protocol on User-to-Network Interface with Advanced Load Balancing
- Private Host SVI

Layer 3 and Layer 4 Features

- Flexible QinQ mapping and termination
- Configuring Layer 3 and Layer 4 ACL on Service Instance

Multicast Features

- Multicast replication: Layer 2 and Layer 3
- Internet Group Management Protocol (IGMP) snooping
- IGMP snooping for QinQ
- Multicast groups

High Availability Features

- On-Board Failure Logging (OBFL)
- Online insertion and removal (OIR) of the ES20 line card
- Nonstop forwarding (NSF)

- Stateful switchover (SSO)
- Route Processor Redundancy (RPR)
- Route Processor Redundancy + (RPR+)
- Bi-directional Forwarding Detection (BFD) with IS-IS
- BFD Support with Fast Reroute (FRR)
- BFD Support with Border Gateway Protocol (BGP)
- BFD Support with Open Shortest Path First (OSPF)
- In-Service Software Upgrade (ISSU) with Enhanced Fast Software Upgrade (eFSU)
- Unidirectional Link Detection (UDLD)
- IEEE 802.1ag-2007 Compliant CFM - Bridge Domain Support

MPLS Features

- Unicast switching, with specific support for up to six label push operations, one label pop operation (two label pop operations in case of Explicit Null), or one label swap with up to five label push operations, at each MPLS switch node
- Support for Explicit Null label to preserve CoS information when forwarding packets from provider (P) to provider edge (PE) routers
- Support for Implicit Null label to request that penultimate hop router forward IP packets without labels to the router at the end of the label switch path (LSP)
- 1000 VRFs
- 1M VRF Routes (1000 routes/VRF)
- Traffic engineering (TE)
- DiffServ TE
- FRR
- Any Transport over MPLS (AToM) support—EoMPLS only, including:
  - EoMPLS VLAN-to-VLAN transport
  - EoMPLS PW into VRF
  - EoMPLS VLAN-to-Port transport
  - EoMPLS Port-to-Port transport
  - EoMPLS VLAN to LSP/TE tunnel mapping
  - EoMPLS SVI-based EoMPLS plus local switching
  - EoMPLS IEEE 802.1D Spanning Tree Protocol (STP)
  - EoMPLS MAC learning and forwarding
- AToM Uplink
- Virtual Private LAN Service (VPLS) support, including:
  - H-VPLS with MPLS edge—H-VPLS with MPLS edge requires either an optical service module (OSM), Cisco SIP-400, Cisco SIP-600, or ES20 line card in both the downlink (facing UPE) and uplink (MPLS core). For more information about configuring H-VPLS, see "Configuring Virtual Private LAN Service" section on page 2-302.

- H-VPLS with QinQ edge—Requires ES20 line card in the uplink, and any LAN port or ES20 line card on the downlink
- Up to 4000 VPLS domains
- Up to 60 VPLS peers per domain(110 from Release12.2(33)SRD onwards)
- Up to 30,000 pseudo-wires, used in any combination of domains and peers up to the 4000-domain or 60-peer maximums. For example, support of up to 4000 domains with 7 peers or up to 60 peers in 500 domains
- Tunnel selection

Layer 2 Protocols and Encapsulation

Layer 2 Gigabit Ethernet support, including:

- IEEE 802.3z 1000 Mbps Gigabit Ethernet
- IEEE 802.3ab 1000BASET Gigabit Ethernet
- IEEE 802.3ae 10 Gbps Ethernet (1-Port 10-Gigabit Ethernet SPA only)
- Jumbo frame (up to 9216 bytes)
- ARPA, IEEE 802.3 SAP, IEEE 802.3 SNAP, QinQ
- IEEE 802.1Q VLANs
- Autonegotiation support including IEEE 802.3 flow control and pause frames
- Gigabit Ethernet Channel (GEC)
- IEEE 802.3ad link aggregation
- Address Resolution Protocol (ARP) and Reverse ARP (RARP)
- Hot Standby Router Protocol (HSRP)
- Virtual Router Redundancy Protocol (VRRP)

QoS Features

This section provides a list of the Quality of Service (QoS) features that are supported by the ES20 line card.

- Modular QoS CLI (MQC) support
- QoS marking ingress, egress
- QoS priority bit manipulation
- Queue count
- Weighted Random Early Detection (WRED) per queue
- Access Control List (ACL) count
- ACLs per port
- ACLs per VLAN
- ACLs per subinterface
- Input policing (port)

- Input policing (VLAN)
- Egress traffic shaping
- Hiearchical traffic shaping (2 levels): Egress service instance
- Hierarchical traffic shaping (2 levels): Interface
- Hierarchical Traffic Shaping (2 levels): Subinterface
- Hierarchical Traffic Shaping (2 levels): Subinterface Groups
- Hierarchical Traffic Shaping: Class-Based Queuing (CBQ), Low Latency Queueing (LLQ)
- LLQ
- Class-based Weighted Fair Queueing (CBWFQ)
- CoS mapping to Differentiated Services Code Point (DSCP) and Multiprotocol Label Switching Experimental bit (MPLS EXP)
- DSCP mapping to MPLS EXP and CoS
- Dual Rate Three Color (2R3C) Ingress Service Policing
- Shaping Support on the Main Interface
- Bandwidth Remaining Ratio (BRR)
- Qos over EVC Group

Accounting and Management Features

- Ingress packet counter
- Egress packet counter
- Cisco Element Manager Framework (CEMF)
- VPN Solution Center (VPNSC)
- Online diagnostics
- Offline diagnostics
- Source address MAC accounting (per port byte and packet per Source Accounting (SA))
- Source address MAC accounting (per port)
- Destination MAC accounting (per port byte and packet per Destination Accounting (DA))
- Destination MAC accounting (per port)
- VLAN accounting
- Sampled Netflow
- BGP policy accounting

Cisco 7600 ES20 Ethernet Line Card Restrictions

This section documents unsupported features and feature restrictions for the ES20 line card on the Cisco 7600 series router.

As of Cisco IOS Release 12.2(33)SRB, ES20 line card:

- Is not supported by the Supervisor Engine 32e.
- Is supported by the Supervisor Engine 720 PFC3B and Supervisor Engine 720 PFC3BXL.
- Is not supported with a Supervisor Engine 720 PFC3A or in PFC3A mode.

For more information about the requirements for Policy Feature Cards (PFCs) on the Cisco 7600 series router, refer to the *Release Notes for Cisco IOS Release 12.2SR for the Cisco 7600 Series Routers* at http://www.cisco.com/en/US/docs/ios/12_2sr/release/notes/122SRrn.html

Switch Port Analyzer (SPAN) Restrictions

Follow these SPAN restrictions when you configure a ES20 line card on the Cisco 7600 series router:

- SPAN is not supported on EVC ports.
- Traffic is incorrectly relayed to a local SPAN port instead of the SPAN destination port when the following conditions occur:
  - When the SPAN source and SPAN destination ports belong to ports [0-9] or both ports belong to ports [10-19]
  - The SPAN source port has a QoS policy-map attached and SPAN traffic matches that policy-map except for class-default.

Using ports [0-9] as SPAN source and [10-19] as SPAN destination or vice versa to avoid the above mentioned limitations.

- You cannot use Gi|Te x/0/1 as a SPAN source port when the diagnostic test **TestMacNotification** is enabled.

For information the diagnostic tests, refer to Cisco 7600 Series Cisco IOS Software Configuration Guide, 12.2SX - Online Diagnostic Tests at:

http://www.cisco.com/en/US/docs/routers/7600/ios/12.2SR/configuration/guide/diagtest.html

Supported MIBs

The following MIBs are supported in Cisco IOS Release 12.2SRB and later releases:

- CISCO-ENTITY-EXT-MIB
- CISCO-ENTITY-FRU-CONTROL-MIB
- ENTITY-MIB
- OLD-CISCO-CHASSIS-MIB
- Class-Based MIB (Cisco Classed-Based QoS MIB)
- IF-MIB (Interface MIB)
- Bridge Domain MIB

For more information about MIB support on a Cisco 7600 series router, refer to the *Cisco 7600 Series Internet Router MIB Specifications Guide*, at the following URL:

http://www.cisco.com/en/US/products/hw/routers/ps368/prod_technical_reference_list.html

To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL:

http://tools.cisco.com/ITDIT/MIBS/servlet/index

If Cisco MIB Locator does not support the MIB information that you need, you can also obtain a list of supported MIBs and download MIBs from the Cisco MIBs page at the following URL:

http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

To access Cisco MIB Locator, you must have an account on Cisco.com. If you have forgotten or lost your account information, send a blank e-mail to cco-locksmith@cisco.com. An automatic check will verify that your e-mail address is registered with Cisco.com. If the check is successful, account details with a new random password will be e-mailed to you. Qualified users can establish an account on Cisco.com by following the directions found at this URL:

https://tools.cisco.com/RPF/register/register.do

Displaying the Cisco 7600 Series Ethernet Services 20G Line Card Type

To verify the ES20 line card hardware type that is installed in your Cisco 7600 series router, you can use the **show module** command. There are other commands on the Cisco 7600 series router that also provide ES20 line card hardware information, such as the **show idprom** command and **show diagbus** command. For a summary of some other ES20 line card commands, see Chapter A, "Command Summary for the Cisco 7600 Series Ethernet Services 20G Line Card."

The following example shows output from the **show module** command on the Cisco 7600 series router with an ES20 line card installed in slot 2:

```
Router# show module 2
Mod Ports Card Type                              Model              Serial No.
--- ----- -------------------------------------- ------------------ -----------
  2    0  ESM20G                                 7600-ES20-BASE     JAB1030007C

Mod MAC addresses                       Hw    Fw           Sw           Status
--- ---------------------------------- ------ ------------ ------------ -------
  2  00e0.aabb.cc00 to 00e0.aabb.cc00   1.0   12.2(2006032 12.2(2006110 PwrDown

Mod  Sub-Module                  Model              Serial       Hw     Status
---- --------------------------- ------------------ ----------- ------- -------
  2  ESM20G/PFC3C Distributed Fo 7600-ES20-D3C      JAB1030008H  1.0    PwrDown

Mod  Online Diag Status
---- -------------------
  2  Not Applicable
Router#
```

The following example shows sample output for an ES20 line card installed in slot 2 of the router:

```
Router# show idprom module 2
IDPROM for module #2
(FRU is 'ESM20G')
OEM String = 'Cisco Systems, Inc'
Product Number = '7600-ESM-BASE'
Serial Number = 'JAB1030007F'
Manufacturing Assembly Number = '73-10437-04'
Manufacturing Assembly Revision = '04'
Hardware Revision = 0.48
Current supplied (+) or consumed (-) = -3.63A

Router#
```

CHAPTER 2

Configuring the Cisco 7600 Series Ethernet Services 20G Line Card

This chapter provides information about configuring the Cisco 7600 Series Ethernet Services 20G (ES20) line card on the Cisco 7600 series router. It includes the following sections:

- Required Configuration Tasks, page 2-2
- Configuring High-Availability Features, page 2-3
  - Configuring UDE on ES20 Line Cards, page 2-3
  - Configuring Unidirectional Link Detection (UDLD) on Ports with EVCs, page 2-8
  - Configuring IEEE 802.1ag-2007 Compliant CFM, page 2-13
  - Support for IEEE 802.1ad, page 2-30
- Configuring Layer 2 Features, page 2-68
  - Cross-Bundling, page 2-69
  - Multichassis Support for Link Aggregation Control Protocol, page 2-74
  - Pseudo MLACP Support on Cisco 7600, page 2-93
  - Configuring Flexible QinQ Mapping and Service Awareness on 7600-ESM-2X10GE and 7600-ESM-20X1GE, page 2-110
  - Configuring Flexible Service Mapping Based on CoS and Ethertype, page 2-119
  - Configuring MultiPoint Bridging over Ethernet on 7600-ESM-2X10GE and 7600-ESM-20X1GE, page 2-128
  - Configuring Gigabit Ethernet Link Aggregation with Advanced Load Balancing, page 2-134
  - Configuring Virtual Private LAN Service (VPLS) with Port-Channel as a Core Interface, page 2-141
  - Configuring the Backup Interface for Flexible UNI, page 2-150
  - Configuring Broadcast Storm Control on Switchports and Ports with Ethernet Virtual Connections, page 2-160
  - Traffic Storm Control on ES20 Switchports, page 2-161
  - Traffic Storm Control on ES20 with EVCs, page 2-161
  - Configuring Asymmetric Carrier-Delay, page 2-165
  - Configuring MST on EVC Bridge Domain, page 2-167
  - DHCP Snooping with Option-82 on EVC, page 2-174

  - Configuring MAC Address Security for EVC Bridge-Domain, page 2-178
  - Configuring Static MAC on Ethernet Flow Point and Pseudowire, page 2-193
  - Configuring Resilient Ethernet Protocol, page 2-201
  - Configuring CFM over EFP Interface with cross connect, page 2-228
  - Configuring Reverse Layer 2 Gateway Ports for the Cisco 7600 Router, page 2-245
  - Configuring Private Host Switch Virtual Interface (VLAN and VPLS), page 2-254
  - IPv6 Policy Based Routing, page 2-286
- Configuring Multicast Features, page 2-258
  - Configuring IGMP/PIM Snooping for VPLS Pseudowire on 7600-ESM-2X10GE and 7600-ESM-20X1GE, page 2-258
  - Configuring Link State Tracking (LST), page 2-260
  - Configuring Multicast VLAN Registration, page 2-262
- Configuring Layer 3 and Layer 4 Features, page 2-265
  - Configuring Layer 3 and Layer 4 Access Control List on a Service Instance, page 2-265
  - VRF aware IPv6 tunnel, page 2-271
- Configuring MPLS Features, page 2-288
  - Configuring Any Transport over MPLS, page 2-289
  - Configuring MPLS Traffic Engineering Class-Based Tunnel Selection, page 2-293
  - Configuring Virtual Private LAN Service, page 2-302
  - Configuring SVI-Based IP/Routed Interworking, page 2-304
- Resetting a Cisco 7600 Series Ethernet Services 20G Line Card, page 2-307

For information about managing your system images and configuration files, refer to the *Cisco IOS Configuration Fundamentals Configuration Guide* and *Cisco IOS Configuration Fundamentals Command Reference* publications that correspond to your Cisco IOS software release.

For more information about some of the commands used in this chapter, see

Chapter A, “Command Summary for the Cisco 7600 Series Ethernet Services 20G Line Card,” and the *Cisco IOS Release 12.2 SR Command References at* http://www.cisco.com/en/US/products/ps6922/prod_command_reference_list.html

Also refer to the related Cisco IOS software command reference and master index publications. For more information about accessing these publications, see the “Related Documentation” section on page -xviii.

Required Configuration Tasks

As of Cisco IOS Release 12.2SRB, there are not many features that require direct configuration on the ES20 line card. You do not need to attach to the ES20 line card itself to perform any configuration.

Identifying Slots and Subslots for the Cisco 7600 Series Ethernet Services 20G Line Card

The ES20 line card supports In-Service Software Upgrade (ISSU) with Enhanced Fast Software Upgrade (eFSU). ISSU allows for the upgrade and downgrade of Cisco IOS images at different release levels on the active and standby supervisors. ISSU procedure also applies to upgrade and downgrade of line card images. A new line card image is loaded, as necessary, when the supervisor engine software is upgraded or downgraded.

For more information, see the *Cisco 7600 Series Cisco IOS Software Configuration Guide, 12.2SR* at http://www.cisco.com/en/US/docs/routers/7600/ios/12.2SR/configuration/guide/swcg.html

Configuring High-Availability Features

This section provides information about configuring high-availability features specific to the ES20 line card.

Configuring UDE on ES20 Line Cards

Unidirectional Ethernet (UDE) feature allows interfaces to operate as unidirectional links, that is, either in receive (Rx) only mode or transmit (Tx) only mode. Unidirectional Ethernet uses only one strand of fiber for either transmitting or receiving one-way traffic for Gig Ports instead of two strands of fiber for a full duplex operation.

UDLR mechanism allows a set of feeds and receivers, which are directly connected by a unidirectional link, to send datagrams as if they were all connected by a bidirectional link.

Unidirectional Routing is used in applications with a great amount of data traffic flowing in one direction and little control traffic flowing in the opposite direction.

When an interface is configured as unidirectional send-only or receive-only, the following actions take place:

- For ports configured as send-only, the port ONLY transmits data and ignores any received data. Similarly receive-only ports do not transmit data.
- UDLD is automatically disabled on the interface.
- Autonegotiation is disabled on the interface.

Restrictions and Usage Guidelines

The following restrictions apply to the UDE links on ES20 line cards:

- Uni Directional Link Routing (UDLR) is configured only on routed ports. Configure the IPv4 address on the UDLR tunnel. Each UDE can either be a switched port or a routed port and has a separate UDLR tunnel. UDLR handles bidirectional communication over the back channel.
- Configuring unidirectional links may cause STP Loops. You must configure protocols correctly to avoid problems with the network.

- For unidirectional links, you should manually configure the encapsulation and trunk mode to fixed values on each side. The protocol is not aware of the link type and will continue to try and negotiate if it is configured to do so. If both sides of a unidirectional link are negotiating, it is possible to get a trunk mismatch where the receive-only side becomes a trunk while the send-only side is access.
- For send-only unidirectional links, switches cannot receive any CDP information about neighbors and VLAN mismatches cannot be detected.
- VTP will not work if the VTP server is downstream of the unidirectional link. VTP pruning on send-only unidirectional links should preferably be disabled.
- Dot1x is incompatible with ULDR.
- If the link between a switch and a host is made unidirectional, IGMP snooping will not work because either the host will not receive IGMP queries from the switch or the switch will not receive IGMP reports from the host.
- If two network devices are connected by a unidirectional link, then ARP requests and the response mechanism will not work. Additionally, static entries need to be created for proper functionality of protocols depending on such a mechanism.
- Link Detection does not work on unidirectional interfaces.
- Unidirectional Ethernet with EtherChannel configuration is not supported on ES20 line cards.
- Receive-only transceivers are not supported.
- UDE is only supported on a single fiber .
- ISIS does not work with UDE/UDLR.
- Auto-rp discovery packets are not received on the UDE receive-only port if UDE is configured with SVI. UDE links configured on routed ports do not have this issue.
- The **loopback mac** command should not be configured explicitly when UDE is configured on an ES20 port. Similarly UDE should not be configured if **loopback mac** is configured on an ES20 port.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface**
4. **unidrectional {send-only | receive-only}**
5. **ip-address** ***ip_address mask***

DETAILED STEPS

	Command	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`**
	Example:
	`Router(config)# interface tengigabitethernet 1/0/0`
		Enters interface mode.

Step 4	**`unidirectional {send-only | receive-only}`**
	Example:
	`Router(config-if)# unidirectional send-only`
	or
	`Router(config-if)# unidirectional receive-only`
		Configures the interface as a unidirectional send-only or a unidirectional receive-only link.
		Note To enable UDE, a link can be configured either as send-only or as receive-only, but not both.

Step 5	**`ip-address {ip_address mask}`**
	Example:
	`Router(config-if)#` **`ip address`** *`11.0.0.2 255.201.220.10`*
		Assigns an IP address and subnet mask to the unidirectional link.

Examples

Router A Configuration

In this example, interface 10.1.0.1 on Router A is configured as the send-only port while the tunnel running from 11.0.0.1 to 11.0.0.2 is configured as the receive-only interface.

```
interface tengigabitethernet 1/0/0
unidirectional send-only
ip address 10.1.0.1 255.255.0.0
ip pim sparse-dense-mode
!
! Configure tunnel as receive-only UDLR tunnel!
interfacetunnel 0
tunnel source 11.0.0.1
tunnel destination 11.0.0.2
tunnel udlr receive-only tengigabitethernet 1/0/0
```

Router B Configuration

In this example, interface 10.1.0.2 on Router B is configured as the receive-only port while the tunnel running from 11.0.0.2 to 11.0.0.1 is configured as the receive-only interface.

```
Config e.g 1
interface tengigabitethernet 1/0/0
unidirectional receive-only
ip address 10.1.0.2 255.255.0.0
ip pim sparse-dense-mode
!
! Configure tunnel as send-only UDLR tunnel.
!
interface tunnel 0
tunnel source 11.0.0.2
tunnel destination 11.0.0.1
tunnel udlr send-only tengigabitethernet 1/2
tunnel udlr address-resolution
Config e.g 2
interface GigabitEthernet1/0/0
switchport
switchport access vlan 100
switchport mode access
no ip address
speed nonegotiate
unidirectional send-only

interface Vlan100
ip address 10.0.1.1 255.255.255.0
ip pim sparse-mode
```

Switched Port UDE Configuration

This example shows UDE configuration on a switched port with an SVI interface, with OSPF enabled.

```
Topology :

    [UDE-R1]-----UDE--------->[UDE-R2]
           <-------UDLR-------

UDE-R1#sh run int gig2/0/5
Building configuration...

Current configuration : 165 bytes
!
interface GigabitEthernet2/0/5
 switchport
 switchport access vlan 300
 switchport mode access
 speed nonegotiate
 no mls qos trust
 unidirectional send-only
end

UDE-R1#sh run int tunnel 10
Building configuration...

Current configuration : 150 bytes
!
interface Tunnel10
 ip address 70.10.10.1 255.255.255.0
 tunnel source 50.0.0.1
 tunnel destination 50.0.0.2
 tunnel udlr receive-only Vlan300
end

UDE-R1#sh run int vlan 300
Building configuration...

Current configuration : 104 bytes
!
interface Vlan300
 ip address 90.90.90.99 255.255.255.0

end

router ospf 1
 log-adjacency-changes
 network 20.0.0.0 0.0.0.255 area 0
 network 90.90.90.0 0.0.0.255 area 0


############ config on R2####################

UDE-R2#sh run int gig2/0/1
Building configuration...

Current configuration : 170 bytes
!
interface GigabitEthernet2/0/1
 switchport
 switchport access vlan 300
 switchport mode access
 speed nonegotiate
 mls qos trust dscp
 unidirectional receive-only
end

UDE-R2#sh run int tunnel 10
Building configuration...

Current configuration : 179 bytes
!
interface Tunnel10
 ip address 70.10.10.2 255.255.255.0
 tunnel source 50.0.0.2
 tunnel destination 50.0.0.1
 tunnel udlr send-only Vlan300
 tunnel udlr address-resolution
end

UDE-R2#sh run int vlan 300
Building configuration...

Current configuration : 82 bytes
!
interface Vlan300
 ip address 90.90.90.90 255.255.255.0
 end

router ospf 1
 log-adjacency-changes
 network 30.0.0.0 0.0.0.255 area 0
 network 90.90.90.0 0.0.0.255 area 0
```

Verification

Use the following commands to verify operation.

Command	Purpose

`Router#`**`sh interface`** *`interface-id`* **`unidirectional`**
`Example:`
`Router#` **`sh interface`** `gigabitEthernet 2/0/5 unidirectional`
`Unidirectional configuration mode: send only`
`CDP neighbour unidirectional configuration mode: off`
	Displays information about the UDE configuration on a specific interface.

Configuring Unidirectional Link Detection (UDLD) on Ports with EVCs

UDLD (Unidirectional Link Detection) is a Layer 2 protocol that interacts with a Layer 1 protocol to determine the physical status of a link. At Layer 1, physical signaling and fault detection is auto-negotiated. UDLD detects the neighbor link, identifies, and disables the wrongly connected LAN ports. When you enable auto-negotiation and UDLD, Layer 1 and Layer 2 detections prevent physical and logical unidirectional connections, and malfunctioning of other protocols.

A unidirectional link occurs when the neighbor link receives the traffic transmitted by the local device, but the local device does not receive the transmitted traffic from its neighbor. If auto-negotiation is active, and one of the fiber strands in a pair is disconnected, the link is disabled. The logical link is undetermined, and UDLD does not take any action. At Layer 1, if both fibers are normal, UDLD at Layer 2 determines if the fibers are accurately connected, and traffic is relayed bidirectionally between the right neighbors. In this scenario, auto-negotiation operates in Layer 1, and the link status is unchecked.

The UDLD protocol monitors physical configuration of the cables, and detects unidirectional links of devices connected to LAN ports via Ethernet cables. When a unidirectional link is detected, UDLD disables the affected LAN port, and alerts the user.

The Cisco 7600 series router periodically transmits UDLD packets to neighboring devices on LAN ports with UDLD. If the packets are returned within a specific time frame, and there is no acknowledgement, the link is flagged as unidirectional, and the LAN port is disabled.

Restrictions and Usage Guidelines

Follow these restrictions and usage guidelines while configuring UDLD on ports with EVCs:

- You can configure UDLD only on a port.
- To identify and disable the unidirectional links, devices at both ends must support UDLD.
- Service bridge domain should be available on the router.
- Any of the supported EVC encapsulation can be configured.
- Cisco IOS Release 15.1(1)S supports EVC port-channels.

Note If UDLD is enabled on an EVC port with service type **connect** or **xconnect** and encapsulation type **default** or **untagged**, the port is disabled.

For more information on UDLD, see the *Cisco 7600 Series Cisco IOS Software Configuration Guide, 12.2SR* at the following URL:

http://www.cisco.com/en/US/docs/routers/7600/ios/12.2SR/configuration/guide/udld.html

Configuring UDLD Aggressive Mode

As UDLD aggressive mode is disabled by default, you can configure UDLD aggressive mode in point-to-point links between network devices that support UDLD aggressive mode.

When UDLD aggressive mode is enabled:

- A port on a bidirectional link with UDLD neighbor relationship does not receive UDLD packets.
- UDLD tries to reestablish the connection with the neighbor.
- After eight failed retries, the port is disabled.

To prevent spanning tree loops, ensure that you set the non aggressive UDLD value interval to 15 seconds. This disables the unidirectional link before blocking the port transitions in the forwarding state (with default spanning tree parameters).

The benefits of enabling UDLD aggressive mode are:

- Port on one side of a link is disabled (both Tx and Rx).
- One side of a link is enabled even if the other side of the link fails.

In the above scenario, UDLD aggressive mode disables the port that prevents traffic from being discarded.

If UDLD...	Then the...
Detects a unidirectional link,	interface with its EVCs are disabled.
Is enabled on a port with an EVC bridge-domain, and encapsulation value set to default or untagged,	selected EVC is not shut down, and prevents the port from being disabled.

Enabling UDLD on Ports With EVC Configured

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **{udld | no udld} enable aggressive**
4. **exit**

DETAILED STEPS

	Command	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	`{udld | no udld} enable aggressive`
	Example:
	`Router# udld enable aggressive`
		Enables the UDLD aggressive mode.

Step 4	`exit`	Exits configuration mode.

SUMMARY STEPS

1. **interface** *type/ slot/ port*
2. **{udld port | no udld port } aggressive**
3. **show udld** *type/ slot/ port*
4. **exit**

DETAILED STEPS

	Command	Purpose
Step 1	`interface` *type/ slot/ port* **Example:** `Router(config)# gigethernet 1/0/0`	Selects the LAN port to configure.

Step 2	`{udld port | no udld port } aggressive`
	Example:
	`Router(config-if)# udld port aggressive`
	`Router(config-if)# no udld port aggressive`
		Enables a UDLD on a specific LAN port. Enter the aggressive keyword to enable **aggressive** mode. On a fiber-optic LAN port, this command overrides the **udld enable** global configuration command.
		Or
		Disables a UDLD on a non- fiber-optic LAN port.

	Command	Purpose
Step 3	`show udld` *type/ slot/ port* **Example:** `Router# show udld 1/0/0`	Verifies the configuration.
Step 4	`exit`	Exits the configuration mode.

Disabling Individual UDLD on Ports With EVC Configured

SUMMARY STEPS

1. **interface** *type/ slot/ port*
2. **{udld port | no udld port }** disable
3. **show udld** *type/ slot/ port*
4. **exit**

DETAILED STEPS

	Command	Purpose

Step 1	`interface` *type/ slot/ port*}
	Example:
	`Router(config)# gigethernet 1/0/0`
		Selects the LAN port to configure.

Step 2	`{udld port	no udld port } disable`
		Example:
		`Router(config-if)# udld port disable`
		`Router(config-if)# no udld port disable`
			Disables a UDLD on the LAN port.
			Or
			Reverts to the **udld enable** global configuration command setting.
			Note This command is supported only on fiber-optic LAN ports.

Step 3	`show udld` *type/ slot/ port* **Example:** `Router# show udld 1/0/0`	Verifies the configuration.
Step 4	`exit`	Exits the configuration mode.

Resetting Disabled UDLD on Ports With EVC Configured

SUMMARY STEPS

1. **udld reset**

DETAILED STEPS

	Command	Purpose

Step 1	`udld reset`
	
	Example:
	`Router# udld reset`
		Resets all the LAN ports disabled by UDLD.

Example

This example displays the global configuration values at router 1:

```
Router(config)#udld enable
```

This example displays the ESM20 port at router 1:

```
Router(config)# inter gi 2/0/1
Router(config-if)# udld port aggressive
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 100
Router(config-if-srv)# rewrite ingess tag translate 1-to2 dot1q 5 second-dot1q 5 symmetric
Router(config-if-srv)# bridge-domain 100
```

This example displays the configuration for a port that is part of a port channel:

```
Router(config)#interface Port-channel1
Router(config-if)#no ip address
Router(config-if)#service instance 1 ethernet
Router(config-if)#encapsulation untagged
Router(config-if)#bridge-domain 100

Router(config)#interface GigabitEthernet3/0/13
Router(config-if)#ip arp inspection limit none
Router(config-if)#no ip address
Router(config-if)#udld port aggressive
Router(config-if)#no mls qos trust
Router(config-if)#channel-group 1 mode on
```

Verification

Use the **show udld** and **show udld** *interface* commands to verify the UDLD configuration:

```
Router(config)show udld gi 3/0/13
Interface Gi1/3
---Port enable administrative configuration setting: Enabled / in aggressive mode
Port enable operational state: Enabled / in aggressive mode
Current bidirectional state: Bidirectional
Current operational state: Advertisement - Single neighbor detected
Message interval: 15
Time out interval: 5
Entry 1
    ---
    Expiration time: 37
    Cache Device index: 1
    Current neighbor state: Bidirectional
    Device ID: 011932118C0
```

```
Port ID: Gi1/1
Neighbor echo 1 device: 0FF71CA880
Neighbor echo 1 port: Gi1/3

Message interval: 15
Time out interval: 5
CDP Device name: rish2
```

ISSU Support for ES20 Line Card

The ES20 line card supports In-Service Software Upgrade (ISSU) with Enhanced Fast Software Upgrade (eFSU). ISSU allows for the upgrade and downgrade of Cisco IOS images at different release levels on the active and standby supervisors. ISSU procedure also applies to upgrade and downgrade of line card images. A new line card image is loaded, as necessary, when the supervisor engine software is upgraded or downgraded.

Configuring IEEE 802.1ag-2007 Compliant CFM

A Metro Ethernet network consists of networks from multiple operators supported by one service provider and connects multiple customer sites to form a virtual private network (VPN). Networks provided and managed by multiple independent service providers have restricted access to each other's equipment. Because of the diversity in these multiple-operator networks, failures must be isolated quickly. As a Layer 2 network, Ethernet must be capable of reporting network faults at Layer 2.

IEEE 802.3ah is a point-to-point and per- physical- wire OAM protocol that detects and isolates connectivity failures in the network. IEEE 802.1ag draft 8.1 *Metro Ethernet Connectivity Fault Management (CFM)* incorporates several OAM facilities that allow you to manage Metro Ethernet networks, including an Ethernet continuity check, end-to-end Ethernet traceroute facility using Linktrace message (LTM), Linktrace reply (LTR), Ethernet ping facility using Loopback Message (LBM), and a Loopback Reply (LBR). These Metro Ethernet CFM protocol elements quickly identify problems in the network.

Ethernet Connectivity Fault Management (CFM) is an end-to-end per-service-instance Ethernet layer operations, administration, and maintenance (OAM) protocol. It includes proactive connectivity monitoring, fault verification, and fault isolation for large Ethernet metropolitan-area networks (MANs) and WANs. Connectivity Fault Management (CFM) is the indispensable capability that service providers require to deploy large-scale, multivendor Metro Ethernet services. This feature upgrades the implementation of CFM to be compliant with the IEEE 802.1ag with the current standard, 802.1ag-2007 and implementation of CFM over L2VFI (Layer 2 Virtual Forwarding Instance Information), cross connect, EVC, and Switchport.

Key CFM mechanisms are:

- Maintenance domains (MDs) that break up the responsibilities for the network administration of a given end-to-end service.
- Maintenance associations (MAs) that monitor service instances within a specified MD.
- Maintenance points, (MPs or MIPs), such as Maintenance end points (MEP's) that transmit and receive CFM protocol messages, and MIPs that catalog information received from MEPs, and respond to Linktrace and Loopback messages.
- Protocols (Continuity Check, Loopback, and Linktrace) that are used to manage faults.

For more information on CFM, see *Cisco IOS Carrier Ethernet Configuration Guide, Release 12.2SR* at

http://www.cisco.com/en/US/docs/ios-xml/ios/cether/configuration/12-2sr/ce-12-2sr-book.html

For more information about the commands used in this section, see *Cisco IOS Ethernet Command Reference Guide* at http://www.cisco.com/en/US/docs/ios/cether/command/reference/ce_book.html

Supported Line Cards

Use the **ethernet cfm global** command to enable the CFM D8.1 feature on the following line cards:

- ES20 and ES40:Switchports, routed ports, and EVC BD.
- SIP400:Routed ports, and Layer 2 Virtual Forwarding Instance (L2VFI).
- SIP600:Switchports, and routed ports.
- 67xx: Switchports, and routed ports.

The complete support matrix for the CFM D8.1 feature is given in Table 2-1and Table 2-2.

Note Table 2-1 and Table 2-2 are part of the same table. The table is split into two for better readability.

Table 2-1 Supported Matrix1

Line card	CFM on Switchport or CFM on Switch + BD for SVI Based EoMPLS for VPLS	CFM on Routed Port	CFM on Service Instance with BD for SVI based EoMPLS for VPLS	CFM on Switchport or CFM on Switch + BD
WS-SUP720-3BXL	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP
WS-SUP720-3B	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP
RSP720-3CXL-10GE	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP
RSP720-3C-10GE	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP
RSP720-3CXL-GE	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP

Line card	CFM on Switchport or CFM on Switch + BD for SVI Based EoMPLS for VPLS	CFM on Routed Port	CFM on Service Instance with BD for SVI based EoMPLS for VPLS	CFM on Switchport or CFM on Switch + BD
RSP720-3C-GE	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP
WS-SUP32-GE-3B	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP
WS-SUP32-10GE-3B	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP
WS-X6148A	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP
WS-X6148-FE-SFP	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP
WS-X6516A-GBIC	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP
WS-X6524-100FX-MM	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP
WS-X6548-RJ-21	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP
WS-X6548-GE-TX	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP

Line card	CFM on Switchport or CFM on Switch + BD for SVI Based EoMPLS for VPLS	CFM on Routed Port	CFM on Service Instance with BD for SVI based EoMPLS for VPLS	CFM on Switchport or CFM on Switch + BD
WS-X6704-10GE	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP
WS-X6708-10G-3C	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP
WS-X6708-10G-3CXL	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP
WS-X6724-SFP	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP
WS-X6748-GE-TX	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP
WS-X6748-SFP	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Applicable	Up MEP Down MEP Port MEP
SIP-400 + V2 GE SPAs or SIP-400 + WAN SPA	Not Supported (SIP-400 + WAN SPA or SIP-400 + v2 GE SPA as uplink) No Transparency with CFM Enabled on the box	Not Supported	Not Supported	Not Supported

Line card	CFM on Switchport or CFM on Switch + BD for SVI Based EoMPLS for VPLS	CFM on Routed Port	CFM on Service Instance with BD for SVI based EoMPLS for VPLS	CFM on Switchport or CFM on Switch + BD
SIP-400 + V2 FE SPA or SIP-400 + WAN SPA	Not Supported SIP-400 + WAN SPA or SIP-400 + V2 GE SPA as uplink No Transparency with CFM Enabled on the box	Not Supported	Not Supported	Not Supported
SIP-600 + V2 GE or V2 10GE SPA or WAN SPA	Up MEP Down MEP Port MEP	Down MEP Port MEP	Not Supported	Up MEP Down MEP Port MEP
ES20-GE or ES20-10GE	Up MEP Down MEP Port MEP	Down MEP Port MEP	Up MEP Down MEP	Up MEP Down MEP Port MEP
ES+ GE /10GE	Up MEP Down MEP Port MEP	Down MEP Port MEP	Up MEP Down MEP	Up MEP Down MEP Port MEP

Table 2-2 Supported Matrix 2

Line card	CFM on Service Instance + xconnect	CFM on Service Instance + BD for SVI based EoMPLS for VPLS	CFM on L2-VFI	CFM on Routed Port
WS-SUP720-3BXL	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP
WS-SUP720-3B	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP
RSP720-3CXL-10GE	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP
RSP720-3C-10GE	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP
RSP720-3CXL-GE	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP
RSP720-3C-GE	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP
WS-SUP32-GE-3B	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP
WS-SUP32-10GE-3B	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP
WS-X6148A	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP

Line card	CFM on Service Instance + xconnect	CFM on Service Instance + BD for SVI based EoMPLS for VPLS	CFM on L2-VFI	CFM on Routed Port
WS-X6148-FE-SFP	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP
WS-X6516A-GBIC	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP
WS-X6524-100FX-MM	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP
WS-X6548-RJ-21	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP
WS-X6548-GE-TX	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP
WS-X6704-10GE	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP
WS-X6708-10G-3C	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP
WS-X6708-10G-3CXL	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP
WS-X6724-SFP	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP
WS-X6748-GE-TX	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP
WS-X6748-SFP	Not Applicable	Not Applicable	Not Applicable	Down MEP Port MEP

Line card	**CFM on Service Instance + xconnect**	**CFM on Service Instance + BD for SVI based EoMPLS for VPLS**	**CFM on L2-VFI**	**CFM on Routed Port**
SIP-400 + V2 GE SPAs or SIP-400 + WAN SPA	Not Supported No Transperency	Not Supported No Transperency	Down MEP	Down MEP Port MEP
SIP-400 + V2 FE SPA or SIP-400 + WAN SPA	Not Supported	Not Supported No Transperency	Down MEP	Down MEP Port MEP
SIP-600 + V2 GE or V2 10GE SPA or WAN SPA	Not Supported	Not Supported	Down MEP	Down MEP Port MEP
ES20-GE or ES20-10GE	Up MEP Down MEP	Up MEP Down MEP	Down MEP	Down MEP Port MEP
ES+ GE /10GE	Up MEP Down MEP	Up MEP Down MEP	Down MEP	Down MEP Port MEP

Scalable Limits

Table 2-3 maps the supported interfaces with the CFM points and their scalability values:

Table 2-3 Scalable Limits

Interfaces	CFM Points	Scalability Values
Switchports and EVC Bridge Domain (BD)	Up MEP Down MEP MIP Port MEP Remote MEP	8K MEPs per box (4K MEPs per LC) at 10 sec CC interval or higher CC intervals. 1K MEPs at 1 sec CC interval or higher CC intervals. 100 MEPs at 100 msec CC interval or higher CC intervals.
Routed Ports	Down MEP Port MEP Remote MEP	1K MEPs at 1 sec CC interval or higher CC intervals. 100 MEPs at 100 msec CC interval or higher CC intervals. 4K MEPs per box at 10 sec CC interval or higher CC intervals.

Restrictions and Usage Guidelines

When configuring CFM D8.1, follow these restrictions and usage guidelines:

- Hardware EoMPLS is not supported.
- Supports interworking between routed ports, switch ports, and EVC BD.
- CFM D8.1 QinQ configuration on a subinterface is not supported.
- You can ping or traceroute to a MEP where Continuity Check (CC) is disabled. However, you cannot use ping and traceroute for an down MEP on a STP blocked port configured on either a supervisor port or a LAN port.
- CFM is not supported with a EVC manual load balancing configuration on a EVC bridge-domain and a EVC cross-connect interface.Though configuration is not rejected, the feature may not work as expected.
- With lower CC intervals, CC packets are transmitted in bursts. Ensure that you appropriately configure the MLS rate limiters to avoid flapping of remote MEPs.
- Ping and traceroute on trunk ports for Port-MEP's and down MEP's configured on native vlan is supported only on ES20 and ES40 line cards.
- In 802.3ah E-OAM, the remote-loopback TEST status is not retained across switchovers. The remote loopback works with a longer OAM timeout value that is greater than 10 seconds.
- Migrating CFM D1.0 to D8.1 works with a reduced scale of 2k MEPs on the routed ports. If there is an EVC service configured within a domain in D1, the link fails while migrating to D8.1. To avoid this, ensure that you configure the VLAN and the EVC within the domain in D1, as shown in the next example.

Sample D1 configuration during migration:

```
ethernet cfm domain 2OUT493 level 2 direction outward
service 1 evc 493
```

Sample configuration to avoid the migration issue:

```
ethernet cfm domain 2OUT493 level 2 direction outward
service 1 evc 493
service 1 vlan 493
```

SUMMARY STEPS (COMMON CONFIGURATIONS FOR EVC, SWITCHPORT, AND ROUTED PORTS)

1. **enable**
2. **configure terminal**
3. **ethernet cfm domain** *domain-name* **level** *level-id*
4. **service** { *short-ma-name* | **number** *MA-number* | **vlan-id** *primary-vlan-id* | **vpn-id** *vpn-id* } {**vlan** *vlan-id* | **port** | **evc** *evc-name* } **direction** {***up*** | ***down***}
5. **continuity-check**
6. **continuity-check** {**interval** *CC-interval* }
7. **end**

DETAILED STEPS (COMMON CONFIGURATIONS FOR EVC, SWITCHPORT, AND ROUTED PORTS)

	Command	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	**ethernet cfm domain** ***domain-name*** **level** ***level-id*** **Example:** `PE1(config)#ethernet cfm domain L4 level 4`	Defines a CFM maintenance domain at a particular maintenance Level. It sets the router into config-ecfm configuration mode, where parameters specific to the maintenance domain can be set.

Step 4	**service {** *short-ma-name* | **number** *MA-number* | **vlan-id** *primary-vlan-id* | **vpn-id** *vpn-id* **} {vlan** *vlan-id* | **port** | **evc** *evc-name* **} direction {***up* | *down***}**
	Example:
	`Router(config-ecfm)#service s41 evc 41 vlan 41`
		Configures the maintenance association and sets a universally unique ID for a customer service instance (CSI) or the maintenance association number value, primary VLAN ID and VPN ID within a maintenance domain in Ethernet connectivity fault management (CFM) configuration mode or the direction. The default value for direction is up.

	Command	Purpose
Step 5	**continuity-check** **Example:** `Router(config-ecfm-srv)#continuity-check`	Configures the transmission of continuity check messages (CCMs), in Ethernet connectivity fault management (CFM) service configuration mode.

Step 6	**continuity-check {interval *CC-interval* }**
	Example:
	`Router(config-ecfm-srv)#continuity-check interval 10s`
		Configures the per-service parameters and sets the interval at which Continuity Check Messages are transmitted.
		• The supported interval values are:
		– 100ms 100 ms
		– 10m 10 minutes
		– 10ms 10 ms
		– 10s 10 seconds
		– 1m 1 minute
		– 1s 1 second
		– 3.3ms 3.3 ms
		– The default is 10seconds.

Step 7	`end`	Exits the interface.

SUMMARY STEPS TO CONFIGURE CFM MEP AND MIP ON A EVC

1. **enable**
2. **configure terminal**
3. **interface**
4. **service instance** {id} **ethernet** {evc-name}
5. **encapsulation** {*encapsulation-type*}
6. **bridge-domain** {*number*}
7. **cfm mep domain** {*domain-name*} **mpid** {*id*}
8. **cfm mip level** {*level*}
9. **cfm encapsulation**
10. **end**

DETAILED STEPS TO CONFIGURE CFM MEP AND MIP ON A EVC

	Command	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** **Example:** `Router(config)# interface tengigabitethernet 1/0/0`	Enters the interface mode.

Step 4	**service instance** {id} **ethernet** {*evc-name*}
	Example:
	`Router(config-interface)#service instance 41 ethernet 41`
		Configures the service instance and the ethernet virtual connections.

Step 5	**encapsulation** {*encapsulation-type*}
	Example:
	`Router(config-if-srv)#encapsulation dot1q 41`
		Configures the encapsulation type.

Step 6	**bridge-domain {***number***}**
	Example:
	`Router(config-if-srv)#bridge-domain 41`
		Configures the bridge domain values.The default domain number is zero; this is the domain number required when communicating to IEEE bridges that do not support this domain extension.

Step 7	**cfm mep domain** {*domain-name*} **mpid** {id}
	Example:
	`Router(config-if-srv)#cfm mep domain L4 mpid 4001`
		Configures the MEP domain and the ID.

Step 8	**cfm mip level** *{level}*
	Example:
	`PE1(config-if-srv)#cfm mip level 4`
		Automatically creates a MIP in the Ethernet interface and sets the maintenance level number. The acceptable range of maintenance levels is 0-7.

	Command	Purpose

Step 9	**`cfm encapsulation`**
	Example:
	`PE1#(config-if-srv)#cfm encapsulation dot1q 100 second-dot1q 200`
		Configures the CFM encapsulation type.

Step 10	**`end`**
	Example:
	`PE1#(config-if-srv)#exit`
		Exits the service instance interface mode.

SUMMARY STEPS TO CONFIGURE CFM MEP AND MIP ON A SWITCH PORT

1. **enable**
2. **configure terminal**
3. **interface**
4. **switchport**
5. **switchport mode** {*trunk*}
6. **ethernet cfm mep domain** *domain-name* **mpid** *mpid* {**vlan** *vlan-id* | **port**}

   or
7. **ethernet cfm mip level** {0 to 7} {**vlan** *vlan-id* }
8. **end**

DETAILED STEPS TO CONFIGURE CFM MEP AND MIP ON A SWITCHPORT

	Command	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`**
	Example:
	`Router(config)# interface tengigabitethernet 1/0/0`
		Enters the interface mode.

	Command	Purpose
Step 4	**switchport** **Example:** `Router(config-interface)#switchport`	Configures the Layer 3 mode into Layer 2 mode for Layer 2 configuration.

Step 5	**switchport mode** {trunk}
	Example:
	`Router(config-if)#switchport mode trunk`
		Configures a trunking VLAN Layer 2 interface.

Step 6	**ethernet cfm mep domain** *domain-name* **mpid** *mpid* {**vlan** *vlan-id* | **port**} **Example:** `Router(config-if)#ethernet cfm mep domain L4 mpid 1 vlan 41`	Sets a port as internal to a maintenance domain, and defines it as a maintenance endpoint. It sets the device into config-if-ecfm-mep configuration mode, where parameters specific to the MEP can bet set. • *domain-name*: String, maximum length of 43 characters • *mpid*: 1 to 8191 • *vlan-id*: 1 to 4094 • *port*: a port MEP, untagged and valid only for outward direction to configure MEP with no VLAN association.

or

Step 7	**ethernet cfm mip level** {0 to 7} {**vlan** vlan-id } **Example:** `PE1(config-if)#ethernet cfm mip level 4 vlan 10`	Sets a port as internal to a maintenance domain, and defines it as a maintenance intermediate point.

Step 8	**end**
	Example:
	`PE1(config-if)#end`
		Exits the service instance interface mode.

SUMMARY STEPS TO CONFIGURE CFM MEP ON A ROUTED PORT

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet**
4. **no ip address**

5. **no mls qos trust**
6. **ethernet cfm mep domain** *domain-name* **mpid** *mpid* {**vlan** *vlan-id*}
7. **interface gigabitethernet**
8. **encapsulation dot1Q vlan-id**
9. **end**

DETAILED STEPS TO CONFIGURE CFM MEP ON A ROUTED PORT

	Command	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`**
	Example:
	`Router(config)# interface tengigabitethernet 1/0/0`
		Enters the interface mode.

Step 4	**`no ip address`**
	Example:
	`Router(config-interface)# no ip address`
		Removes the configured IP address or disables IP processing.

Step 5	**`no mls qos trust`**
	Example:
	`Router(config-if)#no mls qos trust`
		Configures the multilayer switching (MLS) quality of service (QoS) port trust state and traffic by examining the class of service (CoS) or differentiated services code point (DSCP) value. Use the **no** form of this command to return a port to its untrusted state.

	Command	Purpose
Step 6	**ethernet cfm mep domain** *domain-name* **mpid** *mpid* {**vlan** *vlan-id* } **Example:** `Router(config-if)#ethernet cfm mep domain routed mpid 4001 vlan 4001`	Sets a port as internal to a maintenance domain, and defines it as a maintenance end point. It sets the device into config-if-ecfm-mep configuration mode, where parameters specific to the MEP can be set. • *domain-name*: String, maximum length of 43 characters • *mpid*: 1 to 8191 • *vlan-id*: 1 to 4094
Step 7	**interface gigabitethernet subinterface** **Example:** `Router(config)# interface tengigabitethernet subinterface 1/0/0.1`	Configures the subinterface.
Step 8	**encapsulation dot1Q** ***vlan-id*** **Example:** `PE1(config-if)#encapsulation dot1Q vlan-id 10`	Configures the IEEE 802.1Q encapsulation of traffic on a specified subinterface in a virtual LAN (VLAN) on a routed port. The acceptable range of a VLAN is from 1 to 4094.

Step 9	**end**
	Example:
	`PE1(config-if)#end`
		Exits the service instance interface mode.

Verification

Use the following commands to verify operation.

Command	Purpose
show ethernet cfm maintenance-points local	Displays the local maintenance points.
show ethernet cfm maintenance-points remote	Displays the remote maintenance end points.
show ethernet cfm errors	Displays all the CFM Continuity Check error conditions logged on the device.
show ethernet cfm mpdb	Displays the remote maintenance points.

The following example shows a configuration of MEP in a switchport:

```
ethernet cfm domain L4 level 4
service s41 evc 41 vlan 41
continuity-check
int TenGigabitEthernet2/0/0
switchport
switchport mode trunk
ethernet cfm mep domain L4 mpid 1 vlan 41
```

The following example shows a configuration of MIP in a switchport:

```
ethernet cfm domain L4 level 4
service s41 evc 41 vlan 41
continuity-check
int TenGigabitEthernet2/0/0
switchport
switchport mode trunk
ethernet cfm mip level 4 vlan 41
```

The following example shows a configuration of MEP in a EVC bridge domain:

```
ethernet cfm domain L4 level 4
service s41 evc 41 vlan 41
continuity-check
int TenGigabitEthernet4/0/0
service instance 41 ethernet 41
encapsulation dot1q 41
bridge-domain 41
cfm mep domain L4 mpid 4001
```

The following example shows a configuration of MIP in a EVC bridge domain:

```
ethernet cfm domain L4 level 4
service s41 evc 41 vlan 41
continuity-check
int TenGigabitEthernet4/0/0
service instance 41 ethernet 41
encapsulation dot1q 41
bridge-domain 41
cfm cfm mip level 4
```

The following example shows a configuration of MEP on a routed port:

```
ethernet cfm domain routed level 5
 service s2 evc 2 vlan 2 direction down
  continuity-check
interface GigabitEthernet8/0/0
 no ip address
 no mls qos trust
 ethernet cfm mep domain routed mpid 4001 vlan 4001
interface GigabitEthernet8/0/0.10
 encapsulation dot1Q 10
```

The following example shows CFM configuration over a EVC with cross connect in the global domain configuration mode:

```
ethernet cfm domain L6 level 6
service xconn evc xconn
continuity-check
```

The following example shows CFM configuration over a EVC with cross connect in the interface configuration mode:

```
ethernet cfm domain L6 level 6
 service s100 evc 100
  continuity-check
interface Port-channel10
 no ip address
 service instance 100 ethernet 100
  encapsulation dot1q 200
  xconnect 3.3.3.3 1 encapsulation mpls
  cfm mep domain L6 mpid 602
  cfm mip level 7
 !
```

The following example shows CFM configuration on a L2VFI:

```
Router(config)# l2 vfi vfi2 manual evc2
Router(config-vfi)# vpn id 2
Router(config-vfi)# bridge-domain 2 vlan
Router(config-vfi)# no shut
Router(config-vfi)# neighbor 5.5.5.5 encap mpls
Router(config-vfi-neighbor)# interface vlan 2
Router(config-if)# xconnect vfi vfi2
Router(config-if)# no shut
Router(config-if)# ethernet cfm domain vik-vfi-ofm level 4
Router(config-ecfm)# service vlan-id 2 evc evc2 vlan 2 direction down
Router(config-ecfm-srv)# continuity-check
Router(config-ecfm-srv)# continuity-check interval 10s
```

Support for IEEE 802.1ad

Provider networks handle traffic from a large number of customers. It is important that one customer's traffic is isolated from the other customer's traffic. IEEE 802.1ad implements standard protocols for double tagging of data. The data traffic coming from the customer side are double tagged in the provider network where the inner tag is the customer-tag (C-tag) and the outer tag is the provider-tag (S-tag). The control packets are tunneled by changing the destination MAC address in the provider network.

Cisco 7600 series routers already support VLAN double tagging through a feature called QinQ. 802.1ad is the standardized version of QinQ. It also extends the support for Layer 2 Protocol Tunneling Protocol (L2PT). By offering transparent Layer 2 connectivity, the service provider does not get involved in the customer's Layer 3 network. This makes provisioning and maintenance simple, and reduces the operational cost.

Prerequisites for IEEE 802.1ad

- The ethertype should be programmable per port.

Restrictions for IEEE 802.1ad

Follow these restrictions and guidelines when you configure 802.1ad:

- The **l2protocol forward** command is available only on the main interface of switchports and L3 ports. The command is not available on the subinterfaces. All the subinterfaces on a port inherit the behavior from the main interface. The **l2protocol forward** command is also available on EVC service instance.
- The **l2protocol peer** and **l2protocol drop** commands are not supported.
- The **l2protocol forward** command on a main interface and on EVCs supports only cdp, dtp, vtp, stp, and dot1x.
- You cannot configure Dot1ad if custom ethertype is configured on port.

- 802.1ad is supported on the following port types:

Port	EVC	Switchport	Layer Interfaces
C-UNI	Ethertype 0x8100 C-VLAN BPDU Any EVCs	Ethertype 0x8100 C-VLAN BPDU Trunk or Access	Ethertype 0x8100 C-VLAN BPDU
S-UNI	Ethertype 0x88a8 S-VLAN BPDU (Only Encapsulation default is supported)	Ethertype 0x88a8 S-VLAN BPDU Access only	Not supported
S-NNI	Ethertype 0x88a8 S-VLAN BPDU Any EVC	Ethertype 0x88a8 S-VLAN BPDU Trunk	Ethertype 0x88a8 S-VLAN BPDU Trunk

Information About IEEE 802.1ad

To configure IEEE 802.1ad support, you should understand the following concepts:

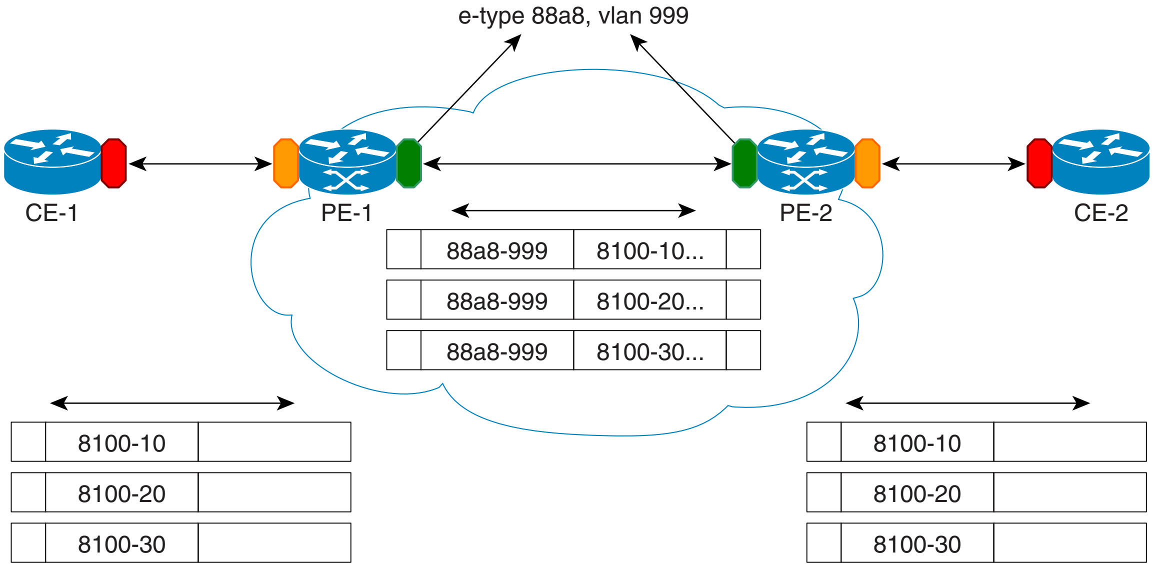
- How Provider Bridges Work
- Guidelines for Handling BPDU
- Interoperability of QinQ and Dot1ad

How Provider Bridges Work

Provider bridges pass the network traffic of many customers, and each customer's traffic flow must be isolated from one another. For the Layer 2 protocols within customer domains to function properly, geographically separated customer sites must appear to be connected through a LAN, and the provider network must be transparent.

The IEEE has reserved 33 Layer 2 MAC addresses for customer devices operating Layer 2 protocols. If a provider bridge uses these standard MAC addresses for its Layer 2 protocols, the customers' and service provider's Layer 2 traffic will be mixed together. Provider bridges solve this traffic-mixing issue by providing Layer 2 protocol data unit (PDU) tunneling for customers using a provider bridge (S-bridge) component and a provider edge bridge (C-bridge) component. Figure 2-1 shows the topology.

Figure 2-1 Layer 2 PDU Tunneling



S-Bridge Component

The S-bridge component is capable of inserting or removing a service provider VLAN (S-VLAN) for all traffic on a particular port. IEEE 802.1ad adds a new tag called a Service tag (S-tag) to all the ingress frames from a customer to the service provider.

The VLAN in the S-tag is used for forwarding the traffic in the service provider network. Different customers use different S-VLANs, which results in each customer's traffic being isolated. In the S-tag, provider bridges use an Ethertype value that is different from the standard 802.1Q Ethertype value, and do not understand the standard Ethertype. This difference makes customer traffic tagged with the standard Ethertype appear as untagged in the provider network so customer traffic is tunneled in the port VLAN of the provider port. The 802.1ad service provider user network interfaces (S-UNIs) and network to network interfaces (NNIs) implement the S-bridge component.

For example, a VLAN tag has a VLAN ID of 1, the C-tag Ethertype value is 8100 0001, the S-tag Ethertype value is 88A8 0001, and the class of service (CoS) is zero.

```
C-tag S-tag
-------------------------------------------------------
-----------------------------------------------
0x8100 | Priority bits | CFI | C-VLAN-ID 0x88A8 | Priority bits | 0 | S-VLAN-ID
-------------------------------------------------------
-----------------------------------------------
```

C-Bridge Component

All the C-VLANs entering on a UNI port in an S-bridge component are provided the same service (marked with the same S-VLAN). Although, C-VLAN components are not supported, a customer may want to tag a particular C-VLAN packet separately to differentiate between services. Provider bridges allow C-VLAN packet tagging with a provider edge bridge, called the C-bridge component of the provider bridge. C-bridge components are C-VLAN aware and can insert or remove a C-VLAN 802.1Q tag. The C-bridge UNI port is capable of identifying the customer 802.1Q tag and inserting or removing an S-tag on the packet on a per service instance or C-VLAN basis. A C-VLAN tagged service instance allows service instance selection and identification by C-VLAN. The 802.1ad customer user network interfaces (C-UNIs) implement the C-component.

MAC Addresses for Layer 2 Protocols

Customers' Layer 2 PDUs received by a provider bridge are not forwarded, so Layer 2 protocols running in customer sites do not know the complete network topology. By using a different set of addresses for the Layer 2 protocols running in provider bridges, IEEE 802.1ad causes customers' Layer 2 PDUs entering the provider bridge to appear as unknown multicast traffic and forwards it on customer ports (on the same S-VLAN). Customers' Layer 2 protocols can then run transparently.

Table 2-4 shows the Layer 2 MAC addresses reserved for the C-VLAN component.

Table 2-4 Reserved Layer 2 MAC Addresses for a C-VLAN Component

Assignment	Value
Bridge Group Address	01-80-c2-00-00-00
IEEE Std 802.3 Full Duplex PAUSE operation	01-80-c2-00-00-01
IEEE Std. 802.3 Slow_Protocols_Multicast address	01-80-c2-00-00-02
IEEE Std. 802.1X PAE address	01-80-c2-00-00-03
Reserved for future standardization - media access method-specific	01-80-c2-00-00-04
Reserved for future standardization - media access method- specific	01-80-c2-00-00-05
Reserved for future standardization	01-80-c2-00-00-06
Reserved for future standardization	01-80-c2-00-00-07
Provider Bridge Group Address	01-80-c2-00-00-08
Reserved for future standardization	01-80-c2-00-00-09
Reserved for future standardization	01-80-c2-00-00-0a
Reserved for future standardization	01-80-c2-00-00-0b
Reserved for future standardization	01-80-c2-00-00-0c
Provider Bridge GVRP Address	01-80-c2-00-00-0d
IEEE Std. 802.1AB Link Layer Discovery Protocol multicast address	01-80-c2-00-00-0e
Reserved for future standardization	01-80-c2-00-00-0f

Table 2-5 shows the Layer 2 MAC addresses reserved for an S-VLAN component. These addresses are a subset of the C-VLAN component addresses, and the C-bridge does not forward the provider's bridge protocol data units (BPDUs) to a customer network.

Table 2-5 Reserved Layer 2 MAC Addresses for an S-VLAN Component

Assignment	Value
IEEE Std 802.3 Full Duplex PAUSE operation	01-80-c2-00-00-01
IEEE Std. 802.3 Slow_Protocols_Multicast address	01-80-c2-00-00-02
IEEE Std. 802.1X PAE address	01-80-c2-00-00-03
Reserved for future standardization - media access method specific	01-80-c2-00-00-04

Table 2-5 Reserved Layer 2 MAC Addresses for an S-VLAN Component

Assignment	Value
Reserved for future standardization - media access method specific	01-80-c2-00-00-05
Reserved for future standardization	01-80-c2-00-00-06
Reserved for future standardization	01-80-c2-00-00-07
Provider Bridge Group Address	01-80-c2-00-00-08
Reserved for future standardization	01-80-c2-00-00-09
Reserved for future standardization	01-80-c2-00-00-0a

Guidelines for Handling BPDU

The general BPDU guidelines are listed here:

UNI-C Ports

The guidelines pertaining to UNI-C ports are:

- VLAN-aware L2 protocols can be peered, tunneled, or dropped.
- Port L2 protocols can either be peered or dropped. They cannot be tunneled.

Table 2-6 shows the Layer 2 PDU destination MAC addresses for customer-facing C-bridge UNI ports, and how frames are processed.

Table 2-6 Layer 2 PDU Destination MAC Addresses for Customer-Facing C-Bridge UNI Ports

Assignment	Protocol	Significance on C-UNI Port	Default Action
01-80-C2-00-00-00	Bridge Group Address (End-to-End BPDUs)	BPDU	Peer
01-80-C2-00-00-01	802.3X Pause Protocol	BPDU	Drop
01-80-C2-00-00-02	Slow Protocol address: 802.3ad LACP, 802.3ah OAM, CDP Pagp, VTP, DTP, UDLD	BPDU	Peer
01-80-C2-00-00-03	802.1X	BPDU	May peer
01-80-C2-00-00-04	Reserved for future media access method	None	Drop
01-80-C2-00-00-05	Reserved for future media access method	None	Drop
01-80-C2-00-00-06	Reserved for future bridge use	None	Drop
01-80-C2-00-00-07	Reserved for future bridge use	None	Drop
01-80-C2-00-00-08	Provider STP (BPDU)	None	Drop
01-80-C2-00-00-09	Reserved for future bridge use	None	Drop
01-80-C2-00-00-0A	Reserved for future bridge use	None	Drop
01-80-C2-00-000-0B	Reserved for future S-bridge purpose	None	Drop

Table 2-6 Layer 2 PDU Destination MAC Addresses for Customer-Facing C-Bridge UNI Ports

Assignment	Protocol	Significance on C-UNI Port	Default Action
01-80-C2-00-00-0C	Reserved for future S-bridge purpose	None	Drop
01-80-C2-00-00-0D	Provider Bridge GVRP address	None	Drop
01-80-C2-00-00-0E	802.1ab-LLDP	BPDU	May peer
01-80-C2-00-00-0F	Reserved for future C-bridge or Q-bridge use	None	Drop
01-80-C2-00-00-10	All bridge addresses	Read Data	Snoop if implemented. Else, discard
01-80-C2-00-00-20	GMRP	Data/BPDU	May peer
01-80-C2-00-00-21	GVRP	Data/BPDU	May peer
01-80-C2-00-00-22 – 2F	Other GARP addresses	Data/BPDU	May peer
01-00-0C-CC-CC-CC	Cisco's CDP DTP VTP PagP UDLD (End-to-End)	BPDU	Peer
01-00-0C-CC-CC-CD	Cisco's PVST(End-to-End)	BPDU	May peer

UNI-S Ports

The guidelines pertaining to UNI-S ports are:

- Packets with C-Bridge addresses (00 - 0F) that are not part of S-Bridge addresses (01 - 0A) are treated as data packet (tunneled).
- VLAN-aware L2 protocols cannot be peered because the port is not C-VLAN aware. They can only be tunneled or dropped.
- Port L2 protocols can be peered, tunneled, or dropped.

Table 2-7 shows the Layer 2 PDU destination MAC addresses for customer-facing S-bridge UNI ports, and how frames are processed.

Table 2-7 Layer 2 PDU Destination MAC Addresses for Customer-Facing S-Bridge UNI Ports

Assignment	Protocol	Significance on S-UNI Port	Default Action
01-80-C2-00-00-00	Bridge Group Address (BPDUs)	Data	Data
01-80-C2-00-00-01	802.3X Pause Protocol	BPDU	Drop
01-80-C2-00-00-02	Slow Protocol address: 802.3ad LACP, 802.3ah	BPDU	Peer
01-80-C2-00-00-03	802.1X	BPDU	Peer
01-80-C2-00-00-04	Reserved for future media access method	BPDU	Drop
01-80-C2-00-00-05	Reserved for future media access method	BPDU	Drop
01-80-C2-00-00-06	Reserved for future bridge use	BPDU	Drop
01-80-C2-00-00-07	Reserved for future bridge use	BPDU	Drop

Table 2-7 Layer 2 PDU Destination MAC Addresses for Customer-Facing S-Bridge UNI Ports

Assignment	Protocol	Significance on S-UNI Port	Default Action
01-80-C2-00-00-08	Provider STP (BPDU)	BPDU	Drop (peer on NNI)
01-80-C2-00-00-09	Reserved for future bridge use	BPDU	Drop
01-80-C2-00-00-0A	Reserved for future bridge use	BPDU	Drop
01-80-C2-00-00-0B	Reserved for future bridge use	Data if not implemented	Drop
01-80-C2-00-00-0C	Reserved for future bridge use	Data if not implemented	Treat as data until implemented
01-80-C2-00-00-0D	Reserved for future GVRP address	Data if not implemented	Treat as data until implemented
01-80-C2-00-00-0E	802.1ab-LLDP	BPDU	May peer
01-80-C2-00-00-0F	Reserved for future C-bridge or Q-bridge use	Data	Data
01-80-C2-00-00-10	All bridge addresses	Data	Data
01-80-C2-00-00-20	GMRP	Data	Data
01-80-C2-00-00-21	GVRP	Data	Data
01-80-C2-00-00-22 – 2F	Other GARP addresses	Data	Data
01-00-0C-CC-CC-CC	Cisco’s CDP DTP VTP PagP UDLD	Data	Data
01-00-0C-CC-CC-CD	Cisco’s PVST	Data	Data

NNI Ports

The Dot1add NNI ports behave in the same way as the customer facing S-bridge ports, with the following exceptions:

- On NNI ports, frames received with DA 01-80-C2-00-00-08 contain STP BPDU. The frames are received and transmitted. On S-UNI ports, any such frames that are received are dropped, and none are sent.
- On NNI ports, frames received with DA 01-80-C2-00-00-02 include CDP Pagp, VTP, DTP, and UDLD protocols.

7600 Action Table

Table 2-8 lists the actions performed on a packet when the packet is received with a specified destination MAC address.

Table 2-8 7600 Action Table

MAC Address	Protocol	C-UNI Action	S-UNI Action	NNI Action
01-80-C2-00-00-00	Bridge Group Address (BPDUs)	Peer	Data	Data
01-80-C2-00-00-01	802.3X Pause Protocol	Drop	Drop	Drop

Table 2-8 7600 Action Table

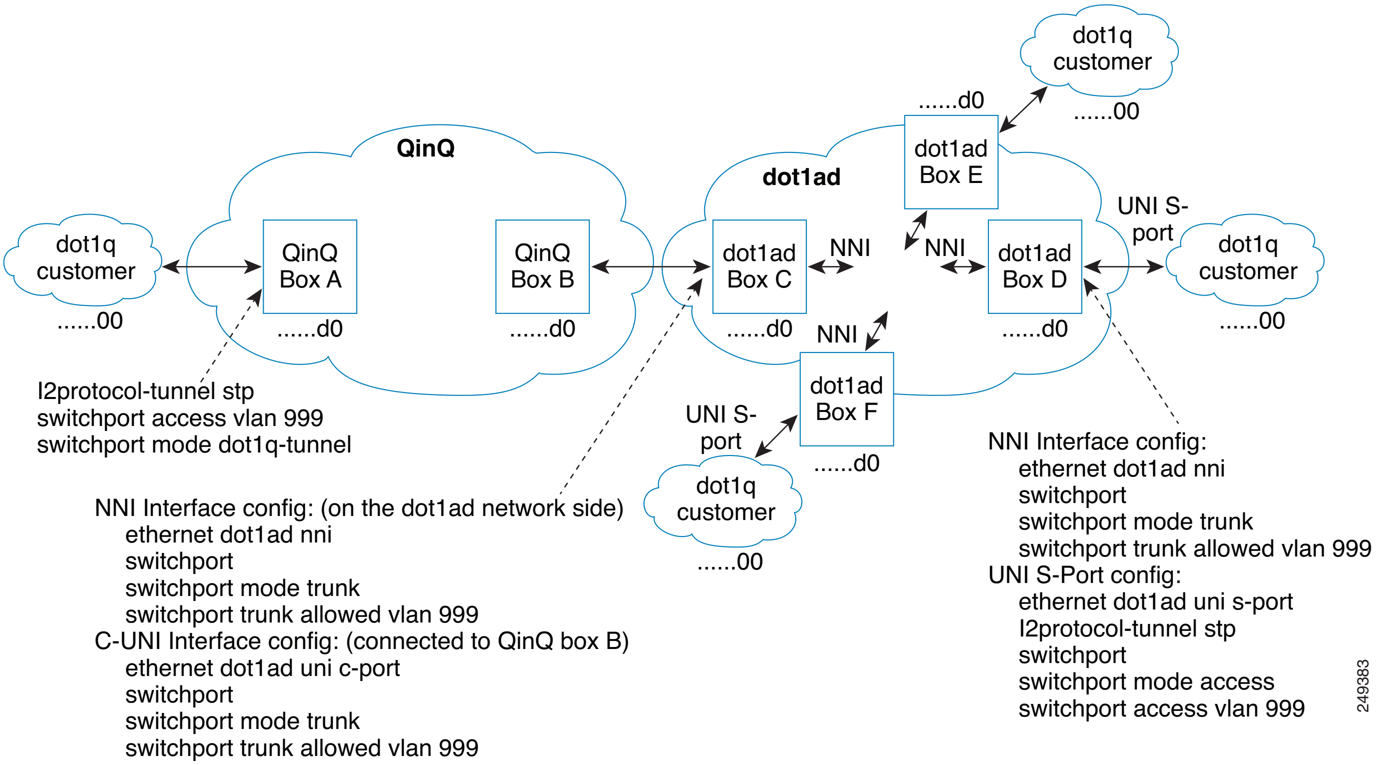
MAC Address	Protocol	C-UNI Action	S-UNI Action	NNI Action
01-80-C2-00-00-02	Slow Protocol address: 802.3ad LACP, 802.3ah	Peer	Peer	Peer
01-80-C2-00-00-03	802.1X	May peer	May peer	May peer
01-80-C2-00-00-04	Reserved	Drop	Drop	Drop
01-80-C2-00-00-05	Reserved	Drop	Drop	Drop
01-80-C2-00-00-06	Reserved	Drop	Drop	Drop
01-80-C2-00-00-07	Reserved	Drop	Drop	Drop
01-80-C2-00-00-08	Provider STP (BPDU)	Drop	Drop	Peer
01-80-C2-00-00-09	Reserved for future bridge use	Drop	Drop	Drop
01-80-C2-00-00-0A	Reserved for future bridge use	Drop	Drop	Drop
01-80-C2-00-00-0B	Reserved for future bridge use	Drop	Data	Data
01-80-C2-00-00-0C	Reserved for future bridge use	Drop	Data	Data
01-80-C2-00-00-0D	Reserved for future GVRP address	Drop	Data	Data
01-80-C2-00-00-0E	802.1ab-LLDP	May peer	Data	Data
01-80-C2-00-00-0F	Reserved for future C-bridge or Q-bridge use	Drop	Data	Data
01-80-C2-00-00-10	All bridge addresses	Snoop if implemented. Else drop	Data	Data
01-80-C2-00-00-20	GMRP	May peer	Data	Data
01-80-C2-00-00-21	GVRP	May peer	Data	Data
01-80-C2-00-00-22 – 2F	Other GARP addresses	May peer	Data	Data
01-00-0C-CC-CC-CC	Cisco's CDP DTP VTP PagP UDLD	Peer	Data	Data
01-00-0C-CC-CC-CD	Cisco's PVST	May peer	Data	Data

Interoperability of QinQ and Dot1ad

The interoperability of QinQ and Dot1ad network enables the exchange of data frames between the networks. The 802.1Q network outer tag VLANs are mapped to the provider S-VLANs of the 802.1ad network.

Figure 2-2 illustrates the interoperability of a Dot1ad network and a QinQ network.

Figure 2-2 Interoperability of Dot1ad Network and a QinQ Network



How to Configure IEEE 802.1ad

This section contains the information about following procedures:

- Configuring a Switchport
- Configuring a Layer 2 Protocol Forward
- Configuring a Switchport for Translating QinQ to 802.1ad
- Configuring a Switchport (L2PT)
- Configuring a Customer-Facing UNI-C Port with EVC
- Configuring a Customer-Facing UNI-C Port and Switchport on NNI with EVC
- Configuring a Customer-Facing UNI-S Port with EVC
- Configuring a Layer 3 Termination
- Displaying a Dot1ad Configuration

Configuring a Switchport

A switchport can be configured as a UNI-C port, UNI-S port, or NNI port.

UNI-C Port

A UNI-C port can be configured as either a trunk port or an access port. Perform the following tasks to configure a UNI-C port as an access port for 802.1ad.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ethernet dot1ad {nni | uni {c-port | s-port}}**
5. **switchport**
6. **switchport mode {access | trunk}**
7. **switchport access vlan** *vlan-id*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `router> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `router# interface gigabitethernet 2/1`	Configures an interface.

Step 4	`ethernet dot1ad {nni | uni {c-port | s-port}}`
	Example:
	`router(config-if)# ethernet dot1ad uni c-port`
		Configures a dot1ad NNI port or UNI port. In this example, it is a UNI-C port.

Step 5	**switchport** **Example:** `router(config-if)# switchport`	Put the interface into Layer 2 mode.

Step 6	**switchport mode {access | trunk}**
	Example:
	`router(config-if)# switchport mode access`
		Sets the interface type. In this example, it is Access.

	Command or Action	Purpose
Step 7	`switchport access vlan` *vlan-id* **Example:** `router(config-if)# switchport access 1000`	Sets the VLAN when an interface is in access mode. In this example, the VLAN is set to 1000.

Step 8	`end`
	Example:
	`router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Perform the following tasks to configure a UNI-C port as a trunk port for 802.1ad.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ethernet dot1ad {nni | uni {c-port | s-port}}**
5. **switchport**
6. **switchport mode {access | trunk}**
7. **switchport trunk allowed vlan** *vlan-list*
8. **end**

DETAILED STEPS

	Command or Action	Purpose

Step 1	`enable`
	Example:
	`router> enable`
		Enables privileged EXEC mode.

Step 2	`configure terminal`
	Example:
	`router# configure terminal`
		Enters global configuration mode.

Step 3	`interface` *type number* **Example:** `router# interface gigabitethernet 2/1`	Configures an interface.

	Command or Action	Purpose

Step 4	**ethernet dot1ad** {**nni** | **uni** {**c-port** | **s-port**}}
	Example:
	`router(config-if)# ethernet dot1ad uni c-port`
		Configures a dot1ad NNI port or UNI port. In this example, it is a UNI-C port.

Step 5	**switchport** **Example:** `router(config-if)# switchport`	Put the interface into Layer 2 mode.

Step 6	**switchport mode {access | trunk}**
	Example:
	`router(config-if)# switchport mode trunk`
		Sets the interface type. In this example, it is Trunk.

Step 7	**switchport trunk allowed vlan** *vlan-list* **Example:** `router(config-if)# switchport trunk allowed vlan 1000, 2000`	Sets the list of allowed VLANs that transmit traffic from this interface in tagged format when in trunking mode.

Step 8	**end**
	Example:
	`router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

UNI-S Port

On a UNI-S port, all the customer VLANs that enter are provided with the same service. The port allows only access configuration. In this mode, the customer's port is configured as a trunk port. Therefore, the traffic entering the UNI-S port is tagged traffic.

Perform the following tasks to configure a UNI-S port as an access port for 802.1ad.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **switchport**
5. **switchport mode {access | trunk}**
6. **ethernet dot1ad {nni | uni {c-port | s-port}}**
7. **switchport access vlan** *vlan-id*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `router> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `router# interface gigabitethernet 2/1`	Configures an interface.
Step 4	**switchport** **Example:** `router(config-if)# switchport`	Put the interface into Layer 2 mode.

Step 5	**switchport mode {access | trunk}**
	Example:
	`router(config-if)# switchport mode access`
		Sets the interface type. In this example, it is Access.

Step 6	ethernet dot1ad {nni | uni {c-port | s-port}}
	Example:
	`router(config-if)# ethernet dot1ad uni s-port`
		Configures a dot1ad NNI port or UNI port. In this example, it is a UNI-S port.

Step 7	**switchport access vlan** *vlan-id* **Example:** `router(config-if)# switchport access 999`	Sets the VLAN when an interface is in access mode. In this example, the VLAN is set to 999.

Step 8	**end**
	Example:
	`router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

NNI Port

NNI port allows only trunk configuration. On an NNI port, the frames received on all the allowed VLANs are bridged to the respective internal VLANs.

Perform the following tasks to configure an NNI port as a trunk port for 802.1ad.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **switchport**
5. **switchport mode {access | trunk}**
6. **ethernet dot1ad {nni | uni {c-port | s-port}}**
7. **switchport trunk allowed vlan** *vlan-list*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `router> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `router# interface gigabitethernet 2/1`	Configures an interface.
Step 4	**switchport** **Example:** `router(config-if)# switchport`	Put the interface into Layer 2 mode.

Step 5	**switchport mode {access | trunk}**
	Example:
	`router(config-if)# switchport mode trunk`
		Sets the interface type. In this example, it is Trunk.

Step 6	`ethernet dot1ad {nni | uni {c-port | s-port}}`
	Example:
	`router(config-if)# ethernet dot1ad nni`
		Configures a dot1ad NNI port or UNI port. In this example, it is an NNI.

	Command or Action	Purpose
Step 7	**switchport trunk allowed vlan** *vlan-list* **Example:** `router(config-if)# switchport trunk allowed vlan 999`	Sets the list of allowed VLANs that transmit traffic from this interface in tagged format when in trunking mode.

Step 8	**end**
	Example:
	`router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Examples

The following example shows how to configure a UNI-C port as an access port. In this example, all the frames that are received are bridged to one internal VLAN 1000. The transmitted frames do not have the access VLAN Dot1q tag.

```
router# configure terminal
router(config)#interface gig2/1
router(config-if)#ethernet dot1ad uni c-port
router(config-if)#switchport
router(config-if)#switchport mode access
router(config-if)#switchport access vlan 1000
```

The following example shows how to configure a UNI-C port as a trunk port. In this example, all the frames that are received on all allowed VLANs (1000 and 2000) are bridged to the respective internal VLANs. The transmitted frames have the respective internal VLAN Dot1q tag.

```
router# configure terminal
router(config)# interface gig2/1
router(config-if)# ethernet dot1ad uni c-port
router(config-if)# switchport
router(config-if)# switchport mode trunk
router(config-if)# switchport access vlan 1000, 2000
```

The following example shows how to configure a UNI-S port. In this example, all the frames that are received are bridged to one internal VLAN (999). The transmitted frames do not have the access VLAN Dot1q tag.

```
router# configure terminal
router(config)#interface gig2/1
router(config-if)#switchport
router(config-if)#switchport mode access
router(config-if)#ethernet dot1ad uni s-port
router(config-if)#switchport access vlan 999
```

The following example shows how to configure an NNI port. Only trunk configuration is allowed on an NNI port. In this example, all the frames that are received on all the allowed VLANs (999) are bridged to the respective internal VLANs. The transmitted frames have the respective internal VLAN Dot1q tag.

```
router# configure terminal
router(config)#interface gig2/1
router(config-if)#switchport
router(config-if)#switchport mode trunk
router(config-if)#ethernet dot1ad nni
router(config-if)#switchport trunk allowed vlan 999
```

The following example shows how to configure Dot1ad on an SVI:

```
router# configure terminal
router(config)#interface gig2/1
router(config-if)#ethernet dot1ad nni
router(config-if)#switchport
router(config-if)#switchport mode trunk
router(config-if)#switchport trunk allowed vlan 999
router(config)#interface vlan 999
router(config-if)#ip address 1.2.3.4 255.255.0.0
```

Configuring a Layer 2 Protocol Forward

Perform the following tasks to configure the Layer 2 protocol forward:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **switchport access valn** *vlan-id*
5. **ethernet dot1ad {nni | uni {c-port | s-port}}**
6. **l2protocol [forward] [***protocol***]**
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `router> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `router(config)# interface gigabitethernet 3/0`	Configures an interface.
Step 4	**switchport access vlan** *vlan-id* **Example:** `router(config)# switchport access vlan 500`	Sets the VLAN when an interface is in access mode.

	Command or Action	Purpose

Step 5	`ethernet dot1ad {nni | uni {c-port | s-port}}`
	Example:
	`router(config-if)# ethernet dot1ad uni s-port`
		Configures a dot1ad NNI port or UNI port. In this example, it is a UNI S-port.

Step 6	**`l2 protocol [forward]`** `[`*protocol*`]`
	Example:
	`router(config-if)# l2 protocol forward vtp`
		Processes or forwards the Layer 2 BPDUs. In this example, all the BPDUs are forwarded except VTP PDUs.

Step 7	**`end`**
	Example:
	`router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Examples

The following example shows how to configure a Layer 2 protocol forward:

```
router# configure terminal
router(config)#interface gig3/0
router(config-if)#switchport access vlan 500
router(config-if)#ethernet dot1ad uni s-port
router(config-if)#l2protocol forward vtp
```

Configuring a Switchport for Translating QinQ to 802.1ad

Translating a QinQ port to 802.1ad involves configuring the port connecting to QinQ port and NNI port.

Perform the following tasks to configure a port connecting to the QinQ port.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **switchport mode {access | trunk}**
5. **switchport trunk allowed vlan** *vlan-list*
6. **end**

DETAILED STEPS

	Command or Action	Purpose

Step 1	`enable`
	Example:
	`router> enable`
		Enables privileged EXEC mode.

Step 2	`configure terminal`
	Example:
	`router# configure terminal`
		Enters global configuration mode.

Step 3	**interface** *type number* **Example:** `router# interface gigabitethernet 1/1`	Configures an interface.

Step 4	**switchport mode {access | trunk}**
	Example:
	`router(config-if)# switchport mode trunk`
		Sets the interface type. In this example, it is Trunk.

Step 5	**switchport trunk allowed vlan** *vlan-list* **Example:** `router(config-if)# switchport trunk allowed vlan 1000`	Sets the list of allowed VLANs that transmit traffic from this interface in tagged format when in trunking mode.

Step 6	`end`
	Example:
	`router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Perform the following tasks to configure an NNI port.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ethernet dot1ad {nni | uni {c-port | s-port}}**
5. **switchport**
6. **switchport mode {access | trunk}**

7. **switchport trunk allowed vlan** *vlan-list*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `router> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `router# interface gigabitethernet 4/1`	Configures an interface.

Step 4	ethernet dot1ad {nni | uni {c-port | s-port}}
	Example:
	`router(config-if)# ethernet dot1ad nni`
		Configures a dot1ad NNI port or UNI port. In this example, it is an NNI.

Step 5	**switchport** **Example:** `router(config-if)# switchport`	Put the interface into Layer 2 mode.

Step 6	**switchport mode {access | trunk}**
	Example:
	`router(config-if)# switchport mode trunk`
		Sets the interface type. In this example, it is Trunk.

Step 7	**switchport trunk allowed vlan** *vlan-list* **Example:** `router(config-if)# switchport trunk allowed vlan 999-1199`	Sets the list of allowed VLANs that transmit traffic from this interface in tagged format when in trunking mode.

Step 8	**end**
	Example:
	`router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Examples

The following example shows how to translate a QinQ port to 802.1ad. In this example, the peer router to gig1/1 multiplexes various customer VLANs into VLAN 1000.

```
router# configure terminal
router(config)#interface gig1/1
router(config-if)#switchport mode trunk
router(config-if)#switchport trunk allowed vlan 1000

router# configure terminal
router(config)#interface gig4/0
router(config-if)#ethernet dot1ad nni
router(config-if)#switchport
router(config-if)#switchport mode trunk
router(config-if)#switchport trunk allowed vlan 1000,1199
```

Configuring a Switchport (L2PT)

Configuring the switchport for L2PT is required to tunnel the STP packets from a customer on the dot1ad network to a customer on the QinQ network.

Perform the following tasks to configure the port connecting to the customer.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **switchport**
5. **ethernet dot1ad {nni | uni {c-port | s-port}}**
6. **no l2 protocol [peer | forward]** [*protocol*]
7. **l2protocol-tunnel [cdp | stp | vtp]**
8. **switchport mode {access | trunk}**
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `router> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**`interface`** *type number* **Example:** `router(config)# interface gigabitethernet 2/1`	Configures an interface.

Step 4	**`switchport`**
	Example:
	`router(config-if)# switchport`
		Put the interface into Layer 2 mode.

Step 5	`ethernet dot1ad {nni	uni {c-port	s-port}}` **Example:** `router(config-if)# ethernet dot1ad uni s-port`	Configures a dot1ad NNI port or UNI port. In this example, it is a UNI S-port.
Step 6	`no l2 protocol [peer	forward] [`*`protocol`*`]` **Example:** `router(config-if)# no l2 protocol forward`	Disables L2 protocol forwarding.
Step 7	**`l2protocol-tunnel [cdp	stp	vtp]`** **Example:** `router(config-if)# l2protocol-tunnel stp`	Enables protocol tunneling for STP.
Step 8	**`switchport mode {access	trunk}`** **Example:** `router(config-if)# switchport mode trunk`	Sets the interface type. In this example, it is Trunk.

Step 9	**`end`**
	Example:
	`router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Perform the following tasks to configure an NNI port.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **switchport**
5. **ethernet dot1ad {nni | uni {c-port | s-port}}**

6. **switchport mode {access | trunk}**
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `router> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `router(config)# interface gigabitethernet 2/1`	Configures an interface.
Step 4	**switchport** **Example:** `router(config-if)# switchport`	Put the interface into Layer 2 mode.

Step 5	ethernet dot1ad {nni | uni {c-port | s-port}}
	Example:
	`router(config-if)# ethernet dot1ad nni`
		Configures a dot1ad NNI or UNI port. In this example, it is an NNI.

Step 6	**switchport mode {access | trunk}**
	Example:
	`router(config-if)# switchport mode trunk`
		Sets the interface type. In this example, it is Trunk.

Step 7	**end**
	Example:
	`router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Examples

The following example shows how to tunnel the STP packets from a customer on the Dot1ad network to a customer on a QinQ network:

```
router# configure terminal
```

```
router(config)#interface gig1/0
router(config-if)#switchport
router(config-if)#ethernet dot1ad uni s-port
router(config-if)#no l2protocol forward
router(config-if)#l2protocol-tunnel stp
router(config-if)#switchport mode access

router# configure terminal
router(config)#interface gig4/0
router(config-if)#switchport
router(config-if)#ethernet dot1ad nni
router(config-if)#switchport mode trunk
```

Configuring a Customer-Facing UNI-C Port with EVC

Perform the following tasks to configure a UNI-C port.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ethernet dot1ad {nni | uni {c-port | s-port}}**
5. **service instance** *id service-type*
6. **encapsulation dot1q** *vlan*-**id second-dot1q {any |** *vlan*-**id} [native]**
7. **bridge-domain** *vlan-id*
8. **service instance** *id service-type*
9. **encapsulation dot1q** *vlan*-**id second-dot1q {any |** *vlan*-**id} [native]**
10. **bridge-domain** *vlan-id*
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `router> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `router(config)# interface gigabitethernet 2/1`	Configures an interface.

Step 4	ethernet dot1ad {nni | uni {c-port | s-port}}
	Example:
	`router(config-if)# ethernet dot1ad uni c-port`
		Configures a dot1ad NNI port or UNI port. In this example, it is a UNI C port.

Step 5	service instance *id service-type* **Example:** `router(config-if)# service instance 1 ethernet`	Configures an Ethernet service instance. In this example, the service instance is 1.
Step 6	**encapsulation dot1q** *vlan-id* **second-dot1q {any |** *vlan-id***} [native]** **Example:** `router(config-if)# encapsulation dot1q 1-100`	Enables IEEE 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.
Step 7	**bridge-domain** *vlan-id* **Example:** `router(config-if)# bridge-domain 1000`	Binds a service instance or a MAC tunnel to a bridge domain.
Step 8	service instance *id service-type* **Example:** `router(config-if)# service instance 2 ethernet`	Configures an Ethernet service instance. In this example, the service instance is 2.
Step 9	**encapsulation dot1q** *vlan-id* **second-dot1q {any |** *vlan-id***} [native]** **Example:** `router(config-if)# encapsulation dot1q 102-4094`	Enables IEEE 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.
Step 10	**bridge-domain** *vlan-id* **Example:** `router(config-if)# bridge-domain 500`	Binds a service instance or a MAC tunnel to a bridge domain.

Step 11	**end**
	Example:
	`router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Perform the following tasks to configure an NNI port.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ethernet dot1ad {nni | uni {c-port | s-port}}**
5. **service instance** *id service-type*
6. **encapsulation dot1q** *vlan*-**id second-dot1q {any |** *vlan*-**id} [native]**
7. **rewrite ingress tag pop 1 symmetric**
8. **bridge-domain** *vlan-id*
9. **service instance** *id service-type*
10. **encapsulation dot1q** *vlan*-**id second-dot1q {any |** *vlan*-**id} [native]**
11. **rewrite ingress tag pop 1 symmetric**
12. **bridge-domain** *vlan-id*
13. **end**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`enable`**
	Example:
	`router> enable`
		Enables privileged EXEC mode.

Step 2	**`configure terminal`**
	Example:
	`router# configure terminal`
		Enters global configuration mode.

Step 3	**`interface`** *`type number`*
	Example:
	`router(config)# interface gigabitethernet 2/1`
		Configures an interface.

Step 4	`ethernet dot1ad {nni | uni {c-port | s-port}}`
	Example:
	`router(config-if)# ethernet dot1ad uni c-port`
		Configures a dot1ad NNI port or UNI port. In this example, it is a UNI C port.

	Command or Action	Purpose

Step 5	`service instance` *`id service-type`*
	Example:
	`router(config-if)# service instance 1 ethernet`
		Configures an Ethernet service instance. In this example, the service instance is 1.

Step 6	**`encapsulation dot1q`** *vlan-id* **`second-dot1q {any |`** *vlan-id***`} [native]`**
	Example:
	`router(config-if)# encapsulation dot1q 1000 second-dot1q 1-100`
		Enables IEEE 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.

Step 7	**`rewrite ingress tag pop 1 symmetric`**
	Example:
	`router(config-if)# rewrite ingress tag pop 1 symmetric`
		Specifies the encapsulation adjustment that is to be performed on the frame ingress to the service instance.

Step 8	**`bridge-domain`** *`vlan-id`*
	Example:
	`router(config-if)# bridge-domain 1000`
		Binds a service instance or a MAC tunnel to a bridge domain.

Step 9	`service instance` *`id service-type`*
	Example:
	`router(config-if)# service instance 2 ethernet`
		Configures an Ethernet service instance. In this example, the service instance is 2.

Step 10	**`encapsulation dot1q`** *`vlan-id`* **`second-dot1q {any |`** *vlan-id***`} [native]`**
	Example:
	`router(config-if)# encapsulation dot1q 500 second-dot1q 102-4904`
		Enables IEEE 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.

Step 11	**`rewrite ingress tag pop 1 symmetric`**
	Example:
	`router(config-if)# rewrite ingress tag pop 1 symmetric`
		Specifies the encapsulation adjustment that is to be performed on the frame ingress to the service instance.

	Command or Action	Purpose
Step 12	**bridge-domain** *vlan-id* **Example:** `router(config-if)# bridge-domain 500`	Binds a service instance or a MAC tunnel to a bridge domain.

Step 13	**end**
	Example:
	`router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Examples

The following example shows how to configure a customer-facing UNI port. In this example, a dot1q frame coming on VLAN 50 matches service instance 1, and on the ingress port, the rewrite command pushes the 1000 outer-vlan.

```
router# configure terminal
router(config)#interface gig1/1
router(config-if)#ethernet dot1ad uni c-port
router(config-if)#service instance 1 ethernet
router(config-if)#encapsulation dot1q 1-100
router(config-if)#bridge-domain 1000
router(config-if)#service instance 2 ethernet
router(config-if)#encapsulation dot1q 102-4904
router(config-if)#bridge-domain 500

router# configure terminal
router(config)#interface gig4/1
router(config-if)#ethernet dot1ad nni
router(config-if)#service instance 1 ethernet
router(config-if)#encapsulation dot1q 1000 second dot1q 1-100
router(config-if)#rewrite ingress tag pop 1 symmetric
router(config-if)#bridge-domain 1000
router(config-if)#service instance 2ethernet
router(config-if)#encapsulation dot1q 500 second dot1q 102-4904
router(config-if)#rewrite ingress tag pop 1 symmetric
router(config-if)#bridge-domain 500
```

Configuring a Customer-Facing UNI-C Port and Switchport on NNI with EVC

Perform the following tasks to configure a UNI-C port.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ethernet dot1ad {nni | uni {c-port | s-port}}**
5. **service instance** *id service-type*
6. **encapsulation dot1q** *vlan*-**id second-dot1q {any |** *vlan*-**id} [native]**
7. **bridge-domain** *vlan-id*

8. **service instance** *id service-type*
9. **encapsulation dot1q** *vlan*-**id second-dot1q {any |** *vlan*-**id} [native]**
10. **bridge-domain** *vlan-id*
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `router> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `router(config)# interface gigabitethernet 2/1`	Configures an interface.

Step 4	ethernet dot1ad {nni | uni {c-port | s-port}}
	Example:
	`router(config-if)# ethernet dot1ad uni c-port`
		Configures a dot1ad NNI port or UNI port. In this example, it is a UNI C port.

Step 5	service instance *id service-type* **Example:** `router(config-if)# service instance 1 ethernet`	Configures an Ethernet service instance. In this example, the service instance is 1.
Step 6	**encapsulation dot1q** *vlan-id* **second-dot1q {any |** *vlan-id*} **[native]** **Example:** `router(config-if)# encapsulation dot1q 1-100`	Enables IEEE 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.
Step 7	**bridge-domain** *vlan-id* **Example:** `router(config-if)# bridge-domain 1000`	Binds a service instance or a MAC tunnel to a bridge domain.

	Command or Action	Purpose
Step 8	`service instance` *id service-type* **Example:** `router(config-if)# service instance 2 ethernet`	Configures an Ethernet service instance. In this example, the service instance is 2.

Step 9	**`encapsulation dot1q`** *vlan-id* **`second-dot1q {any |`** *vlan-id***`} [native]`**
	Example:
	`router(config-if)# encapsulation dot1q 102-4094`
		Enables IEEE 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.

Step 10	**`bridge-domain`** *vlan-id* **Example:** `router(config-if)# bridge-domain 500`	Binds a service instance or a MAC tunnel to a bridge domain.

Step 11	**`end`**
	Example:
	`router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Perform the following tasks to configure an NNI port.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ethernet dot1ad {nni | uni {c-port | s-port}}**
5. **switchport**
6. **switchport mode {access | trunk}**
7. **switchport trunk allowed vlan** *vlan-list*
8. **end**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`enable`**
	Example:
	`router> enable`
		Enables privileged EXEC mode.

Step 2	**`configure terminal`**
	Example:
	`router# configure terminal`
		Enters global configuration mode.

Step 3	**`interface`** *type number* **Example:** `router# interface gigabitethernet 4/1`	Configures an interface.

Step 4	`ethernet dot1ad {nni | uni {c-port | s-port}}`
	Example:
	`router(config-if)# ethernet dot1ad nni`
		Configures a dot1ad NNI port or UNI port. In this example, it is an NNI.

Step 5	**`switchport`**
	Example:
	`router(config-if)# switchport`
		Put the interface into Layer 2 mode.

Step 6	**`switchport mode {access | trunk}`**
	Example:
	`router(config-if)# switchport mode trunk`
		Sets the interface type. In this example, it is Trunk.

Step 7	**`switchport trunk allowed vlan`** *vlan-list* **Example:** `router(config-if)# switchport trunk allowed vlan 1000-500`	Sets the list of allowed VLANs that transmit traffic from this interface in tagged format when in trunking mode.

Step 8	**`end`**
	Example:
	`router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Examples

The following example shows how to configure a customer-facing UNI-C port and switchport on NNI with EVC:

```
router# configure terminal
router(config)#interface gig1/1
router(config-if)#ethernet dot1ad uni c-port
router(config-if)#service instance 1 ethernet
router(config-if)#encapsulation dot1q 1-100
router(config-if)#bridge-domain 1000
router(config-if)#service instance 2 ethernet
router(config-if)#encapsulation dot1q 102-4904
router(config-if)#bridge-domain 500

router# configure terminal
router(config)#interface gig4/0
router(config-if)#switchport
router(config-if)#ethernet dot1ad uni
router(config-if)#switchport mode trunk
router(config-if)#switchport allowed vlan 1000,500
```

Configuring a Customer-Facing UNI-S Port with EVC

Perform the following tasks to configure a UNI-S port.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id service-type*
5. **ethernet dot1ad {nni | uni {c-port | s-port}}**
6. **encapsulation default**
7. **bridge-domain** *vlan-id*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `router> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `router(config)# interface gigabitethernet 2/1`	Configures an interface.
Step 4	`service instance` *id service-type* **Example:** `router(config-if)# service instance 1 ethernet`	Configures an Ethernet service instance. In this example, the service instance is 1.

Step 5	`ethernet dot1ad {nni | uni {c-port | s-port}}`
	Example:
	`router(config-if)# ethernet dot1ad uni s-port`
		Configures a dot1ad NNI port or UNI port. In this example, it is a UNI-S port.

Step 6	**encapsulation default** **Example:** `router(config-if)# encapsulation default`	Configures the default service instance on a port. Anything that does not meet the criteria of other service instances on the same physical interface falls into this service instance.
Step 7	**bridge-domain** *vlan-id* **Example:** `router(config-if)# bridge-domain 1000`	Binds a service instance or a MAC tunnel to a bridge domain.

Step 8	**end**
	Example:
	`router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Perform the following tasks to configure an NNI port.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *id service-type*
5. **ethernet dot1ad {nni | uni {c-port | s-port}}**
6. **encapsulation dot1q** *vlan*-**id second-dot1q {any |** *vlan*-**id} [native]**
7. **rewrite ingress tag pop 1 symmetric**

8. **bridge-domain** *vlan-id*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `router> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `router(config)# interface gigabitethernet 2/1`	Configures an interface.
Step 4	service instance *id service-type* **Example:** `router(config-if)# service instance 1 ethernet`	Configures an Ethernet service instance. In this example, the service instance is 1.

Step 5	ethernet dot1ad {nni | uni {c-port | s-port}}
	Example:
	`router(config-if)# ethernet dot1ad uni c-port`
		Configures a dot1ad NNI or UNI port. In this example, it is a UNI C port.

Step 6	**encapsulation dot1q** *vlan-id* **second-dot1q {any |** *vlan-id***} [native]** **Example:** `router(config-if)# encapsulation dot1q 1000 second-dot1q 1-100`	Enables IEEE 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.
Step 7	**rewrite ingress tag pop 1 symmetric** **Example:** `router(config-if)# rewrite ingress tag pop 1 symmetric`	Specifies the encapsulation adjustment that is to be performed on the frame ingress to the service instance.

	Command or Action	Purpose
Step 8	**bridge-domain** *vlan-id* **Example:** `router(config-if)# bridge-domain 1000`	Binds a service instance or a MAC tunnel to a bridge domain.

Step 9	**end**
	Example:
	`router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Examples

The following example shows how to configure an NNI port:

```
router# configure terminal
router(config)#interface gig1/1
router(config-if)#service instance 1 ethernet
router(config-if)#ethernet dot1ad nni
router(config-if)#encapsulation dot1q 1000
router(config-if)#rewrite ingress tag pop 1 symmetric
router(config-if)#bridge-domain 1000
```

Configuring a Layer 3 Termination

Perform the following tasks to configure a Layer 3 termination.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ethernet dot1ad {nni | uni {c-port | s-port}}**
5. **interface** *type number*
6. **encapsulation dot1q** *vlan*-**id second-dot1q {any |** *vlan*-**id} [native]**
7. **ip address** *ip-address mask*
8. **end**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`enable`**
	Example:
	`router> enable`
		Enables privileged EXEC mode.

Step 2	**`configure terminal`**
	Example:
	`router# configure terminal`
		Enters global configuration mode.

Step 3	**`interface`** *type number* **Example:** `router(config)# interface gigabitethernet 3/0`	Configures an interface.

Step 4	`ethernet dot1ad {nni | uni {c-port | s-port}}`
	Example:
	`router(config-if)# ethernet dot1ad nni`
		Configures a dot1ad NNI or UNI port. In this example, it is an NNI port.

Step 5	**`interface`** *type number* **Example:** `router(config)# interface gigabitethernet 3/0/.1`	Configures an interface.

Step 6	**`encapsulation dot1q`** *vlan-id* **`second-dot1q {any |`** *vlan-id***`} [native]`**
	Example:
	`router(config-if)# encapsulation dot1q 10 second-dot1q 10`
		Enables IEEE 802.1Q encapsulation of traffic on a specified subinterface in a VLAN.

Step 7	**`ip address`**
	Example:
	`router(config-if)# ip address 1.2.3.4 255.255.0.0`
		Sets a primary or secondary IP address for an interface.

Step 8	**`end`**
	Example:
	`router(config-if)# end`
		Returns the CLI to privileged EXEC mode.

Examples

The following example shows how to configure a Layer 3 termination. Note that Layer 3 is supported only on trunk interfaces.

```
router# configure terminal
router(config)#interface gig3/0
router(config-if)#ethernet dot1ad nni
router(config)#interface gig3/0/0.1
router(config-if)#encapsulation dot1q 10 second dot1q 10
router(config-if)#ip address 1.2.3.4 255.255.0.0
```

The following example shows how to configure a Layer 3 termination on an SVI:

```
router# configure terminal
router(config)#interface gig4/1
router(config-if)#ethernet dot1ad nni
router(config-if)#service instance 1 ethernet
router(config-if)#encapsulation dot1q 200 second dot1q 300
router(config-if)#rewrite ingress tag pop 2 symmetric
router(config-if)#bridge-domain 50
router(config-if)#service instance 2 ethernet
router(config-if)#encapsulation dot1q 300
router(config-if)#rewrite ingress tag pop 1 symmetric
router(config-if)#bridge-domain 60

router(config)#interface vlan 50
router(config-if)#ip address 2.3.4.5 255.255.0.0
router(config)#interface vlan 60
router(config-if)#ip address 3.4.5.6 255.255.0.0
```

Displaying a Dot1ad Configuration

You can display a Dot1ad configuration using the **show ethernet dot1ad** command. This command displays the Dot1ad configuration for all interfaces. To display the configuration on a particular interface, use the **show ethernet dot1ad interface** command.

The following example shows how to display a Dot1ad configuration on all interfaces:

```
router# show ethernet dot1ad
Interface: GigabitEthernet4/0/1
DOT1AD C-Bridge Port
L2protocol pass cdp stp vtp dtp pagp dot1x lacp

Interface: GigabitEthernet4/0/2
DOT1AD C-Bridge Port
L2protocol pass cdp stp vtp dtp pagp dot1x lacp
```

Troubleshooting Dot1ad

The following section describes how to troubleshoot Dot1ad.

Note The show commands in these examples should be run from a line card console.

- How do I verify the Dot1ad configuration on a switchport on an X40G card?

  Run the following command to verify the Dot1ad configuration:

```
XYZ-PE1-dfc1#show platform npc switchport interface gi 1/2
 [GigabitEthernet1/2]
```

```
        status [valid, -, applied, enabled]
        src_index [0x1]
        rpcb [0x178BB9C4]
        xlif_id [4097]
        xlif_handle [type:[3] hwidb:[0x20E97F08] if_number:[1121]]
        ft_bits [0x2]
        ing_ctrl_ft_bits [0x2]
        egr_ctrl_ft_bits [0x2]
        port vlan [1]
        mode ingress [NORMAL] egress [NORMAL]
        dot1q_tunnel [No]
        native tagging [No]
        PVLAN isolated or community [No] promiscuous [No]
        ingress vlan-translation [No] BPDU [No]
        egress  vlan-translation [No] BPDU [No]
        dot1ad [Yes] <<<<<<<<<<<<
        ethertype [0x88A8] <<<<<<<<<<<
        Ingress Stat ID: 778698
        Egress Stat ID: 778700
        VLAN List:
        1
        num of vlans [1]
        XYZ-PE1-dfc1#
```

- How do I verify the Dot1ad configuration on the ports with EVCs on an X40G card?

  Run the following command to verify the Dot1ad configuration:

```
XYZ-PE1-dfc1#show platform npc xlif interface gi 1/2 efp 1
EFP XLIF(GigabitEthernet1/2, efp1)[np0] = 4136

 Ingress XLIF table fields

 Feature common enable:  0x1
 Feature enable:         0x1
 Feature bits:           0x1
 Control common bits:    0x0
 Control feature bits:   0x0
 Control rewrite opcode: 0x0
 Reserved 1:             0x0
 Match cond              0x1
 Entry valid:            0x1
 Dbus VLAN:              30
 QoS policy ID:          0
 ACL ID:                 0
 Statistics ID:          450976
 Inner rewrite VLAN:     0
 Outer rewrite VLAN:     0
 QoS flow ID:            0
 Feature data: 00000000 40000000 AAA80000 E0000829
 EFP admin down state  0x0
----- Bridge data ------
 layer2_acl_index:           0x00000000
 evc_feat_data.ip_src_guard     : 0x0
 evc_feat_data.mst_evc          : 0x1
 evc_feat_data.layer2_acl       : 0x0
 EVC - Mac Security:     0x0
 evc_feat_data.sacl       : 0x0
 evc_feat_data.layer2_acl_statid: 0
 PDT: 0xAAA8
 ipsg_label: 0
 block_data: 0x0
 block_l2bpdu: 0x0
 split_h: 0x0
 imp_ltl: 0x0829
 EFP dot1ad port type 0x3       <<<<<<<<
 EFP CDP forward 0x1  <<<<<<<<
 EFP DTP forward 0x0
 EFP VTP forward 0x0
 EFP STP forward 0x0
 EFP DOT1X forward 0x0

 Egress XLIF table fields

 Feature common enable:  0x1
 Feature enable:         0x1
 Feature bits:           0x01
 Control common bits:    0x00
 Control feature bits:   0x00
 Control rewrite opcode: 0x00
 Port:                   0x1
 Match cond              0x1
 Entry valid:            0x1
 Dbus VLAN:              30
 QoS policy ID:          0
 ACL ID:                 0
 Statistics ID:          450980
 Inner rewrite VLAN:     0
 Outer rewrite VLAN:     0
 QoS flow ID:            0
 IP Session en :         0
 Multicast  en :         0
 Feature data 0          0x00000000
 Intf etype:             0x00008064
 Post Filter Opcode      0x00000008
 Pre Filter Opcode       0x00000000
 Pre Tag Outer           0x00000000
 Pre Tag Inner           0x00000000
 Post Filter Vlan high   0x00000064
 Post Filter Vlan low    0x00000064
 Post Filter Vlan outer  0x00000000
 EVC - MST:              0x1
 EVC etype               0x8100
 CFM MEP Level           0x00000008
 CFM MIP Level           0x00000008
 CFM disable             0x0
 MIP filtering           0x0
 block_data:            0x0
 block_l2bpdu:           0x0
 sacl:                   0x0
 sacl index:             0x0000
 sacl statid:            0x00000
XYZ-PE1-dfc1#
XYZ-PE1-dfc1#
```

- How do I verify the L2protocol forwarding on a regular L3 switchports?

  Run the following command to verify the L2protocol forwarding:

  ```
  XYZ-PE1-dfc1# show platform npc xlif 0 port_sram 1

  ........................

   dot1ad port type:      0x0002  <<<<<<<<<
   l2proto cdp fwd:       0x0001  <<<<<<<<<
   l2proto dtp fwd:       0x0000
   l2proto vtp fwd:       0x0000
   l2proto stp fwd:       0x0000
  ```

```
l2proto dot1x fwd:     0x0000

         ..............................................
```

- How do I verify the Dot1ad configuration on ES20 cards?
  - For switchports, run the following command:

    ```
    XYZ-PE1-dfc1#show platform hardware dot1ad l2protocfg port <port-num>
    ```

  - For EVCs, run the following command:

    ```
    XYZ-PE1-dfc1# show platform soft efp-client interface gi x/0/y efp-id l2protocfg
    ```

  - To display the default values, run the following commands:

    ```
    XYZ-PE1-dfc1#show platform hardware dot1ad l2protocfg defaults ?
      <0-2>  0=c-uni, 1=s-uni, 2=nni

    XYZ-PE1-dfc1#show platform hardware dot1ad l2protocfg defaults 0 ?
      <0-2>  0=L3, 1=BD, 2=XCON

    XYZ-PE1-dfc1#show platform hardware dot1ad l2protocfg defaults 0 2
    Raw Data :000FFF77 FFFCFF51
     L2 Proto Configs :
      Protocol         IEEE       CISCO
     ------------------------------------
        CDP       :    FRWD       FRWD
        VTP       :    FRWD       FRWD
        DTP       :    FRWD       FRWD
       Others     :    PEER       PEER

     802.1d protocols : 01:80:C2:00:00:XX

     XX | Config     XX | Config     XX | Config     XX | Config
     -----------     -----------     -----------     -----------
     00 : PEER       01 : DROP       02 : PEER       03 : PEER
     04 : FRWD       05 : FRWD       06 : FRWD       07 : FRWD
     08 : DROP       09 : FRWD       0A : FRWD       0B : FRWD
     0C : FRWD       0D : FRWD       0E : FRWD       0F : FRWD

     All Bridge (0180C2000010)= FRWD
     Group = PEER
     PVST = FRWD
    ```

Configuring Layer 2 Features

This section provides ES20 line card-specific information about configuring the Layer 2 interworking features on the Cisco 7600 series router. It includes the following topics:

- Cross-Bundling, page 2-69
- Configuring Flexible QinQ Mapping and Service Awareness on 7600-ESM-2X10GE and 7600-ESM-20X1GE, page 2-110
- Configuring Flexible Service Mapping Based on CoS and Ethertype, page 2-119
- Configuring the Backup Interface for Flexible UNI, page 2-150Cross-Bundling, page 2-69
- Multichassis Support for Link Aggregation Control Protocol, page 2-74
- Configuring the Backup Interface for Flexible UNI, page 2-150
- Troubleshooting, page 2-156

- Configuring Broadcast Storm Control on Switchports and Ports with Ethernet Virtual Connections, page 2-160
- Configuring Asymmetric Carrier-Delay, page 2-165
- Configuring MST on EVC Bridge Domain, page 2-167

Cross-Bundling

Follow these restrictions and guidelines during cross-bundling various linecards:

- ES20 and ES+ cross-bundling is not supported.
- Any LAN card, and ES20/ES+ cross-bundling is not supported.

Configuring EVC EtherChannel and LACP over EVC Port Channel

An Ethernet link bundle or port channel is an aggregation of up to eight physical Ethernet links to form a single logical link for L2/L3 forwarding. Bundled Ethernet ports are used to increase the capacity of the logical link and provide high availability and redundancy. The EVC EtherChannel feature provides support for EtherChannels on Ethernet Virtual Connection Services (EVCS) service instances.

For more information on EtherChannels, and how to configure EtherChannels on Layer 2 or Layer 3 LAN ports, refer:
http://www.cisco.com/en/US/docs/routers/7600/ios/12.2SXF/configuration/guide/channel.html

The EVC EtherChannel feature supports MPBE, local connect, and cross connect service types. IEEE 802.3ad/Link Aggregation Control Protocol (LACP) provides an association of port channels. This feature supports service instances over bundled Ethernet links.

Ethernet flow points (EFPs) are configured on a port channel. The traffic, carried by the EFPs, is load balanced across member links. EFPs on a port channel are grouped and each group is associated with one member link. Ingress traffic for a single EVC can arrive on any member of the bundle. All egress traffic for an EFP uses only one of the member links. Load balancing is achieved by grouping EFPs and assigning them to a member link. In default load balancing, the user has no control over how the EFPs are grouped together, and sometimes the EFP grouping is not ideal. As a workaround, use manual load balancing to control the EFP grouping.

The scalability for a link-bundling EVC is 16k per chassis. Port Channel EVC scalability for ES20 cards is dependent on the same factors as EVCs configured on physical interfaces, with the number of member links and their distribution across PXFs as an additional parameter. EVC port channel QoS leverages EVC QoS infrastructure

Restrictions and Usage Guidelines

When configuring EVC EtherChannel, follow these restrictions and usage guidelines:

- All member links of the port channel are on Cisco 7600-ES20-GE line cards.
- Bridge-domain, cross connect, connect EVCs, switchports, and IP subinterfaces are allowed over the port-channel interface and the main interface.

Note For a port with a switchport, you can use the service instance ethernet command to create a service instance to support OAM requirements and not data traffic.

- The EFP limit decreases by the number of member links on the NP. Egs: If there are 4 members within the same NP, the EVC limit on the NP decreases to 2000 that ie (8000/4).
- If you configure a physical port as part of a channel group, you cannot configure EVCs on that physical port.
- SPAN is not supported on an EVC source and destination.
- You cannot configure switchport on a physical port that is part of an EVC port channel.
- You can apply QoS policies on EVCs on a port channel, except that ingress microflow policing is not supported. For more information on configuring QoS with EVCs, see "Configuring QoS on the Cisco 7600 Series Ethernet Services 20G Line Card".
- For HQoS and flat policies applied on an EVCS at the egress, the total guaranteed bandwidth of all policies on the EVCS belonging to the port channel cannot exceed the link rate (1 Gbps for 7600-ESM-20X1GE and 10 Gbps for 7600-ESM-2X10GE).
- You cannot use the **bandwidth percent** command on EVC port channels with Cisco 7600-ES20-GE line cards.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. ***interface port channel***
4. [**no**] **`ip address`**
5. [**no**] **service instance** *id* {**Ethernet** *service-name*}
6. **encapsulation dot1q** *vlan-id*
7. **rewrite ingress tag** {**push {dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {dot1q *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} [**symmetric**]
8. [no] **bridge-domain** *bridge-id*
9. **channel-group 5 mode** *on* | *off*
   or
   channel-group 5 mode *active* | *passive*

Note The channel-group command options on/off are applicable when configuring port channel over EVC and the options active/passive are applicable when configuring port channel over EVC with LACP.

DETAILED STEPS

	Command	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)# **interface port channel** *number* **Example:** Router(config)# interface port channel 1	Creates the port channel interface.
Step 4	Router(config-if)# **ip address** *ip_address mask* **Example:** Router(config-if)# ip address 10.0.0.11 255.255.255.0	Assigns an IP address and subnet mask to the EtherChannel.
Step 5	[**no**] **service instance** id {**Ethernet** [service-name]} **Example:** Router(config-if)# service instance 101 ethernet	Creates a service instance (an instance of an EVC) on an interface and sets the device into the config-if-srv submode.
Step 6	**encapsulation dot1q** *vlan-id* **Example:** Router(config-if-srv)# encapsulation dot1q 13	Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.
Step 7	**rewrite ingress tag** {**push {dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {dot1q *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} [**symmetric**] **Example:** Router(config-if-srv)# rewrite ingress tag push dot1q 20	Specifies the tag manipulation that is to be performed on the frame ingress to the service instance.

	Command	Purpose
Step 8	[no] **bridge-domain** *bridge-id* **Example:** `Router(config-subif)# bridge domain 12`	Binds the service instance to a bridge domain instance where *bridge-id* is the identifier for the bridge domain instance.

Step 9	**channel-group 5 mode** *on* |*off*
	Example:
	`Router(config-if)# channel-group 5 mode on`
	OR
	channel-group 5 mode *active* |*passive*
	Example:
	`Router(config-if)# channel-group 5 mode active`
		Enables EVC port channel.
		Enables LACP on the configured EVC port channel.

Examples

In this example, a single port channel interface is created with three possible member links from slots 1 and 2.

```
Router(config)# interface port channel5
Router(config-if)# no ip address
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 350
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
Router(config-subif)# bridge-domain 350
 !
Router(config-if)# service instance 2 ethernet
Router(config-if-srv)# encapsulation dot1q 400
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
Router(config-subif)# bridge-domain 350

Router(config-if)# service instance 3 ethernet
Router(config-if-srv)# encapsulation dot1q 500
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
Router(config-subif)# bridge-domain 370
 !
Router(config)# interface port channel5.1
Router(config-if-srv)# encapsulation dot1Q 500 second-dot1q 300
Router(config-if)# ip address 60.0.0.1 255.0.0.0
!
Router(config)# interface GigabitEthernet1/0/0
Router(config-if)# channel-group 5 mode on

Router(config)# interface GigabitEthernet1/0/1
Router(config-if)# channel-group 5 mode on

Router(config)# interface GigabitEthernet2/0/1
Router(config-if)# channel-group 5 mode on
```

The next example shows scalable Eompls and EVC connect sample configuration.

```
Router#enable
Router#configure terminal
Router(config)#interface GigabitEthernet 3/0/0
Router(config-if)#service instance 10 ethernet
Router(config-srv)#encapsulation dot1q 20
Router(config-if-srv)#rewrite ingress tag pop 1 sym
Router(config-if-srv)#exit
Router(config-if)#exit
Router(config)#interface GigabitEthernet 3/0/1
Router(config-if)#service instance 12 ethernet
Router(config-srv)#encapsulation dot1q 30
Router(config-if-srv)#rewrite ingress tag pop 1 sym
Router(config-if-srv)#exit
Router(config-if)#exit
Router(config)#connect TEST GigabitEthernet 3/0/0 10 GigabitEthernet 3/0/1 12
Router#sh connection all

ID   Name            Segment 1              Segment 2                  State
================================================================================
57   TEST            Gi3/0/0:10             Gi3/0/1:12                 UP
```

Here is a typical QoS configuration.

```
Router(config)# interface port channel10
Router(config-if)# no ip address
Router(config-if)# mls qos trust cos
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 11
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
Router(config-if)# service-policy input x
Router(config-if)# service-policy output y
Router(config-subif)# bridge-domain 1500
```

Here is the configuration for LACP over a configured EVC port channel, on an interface:

```
Router(config-if)# channel-group 5 mode ?
active Enable LACP unconditionally
auto Enable PAgP only if a PAgP device is detected
desirable Enable PAgP unconditionally
on Enable EtherChannel only
passive Enable LACP only if a LACP device is detected
Router(config-if)#channel-group 5 mode active
Router(config-if)#channel-group 5 mode passive
```

Here are LACP EVC Port channel for Fast Switchover (1:1 redundancy) sample configuration outputs.

Port channel Configuration

```
Router(config-if)#interface port channel102
Router(config-if)#mtu 9216
Router(config-if)#no ip address
Router(config-if)#lacp fast-switchover
Router(config-if)#lacp max-bundle 1
Router(config-if)#service instance 50 ethernet
Router(config-if)# encapsulation dot1q 50
Router(config-if)# rewrite ingress tag pop 1 symmetric
Router(config-if)# service-policy output lacp-parent
Router(config-if)# bridge-domain 50
```

Member links configuration

```
Router(config-if)#interface GigabitEthernet3/0/12
Router(config-if)#mtu 9216
Router(config-if)#no ip address
Router(config-if)#no mls qos trust
Router(config-if)#lacp rate fast
Router(config-if)#channel-protocol lacp
Router(config-if)#channel-group 102 mode active
```

Verification

Use the following commands to verify operation.

Command	Purpose
Use the following commands to verify EVC configuration	
`Router# show ethernet service evc [id evc-id | interface interface-id] [detail]`	Displays information pertaining to a specific EVC if an EVC ID is specified, or pertaining to all EVCs on an interface if an interface is specified. The **detailed** option provides additional information on the EVC.
`Router# show ethernet service instance [id instance-id interface interface-id | interface interface-id] [detail]`	Displays information about one or more service instances: If a service instance ID and interface are specified, only data pertaining to that particular service instance is displayed. If only an interface ID is specified, displays data for all service instances on the given interface.
`Router# show ethernet service interface [interface-id] [detail]`	Displays information in the Port Data Block (PDB).
Use the following commands to verify LACP over EVC	
`Router#show etherchannel 15 port channel`	Displays details for port channel 15. This command is common to EVC port channel, switchport port channel and L3 Port channel. The CLI is run at the RP.

Multichassis Support for Link Aggregation Control Protocol

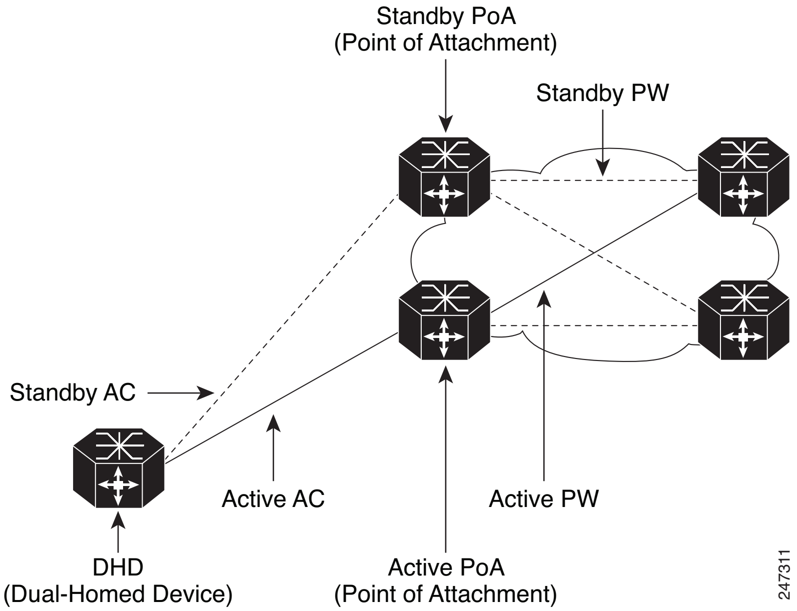
Configured at the edge of a provider's network, Multichassis Link Aggregation Control Protocol (MLACP) features performs the following actions:

- Uses dual-homed devices (DHD) to provide network redundancy between two or more service provider networks.
- Allows the LACP state machine and protocol to operate in a dual- homed mode.

Each switch is a point of attachments (PoA), where one PoA is active, and the other is a standby, and the active PoA executes the multichassis link aggregation group with a DHD. A virtual LACP peer on the PoA is created giving the impression that a DHD is connected to one node.

Figure 2-3 shows the placement of PoAs and DHDs in an MLACP configuration.Figure 2-2

Figure 2-3 Placement of PoAs and DHDs in an MLACP Implementation



The status of the PoAs during traffic relay are:

- The two PoAs form a redundancy group, and only one of the PoAs is active at any given time.
- Only two PoAs form a redundancy group; however, you can configure a maximum of 50 redundancy groups connecting to other DHDs.
- Active links exist only between a DHD and active PoAs. None of the links between the DHD and the standby PoA relay traffic other than Bridge Protocol Data Unit (BPDU)s.
- The state of the etherchannel interface on a standby PoA is UP.

A switchover from an active PoA to a standby PoA occurs when there is a failure on the:

- Uplink port on the DHD
- DHD’s uplink
- Downlink port on an active PoA
- Active PoA node
- Active PoA uplinks
- Cable failure

The default switchover mechanism uses dynamic port priority changes on the port channel and member *link(s)* to provide revertive mode and nonrevertive mode options. The default operation in a *multi-chassis LACP* is revertive.

Bruteforce is a switchover mechanism where the member link is in a err-disable state after a switchover. To recover the port channel and enable the member link on a new standby PoA, use the **err disable recovery cause mlacp-minlink** command in the global configuration mode.

Use the **lacp max-bundle** command in the following modes:

- PoA based: Command is executed on the PoA.

- Shared based: Command is executed on the PoA and DHD.
- DHD based: Command is executed on the DHD.

Use the **lacp max-bundle** command on all the PoAs to operate in the PoA control and shared control modes. The max-bundle value argument should not be less than the total number of links in the Link Aggregation Group (LAG) that are connected to the PoA. Each PoA may be connected to the DHD with a different number of links for the LAG and, therefore, configured with a different value for the max-bundle value argument.

Note **The lacp failover brute-force** command cannot be used with a nonrevertive configuration.

Requirements and Restrictions

Follow these requirements and restrictions when configuring the MLACP feature in a ES20 line card:

- Supported only on ES20 and ES40 line cards, all member links on a port-channel should be on same type of line card.
- Cisco IOS Release 12.2(33)SRE supports service instances only on an MLACP port-channel.
- A PoA may be active for one port-channel, and standby for a different port-channel.
- The maximum number of port-channels supported on a PoA is 256.
- In any LACP configuration, ensure that the numerical value of the system-priority of the virtual LACP instance on the PoAs is lower (higher priority) than that on the DHD for all control variants.
- It is not recommended to configure different max bundle configurations on a PoA. For example, if DHD 1 to PoA has 4 links, PoA2 should also have 4 links.
- Links can be successfully aggregated based on the following constraints:
  - Links should be from the same line card type.
  - QoS should be validated.
  - Port-channel hashing should be identical for two links.
  - Flowcontrol should match.
- When Cisco 7600 routers are used to form a redundancy group within a PoA, the member links should adhere to the constraints listed in the previous paragraph. These constraints are not validated across PoAs and you should ensure that configuration between the two PoAs are identical.
- Ensure that the etherchannel usage configuration is identical on the two PoAs.
- The maximum bundle value on a PoA is 8.
- A maximum of two PoAs in a redundancy group and 50 redundancy groups per node are supported.
- Multiple Spanning Tree (MST) on an EVC is not supported on MLACP etherchannel ports.
- Reverse Layer 2 Gateway Protocol (RL2GP) with MLACP is not supported.
- DHD port-channel cannot use Spanning Tree Protocol (STP) or Resilient Ethernet Protocol (REP) or Reverse Layer 2 Gateway Protocol (RL2GP) as a redundancy option. DHD port-channel disables the STP enabled by default.
- Subinterfaces on port-channels are not supported.
- You can configure the **channel-group** command as **active** and configuring the **channel-group** command as **passive** is not supported.

- As the l**acp direct-loadswap** command is not applicable on a PoA, member links on a PoA are not protected with links on the same PoA.
- We do not recommend you to have different bundle configurations on a DHD. For example, if DHD 1 to PoA1 has four links, DHD 1 to PoA 2 should also have the same number of links.
- Use the **port-channel min-link** command to configure each PoA with the minimum number of links. This maintains the LAG in an active state.
- The **lacp max-bundle** command must be used on all the PoAs to operate in PoA control and shared control modes. The value of the *max-bundle* should not be less than the total number of interfaces in the LAG that are connected to the PoA.
- If you use the **lacp failover** command with *brute force*, then after the switchover, the port-channel member link moves to a errdisabled state.By default, the interval is 300 seconds (tunable range is 30 seconds to 300 seconds).To recover the port-channel, use the **errdisable recovery cause mlacp-minlink** command. EVC with connect as forwarding function is not supported.
- **The lacp failover non-revertive** and **lacp failover brute-force** commands are mutually exclusive within the same port-channel.
- Connectivity Fault Management configuration on an MLACP port-channel is not permissible.
- For best switchover perforamance, configure LACP fast-switchover in PoAs and DHDs.
- You cannot use MLACP port-channel for IP forwarding.
- You cannot configure REP on a MLACP port-channel.
- Use the **errdisable recovery cause mlacp-minlink** command to auto-recover the port-channel after timer expiration.
- The core interfaces in a VPLS core should be a ES20 or ES40 line card.

The recommended configuration sequence is:

- Configure interchassis group and MLACP commands.
- Configure MLACP interchassis group and other port-channel commands.
- Add member links.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **redundancy**
4. **interchassis group** *{number}*
5. **monitor peer** {*BFD*}
6. **member IP** {*IP address*}
7. **mlacp node-id** {*number*}
8. ***mlacp system-mac*** *{IP address}*
9. **mlacp system-priority** *priority*
10. **backbone interface** *any interface*
11. **exit**
12. **interface port-channel** {*port-channel number*}

13. **lacp max-bundle** {*max-bundle value*}
14. **lacp failover** { non-revertivel brute force }
15. **mlacp interchassis group** {*group-id*}
16. **backbone int member**
17. **exit**

DETAILED STEPS

	Command	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	**`redundancy`**
	Example:
	`Router(config)# redundancy`
		Enters redundancy configuration mode.

Step 4	**`interchassis group`** {*`number`*}
	Example:
	`Router(configure-red)# interchassis group 400`
		Configures an interchassis group within the redundancy configuration mode and assigns a group number.

Step 5	**`monitor peer`** {*`BFD`*}
	Example:
	`Router(configure-red)#`
		Configures the BFD option to monitor the state of the peer. The default option is route-watch.

Step 6	**`member ip`** {*`IP address`*}
	Example:
	`Router(configure-red)# member ip 172.3.3.3`
		Configures the IP address of the mlacp peer member group.

Step 7	**`mlacp node-id {`*`number`*`}`**
	Example:
	`Router(config-r-ic)# mlacp node-id 5`
		Defines the node ID to be used in the LACP port-id field. Valid value range is 0 - 7, and the value should be different from the peer values.

	Command	Purpose

Step 8	**mlacp system-mac *{address}***
	 Example:
	 `Router(config-r-ic)# mlacp aaaa.aaaa.aaab`
		Defines and advertises the system MAC address value to the MLACP members of the redundancy group.

Step 9	**mlacp system-priority *priority*** **Example:** `Router(config-r-ic)# mlacp system-priority 100`	Defines the system priority advertised to the other MLACP members of the redundancy group. System priority values are from 1 to 65535, the default value being 32768. The assigned values should be lower than the DHD.
Step 10	backbone interface *any interface* **Example:** `Router(config-r-ic)#  backbone interface GigabitEthernet2/3`	Defines the backbone interface for the MLACP configuration.
Step 11	exit **Example:** `Router(config-r-ic)#exit`	Exits the redundancy mode.

Step 12	**interface port-channel** {*port-channel number*}
	 Example:
	 `Router# interface Port-channel1`
		To identify the PoA uplink failure, configure the port-channel interface or any physical interface.

Step 13	**lacp max-bundl**e {*max-bundle value*}
	 Example:
	 `Router (config-int)# lacp max-bundle 4`
		Configures the max-bundle links that are connected to the PoA. The value of the *max-bundle links* argument should not be less than the total number of links in the LAG that are connected to the PoA.

Step 14	**lacp failover** { *non-revertive| brute force*}
	 Example:
	 `P19_C7609-S(config-if)#lacp failover ?`
	 ` brute-force    Brute force interface failover`
	 `non-revertive  Non revertive interface failover`
		Sets the MLACP switchover to nonrevertive or brute force. Default value is revertive. If you configure brute force, a minimum link or last link failure for every MLACP failure occurs or the dynamic lag priority value is modified.

Step 15	**mlacp interchassis group** *{group-id}*
	 Example:
	 `Router(config-red)#interchassis group 230`
		Specifies that the port-channel is an MLACP port-channel. The **group-id** should match the configured redundancy group.

	Command	Purpose
Step 16	**backbone int** *member* **Example:** `Router(config-r-ic)# backbone interface GigabitEthernet2/4`	Sets the backbone interface member.
Step 17	**exit**	Exits the port-channel interface mode.

Examples

The following is a configuration example for Virtual Private Wire Services (VPWS):

ACTIVE POA

```
redundancy
interchassis group 100
  monitor peer bfd
  member ip 172.3.3.3
  backbone interface GigabitEthernet2/3
  backbone interface GigabitEthernet2/4
  mlacp system-priority 200
  mlacp node-id 0
!
interface Port-channel1
 no ip address
 load-interval 30
 speed nonegotiate
 port-channel min-links 4
 lacp failover brute-force
 lacp fast-switchover
 lacp max-bundle 4
 mlacp lag-priority 28000
 mlacp interchassis group 100
 service instance 2 ethernet
  encapsulation dot1q 2
  rewrite ingress tag pop 1 symmetric
  xconnect 172.2.2.2 2 pw-class mlacp
   backup peer 172.4.4.4 2 pw-class mlacp
 !
pseudowire-class mlacp
 encapsulation mpls
 status peer topology dual-homed

mpls ldp graceful-restart
!
!
interface Loopback0
 ip address 172.1.1.1 255.255.255.255
!
interface GigabitEthernet2/3
 ip address 120.0.0.1 255.255.255.0
 carrier-delay msec 0
 mpls ip
 bfd interval 100 min_rx 100 multiplier 3
!
interface GigabitEthernet2/9
 no ip address
 speed 1000
 channel-group 1 mode active
```

Use the **show lacp multi-chassis group** command to display the interchassis redundancy group value and the operational LACP parameters.

```
MLACP-PE1# show lacp multi-chassis group 100
Interchassis Redundancy Group 100
Operational LACP Parameters:
RG State:     Synchronized
System-Id:    200.000a.f331.2680
ICCP Version: 0
Backbone Uplink Status: Connected
Local Configuration:
Node-id:   0
System-Id: 200.000a.f331.2680

Peer Information:
State:        Up
Node-id:      7
System-Id:    2000.0014.6a8b.c680
ICCP Version: 0

State Flags: Active            - A
             Standby           - S
             Down              - D
             AdminDown         - AD
             Standby Reverting - SR
             Unknown           - U

mLACP Channel-groups
Channel    State      Priority     Active Links   Inactive Links
 Group   Local/Peer  Local/Peer     Local/Peer      Local/Peer
   1        A/S     28000/32768        4/4             0/0
```

Use the **show lacp multi-chassis portchannel** command to display the interface port-channel value channel group, LAG state, priority, inactive links peer configuration, and standby links.

```
MLACP-PE1# show lacp multi-chassis port-channel 1
Interface Port-channel1
Local Configuration:
Address: 000a.f331.2680
Channel Group: 1
State: Active
LAG State: Up
Priority: 28000
Inactive Links: 0
Total Active Links: 4
           Bundled: 4
          Selected: 4
           Standby: 0
        Unselected: 0

Peer Configuration:
Interface: Port-channel1
Address: 0014.6a8b.c680
Channel Group: 1
State: Standby
LAG State: Up
Priority: 32768
Inactive Links: 0
Total Active Links: 4
                           Bundled: 0
          Selected: 0
           Standby: 4
        Unselected: 0
```

Use the **show mpls ldp iccp** command to display the LDP session and ICCP state information.

```
MLACP-PE1# show mpls ldp iccp
ICPM RGID Table
  iccp:
    rg_id: 100, peer addr: 172.3.3.3
    ldp_session 0x3, client_id 0
    iccp state: ICPM_ICCP_CONNECTED
    app type: MLACP
        app state: ICPM_APP_CONNECTED, ptcl ver: 0
ICPM RGID Table total ICCP sessions: 1
ICPM LDP Session Table
  iccp:
    rg_id: 100, peer addr: 172.3.3.3
    ldp_session 0x3, client_id 0
    iccp state: ICPM_ICCP_CONNECTED
    app type: MLACP
        app state: ICPM_APP_CONNECTED, ptcl ver: 0
ICPM LDP Session Table total ICCP sessions: 1
```

Use the **show mpls l2transport** command to display the local interface and session details, destination address, and status.

```
MLACP-PE1# show mpls l2transport vc 2

Local intf     Local circuit              Dest address    VC ID      Status
-------------  -------------------------- --------------- ---------- ----------
Po1            Eth VLAN 2                 172.2.2.2       2          UP
Po1            Eth VLAN 2                 172.4.4.4       2          STANDBY
```

Use the **show etherchannel summary** command to display the status and identity of the MLACP member links.

```
MLACP-PE1# show etherchannel summary
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 2
Number of aggregators:           2

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(RU)         LACP      Gi2/9(P)    Gi2/20(P)   Gi2/31(P)
```

Use the **show lacp internal** command to display the device, port, and member- link information.

```
MLACP-PE1# 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 1
                            LACP port     Admin     Oper    Port        Port
Port      Flags   State     Priority      Key       Key     Number      State
Gi2/9     SA      bndl-act  28000         0x1       0x1     0x820A      0x3D
Gi2/20    SA      bndl-act  28000         0x1       0x1     0x8215      0x3D
```

```
Gi2/31    SA      bndl-act  28000         0x1       0x1      0x8220      0x3D
Gi2/40    SA      bndl-act  28000         0x1       0x1      0x8229      0x3D

Peer (MLACP-PE3) mLACP member links

Gi3/11    FA      hot-sby   32768         0x1       0x1      0xF30C      0x5
Gi3/21    FA      hot-sby   32768         0x1       0x1      0xF316      0x5
Gi3/32    FA      hot-sby   32768         0x1       0x1      0xF321      0x7
Gi3/2     FA      hot-sby   32768         0x1       0x1      0xF303      0x7
```

POA2

```
redundancy
 interchassis group 100
  monitor peer bfd
  member ip 172.1.1.1
  backbone interface GigabitEthernet3/3
  backbone interface GigabitEthernet3/5
  mlacp system-priority 2000
  mlacp node-id 7
!
interface Port-channel1
 no ip address
 load-interval 30
 speed nonegotiate
 port-channel min-links 4
 lacp failover brute-force
 lacp fast-switchover
 lacp max-bundle 4
 mlacp interchassis group 100
 service instance 2 ethernet
  encapsulation dot1q 2
  rewrite ingress tag pop 1 symmetric
  xconnect 172.2.2.2 2 pw-class mlacp
   backup peer 172.4.4.4 2 pw-class mlacp
!
pseudowire-class mlacp
 encapsulation mpls
 status peer topology dual-homed

mpls ldp graceful-restart
!
!
interface Loopback0
 ip address 172.3.3.3 255.255.255.255
!
interface GigabitEthernet3/2
 channel-group 1 mode active
!
interface GigabitEthernet3/3
 ip address 123.0.0.2 255.255.255.0
 mpls ip
 mpls label protocol ldp
 bfd interval 100 min_rx 100 multiplier 3
!
```

Use the **show lacp multi-chassis group** command to display the LACP parameters, local configuration, status of the backbone uplink, peer information, node ID, channel, state, priority active, and inactive links.

```
MLACP-PE3# show lacp multi-chassis group 100
Interchassis Redundancy Group 100
Operational LACP Parameters:
RG State:     Synchronized
```

```
System-Id:    200.000a.f331.2680
ICCP Version: 0
Backbone Uplink Status: Connected
Local Configuration:
Node-id:   7
System-Id: 2000.0014.6a8b.c680

Peer Information:
State:        Up
Node-id:      0
System-Id:    200.000a.f331.2680
ICCP Version: 0

State Flags: Active            - A
             Standby           - S
             Down              - D
             AdminDown         - AD
             Standby Reverting - SR
             Unknown           - U

mLACP Channel-groups
Channel    State      Priority     Active Links   Inactive Links
 Group   Local/Peer  Local/Peer     Local/Peer      Local/Peer
   1        S/A     32768/28000        4/4             0/0
```

Use the **show lacp multi-chassis portchannel** command to display the interface port-channel value channel group, LAG state, priority, inactive links peer configuration, and standby links.

```
MLACP-PE3# show lacp multi-chassis port-channel 1
Interface Port-channel1
Local Configuration:
Address: 0014.6a8b.c680
Channel Group: 1
State: Standby
LAG State: Up
Priority: 32768
Inactive Links: 0
Total Active Links: 4
           Bundled: 0
          Selected: 0
           Standby: 4
        Unselected: 0

Peer Configuration:
Interface: Port-channel1
Address: 000a.f331.2680
Channel Group: 1
State: Active
LAG State: Up
Priority: 28000
Inactive Links: 0
Total Active Links: 4
                           Bundled: 4
          Selected: 4
           Standby: 0
        Unselected: 0
```

Use the **show mpls ldp iccp** command to display the LDP session and ICCP state information.

```
MLACP-PE3# show mpls ldp iccp
ICPM RGID Table
  iccp:
    rg_id: 100, peer addr: 172.1.1.1
    ldp_session 0x2, client_id 0
    iccp state: ICPM_ICCP_CONNECTED
    app type: MLACP
```

```
        app state: ICPM_APP_CONNECTED, ptcl ver: 0
ICPM RGID Table total ICCP sessions: 1
ICPM LDP Session Table
  iccp:
    rg_id: 100, peer addr: 172.1.1.1
    ldp_session 0x2, client_id 0
    iccp state: ICPM_ICCP_CONNECTED
    app type: MLACP
        app state: ICPM_APP_CONNECTED, ptcl ver: 0
ICPM LDP Session Table total ICCP sessions: 1

MLACP-PE3# sh mpls l2transport vc 2

Local intf     Local circuit              Dest address    VC ID      Status
------------- -------------------------- --------------- ---------- ----------
Po1            Eth VLAN 2                 172.2.2.2       2          STANDBY
Po1            Eth VLAN 2                 172.4.4.4       2          STANDBY
```

Use the **show etherchannel summary** command to display the status and identity of the MLACP member links.

```
MLACP-PE3# show etherchannel summary
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 2
Number of aggregators:           2

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(RU)         LACP      Gi3/2(P)    Gi3/11(P)   Gi3/21(P)
                                 Gi3/32(P)
```

Use the **show lacp internal** command to display the device, port, and member- link information.

```
MLACP-PE3# show lacp 1 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 1
                            LACP port     Admin     Oper    Port        Port
Port      Flags   State     Priority      Key       Key     Number      State
Gi3/2     FA      bndl-sby  32768         0x1       0x1     0xF303      0x7
Gi3/11    FA      bndl-sby  32768         0x1       0x1     0xF30C      0x5
Gi3/21    FA      bndl-sby  32768         0x1       0x1     0xF316      0x5
Gi3/32    FA      bndl-sby  32768         0x1       0x1     0xF321      0x7

Peer (MLACP-PE1) mLACP member links

Gi2/20    SA      bndl      28000         0x1       0x1     0x8215      0x3D
Gi2/31    SA      bndl      28000         0x1       0x1     0x8220      0x3D
Gi2/40    SA      bndl      28000         0x1       0x1     0x8229      0x3D
Gi2/9     SA      bndl      28000         0x1       0x1     0x820A      0x3D
MLACP-PE3#
```

The following is a configuration example for a Virtual Private Lan Service (VPLS):

Active POA

```
redundancy
interchassis group 100
  monitor peer bfd
  member ip 172.3.3.3
  backbone interface GigabitEthernet2/3
  backbone interface GigabitEthernet2/4
  mlacp system-priority 200
  mlacp node-id 0
!
interface Port-channel1
 no ip address
 speed nonegotiate
 port-channel min-links 2
 lacp fast-switchover
 lacp max-bundle 4
 mlacp lag-priority 28800
 mlacp interchassis group 100
 service instance 4000 ethernet
  encapsulation dot1q 4000
  rewrite ingress tag pop 1 symmetric
  bridge-domain 4000
!
l2 vfi VPLS manual
 vpn id 4000
 neighbor 172.2.2.2 encapsulation mpls
 neighbor 172.4.4.4 encapsulation mpls
 status decoupled
!
interface Vlan4000
 xconnect vfi VPLS
!
mpls ldp graceful-restart
!
interface Loopback0
 ip address 172.1.1.1 255.255.255.255
!
interface GigabitEthernet2/3
 ip address 120.0.0.1 255.255.255.0
 carrier-delay 0
 mpls ip
 bfd interval 100 min_rx 100 multiplier 3
!
interface GigabitEthernet2/9
 channel-group 1 mode active
!
```

Use the **show lacp mg** command to display the LACP parameters, local configuration, status of the backbone uplink, peer information, node ID, channel, state, priority active, and inactive links.

```
MLACP-PE1# show lacp multi-chassis group 100
Interchassis Redundancy Group 100

Operational LACP Parameters:
RG State:     Synchronized
System-Id:    200.000a.f331.2680
ICCP Version: 0
Backbone Uplink Status: Connected
Local Configuration:
Node-id:   0
```

```
System-Id: 200.000a.f331.2680

Peer Information:
State:        Up
Node-id:      7
System-Id:    2000.0014.6a8b.c680
ICCP Version: 0

State Flags: Active            - A
             Standby           - S
             Down              - D
             AdminDown         - AD
             Standby Reverting - SR
             Unknown           - U

mLACP Channel-groups
Channel    State      Priority     Active Links   Inactive Links
 Group   Local/Peer  Local/Peer     Local/Peer      Local/Peer
   1        A/S     28000/32768        4/4             0/0
```

Use the **show lacp multi-chassis portchannel** command to display the interface port-channel value channel group, LAG state, priority, inactive links peer configuration, and standby links.

```
MLACP-PE1# show lacp multi-chassis port-channel 1
Interface Port-channel1
Local Configuration:
Address: 000a.f331.2680
Channel Group: 1
State: Active
LAG State: Up
Priority: 28000
Inactive Links: 0
Total Active Links: 4
           Bundled: 4
          Selected: 4
           Standby: 0
        Unselected: 0

Peer Configuration:
Interface: Port-channel1
Address: 0014.6a8b.c680
Channel Group: 1
State: Standby
LAG State: Up
Priority: 32768
Inactive Links: 0
Total Active Links: 4
                           Bundled: 0
          Selected: 0
           Standby: 4
        Unselected: 0
```

Use the **show mpls ldp iccp** command to display the LDP session and ICCP state information.

```
MLACP-PE1# show mpls ldp iccp
ICPM RGID Table
  iccp:
    rg_id: 100, peer addr: 172.3.3.3
    ldp_session 0x3, client_id 0
    iccp state: ICPM_ICCP_CONNECTED
    app type: MLACP
        app state: ICPM_APP_CONNECTED, ptcl ver: 0
ICPM RGID Table total ICCP sessions: 1
ICPM LDP Session Table
  iccp:
    rg_id: 100, peer addr: 172.3.3.3
    ldp_session 0x3, client_id 0
    iccp state: ICPM_ICCP_CONNECTED
    app type: MLACP
        app state: ICPM_APP_CONNECTED, ptcl ver: 0
ICPM LDP Session Table total ICCP sessions: 1
```

Use the **show mpls l2transport** command to display the local interface and session details, destination address, and the status.

```
MLACP-PE1# show mpls l2transport vc 4000

Local intf     Local circuit              Dest address    VC ID      Status
-------------  -------------------------- --------------- ---------- ----------
VFI VPLS       VFI                        172.2.2.2       4000       UP
VFI VPLS       VFI  172.4.4.4  4000     UP
```

Use the **show etherchannel summary** command to display the status and identity of the MLACP member links.

```
MLACP-PE1# show etherchannel summary
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 2
Number of aggregators:           2

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(RU)         LACP      Gi2/9(P)    Gi2/20(P)   Gi2/31(P)
                                 Gi2/40(P)
```

Use the **show lacp internal** command to display the device, port, and member-link information.

```
MLACP-PE1# 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 1
                            LACP port     Admin     Oper    Port        Port
Port      Flags   State     Priority      Key       Key     Number      State
Gi2/9     SA      bndl-act  28000         0x1       0x1     0x820A      0x3D
Gi2/20    SA      bndl-act  28000         0x1       0x1     0x8215      0x3D
Gi2/31    SA      bndl-act  28000         0x1       0x1     0x8220      0x3D
Gi2/40    SA      bndl-act  28000         0x1       0x1     0x8229      0x3D

Peer (MLACP-PE3) mLACP member links

Gi3/11    FA      hot-sby   32768         0x1       0x1     0xF30C      0x5
Gi3/21    FA      hot-sby   32768         0x1       0x1     0xF316      0x5
Gi3/32    FA      hot-sby   32768         0x1       0x1     0xF321      0x7
Gi3/2     FA      hot-sby   32768         0x1       0x1     0xF303      0x7
```

Configuration example on a standby PoA:

```
redundancy
 interchassis group 100
  monitor peer bfd
  member ip 172.1.1.1
  backbone interface GigabitEthernet3/3
  backbone interface GigabitEthernet3/5
  mlacp system-priority 2000
  mlacp node-id 7
!
interface Port-channel1
 no ip address
 speed nonegotiate
 port-channel min-links 2
 lacp fast-switchover
 lacp max-bundle 4
 mlacp lag-priority 28800
 mlacp interchassis group 100
 service instance 4000 ethernet
  encapsulation dot1q 4000
  rewrite ingress tag pop 1 symmetric
  bridge-domain 4000
!
l2 vfi VPLS manual
 vpn id 4000
 neighbor 172.2.2.2 encapsulation mpls
 neighbor 172.4.4.4 encapsulation mpls
 status decoupled
!
interface Vlan4000
 xconnect vfi VPLS
!
mpls ldp graceful-restart
!
!
interface Loopback0
 ip address 172.3.3.3 255.255.255.255
!
interface GigabitEthernet3/2
 channel-group 1 mode active
!
interface GigabitEthernet3/3
 ip address 123.0.0.2 255.255.255.0
 mpls ip
 mpls label protocol ldp
 bfd interval 100 min_rx 100 multiplier 3
!
```

Use the **show lacp multi-chassis group** *interchassis group number* command to display the LACP parameters, local configuration, status of the backbone uplink, peer information, nodeID, channel, state, priority, active, and inactive links.

```
MLACP-PE3# show lacp multi-chassis group 100
Interchassis Redundancy Group 100

Operational LACP Parameters:
RG State:     Synchronized
System-Id:    200.000a.f331.2680
ICCP Version: 0
Backbone Uplink Status: Connected
Local Configuration:
Node-id:   7
System-Id: 2000.0014.6a8b.c680
```

```
Peer Information:
State:        Up
Node-id:      0
System-Id:    200.000a.f331.2680
ICCP Version: 0

State Flags: Active            - A
             Standby           - S
             Down              - D
             AdminDown         - AD
             Standby Reverting - SR
             Unknown           - U

mLACP Channel-groups
Channel    State      Priority     Active Links   Inactive Links
 Group   Local/Peer  Local/Peer     Local/Peer      Local/Peer
   1        S/A     32768/28000        4/4             0/0
```

Use the **show lacp multi-chassis portchannel** command to display the interface port-channel valuechannel group, LAG state, priority, inactive links peer configuration, and standby links.

```
MLACP-PE3# show lacp multi-chassis port-channel 1
Interface Port-channel1
Local Configuration:
Address: 0014.6a8b.c680
Channel Group: 1
State: Standby
LAG State: Up
Priority: 32768
Inactive Links: 0
Total Active Links: 4
           Bundled: 0
          Selected: 0
           Standby: 4
        Unselected: 0

Peer Configuration:
Interface: Port-channel1
Address: 000a.f331.2680
Channel Group: 1
State: Active
LAG State: Up
Priority: 28000
Inactive Links: 0
Total Active Links: 4
                           Bundled: 4
          Selected: 4
           Standby: 0
        Unselected: 0

MLACP-PE3# show mpls ldp iccp
ICPM RGID Table
  iccp:
    rg_id: 100, peer addr: 172.1.1.1
    ldp_session 0x2, client_id 0
    iccp state: ICPM_ICCP_CONNECTED
    app type: MLACP
        app state: ICPM_APP_CONNECTED, ptcl ver: 0
ICPM RGID Table total ICCP sessions: 1
ICPM LDP Session Table
  iccp:
    rg_id: 100, peer addr: 172.1.1.1
    ldp_session 0x2, client_id 0
    iccp state: ICPM_ICCP_CONNECTED
    app type: MLACP
        app state: ICPM_APP_CONNECTED, ptcl ver: 0
ICPM LDP Session Table total ICCP sessions: 1


MLACP-PE3# sh mpls l2transport vc 2

Local intf     Local circuit              Dest address    VC ID      Status
-------------  -------------------------- --------------- ---------- ----------
VFI VPLS       VFI                        172.2.2.2       4000       UP
VFI VPLS       VFI  172.4.4.4  4000     UP

Use the show etherchannel summary command to display the status and identity of the MLACP
member
links.
MLACP-PE3#show etherchannel summary
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

Number of channel-groups in use: 2
Number of aggregators:           2

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(RU)         LACP      Gi3/2(P)    Gi3/11(P)   Gi3/21(P)
                                 Gi3/32(P)
```

Use the **show lacp internal** command to display the device, port, and member- link information.

```
MLACP-PE3# show lacp 1 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 1
                            LACP port     Admin     Oper    Port        Port
Port      Flags   State     Priority      Key       Key     Number      State
Gi3/2     FA      bndl-sby  32768         0x1       0x1     0xF303      0x7
Gi3/11    FA      bndl-sby  32768         0x1       0x1     0xF30C      0x5
Gi3/21    FA      bndl-sby  32768         0x1       0x1     0xF316      0x5
Gi3/32    FA      bndl-sby  32768         0x1       0x1     0xF321      0x7

Peer (MLACP-PE1) mLACP member links

Gi2/20    SA      bndl      28000         0x1       0x1     0x8215      0x3D
Gi2/31    SA      bndl      28000         0x1       0x1     0x8220      0x3D
Gi2/40    SA      bndl      28000         0x1       0x1     0x8229      0x3D
Gi2/9     SA      bndl      28000         0x1       0x1     0x820A      0x3D
MLACP-PE3#
```

Troubleshooting

Table 2-9 provides troubleshooting solutions for Multichassis Support for LACP feature.

Table 2-9 Troubleshooting Multichassis Support for LACP

Problem	Solution
If the MLACP or ICCP is down or if the the links between the POA are down or theICCP is in a different state	Use the **debug lacp multi-chassis** command to verify the MLACP interface, database, interchassis redundancy group and user interface. Share the output with TAC for further investigation.
Connection timer has expired or ICCP is disconnected or attempting to establish an ICCP connection	Use the **show lacp multi-chassis group** command to verify if the connection timer is enabled. Share the output with TAC for further investigation.
LAG state failure or RG is synchronized or timed out LAG is enabled for member links, but the hold down timer is running or RG is waiting Member links are not enabled or link failure detected	Use the **show lacp multi-chassis port-channel** commandand share the output with TAC for further investigation.
Traffic loss after reversal	Use the **test lacp multichassis reversion-delay** *delay_in_seconds* command to extend the amount of time before mLACP reverts to the previous active state after failure recovery. This allows the pseudowires additional time to come UP before the PoA reverts to active state.
Failover performance	Complete these steps: **1.** Reduce the pesudowire number to 100and check if the performance issue is corrected. **2.** Check the DHD configuration. Remove any min-links configuration from the DHD as it slows down failovers and provides no additional value. **3.** Ensure that **lacp fast-switchover** is configured on DHD and PoAs.
Unexpected active/standby roles or priorities	Complete these steps: **1.** Use the **show run int port-ch and show lacp mp** command to check if the configured priorities match the operational priorities. If they don’t match, it implies that there was a failover. **2.** Use the **monitor event-trace mlacp** command on the SP to verify the reasons for any failovers that caused swapping of active/standby roles or dynamic priority changes.

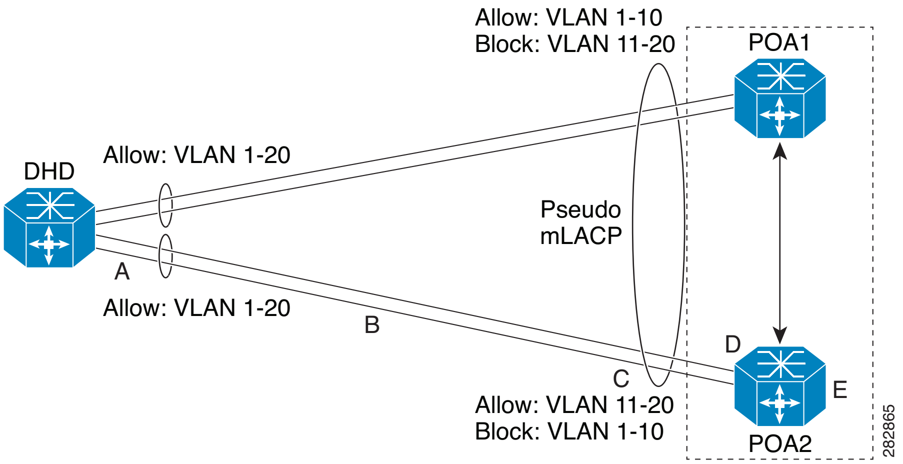
Pseudo MLACP Support on Cisco 7600

In dual homing, a device is connected to the network using two independent access points or points of attachments (POAs). One POA is the primary connection and the other is a standby connection that is activated in the event of a failure of the primary connection. The Multi-chassis Link Aggregation Protocol (MLACP) solution is an active and standby Provider Edge (PE) redundancy mechanism. The Pseudo MLACP (PMLACP) feature introduced in Cisco IOS release 15.1(3)S, provides a flexible dual homing redundancy mechanism where both the connections are in the active mode (active-active mode). In PMLACP implementation, a PMLACP application is implemented on the PE router. Both the POA ports are placed in active mode with manual VLAN load balancing.

PMLACP provides higher bandwidth utilization than MLACP and other active and standby link level schemes. PMLACP provides VLAN based redundancy by allowing you to configure one primary and one secondary interface pair for each member VLAN. The POAs determine which POA is active and standby for each VLAN on a Multi-Chassis Link Aggregation (MLAG) and only the active POA forwards frames for the respective VLAN. Additionally PMLACP allows maximum flexibility for the PE-CE inter operability in terms of dual-homing redundancy and failover recovery.

Figure 2-4 explains the PMLACP implementation with manual VLAN load-balancing configuration.

Figure 2-4 PMLACP Implementation

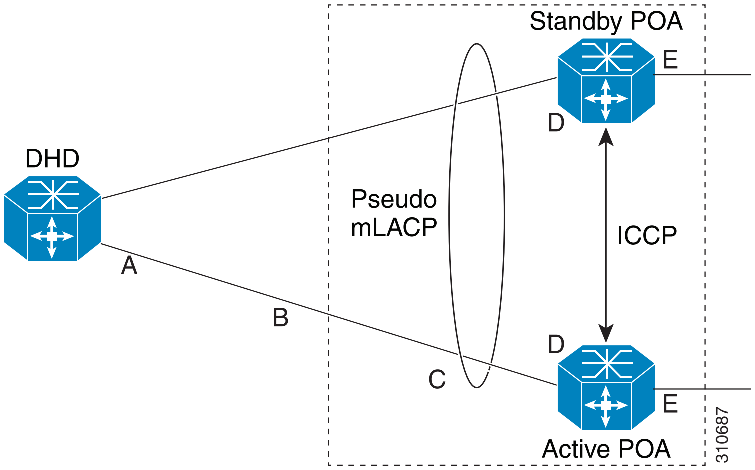


In the illustration, POA ports are configured for a PMLACP role, and ports are configured in active-active mode with manual VLAN load-balancing. The POAs are configured to allow certain VLANs on one of their downlinks but not the other VLANs. The POA activates its uplinks for locally active VLANs. DHD is configured to enable all VLANs on both its uplinks. Traffic from DHD is initially flooded on both uplinks until DHD learns which uplink is active for which VLANs.

Failover Operations

The PMLACP feature provides network resiliency by protecting against port, link, and node failures. Figure 2-5 explains the failure points in a network.

Figure 2-5 PMLACP Failover Protection



These failures can be categorized into five types.

- A—Failure of the uplink port on the DHD
- B—Failure of the ethernet link
- C—Failure of the downlink port on the POA
- D—Failure of the POA node
- E—Failure of the active POA uplinks

The failover operations are triggered by three different events.

- Access side link or port failure (failure types A- C): PMLACP on the failing POA initiates a failover to the peer for any VLANs that were active on the failed link or links. This failover is initiated by sending an MLACP port state Type Length Value (TLV) message, indicating that the port state is down.
- Node failure (failure type D): PMLACP on the surviving POA receives a node failure notification and initiates a failover of all VLANs in standby mode on all shared MLAGs.
- POA uplink failure (failure type E): The failing POA sends a message to the peer about the core isolation using the MLACP system state TLV, indicating that the POA is isolated. It will then place all VLANs in the blocking mode.

All the three failover events involve the peer POA receiving a notification of the failure. At this point the receiving standby POA completes the following steps:

1. Unblocks any of the affected VLANs which were in standby or blocked mode.

2. Sends a MAC flush message to the access side network device through a Multiple VLAN Registration Protocol (MVRP) message. This message reflects all the VLANs which are being activated only for the associated interface. When DHD receives the MVRP message, DHD responds by flushing the MAC address tables for those VLANs.
3. Triggers the core network edge MAC flushing.

Failure Recovery

PMLACP uses revertive mode after a failure recovery to support the active-active model. The reversal process is also similar to the failover process. The standby POA initates the reversal for each VLAN by indicating that the POA is relinquishing its active role for the VLAN. This is done though an ICCP PLACP interface state TLV message, which indicates that it is no longer in active mode for the affected VLANs. Upon TLV receipt, the recovering POA unblocks the affected VLANs and triggers the MAC flushes towards access side and core side.

Revertive mode is enabled by default. If you want to choose when to trigger reversion after the failover recovery, you can configure non revertive mode. The non revertive mode is enabled by configuring the command **lacp failover non-revertive** under port channel.

Restrictions for PMLACP

Follow this restrictions and usage guidelines while configuring PMLACP.

- PMLACP is supported on ES+ and ES 20 line cards.
- PMLACP is supported on SUP 720 and RSP 720.
- PMLACP configuration on a port channel supports only service instances.
- If PMLACP is enabled on a port channel, Resilient Ethernet Protocol (RTP), Spanning Tree Protocol (STP), Link Aggregation Control Protocol (LACP), VLAN Trunking Protocol (VTP), or other layer 2 control protocols are not supported.
- The ethernet VLAN color blocking needs to be configured on all VLANs under the port channel if it has EVC xconnect or MTP configured on it. Use the **ethernet vlan color-block vlan all** command for configuring it.
- Both POAs must contain the same configuration of manual-load balance VLAN list and LAG.
- The bridge-domain that is configured under a PMLACP port channel EVC should not be part of any other non PMLACP interfaces.
- Only one port channel of MLACP or PMLACP type is supported on a single redundancy group (RG). There can be one MLACP port channel and another PMLACP port channel on a single RG, but not two port channels of the same type.
- Active VLAN list configuration needs to be the same on both POAs.
- The port-channel configuration on both POAs must be the same, but port-channel members need not be the same.
- The recommended configuration sequence for PMLACP is:
  - Configure interchassis group and PMLACP commands.
  - Configure MLACP interchassis group and other port channel commands.
  - Add member links.

Configuring PMLACP on Cisco 7600

Complete the following steps to configure PMLACP on the Cisco 7600 router.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** *pw-class-name*
4. **encapsulation mpls**
5. **status peer topology dual-homed**
6. **exit**
7. **l2 vfi** *name* **manual**
8. **vpn id** *vpn-id*
9. **neighbor** *remote-id* **encapsulation mpls**
10. **exit**
11. **redundancy**
12. **interchassis group** *number*
13. **monitor peer bfd**
14. **member IP** *IP-address*
15. **mlacp node-id** *number*
16. **mlacp system-priority** *priority*
17. **backbone interface** *interface*
18. **exit**
19. **interface port-channel** *port-channel number*
20. **no ip address**
21. **mlacp interchassis group** *group-id*
22. **mlacp mode** *active-active*
23. **mlacp load-balance primary vlan** *range*
24. **mlacp load-balance secondary vlan** *range*
25. **ethernet vlan color-block all**
26. **service instance** *id* **ethernet**
27. **encapsulation dot1q** *vlan id*
28. **rewrite ingress tag pop {1 | 2} symmetric**
29. **xconnect** *peer-id vc-id* **pw-class** *pw-class-name*

    or

    brige-domain *bridge-domain-id*
30. **backup peer** *peer-id vc-id* **pw-class** *pw-class-name*

31. **exit**
32. **interface vlan** *bridge-domain-id*
33. **xconnect vfi vfi-name**
34. **end**

DETAILED STEPS

	Command	Purpose

Step 1	**`enable`**
	Example:
	`Router> enable`
		Enables privileged EXEC mode, and if prompted enter your password.

Step 2	**`configure terminal`**
	Example:
	`Router# configure terminal`
		Enters global configuration mode.

Step 3	**`pseudowire-class`** *`pw-class-name`*
	Example:
	`Router(config)# pseudowire-class vpws`
		Specifies the name of a pseudowire class and enters pseudowire class configuration mode.

Step 4	**`encapsulation mpls`**
	Example:
	`Router(config-pw-class)# encapsulation mpls`
		Specifies that MPLS is used as the data encapsulation method for tunneling Layer 2 traffic over the pseudowire.

Step 5	**`status peer topology dual-homed`**
	Example:
	`Router(config-pw-class)# status peer topology dual-homed`
		Enables the reflection of the attachment circuit status on both the primary and secondary pseudowires. This configuration is necessary if the peer PEs are connected to a dual-homed device.

Step 6	**`exit`**
	Example:
	`Router(config-pw-class)# exit`
		Exits pseudowire class configuration mode.

Step 7	**`l2 vfi`** *`name`* **`manual`**
	Example:
	`Router(config)# l2 vfi vpls manual`
		Creates a named Layer 2 Virtual Forwarding Instance (VFI) and enables the Layer 2 VFI manual configuration mode.
		Note Perform steps 7 to 10 only if you are configuring PMLACP over VPLS. Else go to step 11.

	Command	Purpose
Step 8	**vpn id** *vpn-id* **Example:** `Router(config-vfi)# vpn id 17`	Configures a VPN ID for the VPLS domain.
Step 9	**neighbor** *remote-id* **encapsulation mpls** **Example:** `Router(config-vfi)# neighbor 1.5.1.1 encapsulation mpls`	Specifies the remote peering router ID, which is the IP address of the router, and the tunnel encapsulation type for the emulated VC.
Step 10	**exit** **Example:** `Router(config-vfi)# exit`	Exits the L2 VFI manual configuration mode.
Step 11	**redundancy** **Example:** `Router(config)# redundancy`	Enters redundancy configuration mode.
Step 12	**interchassis group** *number* **Example:** `Router(configure-red)# interchassis group 100`	Configures an interchassis group within the redundancy configuration mode and assigns a group number.
Step 13	**monitor peer bfd** **Example:** `Router(configure-r-ic)# monitor peer bfd`	Configures the BFD option to monitor the state of the peer. **Note** The **monitor peer bfd** command is optional. If this command is not specified, the default option is route-watch.
Step 14	**member ip** *IP-address* **Example:** `Router(configure-r-ic)# member ip 172.3.3.3`	Configures the IP address of the MLACP peer member group.
Step 15	**mlacp node-id** *node-id* **Example:** `Router(config-r-ic)# mlacp node-id 5`	Specifies the node ID to be used in the LACP port-id field. **node-id** — Valid range is 0 - 7, and the value should be different from the peer values.

	Command	Purpose
Step 16	**mlacp system-priority** *priority* **Example:** `Router(config-r-ic)# mlacp system-priority 100`	Specifies the system priority advertised to the other MLACP members of the redundancy group. **priority** — Acceptable range is 1 to 65535. The default value is 32768. The assigned values should be lower than the DHD.
Step 17	**backbone interface** *interface* **Example:** `Router(config-r-ic)# backbone interface GigabitEthernet2/3`	Specifies the backbone interface for the MLACP configuration.
Step 18	**exit** **Example:** `Router(config-r-ic)# exit`	Exits the redundancy mode.
Step 19	**interface port-channel** *number* **Example:** `Router(config)# interface Port-channel 10`	Specifies the port-channel interface.
Step 20	**no ip address** **Example:** `Router(config-if)# no ip address`	Removes the IP address from the interface.
Step 21	**mlacp interchassis group** *group-id* **Example:** `Router(config-if)# mlacp interchassis group 100`	Specifies that the port-channel is an MLACP port-channel. The **group-id** should match the configured redundancy group.
Step 22	**mlacp mode active-active** **Example:** `Router(config-if)# mlacp mode active-active`	Specifies the MLACP mode as active-active.
Step 23	**mlacp load-balance primary vlan** *range* **Example:** `Router(config-if)# mlacp load-balance primary vlan 100-109`	Specifies the primary VLAN range for manual load balancing. **range** — Specifies the VLAN ID range. Values range from 1 to 4094.

	Command	Purpose
Step 24	**mlacp load-balance secondary vlan** ***range*** **Example:** `Router(config-if)# mlacp load-balance secondary vlan 110-120`	Specifies the secondary VLAN range for manual load balancing.
Step 25	**ethernet vlan color-block all** **Example:** `Router(config-if)# ethernet vlan color-block all`	Blocks VLANs on EVCs with connect and cross-connect. devices. **Note** This configuration is required if EVC cross connect or MTP is used on the PMLACP port channel.
Step 26	**service instance** *id* ethernet **Example:** `Router(config-if)# service instance 101 ethernet`	Creates a service instance on an interface.
Step 27	**encapsulation dot1q** *vlan-id* **Example:** `Router(config-if-srv)# encapsulation dot1q 100`	Configures the encapsulation. Defines the matching criteria to be used in order to map the ingress dot1q frames on an interface to the appropriate service instance.
Step 28	**rewrite ingress tag pop {1 | 2} symmetric** **Example:** `Router(config-if-srv)# rewrite ingress tag pop 1 symmetric`	Specifies the tag manipulation that is to be performed on the frame in ingress direction to the service instance.
Step 29	**xconnect** *peer-id vc-id* **pseudowire-class** *pw-classname* or **brige-domain** bridge-domain-*id* **Example:** `Router(config-if-srv)# xconnect 3.3.3.3 90 pseudowire-class vpws`	Binds the 802.1Q VLAN attachment circuit to a virtual circuit (VC). Binds the attachment circuit to a pseudowire VC. • **peer-id—** specifies the IP address of the peer PE router. • **vc-id**— specifies the 32-bit value that identifies the VC between the peer PE routers at each endpoint of the VC. You must configure the same VC ID on the peer PE router. • **pw-classname**— Specifies the pseudowire class. **Note** Use the **bridge-domain** command if you are configuring PMLACP on VPLS.

	Command	Purpose
Step 30	**backup peer** *peer-id vc-id* **pseudowire-class** *pw-classname* **Example:** `Router(config-if-srv)# backup peer 4.3.3.3 90 pseudowire-class vpws`	Specifies a redundant peer for a pseudowire virtual circuit.
Step 31	**exit** **Example:** `Router(config-if-srv)# end`	Exits from the interface configuration mode.
Step 32	**interface vlan** *bridge-domain-id* **Example:** `Router(config-if)# interface vlan 201`	Creates or accesses a dynamic switched virtual interface (SVI). **Note** You need to perform steps 32 and 33 only if you are configuring VPLS.
Step 33	**xconnect vfi** *vfi-name* **Example:** `Router(config-if)# xconnect vfi vpls`	Specifies the Layer 2 VFI that you are binding to the VLAN port.

Step 34	**end**
	Example:
	`Router(config-if)# end`
		Exits the port-channel interface mode.

Configuration Examples

This is a configuration example for PMLACP with EVC xconnect on two POAs, A and B. In this example primary VLAN range is configured as 100-109 on router A and 110-120 on router B. The VLAN range is interchanged so that the primary VLAN range of router A becomes the secondary VLAN range in router B and the secondary VLAN range of router A becomes the primary VLAN range in router B.

```
RouterA> enable
RouterA# configure terminal
RouterA(config)# pseudowire-class vpws
RouterA(config-pw-class)# encapsulation mpls
RouterA(config-pw-class)# status peer topology dual-homed
RouterA(config-pw-class)# exit
RouterA(config)# l2 vfi vpls manual
RouterA(config-vfi)# vpn id 100
RouterA(config-vfi)# neighbor 3.3.3.3 encapsulation mpls
RouterA(config-vfi)# exit
RouterA(config)# redundancy
RouterA(config-red)# interchassis group 100
RouterA(config-r-ic)# monitor peer bfd
RouterA(config-r-ic)# member ip 2.2.2.2
RouterA(config-r-ic)# backbone interface GigabitEthernet8/0/10
RouterA(config-r-ic)# mlacp system-priority 100
RouterA(config-r-ic)# mlacp node-id 1
```

```
Router(config)# interface Port-channel10
RouterA(config-if)# no ip address
RouterA(config-if)# mlacp interchassis group 100
RouterA(config-if)# mlacp mode active-active
RouterA(config-if)# mlacp load-balance primary vlan 100-109
RouterA(config-if)# mlacp load-balance secondary vlan 110-120
RouterA(config-if)# ethernet vlan color-block all
RouterA(config-if)# service instance 10 ethernet
RouterA(config-if-srv)# encapsulation dot1q 100
RouterA(config-if-srv)# rewrite ingress tag pop 1 symmetric
RouterA(config-if-srv)# xconnect 3.3.3.3 90 pseudowire-class vpws
RouterA(config-if-srv)# backup peer 4.3.3.3 91
RouterA(config-if)# service instance 11 ethernet
RouterA(config-if-srv)# encapsulation dot1q 101
RouterA(config-if-srv)# rewrite ingress tag pop 1 symmetric
RouterA(config-if-srv)# bridge-domain 201
RouterA(config-if-srv)# exit
RouterA(config-if)# exit
RouterA(config)# interface vlan 201
RouterA(config-if)# no shutdown
RouterA(config-if)# xconnect vfi vpls
RouterA(config-if)# end

RouterB> enable
RouterB# configure terminal
RouterB(config)# pseudowire-class vpws
RouterB(config-pw-class)# encapsulation mpls
RouterB(config-pw-class)# status peer topology dual-homed
RouterB(config-pw-class)# exit
RouterB(config)# l2 vfi vpls manual
RouterB(config-vfi)# vpn id 100
RouterB(config-vfi)# neighbor 3.3.3.3 encapsulation mpls
RouterB(config-vfi)# exit
RouterB(config)# redundancy
RouterB(config-red)# interchassis group 100
RouterB(config-r-ic)# monitor peer bfd
RouterB(config-r-ic)# member ip 1.1.1.1
RouterB(config-r-ic)# backbone interface GigabitEthernet8/0/10
RouterB(config-r-ic)# mlacp system-priority 100
RouterB(config-r-ic)# mlacp node-id 2
Router(config)# interface Port-channel 10
RouterB(config-if)# no ip address
RouterB(config-if)# mlacp interchassis group 100
RouterB(config-if)# mlacp mode active-active
RouterB(config-if)# mlacp load-balance primary vlan 110-120
RouterB(config-if)# mlacp load-balance secondary vlan 100-109
RouterB(config-if)# ethernet vlan color-block all
RouterB(config-if)# service instance 10 ethernet
RouterB(config-if-srv)# encapsulation dot1q 100
RouterB(config-if-srv)# rewrite ingress tag pop 1 symmetric
RouterB(config-if-srv)# xconnect 3.3.3.3 90 pseudowire-class vpws
RouterB(config-if-srv)# backup peer 4.3.3.3 91
RouterB(config-if)# service instance 11 ethernet
RouterB(config-if-srv)# encapsulation dot1q 101
RouterB(config-if-srv)# rewrite ingress tag pop 1 symmetric
RouterB(config-if-srv)# bridge-domain 201
RouterB(config-if-srv)# exit
RouterB(config-if)# exit
RouterB(config)# interface vlan 201
RouterB(config-if)# no shutdown
RouterB(config-if)# xconnect vfi vpls
RouterB(config-if)# end
```

Verification

Use the **show lacp multi-chassis load-balance port-channel** *number* command to verify the PMLACP configuration information on the port channel interface.

```
PE1# show lacp multi-chassis load-balance port-channel 10
Interface Port-Channel 10
        Local Configuration:
                P-mLACP Enabled:      Yes
                Redundancy Group:     100
                Revertive Mode:       Non-Revertive
                Primary VLANs:        4001-4002,4004-4005,4007-4010
                Secondary VLANs:      4012-4013,4015-4016,4018-4021
Local Interface State:
                Interface ID: 10
                Port State:           Up
                Primary VLAN State:   Standby
                Secondary VLAN State: Standby
Peer Interface State:
                Interface ID: 10
                Primary VLAN State:   Active
                Secondary VLAN State: Active
```

Use the **show lacp multi-chassis group** command to display the interchassis redundancy group and the operational LACP parameters.

```
PE1# show lacp multi-chassis group

Interchassis Redundancy Group 100
Operational LACP Parameters:
                RG State:     Synchronized
                System-Id:    32768.001b.0de6.3080
                ICCP Version: 0
        Backbone Uplink Status: Connected
        Local Configuration:
                Node-id:   1
                System-Id: 32768.001b.0de6.3080
Peer Information:
                State:        Up
                Node-id:      2
                System-Id:    32768.f866.f2d2.6680
                ICCP Version: 0
State Flags: Active - A
             Standby           - S
             Down              - D
             AdminDown         - AD
             Standby Reverting - SR
             Unknown           - U
mLACP Channel-groups
Channel    State      Priority     Active Links   Inactive Links
 Group   Local/Peer  Local/Peer     Local/Peer      Local/Peer
   10       A/A     32768/32768        2/2             0/0

Redundancy Group 100 (0x64)
  Applications connected: mLACP, Pseudo-mLACP
  Monitor mode: BFD
  member ip: 2.2.2.2 "PE2", CONNECTED
    BFD neighbor: GigabitEthernet2/9, next hop 192.168.41.2, UP
    mLACP state: CONNECTED
    Pseudo-mLACP state: CONNECTED

backbone int GigabitEthernet8/0/9: UP (IP)
```

```
ICRM fast-failure detection neighbor table
  IP Address       Status Type Next-hop IP      Interface
  ==========       ====== ==== ===========      =========
  2.2.2.2          UP     BFD  192.168.41.2     GigabitEthernet2/9
```

Use the **show lacp multi-chassis load-balance group** command to display the PMLACP configuration information including redundancy group, link states and interface status.

```
PE2#sh lacp multi-chassis load-balance group
Interchassis Redundancy Group 100
                RG State:       Synchronized
                ICCP Version:   0
        Backbone Uplink Status: Connected
        Local Configuration:
                Node-id:        2
        Peer Information:
                State:          Up
                Node-id:        1
                ICCP Version:   0
States:      Active     - ACT           Standby    - SBY
             Down       - DN            AdminDown  - ADN
             Unknown    - UN            Reverting  - REV
P-mLACP Interfaces
Interface    Port State     Local VLAN State      Peer VLAN State
   ID          Local       Primary/Secondary     Primary/Secondary
   10          ADN              ADN/ADN                DN/DN
   34           UP              ACT/SBY               ACT/SBY
```

Troubleshooting Tips

Table 2-10 Troubleshooting Tips

Command	Purpose
debug lacp load-balance [all | database | redundancy-group | vlan]	Enables debugging of the PMLACP activity. Use this command from the switch processor (SP).
debug redundancy interchassis [all | application | error | event | monitor]	Enables debugging of the interchassis redundancy manager.
debug mpls ldp iccp	Enables debugging of the Inter Chassis Control Protocol (ICCP). Use this command from the RP.

Configuring Custom Ethertype for EVC Interfaces

Custom Ethertype feature allows you to customize the Ethernet settings on an ES20 line card. This feature enables you to configure an ethertype with the outer tag for dot1Q and QinQ packets. Both EVCs (802.1Q and QinQ) and QinQ routed subinterfaces support custom ethertype. By default Cisco 7600 series router supports Ethertype 0x8100 for dot1Q and Q-in-Q outer tag. You can use the custom ethertype feature to configure the following ethertypes for each port on ES20 line cards:

- 0x8100 – 802.1q

- 0x9100 – Q-in-Q
- 0x9200 – Q-in-Q
- 0x88a8 – 802.1ad

Use this command to configure a custom ethertype on a physical port.

dot1q tunneling ethertype <*0x88A8* | *0x9100* | *0x9200*>

In this sample configuration, Ethertype is set to 0x9100, service instance is created, and rewrite is initiated:

```
interface GigabitEthernet 1/0/0
    dot1q tunneling ethertype 0x9100
    service instance <number> ethernet
        encapsulation dot1q <vlan 1> [second-dot1q <vlan 2>]
        Rewrite <Rewrite>
```

Note 802.1q (0x8100) is the default ethertype setting.

Supported Rewrite Rules for a Custom Ethertype Configuration

Rewriting allows you to add or remove VLAN tags in the packets that are transferred between two customer sites within a service provider network.

These rewrites are supported on a Network Network Interface (NNI):

- Non range on C-Tag on an NNI
- Range on C-Tag on an NNI

Supported Rewrites for Non Range on a C-Tag on an NNI

When a custom ethertype is configured within the NNI physical interface, and VLAN range is not specified, the following rewrites are supported for a provider bridge:

- For **encapsulation untagged**:
  - No rewrite
  - Rewrite ingress tag push **dot1q** *vlan1* [*second-dot1q vlan2*] **symmetric**
- For **encapsulation default**:
  - No rewrite
- For **encapsulation dot1q vlan**:
  - No rewrite
  - Rewrite ingress tag pop 1 **symmetric**
  - Rewrite ingress tag translates 1-to-1 **dot1q** *vlan* **symmetric**
  - Rewrite ingress tag translate 1-to-2 **dot1q** *vlan 1* **second-dot1q** *vlan 2* **symmetric**
- For **encapsulation dot1q** *vlan1* second-**dot1q** *vlan2*
  - No rewrite
  - Rewrite ingress tag pop 1 **symmetric**
  - Rewrite ingress tag pop 2 **symmetric**
  - Rewrite ingress tag translate 1-to-1 **dot1q** *vlan* **symmetric**

- Rewrite ingress tag translate 1-to-2 **dot1q** *vlan 1* second **dot1q** *vlan 2* **symmetric**
- Rewrite ingress tag translate 2-to-1 **dot1q** *vlan* **symmetric**
- Rewrite ingress tag translate 2-to-2 **dot1q** *vlan 1* **second-dot1q** *vlan 2* **symmetric**

Supported Rewrites for Range on C-Tag with an NNI

When a VLAN range is specified on the C-Tag, push Rewrites are not supported. These rewrites are supported for a VLAN range on C-Tag:

- For **encapsulation dot1q** *vlan1 – vlan2:*
  - No rewrite
- For **encapsulation dot1q** *vlan1* **second-dot1q** *vlan2 – vlan3*:
  - No rewrite
  - Rewrite ingress tag pop 1 *symmetric*
  - Rewrite ingress tag translate 1-to-1 **dot1q** *vlan* **symmetric**
  - Rewrite ingress tag translate 1-to-2 **dot1q** *vlan 1* **second-dot1q** *vlan 2* **symmetric**

Note To avoid hierarchical provider bridges during custom ethertype configuration, NNI interface does not support **ingress push** rewrite except for **encap untagged**.

Restrictions and Usage Guidelines

Follow these restrictions and usage guidelines while configuring custom ethertype:

- Cisco IOS Release 15.1(1)S supports custom ethertype on port-channels.
- Custom ethertype is configured within a physical interface and is applicable to all service instances and subinterfaces of the physical interface.
- Mixed custom ethertype is not supported. If you configure the ingress interface values as 8100/9100/9200/88a8, then traffic other than the corresponding ethertype in the outer tag is not supported. If a packet is received with outer-tag other than the configured custom ethertype, then it is treated as untagged. However, when a packet with outer tag 8100 is received, it is treated as tagged packet, irrespective of the configured custom ethertype.
- When a custom ethertype is configured on the egress port with connect/cross connect configuration, the egress port does not filter the VLAN, and mismatched VLANs are relayed through the egress port.
- If a custom ethertype is configured on the port-channel, the same ethertype is implicitly configured for all the other member interfaces.
- You cannot configure custom ethertype explicitly under a member interface of a port-channel.
- An interface configured with custom ethertype cannot be a part of port-channel.

This example shows ingress and egress port configuration:

```
Ingress configuration
Int Gi1/1
Dot1q tunnel ethertype 0x9100
service instance 1 ethernet
Encap dot1q [second-dot1q]
```

```
Egress configuration
Int Gi1/1
Dot1q tunneling ethertype 0x9200
service instance 1 eth
Encap untagged
```

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *{ethernet |tengigabit Ethernet}*
4. **dot1q tunneling ethertype** [*0x9100*| *0x9200*| *0x88A8*]
5. **service instance Ethernet** *id [Ethernet service name*]
6. [**no**] **encapsulation untagged, dot1q {**any | *vlan-id[vlan-id[vlan-id]*} second-dot1q {any | *vlan-id[vlan-id[vlan-id*]]}
7. **Rewrite ingress tag** {**push {dot1q** vlan-id | **dot1q** vlan-id **second-dot1q** vlan-id **dot1q** vlan-id} | **pop {1 | 2} | translate {1-to-1 {dot1q** vlan-id}| **2-to-1 dot1q** vlan-id }| **1-to-2 {dot1q** vlan-id **second-dot1q** vlan-id **dot1q** vlan-id} | **2-to-2 {dot1q** vlan-id **second-dot1q** vlan-id **dot1q** vlan-id**}} symmetric**
8. **exit**

DETAILED STEPS

	Command	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`** `slot/subslot/port[.subinterface-number]`
	`or`
	`interface tengigabitethernet` `slot/subslot/port[.subinterface-number]`
	Example:
	`Router# int Gigabit 1/0/0`
	`Router# int Gigabit slot/port ES20`
	`Router# interface Tengigabit slot/subslot/port`
		Specifies the Gigabit Ethernet or the 10-Gigabit Ethernet interface to configure.

	Command	Purpose
Step 4	**dot1q tunneling ethertype** [0x9100| 0x9200| 0x88A8] **Example:** Router# dot1q tunneling ethertype [0x9100 | 0x9200 | 0x88A8]	Configure Custom Ethertype as 9100, 9200, or 88A8 within the physical interface because all service instances on physical interface use the configured Ethertype.
Step 5	**service instance Ethernet** id [Ethernet service name] **Example:** Router# service instance Ethernet <id> [Ethernet service name]	Creates a service instance (an instance of an EVC) on an interface and sets the device into the config-if-srv submode.
Step 6	[no] encapsulation untagged, dot1q {any | “<vlan-id>[,<vlan-id>[-<vlan-id>]]”} second-dot1q {any | “<vlan-id>[,<vlan-id>[-<vlan-id>]]”} **Example:** Router# encap dot1q 100 second dot1q 200	Defines the matching criteria that maps the ingress dot1q, QinQ, or untagged frames on an interface for the appropriate service instance.
Step 7	**Rewrite ingress tag** {**push** {**dot1q** vlan-id | **dot1q** vlan-id **second-dot1q** vlan-id **dot1q** vlan-id} | **pop** {1 | 2} | **translate** {1-to-1 {**dot1q** vlan-id}| 2-to-1 **dot1q** vlan-id }| 1-to-2 {**dot1q** vlan-id **second-dot1q** vlan-id **dot1q** vlan-id} | 2-to-2 {**dot1qvlan-id second-dot1q** vlan-id **dot1q** vlan-id}} **symmetric** **Example:** Router(config-if-srv)# Rewrite ingress tag push dot1q 20	Specifies the Rewrite operation.
Step 8	exit **Example:** Router(config-if-srv)#exit	Exits the configuration mode.

Examples

Single Tag Encap with Connect with Custom Ethertype Configured

In this example, custom ethertype is configured on a *single tag encap* using the connect configuration:

```
Router#sh running-config int Gi1/0/0
//Building configuration...
interface GigabitEthernet1/0/0
 no ip address
 dot1q tunneling ethertype 0x9100
 no mls qos trust
 service instance 1 ethernet
  encapsulation dot1q 10
```

```
interface GigabitEthernet1/0/2
 no ip address
 dot1q tunneling ethertype 0x9100
 mls qos trust dscp
 service instance 1 ethernet
  encapsulation dot1q 10
Router)#connect LC1 GigabitEthernet 1/0/0 1 GigabitEthernet 1/0/2 1
```

Single Tag Encap with Bridge Domain

In this example, custom ethertype is configured on a *single tag encap* using bridge domain configuration:

```
interface GigabitEthernet1/0/0
 no ip address
 dot1q tunneling ethertype 0x9100
 no mls qos trust
 service instance 1 ethernet
  encapsulation dot1q 10
  bridge-domain 100

interface GigabitEthernet1/0/2
 no ip address
 dot1q tunneling ethertype 0x9100
 mls qos trust dscp
 service instance 1 ethernet
  encapsulation dot1q 10
  bridge-domain 100
```

Single Tag Encap with Cross Connect

In this example, custom ethertype is configured on a *single tag encap* with **cross connect** configuration:

```
interface GigabitEthernet1/0/0
 no ip address
 dot1q tunneling ethertype 0x9100
 no mls qos trust
 service instance 1 ethernet
  encapsulation dot1q 10
  xconnect 3.3.3.3 10 encapsulation mpls

interface GigabitEthernet1/0/1
 ip address 10.10.10.2 255.255.255.0
 no mls qos trust
 mpls label protocol ldp
 mpls ip
```

Custom Ethertype Support with Sub Interfaces

In this example, custom ethertype is configured on a subinterface. Custom ethertype is always configured within the main physical interface and **QinQ encap** is configured within the subinterface.

```
interface GigabitEthernet1/0/0
 no ip address
 dot1q tunneling ethertype 0x9100
 no mls qos trust
end
interface GigabitEthernet1/0/0.10
 encapsulation dot1Q 10 second-dot1q 20
 ip address 20.20.20.2 255.255.255.0
end
```

Verification

Use these commands to verify operations.

Command	Purpose
`Router#` **`show ethernet service instance`** `[`**`id`** *instance-id* `|` **`interface`** *interface-id* `|` **`interface`** *interface-id*`]` `[`**`detail`**`]`	Displays information about: • Specific EVCs if an EVC ID is specified. • All the EVCs on an interface, if an interface is specified. The **detail** option provides additional information about the EVC. This can be given on a route processor of a line card console to determine if the Custom Ethertype is configured within a physical port.

Configuring Flexible QinQ Mapping and Service Awareness on 7600-ESM-2X10GE and 7600-ESM-20X1GE

The Flexible QinQ Mapping and Service Awareness on 7600-ESM-2X10GE and 7600-ESM-20X1GE feature allows service providers to offer triple-play services, residential internet access from a DSLAM, and business Layer 2 and Layer 3 VPN by providing for termination of double-tagged dot1q frames onto a Layer 3 subinterface at the access node.

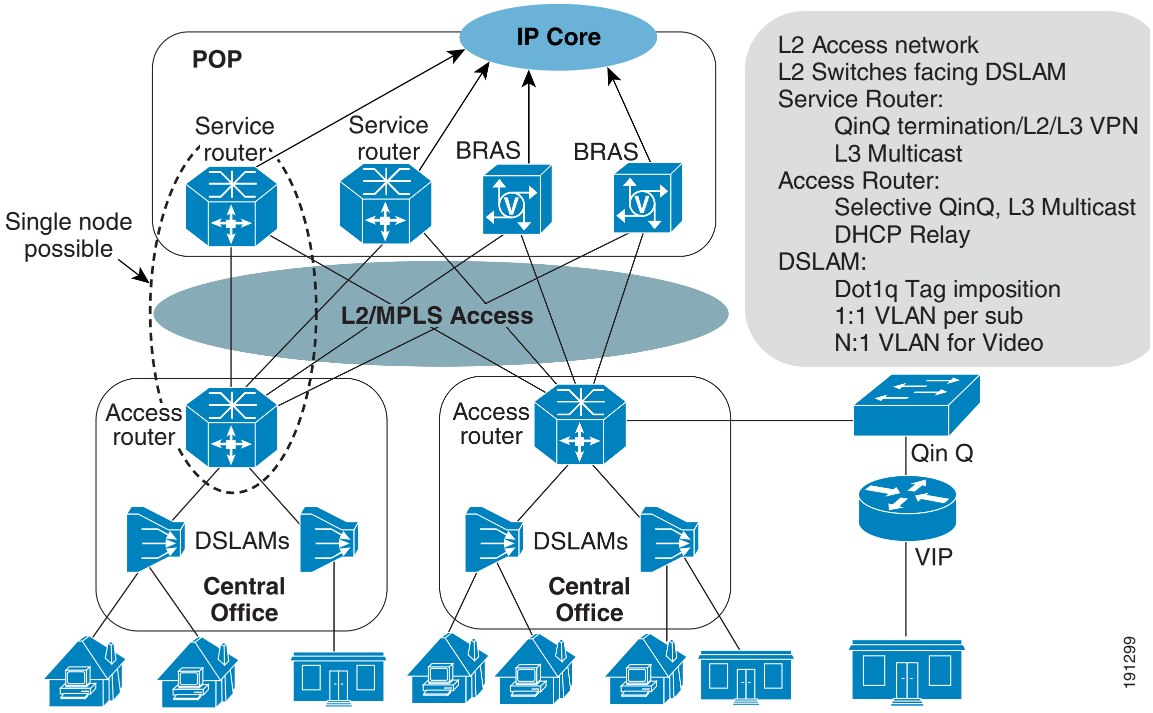
The access node connects to the DSLAM through the 7600-ESM-2X10GE or 7600-ESM-20X1GE. This provides a flexible way to identify the customer instance by its VLAN tags, and to map the customer instance to different services.

Flexible QinQ Mapping and Service Awareness on 7600-ESM-2X10GE and 7600-ESM-20X1GE is supported only through Ethernet Virtual Connection Services (EVCS) service instances.

EVCS uses the concepts of EVCs (Ethernet virtual circuits) and service instances. An EVC is an end-to-end representation of a single instance of a Layer 2 service being offered by a provider to a customer. It embodies the different parameters on which the service is being offered. A service instance is the instantiation of an EVC on a given port on a given router.

Figure 2-6 shows a typical metro architecture where the access switch facing the DSLAM provides VLAN translation (selective QinQ) and grooming functionality and where the service routers (SR) provide QinQ termination into a Layer 2 or Layer 3 service.

Figure 2-6



L2 Access network
L2 Switches facing DSLAM
Service Router:
QinQ termination/L2/L3 VPN
L3 Multicast
Access Router:
Selective QinQ, L3 Multicast
DHCP Relay
DSLAM:
Dot1q Tag imposition
1:1 VLAN per sub
N:1 VLAN for Video

Flexible QinQ Mapping and Service Awareness on 7600-ESM-2X10GE and 7600-ESM-20X1GE provides the following functionality:

- VLAN connect with local significance (VLAN local switching)
  - Single tag Ethernet local switching where the received dot1q tag traffic from one port is cross connected to another port by changing the tag. This is a 1-to-1 mapping service and there is no MAC learning involved.
  - Double tag Ethernet local switching where the received double tag traffic from one port is cross connected to another port by changing both tags. The mapping to each double tag combination to the cross-connect is 1-to-1. There is no MAC learning involved.
  - Hairpinning:is a cross connect between two EFPS on the same port.
- Selective QinQ (1-to-2 translation)
  - Cross connect—Selective QinQ adds an outer tag to the received dot1q traffic and then tunnels it to the remote end with Layer 2 switching or EoMPLS.
- Double tag translation (2-to-2 translation) Layer 2 switching— Two received tagged frames are popped and two new tags are pushed.
  - Cross connect—Selective QinQ adds an outer tag to the received dot1q traffic and then tunnels it to the remote end with Layer 2 switching or EoMPLS.
  - Layer 2 switching—Selective QinQ adds an outer tag to the received dot1q traffic and then performs Layer 2 switching to allow SVI based on the outer tag for configuring additional services.
- Double tag translation (2-to-2 translation) Layer 2 switching— Two received tagged frames are popped and two new tags are pushed.
- Double tag termination (2-to-1 tag translation)

  - Ethernet MultiPoint Bridging over Ethernet (MPBE)—The incoming double tag is uniquely mapped to a single dot1q tag that is then used to do MPBE
  - Double tag MPBE—The ingress line uses double tags in the ingress packet to look up the bridging VLAN. The double tags are popped and the egress line card adds new double tags and sends the packet out.
  - Double tag routing—Same as regular dot1q tag routing except that double tags are used to identify the hidden VLAN.
- Local VLAN significance—VLAN tags are significant only to the port.
- Scalable EoMPLS VC—Single tag packets are sent across the tunnel.
- QinQ policing and QoS
- Layer 2 protocol data unit (PDU) packet—If the Layer 2 PDUs are tagged, packets are forwarded transparently; if the Layer 2 PDUs are untagged, packets are treated per the physical port configuration.

Restrictions and Usage Guidelines

When configuring Flexible QinQ Mapping and Service Awareness on 7600-ESM-2X10GE and 7600-ESM-20X1GE, follow these restrictions and usage guidelines:

- Service Scalability:
  - Service Instances per port / NP: 8, 000
  - Service instances per Line Card: 16, 000
  - Service instances per port-channel: 8000. This is subject to the number of members per NP. This value would reduce by the factor of the member links per NP. If the members links are spread across various NPs, then the max number of service instances per port channel is unchanged.
  - TCAM Entry Usage: The number of TCAMs an EVC uses depends on the encapsulation configured on the TCAM as shown in the following examples.

Example 1

```
service instance 1 eth
encap dot1q 100
```

TCAMS used - 1

Example 2

```
service instance 1 eth
encap dot1q 200 second dot1q 300
```

TCAMs used - 1

Example 3

```
service instance 1 eth
encap dot1q 201, 202
```

TCAMs used - 2 (one for each encapsulation)

Example 4

```
service instance 1 eth
encap dot1q 20-40
```

TCAMs used - 4

```
First entry to match vlans 20-23
Second entry to match vlans 24-31
Third entry to match vlans 32-39
Fourth entry to match vlan 40
```

A range does not always mean multiple TCAMs as shown in the following example where one TCAM entry is used.

Example 5 -

```
service instance 1 ethernet
encap dot1q 8-15
service instance 2 ethernet
encap dot1q 2000 second-dot1q 96-127
```

TCAM usage per EVC : 1

  - Service instances per router: 32, 000
  - Bridge-domains per router: 4, 000
  - Local switching: 16, 000
  - Xconnect: 16, 000
  - Subinterface: 2, 000
  - Number of service instance on a particular domain: 110 per NP

- QoS Scalability:
  - Shaping - parent queue is 2,000 and child queue is 16,000
  - Marking - parent queue is 2,000 and child queue is 16,000
- Modular QoS CLI (MQC) actions supported include:
  - Shaping
  - Bandwidth
  - Two priority queues per policy
  - The **set cos** command, **set cos-inner** command, **set cos cos-inner** command, and **set cos-inner cos** command
  - WRED aggregate
  - Queue-limit

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot/subslot/port[.subinterface-number] or* ***interface tengigabitethernet*** *slot/subslot/port[.subinterface-number]*
4. [**no**] **service instance** *id* {**Ethernet** *service-name*}
5. **encapsulation dot1q** *vlan-id*
6. **rewrite ingress tag** {**push {dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {dot1q *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} [**symmetric**]

DETAILED STEPS

	Command	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** slot/subslot/port[.subinterface-number] or **interface tengigabitethernet** slot/subslot/port[.subinterface-number] **Example:** `Router(config)# interface gigabitethernet 4/0/0`	Specifies the Gigabit Ethernet or the Ten Gigabit Ethernet interface to configure, where: - *slot/subslot/port*—Specifies the location of the interface. - *subinterface-number*—(Optional) Specifies a secondary interface (subinterface) number.

Step 4	[**no**] **service instance** id {**Ethernet** [service-name}
	Example:
	`Router(config-if)# service instance 101 ethernet`
		Creates a service instance (an instantiation of an EVC) on an interface and sets the device into the config-if-srv submode.

Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Router(config-if-srv)# encapsulation dot1q 13`	Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.
Step 6	**rewrite ingress tag** {**push {dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {dot1q *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} [**symmetric**] **Example:** `Router(config-if-srv)# rewrite ingress tag push dot1q 20`	Specifies the tag manipulation that is to be performed on the frame ingress to the service instance.

Examples

Single Tag VLAN Connect

The following example shows a typical configuration of a DSLAM facing port of the first PE router.

```
! DSLAM facing port
Router(config)# interface TenGigabitEthernet1/0/1
Router(config-if)# service instance 100 ethernet
Router(config-if-srv)# encapsulation dot1q 10
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
!L2 facing port
Router(config)# interface TenGigabitEthernet1/0/2
Router(config-if)# service instance 101 ethernet
Router(config-if-srv)# encapsulation dot1q 11
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
! connect service
Router# connect EVC1 TenGigabitEthernet1/0/1 100 TenGigabitEthernet1/0/2 101
```

Double Tag VLAN Connect

In this example, an incoming frame with an outer dot1q tag of 10 and inner tag of 20 enters TenGigabitEthernet1/0/1. It is index directed to TenGigabitEthernet1/0/2 and exits with an outer dot1q tag of 11 and inner tag 21. No MAC learning is involved.

The following example shows a typical configuration of a MPLS core facting port of the first PE router.

```
! DSLAM facing port
Router(config)# interface TenGigabitEthernet1/0/1
Router(config-if)# service instance 100 ethernet
Router(config-if-srv)# encapsulation dot1q second-dot1q 20
Router(config-if-srv)# rewrite ingress tag pop 2 symmetric
!L2 facing port
Router(config)# interface TenGigabitEthernet1/0/2
Router(config-if)# service instance 101 ethernet
Router(config-if-srv)# encapsulation dot1q 11 second-dot1q 21
Router(config-if-srv)# rewrite ingress tag pop 2 symmetric
! connect service
Router# connect EVC1 TenGigabitEthernet1/0/1 100 TenGigabitEthernet1/0/2 101
```

Selective QinQ with Connect

This configuration uses EoMPLS to perform packet forwarding. This is index directed.

```
! DSLAM facing port - single tag packet from link
Router(config)# interface TenGigabitEthernet1/0/1
Router(config-if)# service instance 100 ethernet
Router(config-if-srv)# encapsulation dot1q 10-20,30,50-60
!L2/QinQ facing port double tag packets
Router(config)# interface TenGigabitEthernet1/0/2
Router(config-if)# service instance 101 ethernet
Router(config-if-srv)# encapsulation dot1q 11 second-dot1q any
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
! connecting service instances
! QinQ outer dot1q is 11
Router# connect EVC1 TenGigabitEthernet1/0/1 100 TenGigabitEthernet1/0/2 101
```

Selective QinQ with Xconnect

This configuration uses EoMPLS under single tag subinterface to perform packet forwarding. The following example shows a typical configuration of a MPLS core facing port of the second PE router.

DSLAM facing port

```
! DSLAM facing port
Router(config)# interface TenGigabitEthernet1/0/1
Router(config-if)# service instance 100 ethernet
Router(config-if-srv)# encapsulation dot1q 10-20,30,50-60
Router(config-if-srv)# xconnect 2.2.2.2 999 pw-class vlan-xconnect
!
Router(config)# interface Loopback1
Router(config-if)# ip address 1.1.1.1 255.255.255.255
```

MPLS core facing port

```
! MPLS core facing port
Router(config)# interface TenGigabitEthernet2/0/1
Router(config-if)# ip address 192.168.1.1 255.255.255.0
Router(config-if)# mpls ip
Router(config-if)# mpls label protocol ldp
!
Router(config)# interface Loopback1
Router(config-if)# ip address 2.2.2.2 255.255.255.255
```

CE facing EoMPLS configuration

```
Router(config)# interface TenGigabitEthernet1/0/2
Router(config-if)# service instance 1000
Router(config-if-srv)# encapsulation dot1q 1000 second-dot1q any
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
Router(config-if-srv)#xconnect 1.1.1.1 999 pw-class vlan-xconnect
```

Selective QinQ with Layer 2 Switching

This configuration uses Layer 2 Switching to perform packet forwarding. The forwarding mechanism is the same as MPB-E, only the rewrites for each service instance are different.

DSLAM facing port, single tag incoming

```
Router(config)# interface TenGigabitEthernet1/0/1
Router(config-if)# service instance 100 ethernet
Router(config-if-srv)# encapsulation dot1q 10-20
Router(config-subif)# bridge-domain 11
```

QinQ VLAN

```
Router(config)# interface VLAN11
!QinQ facing port
Router(config)# interface TenGigabitEthernet1/0/2
Router(config-if)# switchport
Router(config-if)# switchport mode trunk
Router(config-if)# switchport trunk vlan allow 11
```

Double Tag Translation (2-to-2 Tag Translation)

In this case, double-tagged frames are received on ingress. Both tags are popped and two new tags are pushed. The packet is then Layer 2 switched to the bridge-domain VLAN.

QinQ facing port

```
Router(config)# interface TenGigabitEthernet1/0/1
Router(config-if)# service instance 100 ethernet
Router(config-if-srv)# encapsulation dot1q 100 second-dot1q 10
Router(config-if-srv)# rewrite ingress tag translate 2-to-2 dot1q 200 second-dot1q 20 
second-dot1q 10
Router(config-subif)# bridge-domain 200
```

QinQ VLAN

```
Router(config)# interface VLAN200
!
Router(config)# interface TenGigabitEthernet1/0/2
Router(config-if)# service instance 101 ethernet
Router(config-if-srv)# encapsulation dot1q 200 second-dot1q 20
Router(config-subif)# bridge-domain 200
```

Double Tag Termination (2 to 1 Tag Translation)

This example falls under the Layer 2 switching case.

Double tag traffic

```
Router(config)# interface TenGigabitEthernet1/0/1
Router(config-if)# service instance 100 ethernet
Router(config-if-srv)# encapsulation dot1q 200 second-dot1q 20
Router(config-if-srv)# rewrite ingress tag pop 2 symmetric
Router(config-subif)# bridge-domain 10

!
Router(config)# interface TenGigabitEthernet1/0/2
Router(config-if)# service instance 101 ethernet
Router(config-if-srv)# encapsulation dot1q 10
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
Router(config-subif)# bridge-domain 10
!
Router(config)# interface TenGigabitEthernet1/0/3
Router(config-if)# service instance 101 ethernet
Router(config-if-srv)# encapsulation dot1q 30
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
Router(config-subif)# bridge-domain 10
```

Verification

Use the following commands to verify operation.

Command	Purpose
Router# **show ethernet service evc** [**id** *evc-id* | **interface** *interface-id*] [**detail**]	Displays information pertaining to a specific EVC if an EVC ID is specified, or pertaining to all EVCs on an interface if an interface is specified. The **detailed** option provides additional information on the EVC.
Router# **show ethernet service instance** [**id** *instance-id* **interface** *interface-id* | **interface** *interface-id*] [**detail**]	Displays information about one or more service instances: If a service instance ID and interface are specified, only data pertaining to that particular service instance is displayed. If only an interface ID is specified, displays data for all service instances s on the given interface.
Router# **show ethernet service interface** [*interface-id*] [**detail**]	Displays information in the Port Data Block (PDB).
Router# **show mpls l2 vc detail**	Displays detailed information related to the virtual connection (VC).
Router# **show mpls forwarding** (Output should have the label entry l2ckt)	Displays the contents of the Multiprotocol Label Switching (MPLS) Label Forwarding Information Base (LFIB).

Troubleshooting

Use these debug commands to troubleshoot Flexible QinQ feature.

Debug commands

Command	Purpose
[no] debug ethernet service evc [id <evc-id>]	Enables EVC debugging on the RP. If no EVC ID is specified, debugging is enabled for all EVCs on the system.
[no] debug ethernet service instance [id <instance-id> interface <interface-id> | interface <interface-id>]	Enables EFP debugging on the RP. If no options are specified, debugging for all EFPs is enabled. If an EFP ID and interface are specified, only those debug messages associatedwith the EFP are displayed as the output. If only an interface is specified, debug messages for all EFPs on that interface is displayed.
[no] debug ethernet service interface [<interface-id>]	Enables PDB debugging.
[no] debug ethernet service api	Enables debugging between Ethernet Services Infrastructure and its clients.
debug ethernet service oam-mgr	Enables OAM Manager debugging, to debug OAM inter-working.
[no] debug ethernet service error	Enables ethernet service error debugging.
[no] debug ethernet service all	Enables EI debugging messages for all PDBs, EVCs and EFPs

Table 2-11 provides the troubleshooting solutions for the Flexible mapping feature.

Table 2-11 Troubleshooting Flexible mapping feature

Problem	Solution
Erroneous TCAM entries.	Use the **show hw-module subslot** *subslot* **tcam** command to verify and the TCAM entries. Share the output with TAC for further investigation.
Incorrect virtual VLAN IDs on a QinQ subinterface.	Use the **test hw-mod subslot** *subslot* command to verify the virtual VLAN ID values on a QinQ subinterface. Share the output with TAC for further investigation.
Wrong interface configured and tag manipulation incorrectly programmed.	Use the command **show platform np interface** *detail* to verfiy the interface and tag details. Share the output with TAC for further investigation.
VLAN ID is incorrectly programmed	Use the command **show hw-module subslot** *subslot* **tcam all_entries vlan** to verify the VLAN ID details. Share the output with TAC for further investigation.
Inner, outer start/end VLANs incorrectly programmed.	Use the **show platform np efp** command to verify the VLAN details. Share the output with TAC for further investigation.
Erroneous TCAM entries on the platform	Use the **show plat soft qos tcamfeature** and **show plat soft qos tcamt** commands to verify the TCAM entries. Share the output with TAC for further investigation.

Configuring Flexible Service Mapping Based on CoS and Ethertype

The Flexible Serivce Mapping based on CoS and Etherytpe feature enhances the current capability of mapping packets to service instance. It uses CoS and Ethertypes to classify traffic into different service instances and consumes less number of VLANs on the module.

Prior to the implementation of this feature, three different VLANs were required to relay voice, data, and video services. This feature distinguishes an EVC based on the CoS value, and implements one EVC with CoS, and another EVC with inner CoS. You can also use the same VLANs for eight different EVCs (CoS values 0-7 = 8 values) saving the usage of VLANs . These EVCs are associated with bridge domain, cross connect (xconnect), and connect.

This feature extends the following capabilities to the current implementation of mapping the service instances:

- Match on a single CoS value (either inner CoS or outer CoS, but not both simultaneously, and applicable only for QinQ).
- Match on a range or list of CoS values when a single VLAN is specified in the encapsulation criteria in dot1q/QinQ EVCs.
- Match support for a single CoS value for a range or list of VLANs. Acceptable range of CoS value is 0-7.
- Match the following supported payload ethertypes
  - IPv4 (etype 0x0800)
  - IPv6 (etype 0x086dd)
  - **pppoe-all** (0x8863 and 0x8864)
- In the case of QinQ, inner VLAN can have a range when the outer VLAN is a single VLAN.
- Match on range or list of CoS values when both outer and inner VLANs are single.

- Match on ethertype is supported both in the case of a single VLAN or in QinQ.
- Supports **pppoe-all** command (matches both 0x8863 and 0x8864).
- Matching on **pppoe-session** and **pppoe-discovery** commands are individually not supported.

You can use CoS and ethertype to classify the traffic into various service instances to reduce the number of vlans.

Restrictions and Usage Guidelines

When configuring Flexible Service Mapping based on CoS and Ethertype, follow these restrictions and guidelines:

- This release supports pppoe-all (matches both pppoe-discovery and pppoe-session), and does not individually support pppoe-discovery and pppoe-session as ethertypes.
- This feature supports both Dot1Q and QinQ.
- Egress behavior implemented for mismatched CoS and Ethertype forwards the packet without re-write. There is no filtering on egress, based on the CoS or Layer 3 Ethertype. Even if CoS or Etherype mismatches, if egress VLAN information matches, the frames are forwarded.
- Neither pppoe-discovery or pppoe-session are supported individually as Ethertypes. Cisco IOS 12.2(33) SRE release supports pppoe-all.
- Service instances on port-channels are supported.
- You cannot match Etherype and CoS for the same service instance.
- In the case of QinQ, a service instance can match on either outer Cos or inner Cos, but not simultaneously for the same service instance.
- You cannot specify a range or list of outer VLANs in double-tag cases.
- MAC learning occurs with bridge-domain, but does not occur with xconnect and connect.
- Egress checking of VLAN matching does not occur with xconnect and local connect.

Summary Steps

1. enable
2. **configure terminal**
3. ***interface gigabitethernet*** *slot/subslot/port or* ***interface tengigabitethernet*** *slot/subslot/port or* ***interface port-channel*** *number*
4. [**no**] **shut**
5. **service instance** *id* {**Ethernet** [*service-name*]}
6. **encapsulation dot1q** *vlan-id* ***{cos | comma| hyphen| etype} or encapsulation dot1q*** *vlan-id* ***second-dot1q {any | vlan-id[****vlan-id[vlan-id]]****} or***
7. ***encapsulation dot1q*** *vlan-id* ***cos [0-7] or encapsulation dot1q*** *vlan-id* ***etype [IPv4|IPv6|pppoe-all] or***
8. ***encapsulation dot1Q vlan-id {****vlan id****} second-dot1q {{any | vlan-id[****vlan-id[vlan-id]****] } } cos {0-7} or***
9. *encapsulation dot1Q vlan-id {vlan id} second-dot1q {{any | vlan-id[vlan-id[vlan-id]] } } etype {etype string} or*
10. ***encapsulation dot1Q vlan-id {****vlan id****} cos [0-7] second-dot1q {{any | vlan-id[****vlan-id[vlan-id]****] } }***

DETAILED STEPS

	Command	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** *slot/port* or **interface tengigabitethernet** *slot/port* or **interface port-channel** *number* **Example:** `Router(config)# interface gigabitethernet4/0/0`	Specifies the Gigabit Ethernet or the Ten Gigabit Ethernet interface to configure, where: - *slot/port*—Specifies the location of the interface. - Creates the port-channel interface.
Step 4	[**no**] **shut** **Example:** Router(config-if)# **no** sh	Initiates the selected interface.

Step 5	**service instance *id* {Ethernet *service-name*}**
	Example:
	`R1(config-if)#serv inst 1 eth`
		Creates a service instance on the selected ethernet interface.

	Note The commands that follow are used for dot1q or QinQ configurations. Read the purpose of each command to determine which to use.	

Step 6	**encapsulation dot1q *vlan-id* {cos|comma|hyphen|etype}**
	Example:
	`R1(config-if-srv)#encap dot1q 100?`
		Defines the matching criteria to map dot1Q ingress frames on an interface to the appropriate service instance.The value of a VLAN ID is an integer in the range from 1 to 4094. Enter hyphens to separate the starting and ending VLAN IDS used to define a range of VLAN IDs. Available options are CoS and ethertype.
		Note If range is used on VLANS, then range on CoS cannot be used, and vice versa.

	or	

	encapsulation dot1q *vlan-id* second-dot1q {any | *vlan-id*[*vlan-id[-vlan-id*]]}
	Example:
	`Router(config-if-srv)# encapsulation dot1q 100 cos 2-5 second-dot1q 60`
		Defines the matching criteria to map Q-in-Q ingress frames on an interface to the appropriate service instance.

	Command	Purpose
or		
	encapsulation dot1q *vlan-id* cos [0-7] **Example:** `Router(config-if-srv)# encapsulation dot1q 100 cos 5-6`	Specifies the CoS value in the match criteria for the ingress frames on the service instance.
or		
	encapsulation dot1q *vlan-id* etype [IPv4 | IPv6 | pppoe-all] **Example:** `Router(config-if-srv)# encapsulation dot1q 100 etype ipv4`	Specifies the payload ethertype value in the match criteria for the ingress frames on the service instance.
		or

	encapsulation dot1Q *vlan id* second-dot1q {{any | vlan-id[vlan-id[vlan-id]}} cos {0-7}
	Example:
	`Router(config-if-srv)# encapsulation dot1q 100 second-dot1q 60 cos 5`
		Specifies the encapsulation for QinQ with inner CoS.

		or

	encapsulation dot1Q vlan-id {*vlan id*} second-dot1q {{any | vlan-id[vlan-id[vlan-id]}} etype {etype string}
	Example:
	`Router(config-if-srv)# encapsulation dot1q 100 second-dot1q 60 etype ipv6`
		Specifies the encapsulation for QinQ with ethertype.

		or

	encapsulation dot1Q vlan-id *{vlan id}* cos [0-7] second-dot1q {{any | vlan-id[vlan-id[vlan-id]} }
	Example:
	`Router(config-if-srv)# encapsulation dot1q 100 cos 2-5 second-dot1q 60`
		Specifies the encapsulation for QinQ with outer CoS.

Support Configurations

The following are the supported Ethertype and CoS configurations:

Supported payload ethertype configurations for a single tag:

```
Router(config)# interface gigabitethernet 1/1
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q vlan id etype ipv4
```

Supported payload Ethertype configurations for a double tag:

```
Router(config)# interface gigabitethernet 1/1
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q vlan id second-dot1q vlan id etype ipv4
```

Supported payload Ethertype configurations for a single tag with a single VLAN:

```
Router(config)# interface gigabitethernet 1/1
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 10 etype ipv4
Router(config-if-srv)# exit
Router(config-if)# service instance 2 ethernet
Router(config-if-srv)# encapsulation dot1q 10 etype ipv6
Router(config-if-srv)# exit
Router(config-if)# service instance 3 ethernet
Router(config-if-srv)# encapsulation dot1q 10 etype pppoe-all
```

Supported payload Ethertype configurations for a single tag with range of VLANs:

```
Router(config)# interface gigabitethernet 1/1
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 11-15 etype ipv4
Router(config-if-srv)# exit
Router(config-if)# service instance 2 ethernet
Router(config-if-srv)# encapsulation dot1q 11-15 etype ipv6
Router(config-if-srv)# exit
Router(config-if)# service instance 3 ethernet
Router(config-if-srv)# encapsulation dot1q 11-15 etype pppoe-all
```

Supported payload Ethertype configurations for a double tag with no range:

```
Router(config)# interface gigabitethernet 1/1
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 10 second-dot1q 1001 etype ipv4
Router(config-if-srv)# exit
Router(config-if)# service instance 2 ethernet
Router(config-if-srv)# encapsulation dot1q 10 second-dot1q 1001 etype ipv6
Router(config-if-srv)# exit
Router(config-if)# service instance 3 ethernet
Router(config-if-srv)# encapsulation dot1q 10 second-dot1q 1001 etype pppoe-all
```

Supported payload Ethertype configurations for double tag with range on inner VLANs:

```
Router(config)# interface gigabitethernet 1/1
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 10 second-dot1q 11-15 etype ipv4
Router(config-if-srv)# exit
Router(config-if-srv)# encapsulation dot1q 10 second-dot1q 11-15 etype ipv6
Router(config-if-srv)# exit
Router(config-if-srv)# encapsulation dot1q 10 second-dot1q 11-15 etype pppoe-all
```

Supported CoS configurations for a single tag:

```
Router(config)# interface gigabitethernet 1/1
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 100 cos 5
Router(config)# interface gigabitethernet 1/1
Router(config-if)# service instance 2 ethernet
Router(config-if-srv)# encapsulation dot1q 100 cos 6-7
Router(config)# interface gigabitethernet 1/1
Router(config-if)# service instance 3 ethernet
Router(config-if-srv)# encapsulation dot1q 100 cos 0-3
```

Supported CoS configurations for a double tag:

Inner Cos:

```
Router(config)# interface gigabitethernet 1/1
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 100 second-dot1q 60 cos 5
```

Outer CoS:

```
Router(config-if)# service instance 2 ethernet
Router(config-if-srv)# encapsulation dot1q 100 cos 5 second-dot1q 200
```

The following example displays EVCs with encap dot1q and CoS within a bridge domain:

```
R1# sh runn int gi 3/0/11
Building configuration...

Current configuration : 84 bytes
!
interface GigabitEthernet3/0/11
 no ip address
 shutdown
 mls qos trust dscp
end
R1# sh runn int gi 3/0/12
Building configuration...

Current configuration : 72 bytes
!
interface GigabitEthernet3/0/12
 no ip address
 no mls qos trust
end

R1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# int gi 3/0/11
R1(config-if)# no sh
R1(config-if)# serv inst 1 eth
R1(config-if-srv)# encap dot1q 100 ?
  ,             comma
  -             hyphen
  cos           cos Vlan
  etype         payload ethertype after Vlan Field
  second-dot1q  inner 802.1Q Virtual LAN or C-VLAN
  <cr>

R1(config-if-srv)# encap dot1q 100 cos ?
  <0-7>  cos values

R1(config-if-srv)# encap dot1q 100 cos 5
R1(config-if-srv)# bridge-domain 202
R1(config-if-srv)# int gi 3/0/12
R1(config-if)# no sh
R1(config-if)# serv inst 1 eth
R1(config-if-srv)# encap dot1q 100 cos 5
R1(config-if-srv)# bridge-domain 202
R1(config-if-srv)# end
R1# sh bridge-domain 202
Bridge-domain 202 (2 ports in all)
State: UP                    Mac learning: Enabled
    GigabitEthernet3/0/11 service instance 1
    GigabitEthernet3/0/12 service instance 1
```

The following example shows EVC with encap dot1q and etype ipv4 with bridge-domain:

```
R1(config)# int gi 3/0/11
```

```
R1(config-if)# serv inst 1 eth
R1(config-if-srv)# encap dot1q 100 etype ?
  ipv4       IPv4
  ipv6       IPv6
  pppoe-all  PPPoE ALL

R1(config-if-srv)# encap dot1q 100 etype ipv4
R1(config-if-srv)# bridge-domain 202
R1(config-if-srv)# int gi 3/0/12
R1(config-if)# serv inst 1 eth
R1(config-if-srv)# encap dot1q 100 etype ipv4
R1(config-if-srv)# bridge-domain 202
R1(config-if-srv)# end
R1# sh bridge-domain 202
Bridge-domain 202 (2 ports in all)
State: UP                    Mac learning: Enabled
    GigabitEthernet3/0/11 service instance 1
    GigabitEthernet3/0/12 service instance 1
```

Supported payload ether type configurations for a single tag:

```
    int g1/0/1
service instance 1 ethernet
encapsulation dot1q  100 etype ipv4
```

Supported payload ether type configurations for a double tag:

```
int g1/0/1
service instance 2 ethernet
encapsulation dot1q 100 second-dot1q 60 etype ipv6

service instance 3 ethernet
encapsulation dot1q 100 second-dot1q 60 etype ipv4

service instance 4 ethernet
encapsulation dot1q 100 second-dot1q 60 etype pppoe-all
```

Supported CoS configurations for a single tag:

```
int g1/0/1
service instance 1 ethernet
encapsulation dot1q 100 cos 5
```

SupportedCoS configurations for a double tag:

```
int g1/0/1
service instance 2 ethernet
encapsulation dot1q 100 second-dot1q 60 cos 5

service instance 3 ethernet
encapsulation dot1q 100 cos 6-7 second-dot1q 60
```

Supported CoS configurations for local connect:

```
Router(config)# interface TenGigabitEthernet2/3
Router(config-if)# no ip address
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 2 second-dot1q 2-3 cos 5
Router(config)# interface TenGigabitEthernet2/4
Router(config-if)# no ip address
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 2 second-dot1q 2-3 cos 5
Router(config-if-srv)# connect local1 te2/3 1 te2/4 1
```

Supported flexible service mapping configurations for cross connect:

```
Router 1(config)# interface TenGigabitEthernet2/3
Router 1(config-if)# no ip address
Router 1(config-if)# service instance 1 ethernet
Router 1(config-if-srv)# encapsulation dot1q 2 second-dot1q 2-3 cos 5
Router 1(config-if-srv)# xconnect 75.1.1.5 10000 encapsulation mpls
!
Router 1(config-if-srv)# end
```

The peer side router configuration is below:

```
Router 2(config)# interface GigabitEthernet3/0/14
Router 2(config-if)# no ip address
Router 2(config-if)# service instance 1 ethernet
Router 2(config-if-srv)# encapsulation dot1q 2 second-dot1q 2-3 cos 5
Router 2(config-if-srv)# xconnect 75.1.1.1 10000 encapsulation mpls
Router 2(config-if-srv)# end
```

Verification

Use the following commands to verify operation.

Command	Purpose
Router# **show ethernet service instance [detail | id** *id* **interface** *type number* [**detail | mac security [address | last violation | statistics] | platform | stats] | interface** *type number* [**detail | platform | stats | summary] | mac security [address | last violation | statistics] | platform | policy-map | stats | summary**]	Displays information about one or more service instances: If a service instance ID and interface are specified, only data pertaining to that particular service instance is displayed. If only an interface ID is specified, displays data for all service instances on the given interface.
Router# **show bridge-domain** [*bridge-id* **[mac security [address | last violation | statistics] | split-horizon [group** {*group-number* **| all | none**}**]] | stats]**	Displays the bridge domain information.

Sample output for the **show ethernet service instance** command:

```
Router# show ethernet service instance id 5 interface gigabitethernet3/1 detail
Service Instance ID: 5
Associated Interface: GigabitEthernet3/1
Associated EVC:
L2protocol drop
CE-Vlans:
Encapsulation: dot1q 11 vlan protocol type 0x8100 cos 3
Interface Dot1q Tunnel Ethertype: 0x8100
State: Up
EFP Statistics:
   Pkts In   Bytes In   Pkts Out  Bytes Out
   2253215  225321500    2248193  224819300
```

Sample output for the **show ethernet service instance stats** command:

```
Router 1# show ethernet service instance interface port-channel stats
Port maximum number of service instances: 8000
Service Instance 1, Interface Port-channel
    Pkts In   Bytes In   Pkts Out  Bytes Out
   2253215  225321500    2248193  224819300
Service Instance 2, Interface Port-channel
   Pkts In   Bytes In   Pkts Out  Bytes Out
   2253215  225321500    2248193  224819300
Service Instance 3, Interface Port-channel
    Pkts In   Bytes In   Pkts Out  Bytes Out
   2253215  225321500    2248193  224819300
Router 1# show ethernet service instance interface port-channel detail
```

```
Service Instance ID: 1
Associated Interface: Port-channel
Associated EVC:
Port-channel load-balance interface: None
L2protocol drop
CE-Vlans:
Encapsulation: dot1q 10 vlan protocol type 0x8100 second-dot1q 50 vlan protocol type 
0x8100 cos 5
Interface Dot1q Tunnel Ethertype: 0x8100
State: Up
EFP Statistics:
   Pkts In   Bytes In   Pkts Out  Bytes Out
   2253215  225321500    2248193  224819300
EFP Microblocks:
****************
Microblock type: Bridge-domain
Bridge-domain: 2301
Service Instance ID: 2
Associated Interface: Port-channel1
Associated EVC:
Port-channel load-balance interface: None
L2protocol drop
CE-Vlans:
Encapsulation: dot1q 101 vlan protocol type 0x8100 second-dot1q 205 vlan protocol type 
0x8100 payload etype pppoe-all
Interface Dot1q Tunnel Ethertype: 0x8100
State: Up
EFP Statistics:
    Pkts In   Bytes In   Pkts Out  Bytes Out
   2253215  225321500    2248193  224819300
EFP Microblocks:
****************
Microblock type: Bridge-domain
Bridge-domain: 2302
Service Instance ID: 3
Associated Interface: Port-channel
Associated EVC:
Port-channel load-balance interface: None
L2protocol drop
CE-Vlans:
Encapsulation: dot1q 5 vlan protocol type 0x8100 cos 6-7
Interface Dot1q Tunnel Ethertype: 0x8100
State: Up
EFP Statistics:
    Pkts In   Bytes In   Pkts Out  Bytes Out
   2253215  225321500    2248193  224819300
EFP Microblocks:
****************
Microblock type: Bridge-domain
Bridge-domain: 2303

Router 1# sh run int port-channel
Building configuration...

Current configuration : 361 bytes
!
interface Port-channel
 no ip address
 load-interval 30
 service instance 1 ethernet
  encapsulation dot1q 10 second-dot1q 50 cos 5
  bridge-domain 2301
 ! service instance 2 ethernet
  encapsulation dot1q 101 second-dot1q 205 etype pppoe-all
  bridge-domain 2302
 ! service instance 3 ethernet
  encapsulation dot1q 5 cos 6-7
  bridge-domain 2303
 !end
```

Configuring MultiPoint Bridging over Ethernet on 7600-ESM-2X10GE and 7600-ESM-20X1GE

The MultiPoint Bridging over Ethernet (MPBE) on 7600-ESM-2X10GE and 7600-ESM-20X1GE feature provides Ethernet LAN switching with MAC learning, local VLAN significance, and full QoS support. MPBE also provides Layer 2 switchport-like features without the full switchport implementation. MPBE is supported only through Ethernet Virtual Connection Services (EVCS) service instances.

EVCS uses the concepts of EVCs (Ethernet virtual circuits) and service instances. An EVC is an end-to-end representation of a single instance of a Layer 2 service being offered by a provider to a customer. It embodies the different parameters on which the service is being offered. A service instance is the instantiation of an EVC on a given port on a given router.

For MPBE, an EVC packet filtering capability prevents leaking of broadcast/multicast bridge-domain traffic packets from one service instance to another. Filtering occurs before and after the rewrite to ensure that the packet goes only to the intended service instance.

You can use MPBE to:

- Simultaneously configure Layer 2 and Layer 3 services such as Layer 2 VPN, Layer 3 VPN, and Layer 2 bridging on the same physical port.
- Define a broadcast domain in a system. Customer instances that are part of a broadcast domain can be in the same physical port or in different ports.
- Configure multiple service instances with different encapsulations and map them to a single bridge domain.
- Perform local switching between service instances on the same bridge domain.
- Span across different physical interfaces using service instances that are part of the same bridge domain.
- Use encapsulation VLANs as locally significant (physical port).
- Replicate flooded packets from the core to all service instances on the bridge domain.
- Configure a Layer 2 tunneling service or Layer 3 terminating service on the bridge domain VLAN.

MPBE accomplishes this by manipulating VLAN tags for each service instance and mapping the manipulated VLAN tags to Layer 2 or Layer 3 services. Possible VLAN tag manipulations include:

- Single tag termination
- Single tag tunneling
- Single tag translation
- Double tag termination
- Double tag tunneling
- Double tag translation
- Selective QinQ translation

Restrictions and Usage Guidelines

When configuring the MultiPoint Bridging over Ethernet on 7600-ESM-2X10GE and 7600-ESM-20X1GE feature, follow these restrictions and usage guidelines:

- Each service instance is considered as a separate circuit on the bridge-domain.
- Encapsulation can be dot1q or QinQ packets.
- 60 MPB VCs per 10G Complex (or 120 MPB VCs per ES20 line card) are supported on one bridge-domain.
- IGMP snooping is supported with MPB VCs.
- Split Horizon is supported with MPB VCs.
- BPDU packets are either tunneled or dropped.
- For ingress policing, only the drop action and the accept action for the **police** command are supported. Marking is not supported as part of the policing.
- Ingress shaping is not supported.
- For ingress marking, supports **match vlan** command, **match vlan-inner** command, **match cos** command, **match cos-inner** command, **set cos** command, and **set cos-inner** command.
- For egress marking, **set cos** command and **set cos-inner** command are supported; **match inner-cos** command and **match inner-vlan** command are not supported.

Summary Steps

1. enable
2. **configure terminal**
3. ***interface gigabitethernet*** *slot/subslot/port[.subinterface-number] or* ***interface tengigabitethernet*** *slot/subslot/port[.subinterface-number]*
4. [**no**] **service instance** *id* {**Ethernet** [*service-name*}
5. **encapsulation dot1q** *vlan-id*
6. **rewrite ingress tag** {**push {dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {dot1q *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} [**symmetric**]
7. [no] **bridge-domain** *bridge-id*

DETAILED STEPS

	Command	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** slot/subslot/port[.subinterface-number] or **interface tengigabitethernet** slot/subslot/port[.subinterface-number] **Example:** `Router(config)# interface gigabitethernet4/0/0`	Specifies the Gigabit Ethernet or the Ten Gigabit Ethernet interface to configure, where: - *slot/subslot/port*—Specifies the location of the interface. - *subinterface-number*—(Optional) Specifies a secondary interface (subinterface) number.

Step 4	[**no**] **service instance** *id* {**Ethernet** [*service-name*}
	Example:
	`Router(config-if)# service instance 101 ethernet`
		Creates a service instance (an instantiation of an EVC) on an interface and sets the device into the config-if-srv submode.

Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Router(config-if-srv)# encapsulation dot1q 10`	Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.
Step 6	[**no**] **rewrite ingress tag** {**push** {**dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {dot1q vlan-id **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} [**symmetric**] **Example:** `Router(config-if-srv)# rewrite ingress tag push dot1q 200`	This command specifies the tag manipulation that is to be performed on the frame ingress to the service instance. **Note** If this command is not configured, then the frame is left intact on ingress (the service instance is equivalent to a trunk port).
Step 7	[no] **bridge-domain** *bridge-id* **Example:** `Router(config-subif)# bridge domain 12`	Binds the service instance to a bridge domain instance where *bridge-id* is the identifier for the bridge domain instance.

Examples

Single Tag Termination Example

In this example, the single tag termination unidentified customers based on a single VLAN tag and maps the single-VLAN tag to the bridge-domain.

```
Router(config)# interface TenGigabitEthernet1/2/0
Router(config-if)# service instance 10 ethernet
Router(config-if-srv)# encapsulation dot1q 10
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
Router(config-if-srv)# bridge domain 12
}
```

Single Tag Tunneling Example

In this single tag tunneling example, the incoming VLAN tag is not removed but continues with the packet.

```
Router(config)# interface TenGigabitEthernet1/2/0
Router(config-if)# service instance 10 ethernet
Router(config-if-srv)# encapsulation dot1q 10
Router(config-if-srv)# bridge-domain 200
```

Single Tag Translation Example

In this single-tag translation example, the incoming VLAN tag is removed and VLAN 200 is added to the packet.

```
Router(config)# interface TenGigabitEthernet3/0/0
Router(config-if)# service instance 10 ethernet
Router(config-if-srv)# encapsulation dot1q 10
Router(config-if-srv)# rewrite ingress tag translate 1-to-1 dot1q 200 symmetric
Router(config-if-srv)# bridge-domain 200
```

Double Tag Termination Configuration Example

In this double-tag termination example, the ingress receives double tags that identify the bridge VLAN; the double tags are stripped (terminated) from the packet.

```
Router(config)# interface TenGigabitEthernet2/0/0
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 10 inner 20
Router(config-if-srv)# rewrite ingress tag pop 2 symmetric
Router(config-if-srv)# bridge-domain 200
Router(config-if)# service instance 2
Router(config-if-srv)# encapsulation dot1q 40 inner 30
Router(config-if-srv)# rewrite ingress tag pop 2 symmetric
Router(config-if-srv)# bridge-domain 200
```

Double-Tag Translation Configuration Example

In this example, double tagged frames are received on ingress. Both tags are popped and two new tags are pushed. The packet is then Layer 2-switched to the bridge-domain VLAN.

```
Router(config)# interface TenGigabitEthernet1/0/0
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 10 second-dot1q 20
Router(config-if-srv)# rewrite ingress tag translate 2-to-2 dot1q 40 second-dot1q 30
symmetric
Router(config-if-srv)# bridge-domain 200
Router(config-if)# service instance 2 ethernet
Router(config-if-srv)# encapsulation dot1q 40 second-dot1q 30
Router(config-if-srv)# rewrite ingress tag translate 2-to-2 dot1q 10 second-dot1q 20
symmetric
Router(config-if-srv)# bridge-domain 200
```

Selective QinQ Configuration Example

In this example, a range of VLANs is configured and plugged into a single MPB VC.

```
Router(config)# interface TenGigabitEthernet1/0/0
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 10-20
Router(config-if-srv)# bridge-domain 200

Router(config)# interface TenGigabitEthernet2/0/0
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 10-30
Router(config-if-srv)# bridge-domain 200
```

Untagged Traffic Configuration Example

In this example, untagged traffic is bridged to the bridge domain and forwarded to the switchport trunk.

```
Router(config)# interface GigabitEthernet2/0/1
Router(config-if)# no ip address
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation untagged
Router(config-if-srv)# bridge-domain 11
Router(config)# interface TenGigabitEthernet1/0/0
Router(config-if)# switchport
Router(config-if)# switchport mode trunk
Router(config-if)# switchport allowed vlan 11
```

MPBE with Split Horizon Configuration Example

In this example, unknown unicast traffic is flooded on the bridge domain except for the interface from which the traffic originated.

```
Router(config)# interface GigabitEthernet2/0/0
Router(config-if)# no ip address
Router(config-if)# service instance 1000 ethernet
Router(config-if-srv)# encapsulation dot1q 100 second-dot1q 10-20
Router(config-if-srv)# bridge-domain 100 split-horizon
Router(config-if)# service instance 1001 ethernet
Router(config-if-srv)# encapsulation dot1q 101 second-dot1q 21-30
Router(config-if-srv)# bridge-domain 101 split-horizon
Router(config-if)# service instance 1010 ethernet
Router(config-if-srv)# encapsulation dot1q 100
Router(config-if-srv)# rewrite ingress tag symmetric translate 1-to-2 dot1q 10
second-dot1q 100 symmetric
Router(config-if-srv)# bridge-domain 10 split-horizon
Router(config-if)# mls qos trust dscp
```

In this example, service instances are configured on Ethernet interfaces and terminated on the bridge domain.

```
Router(config)# interface GigabitEthernet2/0/0
Router(config-if)# service instance 100 ethernet
Router(config-if-srv)# encapsulation dot1q 1000
Router(config-if-srv)# bridge-domain 10

Router(config)# interface GigabitEthernet1/0/0
Router(config-if)# switchport
Router(config-if)# switchport mode trunk
Router(config-if)# switchport trunk allowed vlan 10
```

In this example, VPLS is configured in the core with multiple bridge domains.

```
!
l2 vfi vpls10 manual
 vpn id 10
 neighbor 20.0.0.2 encapsulation mpls
!
l2 vfi vpls100 manual
 vpn id 100
 neighbor 20.0.0.2 encapsulation mpls
!
l2 vfi vpls11 manual
 vpn id 11
 neighbor 20.0.0.2 encapsulation mpls
!
interface Vlan100
 mtu 9216
 no ip address
 xconnect vfi vpls1
end
```

Verification

Use the following commands to verify operation.

.

Command	Purpose
Router# **show ethernet service evc** [**id** *evc-id* | **interface** *interface-id*] [**detail**]	Displays information pertaining to a specific EVC if an EVC ID is specified, or pertaining to all EVCs on an interface if an interface is specified. The **detailed** option provides additional information on the EVC.
Router# **show ethernet service instance** [**id** *instance-id* **interface** *interface-id* | **interface** *interface-id*] [**detail**]	Displays information about one or more service instances: If a service instance ID and interface are specified, only data pertaining to that particular service instance is displayed. If only an interface ID is specified, displays data for all service instances s on the given interface.
Router# **show ethernet service interface** [*interface-id*] [**detail**]	Displays information in the Port Data Block (PDB).
Router# **show mpls l2 vc detail**	Displays detailed information related to the virtual connection (VC).
Router# **show mpls forwarding** (Output should have the label entry l2ckt)	Displays the contents of the Multiprotocol Label Switching (MPLS) Label Forwarding Information Base (LFIB).

Configuring Gigabit Ethernet Link Aggregation with Advanced Load Balancing

When you configure an Ethernet Flow Point (EFP) within a port-channel interface, you can specify a primary and multiple backup member-links to use as the egress interface for that EFP when the interface state is set to UP and it is part of the port-channel group. When the preferred member link is not available (interface is state is set to DOWN or not part of the port-channel group), a backup member link is used to manually load balance the EFP traffic over the port-channel. For each port-channel member link, you can configure a unique link ID within an acceptable range of 1-16 within the channel group. For each of the member links, you can specify a list of Ethernet Virtual Connections (EVC)s in the member link to relay egress traffic.

If none of the backup links are available, or you have not configured the primary or the backup links, the router selects an egress interface for the specific EFP. The backup link is selected based on the order of the configured list of backup link IDs. The first backup link in the list is used if available or the next backup link in the list is used. This continues until an available backup link is found.

In bridge domains, ingress traffic can access any port with an EFP in the same bridge domain and port channel. In local switching (connect) and cross-connect (xconnect), ingress traffic is received at the EVC port specified in the connect or cross-connect configurations.

To associate an EFP, or a set of EFPs to an EFP Port channel member link, you should:

- Assign a link ID to the port-channel member link at the interface configuration level.
- Associate a list of EFPs to an egress member link in the port-channel interface configuration level.

Restriction and Usage Guidelines

Follow these restrictions and guidelines when you configure a Gigabit Ethernet Link Aggregation with Link Aggregation Control Protocol with Advanced Load Balancing:

- When you configure a link ID for a port-channel member link, and configure that member link as the preferred egress link for the same service instances in that port-channel and the traffic is redistributed based on the following scenarios:
  - Service instances configured to be relayed over the preferred egress member links are relayed over the preferred member link. This is an expected behavior.
- If you have not configured the preferred member link, traffic is not redistributed based on the following scenarios:
  - For example, if there are 8 member links in a port-channel, the port manager allocates the load share of the member links as follows:

    Member 1 - Load share bit 0, Member 2 - Load share bit 1,

    Member 3 - Load share bit 2, Member 4 - Load share bit 3,

    Member 5 - Load share bit 4, Member 6 - Load share bit 5,

    Member 6 - Load share bit 6, Member 7 - Load share bit 7.

When you configure Member 1 with link ID 2, the port manager allocates the load share bit of 2 to member 1. So, the new assignments are:

Member 1 - Load share bit 2, Member 3 - Load share bit 0 [Load share of the other members remain the same].

If the platform relays data over an egress link that has the load share bit 2, before the user has configured the link ID = 2 for Member 1, this EFP traffic is relayed over Member 3. After the user configuration happens, member 1 has the load share bit = 2, this traffic is relayed over member 1. The reverse also happens; traffic relayed over member 1 before the user configuration is relayed over member 3.

SUMMARY STEPS TO ASSIGN A LINK ID TO THE PORT- CHANNEL MEMBER LINK

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *type|slot|por*t
4. **channel-group {*group*} mode {active | on | passive} {link {*ID*}**
5. **exit**

DETAILED STEPS TO ASSIGN A LINK ID TO THE PORT- CHANNEL MEMBER LINK

	Command	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** type/slot/port **Example:** `Router(config)#interface gigabitethernet 4/0/0`	Specifies the Gigabit Ethernet interface to configure, where: • *slot/subslot/port*—Specifies the location of the interface.
Step 4	**channel-group *channel-group-number* mode {active | on | passive} link *id*** `Router(config-if)#channel-group 100 mode on link 1`	Assigns a link identifier to load balance the links in the channel group.
Step 5	**exit**	Exits the configuration mode.

SUMMARY STEPS TO CREATE EVCS AND RELAY EGRESS TRAFFIC THROUGH ITS MEMBER LINKS

1. **enable**
2. **configure terminal**
3. **interface PortChannel** *{ID}*
4. **PortChannel load-balance link** {*ID*}
5. **backup link** {*ID1*} [...[*ID8*]]
6. **service-instance** {*service instance list*}

7. **exit**
8. **enable**
9. **configure terminal**
10. **interface gigabitethernet** *type*| *slot*|*port*
11. **channel-group** {*group*} **mode** {*mode*} {**link** {*ID*}}

DETAILED STEPS TO CREATE EVCS AND RELAY EGRESS TRAFFIC THROUGH ITS MEMBER LINKS

	Command	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 PortChannel`**`{ID}`
	Example:
	`Router#(config)#interface PortChannel 100`
		Specifies the port-channel interface.

Step 4	**`PortChannel load-balance link`** `{ID}`
	Example:
	`Router(config-if)#port-channel load-balance link 1`
		Assigns the link ID used for egress load balancing.

Step 5	**`backup link`** `{ID1} [...[ID8]]`
	Example:
	`Router#(config-if-lb)#backup link 3`
		Configures the backup link ID of a member link.

Step 6	**`service-instance`** `{`***`service instance list`***`}`
	Example:
	`Router#(config-if-lb)#service-instance 1`
		Configures a list of ethernet service instances whose egress traffic over the member link identified by Step 4.

Step 7	`exit`
	Example:
	`Router#(config-if-lb)#exit`
		Exits from port-channel load-balance config mode.

	Command	Purpose
Step 8	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 9	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 10	**interface gigabitethernet** slot/subslot/port **Example:** `Router(config)#interface gigabitethernet 4/0/0`	Specifies the Gigabit Ethernet interface to configure, where: • *slot/subslot/port*—Specifies the location of the interface.

Step 11	**channel-group** {*group*} **mode** {*mode*} {**link**| **link** {*ID*}
	Example:
	`Router(config-if)#channel-group 100 mode on link 2`
		Associates the specified member link to the EVC.

Examples

The following example shows the assignment of a link identified to a port-channel member link and its running configuration.

```
On RP
=====
Router-EoM1#show runn int Gig 3/0/2
Building configuration...

Current configuration : 149 bytes
!
interface GigabitEthernet3/0/2
 ip arp inspection limit none
 no ip address
 loopback mac
 no mls qos trust
 channel-group 100 mode on link 2
end

Router-EoM1#show runn int Gig 3/0/3
Building configuration...

Current configuration : 119 bytes
!
interface GigabitEthernet3/0/3
 no ip address
 loopback mac
 no mls qos trust
 channel-group 100 mode on link 3
end
```

```
Router-EoM1#
Router-EoM1#show runn int PortChannel 100
Building configuration...

Current configuration : 350 bytes
!
interface PortChannel100
 no ip address
 PortChannel load-balance link 1
  service-instance 1
  backup link 3
 !
 PortChannel load-balance link 2
  service-instance 2
  backup link 3
 !
 service instance 1 ethernet
  encapsulation dot1q 10
  bridge-domain 100
 !
 service instance 2 ethernet
  encapsulation dot1q 11
  bridge-domain 100
 !
end

Router-EoM1#show etherchannel summary
Flags:  D - down        P - bundled in PortChannel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 2
Number of aggregators:           2

Group  PortChannel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(RD)          -
100    Po100(RU)        -        Gi3/0/1(P)  Gi3/0/2(P)  Gi3/0/3(P)
```

The following example shows the creation of EVCS and relaying the egress traffic through its member links.

```
Router-EoM1#show ethernet service instance load-balance
Manually Assigned Load-Balancing Status for PortChannel100

 Link ID 1: GigabitEthernet3/0/1 (Active)
  Backup: Link ID 3 GigabitEthernet3/0/3
  Service instances: 1

 Link ID 2: GigabitEthernet3/0/2 (Active)
  Backup: Link ID 3 GigabitEthernet3/0/3
  Service instances: 2
Port maximum number of service instances: 8000
Service Instance 1, Interface PortChannel100
   Pkts In   Bytes In   Pkts Out  Bytes Out
         0          0          0          0
```

```
Service Instance 2, Interface PortChannel100
   Pkts In   Bytes In   Pkts Out  Bytes Out
         0          0          0          0

Router-EoM1#show ethernet service instance interface PortChannel 100 lo
Router-EoM1#$et service instance interface PortChannel 100 load-balance
Manually Assigned Load-Balancing Status for PortChannel100

 Link ID 1: GigabitEthernet3/0/1 (Active)
  Backup: Link ID 3 GigabitEthernet3/0/3
  Service instances: 1

 Link ID 2: GigabitEthernet3/0/2 (Active)
  Backup: Link ID 3 GigabitEthernet3/0/3
  Service instances: 2

Router-EoM1#
Router-EoM1#$et service instance interface PortChannel 100 summ
Router-EoM1#$et service instance interface PortChannel 100 summary
Associated interface: PortChannel100
            Total       Up  AdminDo     Down  ErrorDi  Unknown  Deleted  BdAdmDo
bdomain         2        2        0        0        0        0        0        0
xconnect        0        0        0        0        0        0        0        0
local sw        0        0        0        0        0        0        0        0
other           0        0        0        0        0        0        0        0
Router-EoM1#
Router-EoM1#
Router-EoM1#show etherchannel ?
  <1-564>       Channel group number
  detail        Detail information
  load-balance  Load-balance/frame-distribution scheme among ports in
                PortChannel
  port          Port information
  PortChannel  PortChannel information
  protocol      protocol enabled
  summary       One-line summary per channel-group
  |             Output modifiers
  <cr>

Router-EoM1#show etherchannel Por
Router-EoM1#show etherchannel Port-c
Router-EoM1#show etherchannel PortChannel 100 ?
% Unrecognized command
Router-EoM1#show etherchannel Port-channe
Router-EoM1#show etherchannel PortChannel ?
  |  Output modifiers
  <cr>

Router-EoM1#show etherchannel PortChannel
                Channel-group listing:

On LC
======
ESM-20G-3#show platform interface PortChannel 100 efp all
index 0x10000001        if_number 38            efp ID 1
service: bridging
configured member ports:
slot 3 port 2 pseudo_slotunit 140
slot 3 port 3 pseudo_slotunit 138
slot 3 port 4 pseudo_slotunit 136
egress interface: Gi3/0/1
ppe [0]: index 3

index 0x10000002        if_number 38            efp ID 2
```

```
service: bridging
configured member ports:
slot 3 port 2 pseudo_slotunit 141
slot 3 port 3 pseudo_slotunit 139
slot 3 port 4 pseudo_slotunit 137
egress interface: Gi3/0/2
ppe [0]: index 4

 Number of entries: 2
ESM-20G-3#
```

Verification

Use the following commands to verify operation.

Table 2-12 Commands for Displaying Traffic Storm Control Status and Configuration

Command	Purpose
`Router#` **`show ethernet service instance interface`** *`interface`* **`load-balance`**	Displays the current egress memberlink assignments for service instances configured with port-channel load-balancing.
`Router#` **`show ethernet service instance id`** *`<efp>`* **`interface port-channel`** *`<group>`* **`detail`**	Displays detailed status for the specified service instance, including the egress memberlink assignment, if any.

Troubleshooting Load Balancing Features

Table Table 2-13 provides troubleshooting solutions for the LoadBalancing features.

Table 2-13 Troubleshooting Scenarios for Load Balancing features

Problem	Solution
Link group creation command is rejected with an error message “Incomplete command".	Re-configure the link group with the specific link ID and these keywords: • **port-channel load-balance link:**<< Missing link ID>> • **no port-channel load-balance link**: << Missing link ID>> • **default port-channel load-balance link:**<< Missing link ID • **port-channel load-balanc**e:<< Missing 'link' keyword • **port-channel:** << Missing 'load-balance' keyword>>
Error message “Invalid input detected".	Re-configure the link group with valid IDs.

Problem	Solution
Back up link command is rejected and an error message displayed	Ensure that: • The back up link ID does not overlap with the primary link ID. • You have not exceeded the permissible number of back up links. • You have not entered a sub-mode command in a deleted load-balance group.
Invalid input	1. Execute the **show run** command to confirm if duplicate back up link IDs exists between two link groups. 2. Ensure that the configured EFPs have valid IDs. 3. Ensure that you have not configured an existing EFP ID in a different link group.
Member link is disabled	Use the **show etherchannel port-channel** command to verify the load share of each member link. Study the derived output and share the information with TAC for further investigation.
Traffic is not dsitributed equally among all members (Port channel load balancing issue)	Use the **show ethernet service instance interface** *port-channel* **load-balance** command to verify the load balancing information for all the port channels. Share the output with TAC for further investigation.
Traffic is not dsitributed equally among all members (EFP load balancing issues)	Use the **show ethernet service instance id** *efp* interface **port-channel** *group detail* command to verify and display the the load balancing information for the EFPs. Share the output with TAC for further investigation.

Configuring Virtual Private LAN Service (VPLS) with Port-Channel as a Core Interface

Virtual Private LAN Service interconnects geographically disparate LAN segments as a single bridged domain over a packet switched network, such as MPLS Core.

The current Cisco IOS L2VPN implementation builds a point-to-point connection to interconnect the VCs of peer customer sites. To communicate directly among all the L2VPN sites, a distinct emulated VC is created between each pair of peer VCs. For instance, when two sites of the same L2VPN is connected to the same Provider Edge (PE) router, two separate emulated VCs are mapped to the remote site instead of sharing a common emulated VC.

For an L2VPN customer who uses the service provider backbone to interconnect its LAN segments, the multi-access broadcast network is transformed into a fully meshed point-to-point network. This requires extensive reconfiguration on the existing Customer Edge (CE) devices.

In a VPLS deployment model, the service provider backbone network acts as a logical bridge. The topology and signaling of the backbone is transparent to the interconnected LAN segments.

You can use this feature to:

- Configure VPLS/H-VPLS on the port-channel interfaces of the ES20 line card as a core facing interface to provide port-channel member link redundancy and load balancing. The load-balancing is per-flow based, i.e. traffic of a VPLS VC is loadbalanced across member links based on the flow. For more information on the advanced load balancing options, see "Fat Pseudo-Wire Load Balancing" section on page 2-142 and "Provider Router Load Balancing" section on page 2-143.
- Match the capabilities and requirements of the VPLS in a single link. Due to multiple links in a Link Aggregation Group, the packets of a particular flow are always transmitted only to one link.
- Configure VPLS with port-channel interfaces as the core facing interface, where the member links of the port-channel are from a ES20 or a ES40 line card.

TE-FRR Support on VPLS LAG NNI

In an MPLS environment, traffic engineering (TE) provides a fast protection mechanism for link and node failures using fast reroute (FRR). On the Cisco 7600 series router, TE/FRR across port channel bundles is supported using Bidirectional Forwarding Detection (BFD), Reservation Protocol (RSVP) fast hello packets, min-link or max-bundle configuration. The default interval for hello packets is 200 milliseconds. It takes three hello packets (600 milliseconds) to detect the downtime of a bundle.

The Link Aggregation Control Protocol (LACP) fast switchover with fast link detection, takes about 200 to 600 milliseconds from the time a link has failed to the time the line card has processed the membership change request. TE/FRR measurements are highly dependent on LACP convergence, RSVP fast hello intervals, and LTL programming.

Traffic engineering fast reroute (TE-FRR) for VPLS over port-channel (PoCH) is supported in Cisco IOS Release 15.0(1)S.

For more information on MPLS TE- FRR, see the *MPLS Traffic Engineering (TE) - Fast Reroute (FRR) Link and Node Protection* feature guide at the following url:

http://www.cisco.com/en/US/docs/ios/mpls/configuration/guide/mp_te_frr_node_prot.html

Load Balancing

Load balancing ensures that the packets reach their destination in an orderly fashion. You can use the port-channel as the core facing interface in the PE routers to load balance per flow for the VCs mapped to this port-channel. To maintain the load balancing at the PE routers, you can use Fat Pseudo-Wire Load Balancing, page 2-142 and Provider Router Load Balancing, page 2-143.

Fat Pseudo-Wire Load Balancing

Fat pseudo-wire load balancing balances the VPLS VC traffic across the core network. An additional load balance label is inserted along with the VPLS VC labels such as VC label, and IGP label at the PE side. The remote end PE removes the load-balance label on the packet. For a single VC, the load-balance label is calculated based on the flow information of a VC.

At the core router, you can use the fat pseudo-wire to perform the following load balancing types:

- Equal Cost Multi-Path (ECMP): In a core network, multiple ECMP paths are used to reach the remote PE. Application of the load-balance label balances the traffic load across the multiple paths. This is because the load-balance label is different for different flows of a VC,and the hash algorithm using the mpls label for load-balancing generates a different hash to distribute the traffic.
- Port-channel: In a core network, if the selected path is a port-channel, the member links are load balanced due to modifications in the load balance label.

You can use the **platform vfi load-balance-label vlan** [*vlan*|*vlan-vlan*] command to configure the Fat pseudo-wire load balancing per vlan on a PE router. This is irrespective of the core facing interface being a port-channel or a non port-channel.

Provider Router Load Balancing

This is not supported in the ES20 line card. For more information about its support on a ES40 line card, see *Cisco 7600 Series Ethernet Services + Line Card Configuration Guide* located at the following URL:

http://www.cisco.com/en/US/docs/routers/7600/install_config/ES40_config_guide/es40_sw_config.html

BPDU PW Over LAG NNI

BPDU PW can be provisioned over a port channel interface. Provisioning BPDU PW on a port channel enables you to benefit from the link redundancy provided by LAG NNI. The redundancy helps pseudowire to remain always UP.

Effective from Cisco IOS Release 15.1(2)S, this feature is supported on the Cisco 7600 series routers. For configuration information, see Configuring BPDU PW on a Port Channel, page 2-145.

Restrictions and Usage Guidelines

Follow these restrictions and guidelines to configure H-Virtual Private LAN Service (VPLS) within a port-channel core interface:

- Provider Edge (PE) router LAG is supported on the ES-20 line card, for VPLS imposition or disposition functions.
- Provider router load balancing is not supported on the ES20 line card.
- A highly scaled VPLS setup or a highly scaled multicast configuration over VPLS on port-channel interfaces, can impact LACP fast switchover convergence.
- Existing port-channel features are supported.
- A maximum of six VPLS port-channel core interfaces are supported in the core router.
- QoS is not supported on ES20 port-channel interfaces, member-links, and port-channel subinterfaces.
- When a fat pseudowire (P/W) is configured, the core facing interface should be from a ES20 or a ES40 line card.
- A fat P/W is supported through a configurable command and should be uniformly enabled across all the peer PE routers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **platform vfi load-balance-label vlan** [*vlan*|*vlan-vlan*]

   or

   port-channel load-balance src-dst-mixed-ip-port

   or

[no] **port-channel load-balance mpls**

or

[no] **platform mpls load-balance ingress-port**

4. **exit**

DETAILED STEPS

	Command	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 the global configuration mode.

Step 3	**`platform vfi load-balance-label vlan`** [*`vlan`*|*`vlan-vlan`*] **Example:** `Router(config)# platform vfi load-balance-label vlan 5`	Configures fat pseudowire load balance label.
or		

	`[no] port-channel load-balance src-dst-mixed-ip-port`
	Example:
	`Router(config)# port-channel load-balance src-dst-mixed-ip-port`
		Configures port channel load balancing.
		The **src-dst-mixed-ip-port** mode allows load balance of IPV4 packets by source and destination MAC address, source and destinationIP address and TCP/UDP port number.

or		

	`[no] port-channel load-balance mpls [label|label-ip]`
	Example:
	`Router(config)# Router(config)# port-channel load-balance mpls label`
		Configures port channel load balancing. The **mpls** mode uses the MPLS label or IP address during load balancing. Load-balance label balances the traffic across the multiple paths

or		

	`[no] platform mpls load-balance ingress-port`
	Example:
	`Router(config)# platform mpls load-balance ingress-port`
		Configures ingress port-based load balancing on the P-router. Use the **no** form of the command to disable the configuration.

Step 4	`exit`	Exits from the configuration mode.

Verification

You can execute the **show running-config | include load-balance** command to confirm if the load balance label is applied, and the fat pseudowire is enabled on the router.

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# end
Router# show running-config | include load-balance
platform vfi load-balance-label vlan 6-100
port-channel load-balance src-dst-mixed-ip-port
```

Configuring BPDU PW on a Port Channel

Configure BPDU PW on a port channel between two PEs. Before you begin, you need to configure a VFI on a remote peer enabling BPDU PW on it. Complete the following steps:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2 vfi** *name* **manual**
4. **vpn id** *id-number*
5. **forward permit l2protocol all**
6. **neighbor** *remote-router-id vc-id* {**encapsulation** *encapsulation-type* | **pw-class** *pw-name*} [**no-split-horizon**]
7. **end**

DETAILED STEPS

	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	**l2 vfi** *name* **manual** **Example:** `Router(config)# l2 vfi vfi10 manual`	Creates a layer 2 VFI and enters layer 2 VFI manual configuration mode.

Step 4	**vpn id** *id-number* **Example:** `Router(config-vfi)# vpn id 110`	Specifies the VPN ID.
Step 5	**forward permit l2protocol all** **Example:** `Router(config-vfi)# forward permit l2protocol all`	Creates a pseudowire that is to be used to transport BPDU data between the two N-PE routers.
Step 6	**neighbor** *remote-router-id vc-id* {**encapsulation** *encapsulation-type* | **pw-class** *pw-name*} [**no-split-horizon]** **Example:** `Router(config-vfi)# neighbor 10.10.10.2 encapsulation mpls`	Specifies the peer IP address of the redundant N-PE router and the type of tunnel signaling and encapsulation mechanism. Valid encapsulation types are L2TPv3 and MPLS.

Step 7	**end**
	Example:
	`Router(config-vfi)# end`
		Ends the current configuration session and returns to privileged EXEC mode.

This example shows the enabling of BPDU PW on a remote peer:

```
Router> enable
Router# configure terminal
Router(config)# l2 vfi vfi10 manual
Router(config-vfi)# vpn id 110
Router(config-vfi)# forward permit l2protocol all
Router(config-vfi)# neighbor 10.10.10.2 encapsulation mpls
Router(config-vfi)# end
```

Configuring the MPLS Enabled Port Channel

Once you configure the BPDU PW on a peer, configure the MPLS enabled port channel towards the core:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *channel-number*
4. **ip address** *ip-address*
5. **mpls ip**
6. **mls qos trust dscp**
7. **end**

DETAILED STEPS

	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	Creates the EtherChannel (or port channel) virtual interface.
Step 4	**channel-group** *port-channel-number* **mode on** **Example:** Router(config)# channel-group 1 mode on	Assign a Fast Ethernet interface to an EtherChannel group. All possible modes such as, pagp,lacp, and none are valid here.
Step 5	**ip address** *ip-address subnet-mask* **Example:** Router(config-if)# ip address 100.0.0.1 255.255.255.0	Assigns the protocol IP address and subnet mask to the interface.
Step 6	**mpls ip** **Example:** Router(config-if)# mpls ip	Enables MPLS forwarding of IPv4 packets along normally routed paths for the associated interface.
Step 7	**mls qos trust dscp** **Example:** Router(config-if)# mls qos trust dscp	Classifies incoming packets that have packet DSCP values (the most significant 6 bits of the 8-bit service-type field).
Step 8	**end** **Example:** Router(config-if)# end	Ends the current configuration session and returns to privileged EXEC mode.

This example shows the configuration of an MPLS enabled port channel:

```
Router> enable
Router# configure terminal
Router(config)# interface Port-channel 1
Router(config)# channel-group 1 mode on
Router(config-if)# ip address 100.0.0.1 255.255.255.0
Router(config-if)# mpls ip
Router(config-if)# mls qos trust dscp
Router(config-if)# end
```

Binding the VFI to the VLAN

Bind the VFI to the VLAN you configured.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface vlan** *vlan-id*
4. **no ip address**
5. **xconnect vfi** *vfi name*
6. **end**

DETAILED STEPS

	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 vlan** *vlan-id* **Example:** `Router(config-if)# interface vlan 1`	Creates or accesses a dynamic switched virtual interface (SVI).
Step 4	**no ip address** **Example:** `Router(config-if)# no ip address`	Disables IP processing. (You configure a Layer 3 interface for the VLAN if you configure an IP address.)
Step 5	**xconnect vfi** *vfi name* **Example:** `Router(config-if)# xconnect vfi vfi10`	Specifies the Layer 2 VFI that you are binding to the VLAN port.

This example shows an interface VLAN configuration:

```
Router(config-if)# interface vlan 1
Router(config-if)# no ip address
Router(config-if)# xconnect vfi vfi10
```

This example shows how to use the **show vfi** command for VFI statu:

```
Router# show vfi vfi10
VFI name: vfi10, state: up
```

```
VPN ID: 100
Local attachment circuits:
  vlan 1
Neighbors connected via pseudowires:
  100.0.0.1 100.0.1.1 100.0.2.2 100.0.4.4 100.0.7.7
```

Troubleshooting

This section describes how to troubleshoot BPDU PW issues.

<table>
<tr><th>Scenarios/Problems</th><th>Solution</th></tr>
<tr><td>How to verify whether or not the BPDU PW status is in the UP.</td><td>

Use the **show mpls l2transport vc** command:

```
Router# show mpls l2transport vc 210
Local intf     Local circuit              Dest address    VC ID      Status
-------------  -------------------------- --------------- ---------- ----------
VFI 210        VFI                        10.144.144.144  210        UP
```

</td></tr>
<tr><td>How to verify whether or not a port-channel BPDU PW pseudoport is added to the MST tree.</td><td>

Use the **show spanning-tree mst** command:

```
Router# show spanning-tree mst

##### MST0    vlans mapped:   1-4094
Bridge        address 001a.3029.d400  priority      32768 (32768 sysid 0)
Root          address 0026.527c.5300  priority      24577 (24576 sysid 1)
              port    Gi5/3           path cost     200019
Regional Root this switch
Operational   hello time 2 , forward delay 15, max age 20, txholdcount 6
Configured    hello time 2 , forward delay 15, max age 20, max hops    20

Interface                        Role Sts Cost      Prio.Nbr Type
----------------                 ---- --- --------- -------- -------------------
-------------
Gi1/19                           Desg FWD 20000     128.19   P2p
Gi1/20                           Desg FWD 20000     128.20   P2p
Gi5/3                            Root FWD 200000    128.1027 P2p Bound(STP)
Gi6/3                            Altn BLK 200000    128.1283 P2p Bound(STP)
Gi8/0/3                          Desg FWD 20000     128.1796 P2p
Gi8/0/5                          Desg FWD 20000     128.1798 P2p Bound(STP)
Gi8/0/7                          Desg FWD 20000     128.1800 P2p
```

This example shows the detailed output:

```
Router# show spanning-tree mst detail
##### MST0    vlans mapped:   1-4094
Bridge        address 001a.3029.d400  priority      32768 (32768 sysid 0)
Root          address 0026.527c.5300  priority      24577 (24576 sysid 1)
              port    Gi5/3           path cost     200019
Regional Root this switch
Operational   hello time 2 , forward delay 15, max age 20, txholdcount 6
Configured    hello time 2 , forward delay 15, max age 20, max hops    20

GigabitEthernet1/19 of MST0 is designated forwarding
Port info             port id         128.19  priority    128  cost       20000
Designated root       address 0026.527c.5300  priority  24577  cost      200019
Design. regional root address 001a.3029.d400  priority  32768  cost           0
Designated bridge     address 001a.3029.d400  priority  32768  port id   128.19
Timers: message expires in 0 sec, forward delay 0, forward transitions 1
Bpdus sent 140561, received 0

.................................................
```

</td></tr>
</table>

Configuring the Backup Interface for Flexible UNI

The Backup Interface for Flexible UNI feature allows you to configure redundant user-to-network interface (UNI) connections for Ethernet interfaces, which provides redundancy for dual-homed devices.

You can configure redundant (flexible) UNIs on a network provider-edge (N-PE) device in order to supply flexible services through redundant user provider-edge (U-PE) devices. The UNIs on the N-PEs are designated as primary and backup and have identical configurations. If the primary interface fails, the service is automatically transferred to the backup interface.

The primary interface is the interface for which you configure a backup. During operation, the primary interface is active and the backup (secondary) interface operates in standby mode. If the primary interface goes down (due to loss of signal), the router begins using the backup interface.

While the primary interface is active (up) the backup interface is in standby mode. If the primary interface goes down, the backup interface transitions to the up state and the router begins using it in place of the primary. When the primary interface comes back up, the backup interface transitions back to standby mode. While in standby mode, the backup interface is effectively down and the router does not monitor its state or gather statistics for it.

This feature provides the following benefits:

- Supports the following Ethernet virtual circuit (EVC) features:
  - Frame matching: EVC with any supported encapsulation (Dot1q, default, untagged)
  - Frame rewrite: Any supported (ingress and egress with push, pop, and translate)
  - Frame forwarding: MultiPoint Bridging over Ethernet (MPB-E), xconnect, connect
  - Quality of Service (QoS) on EVC
- Supports Layer 3 (L3) termination and L3 Virtual Routing and Forwarding (VRF)
- Supports several types of uplinks: MultiProtocol Label Switching (MPLS), Virtual Private LAN Service (VPLS), and switchports

The Backup Interface for Flexible UNI feature makes use of these Ethernet components:

- Ethernet virtual circuit (EVC)—An association between two or more UNIs that identifies a point-to-point or point-to-multipoint path within the provider network. For more information about EVCs, see the "Configuring Flexible QinQ Mapping and Service Awareness on 7600-ESM-2X10GE and 7600-ESM-20X1GE" section on page 2-110.
- Ethernet flow point (EFP)—The logical demarcation point of an EVC on an interface. An EVC that uses two or more UNIs requires an EFP on the associated ingress interface and egress interface of every device that the EVC passes through.

Restriction and Usage Guidelines

Observe these restrictions and usage guidelines as you configure a backup interface for Flexible UNI on the router:

- Hardware and software support:
  - Supported on Cisco 7600 Series ES+ and ES20 line cards.
  - Supported with the Route Switch Processor 720, Supervisor Engine 720, and Supervisor Engine 32.
  - Requires Cisco IOS Release 12.2SRB1 or later.

- You can use the same IP address on both the primary and secondary interfaces. This enables the interface to support L3 termination (single or double tagged).
- The configurations on the primary and backup interfaces must match. The router does not check that the configurations match; however, the feature does not work if the configurations are not the same.

Note If the configuration includes the **xconnect** command, you must specify a different VCID on the primary and backup interfaces.

- The duplicate resources needed for the primary and secondary interfaces are taken from the total resources available on the router and thus affect available resources. For example, each **xconnect** consumes resources on both the primary and backup interfaces.
- Local switching (**connect**) between primary and backup interfaces uses twice the number of physical interfaces. This limitation is due to lack of support for local switching on EVCs on the same interface.
- Any features configured on the primary and backup interfaces (such as **bridge-domain**, **xconnect**, and **connect**) transition up or down as the interface itself transitions between states.
- Switchover time between primary and backup interfaces is best effort. The time it takes the backup interface to transition from standby to active mode depends on the link-state detection time and the amount of time needed for EVCs and their features to transition to the up state.
- Configuration changes and administrative actions made on the primary interface are automatically reflected on the backup interface.
- The router monitors and gathers statistics for the active interface only, not the backup. During normal operation, the primary interface is active; however, if the primary goes down, the backup becomes active and the router begins monitoring and gathering statistics for it.
- When the primary interface comes back up, the backup interface always transitions back to standby mode. Once the signal is restored on the primary interface, there is no way to prevent the interface from being restored as the primary.

Configuration Instructions

To configure a backup interface for a flexible UNI on an Ethernet port, perform the following steps:

	Command or Action	Purpose

Step 1	`Router(config)# `**`interface`** *`type slot/subslot/port`*
	Example:
	`Router(config)# interface gigabitethernet3/0/0`
		Selects the primary interface. This is the interface you are creating a backup interface for. For example, **interface gigabitEthernet 3/0/0** selects the interface for port 0 of the Gigabit Ethernet card installed in slot 3, subslot 0.
		• *type* specifies the interface type. Valid values are **gigabitethernet** or **tengigabitethernet**.
		• *slot/subslot/port* specifies the location of the interface.

Step 2	`Router(config-if)# `**`backup interface`** *`type interface`*
	Example:
	`Router(config)# backup interface gigabitethernet4/0/1`
		Selects the interface to serve as a backup interface.

	Command or Action	Purpose
Note	You must apply the same configuration to both the primary and backup interfaces or the feature does not work. To configure EVC service instances on the interfaces, use the **service instance**, **encapsulation**, **rewrite**, **bridge-domain**, and **xconnect** commands. For information, see the "Configuring Flexible Service Mapping Based on CoS and Ethertype" section on page 2-119 and the "Configuring Any Transport over MPLS" section on page 2-289.	
Step 3	`Router(config-if)#` **`backup delay`** *`enable-delay disable-delay`* **Example:** Router(config-if)# backup delay 0 0	(Optional) Specifies a time delay (in seconds) for enabling or disabling the backup interface. • *enable-delay* is the amount of time to wait after the primary interface goes down before bringing up the backup interface. • *disable-delay* is the amount of time to wait after the primary interface comes back up before restoring the backup interface to the standby (down) state **Note** For the backup interface for Flexible UNI feature, do not change the default delay period (0 0) or the feature may not work correctly.
Step 4	Router(config-if)# **backup load** *enable-percent disable-percent* **Example:** Router(config-if)# backup load 50 10	(Optional) Specifies the thresholds of traffic load on the primary interface (as a percentage of the total capacity) at which to enable and disable the backup interface. • *enable-percent*—Activate the backup interface when the traffic load on the primary exceeds this percentage of its total capacity. • *disable-percent*—Deactivate the backup interface when the combined load of both primary and backup returns to this percentage of the primary's capacity. Applying the settings from the example to a primary interface with 10-Mbyte capacity, the router enables the backup interface when traffic load on the primary exceeds 5 Mbytes (50%), and disables the backup when combined traffic on both interfaces falls below 1 Mbyte (10%).
Step 5	Router(config-if)# **exit**	Exits interface configuration mode and returns to global configuration mode.
Step 6	Router(config)# **connect primary** *interface srv-inst interface srv-inst* Router(config)# **connect backup** *interface srv-inst interface srv-inst* **Example:** Router(config-if)# connect primary gi3/0/0 2 gi3/0/1 2 Router(config-if)# connect backup gi4/0/0 2 gi4/0/1 2	(Optional) Creates a local connection between a single service instance (*srv-inst*) on two different interfaces. The **connect primary** command creates a connection between primary interfaces, and **connect backup** creates a connection between backup interfaces. In the example, a local connection is configured between service instance 2 on primary interfaces (gi3/0/0 and gi3/0/1) and on backup interfaces (gi4/0/0 and gi4/0/1).

	Command or Action	Purpose
Step 7	Router(config)# **connect primary** *interface srv-inst1 interface srv-inst2* Router(config)# **connect backup** *interface srv-inst1 interface srv-inst2* **Example:** Router(config-if)# connect primary gi3/0/0 2 gi3/0/0 3 Router(config-if)# connect backup gi4/0/0 2 gi4/0/0 3	(Optional) Enables local switching between different service instances (*srv-inst1* and *srv-inst2*) on the same port. Use the **connect primary** command to create a connection on a primary interface, and **connect backup** to create a connection on a backup interface. In the example, we are configuring local switching between service instances 2 and 3 on both the primary (gi3/0/0) and backup interfaces (gi4/0/0).
Step 8	Router(config-if)# **exit**	Exits interface configuration mode.

The following example shows a sample configuration in which:

- gi3/0/1 is the primary interface and gi4/0/1 is the backup interface.
- Each interface supports two service instances (2 and 4), and each service instance uses a different type of forwarding (**bridge-domain** and **xconnect**).
- The **xconnect** command for service instance 2 uses a different VCID on each interface.

```
int gi3/0/1
  backup interface gi4/0/1
  service instance 4 ethernet
    encapsulation dot1q 4
    rewrite ingress tag pop 1 symmetric
    bridge-domain 4
  service instance 2 ethernet
    encapsulation dot1q 2
    rewrite ingress tag pop 1 symmetric
    xconnect 10.0.0.0 2 encap mpls

int gi4/0/1

  service instance 4 ethernet
    encapsulation dot1q 4
    rewrite ingress tag pop 1 symmetric
    bridge-domain 4
  service instance 2 ethernet
    encapsulation dot1q 2
    rewrite ingress tag pop 1 symmetric
    xconnect 10.0.0.0 5 encap mpls
```

Verification

This section lists the commands to display information about the primary and backup interfaces configured on the router. In the examples that follow, the primary interface is gi3/0/0 and the secondary (backup) interface is gi3/0/11.

- To display a list of backup interfaces, use the **show backup** command in privileged EXEC mode. Our sample output shows a single backup (secondary) interface:

```
NPE-11# show backup
Primary Interface     Secondary Interface    Status
-----------------     -------------------    ------
GigabitEthernet3/0/0  GigabitEthernet3/0/11  normal operation
```

- To display information about a primary or backup interface, use the **show interfaces** command in privileged EXEC mode. Issue the command on the interface for which you want to display information. The following examples show the output displayed when the command is issued on the primary (gi3/0/0) and backup (gi3/0/11) interfaces:

```
NPE-11# sh int gi3/0/0
GigabitEthernet3/0/0 is up, line protocol is up (connected)
  Hardware is GigEther SPA, address is 0005.dc57.8800 (bia 0005.dc57.8800)
  Backup interface GigabitEthernet3/0/11, failure delay 0 sec, secondary disable delay
0 sec, kickin load not set, kickout load not set
[…]

NPE-11# sh int gi3/0/11
GigabitEthernet3/0/11 is standby mode, line protocol is down (disabled)
```

If the primary interface goes down, the backup (secondary) interface is transitioned to the up state, as shown in the command output that follows. Notice how the command output changes if you reissue the **show backup** and **show interfaces** commands at this time: the **show backup** status changes, the line protocol for gi3/0/0 is now down (not connected), and the line protocol for gi3/0/11 is now up (connected).

```
NPE-11# !!! Link gi3/0/0 (active) goes down…
22:11:11: %LINK-DFC3-3-UPDOWN: Interface GigabitEthernet3/0/0, changed state to down
22:11:12: %LINK-DFC3-3-UPDOWN: Interface GigabitEthernet3/0/11, changed state to up
22:11:12: %LINEPROTO-DFC3-5-UPDOWN: Line protocol on Interface GigabitEthernet3/0/0,
changed state to down
22:11:13: %LINEPROTO-DFC3-5-UPDOWN: Line protocol on Interface GigabitEthernet3/0/11,
changed state to up

NPE-11# sh backup
Primary Interface     Secondary Interface    Status
-----------------     -------------------    ------
GigabitEthernet3/0/0  GigabitEthernet3/0/11  backup mode

NPE-11# sh int gi3/0/0
GigabitEthernet3/0/0 is down, line protocol is down (notconnect)
  Hardware is GigEther SPA, address is 0005.dc57.8800 (bia 0005.dc57.8800)
  Backup interface GigabitEthernet3/0/11, failure delay 0 sec, secondary disable delay
0 sec,

NPE-11# sh int gi3/0/11
GigabitEthernet3/0/11 is up, line protocol is up (connected)
```

Example

The configuration includes several EVCs (service instances), configured as follows:

- Service instance EVC4 is configured on primary and backup interfaces (links) that terminate in a bridge domain, with a VPLS uplink onto NPE12.
- Service instance EVC2 is configured as scalable Ethernet over MPLS, peering with an SVI VPLS on NPE12.

NPE10 Configuration:

```
int ge2/4.4
  description npe10 to npe11 gi3/0/11 –
backup - bridged
  encap dot1q 4
  ip address 100.4.1.33 255.255.255.0

int ge2/4.2
  description npe10 to npe11 gi3/0/11 –
backup – xconnect
  encap dot1q 2
  ip address 100.2.1.33 255.255.255.0
```

NPE14 Configuration:

```
int ge1/3.4
  description npe14 to npe11 gi3/0/0 –
primary - bridged
  encap dot1q 4
  ip address 100.4.1.22 255.255.255.0

int ge1/3.2
  description npe14 to npe11 gi3/0/0 –
primary - xconnect
  encap dot1q 2
  ip address 100.2.1.22 255.255.255.0
```

72a Configuration:

```
int fa1/0.4
  description 72a to npe12 – bridged
  encap dot1q 4
  ip address 100.4.1.12 255.255.255.0

int fa1/0.2
  description 72a to npe12 - xconnect
  encap dot1q 2
  ip address 100.2.1.12 255.255.255.0
```

```
NPE11 Configuration:
interface gigabitEthernet3/0/0
  backup interface gigabitEthernet3/0/11
  service instance 2 ethernet
    encapsulation dot1q 2
    rewrite ingress tag pop 1 symmetric
    xconnect 12.0.0.1 2 encapsulation mpls
  service instance 4 ethernet
    encapsulation dot1q 4
    rewrite ingress tag pop 1 symmetric
    bridge-domain 4

interface gigabitEthernet3/0/11
  service instance 2 ethernet
    encapsulation dot1q 2
    rewrite ingress tag pop 1 symmetric
    xconnect 12.0.0.1 21 encapsulation
mpls
  service instance 4 ethernet
    encapsulation dot1q 4
    rewrite ingress tag pop 1 symmetric
    bridge-domain 4
```

NPE11 Core facing:

```
interface GE-WAN 4/3
  description npe11 to npe12
  ip address 10.3.3.1 255.255.255.0
  mpls ip
l2 vfi vlan4 manual
  vpn id 4
  neighbor 12.0.0.1 4 encapsulation mpls
interface Vlan 4
  xconnect vfi vlan4
```

NPE12 Core facing:
l2 vfi vlan4 manual
  vpn id 4
  neighbor 11.0.0.1 4 encap mpls
interface Vlan4
  description npe12 to npe11 xconnect
  xconnect vfi vlan4
l2 vfi vlan2 manual
  vpn id 2
  neighbor 11.0.0.1 2 encap mpls
  neighbor 11.0.0.1 21 encap mpls
Interface Vlan2
  xconnect vfi vlan2
interface GE-WAN 9/4
  description npe12 to npe11
  ip address 10.3.3.2 255.255.255.0
  mpls ip
	NPE12 facing the customer:
	interface fastEthernet 8/2
	  description npe12 to 72a
	  switchport
	  switchport trunk encap dot1q
	  switchport mode trunk
	  switchport trunk allowed vlan 2-4

Primary interface is enabled: NPE 11#sh backup Primary interface Secondary interface Status -- -- GigabitEthernet3/0/0GigabitEthernet3/0/11 normal operation NPE-11#sh int gi3/0/0 GigabitEthernet3/0/0 is up, line protocol is up (connected) Hardware is GigEther SPA, address is 0005.dc57.8800(bia 0005.dc57.8800) Backup interface GigabitEthernet3/0/11, failure delay 0 sec, secondary disable delay 0 sec,kicking load not set, kickout load not set, [...] NPE-11#sh int gi3/0/11 GigabitEthernet 3/0/11 is standby mode, line protocol is down (disabled)	Primary link is disabled: NPE 11#!!!Link gi3/0/0 (active) goes down 22:11:11: % LINK-DFC3-3-UPDOWN:Interface GigabitEthernet3/0/0, changed state to down 22:11:12: % LINK-DFC3-3-UPDOWN:Interface GigabitEthernet3/0/0, changed state to up 22:11:12: % LINKPROTO-DFC3-3-5-UPDOWN:Line protocol on Interface GigabitEthernet3/0/0, changed state to down 22:11:13: % LINKPROTO-DFC3-3-5-UPDOWN:Line protocol on Interface GigabitEthernet3/0/11, changed state to up NP-11#sh backup Primary interface Secondary interface Status -- -- GigabitEthernet3/0/0GigabitEthernet3/0/11 backup mode NP-11#sh int gi3/0/0 GigabitEthernet3/0/0 is down, line protocol is down (notconnect) Hardware is GigEther SPA, address is 0005.dc57.8800(bia 0005.dc57.8800) Backup interface GigabitEthernet3/0/11, failure delay 0 sec, secondary disable delay 0 sec NPE-11#sh int gi3/0/11 GigabitEthernet 3/0/11 is up, line protocol is up (connected)

Troubleshooting

Table 2-14 provides troubleshooting solutions for the backup interface of the Flexible UNI feature.

Table 2-14 Troubleshooting Scenarios for backup interface of the Flexible UNI feature

Problem	Solution

The backup interface is in a standby state or the line protocol is down	Use the **show interfaces** command on the specific interface in privileged EXEC mode to display interface and line protocol details. Share the output with TAC for further investigation.
	This sample output of the command displayed when the command on the primary (gi3/0/0) and backup (gi3/0/11) interfaces:
	`NPE-11# show int gi3/0/0`
	`GigabitEthernet3/0/0 is up, line protocol is up (connected)`
	`  Hardware is GigEther SPA, address is 0005.dc57.8800 (bia 0005.dc57.8800)`
	`  Backup interface GigabitEthernet3/0/11, failure delay 0 sec, secondary disable delay`
	`0 sec, kickin load not set, kickout load not set`
	`[...]`
	`NPE-11# show int gi3/0/11`
	`GigabitEthernet3/0/11 is standby mode, line protocol is down (disabled)`

Configuring Layer2 Access Control Lists (ACLs) on an EVC

ACLs (Access Control Lists) perform the following tasks:

- Apply security and QoS at the interface, sub-interface, and service levels.
- Filter the packets in a modular manner.

You can use a collection of sequential ACL rules to filter network traffic. Though the ACLs are applied on a network interface, you can use this feature to apply Layer 2 on different Ethernet Virtual Connections (EVCs). Table 2-15 maps the supported layers with their parameters.

Table 2-15 Mapping between the ACL supported layers to the parameters

Layer	Based on
Layer 2	• MAC Source and destination

Table 2-16 ACL commands

Layer	Action	Command
Layer 2	Create a L2 Access List	*mac access-list extendend {aclname}*
	Apply a Access list within the EVC	mac access- group aclname in

Restrictions and Usage Guidelines

Follow these restrictions and usage guidelines when you configure ACLs on a EVC:

- L2 ACL is supported only on ingress.
- You can apply a single ACL to more than one EFP.
- If a Layer 2 ACL is applied to an EFP with a Layer 2 ACL, the new ACL replaces the previous ACL.
- L2 ACL configuration applied on the EVC interface should contain the source MAC address, destination MAC address, and the address mask.
- You can apply a maximum of 256 unique ACLs on all the EVCs.
- Maximum number of 16 ACEs (Access Control Elements) per ACL are supported.
- The counters are supported per ACL per EVC.
- Per ACL permit or deny counters are not supported on ES20 interfaces.
- ACL configurations with QoS, Vlan, or Ethertype filtering are ignored when applied on an ES20 EVC interface.
- L2 ACL is not supported in an ES20 EVC on the port-channel of an ES20 interface.

SUMMARY STEPS TO CREATE A L2 ACL

1. **enable**
2. **configure terminal**
3. ***mac access-list extendend aclname {permit | deny} {host a.b.c host x.y.z}***
4. exit

DETAILED STEPS TO CREATE A L2 ACL

	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	*mac access-list extendend aclname* {permit | deny} {*host a.b.c host x.y.z*}
	Example:
	`me7600-5(config)#mac access-list extended test-l2-acl`
		Creates a Layer 2 Access List on the selected interface.

Step 4	`exit`	Exits the configuration mode.

SUMMARY STEPS TO APPLY A L2 ACL

1. **enable**
2. **configure terminal**
3. ***interface gigabitethernet*** type/ slot/port *[.subinterface-number] or* ***interface tengigabitethernet*** type/ slot/port *[subinterface-number]*
4. **[no] service instance** *id* **{Ethernet*}***
5. **encapsulation dot1q** ***vlan id***
6. **mac access- group** ***aclname in***
7. **exit**

DETAILED STEPS TO APPLY A L2 ACL

	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** type/ slot/port [subinterface-number] or **interface tengigabitethernet** type/ slot/port [subinterface-number] **Example:** `Router(config)# interface gigabitethernet 4/0/0`	Specifies the Gigabit Ethernet or the Ten Gigabit Ethernet interface to configure, where: • *slot/subslot/port*—Specifies the location of the interface. • *subinterface-number*—(Optional) Specifies a secondary interface (sub-interface) number.

Step 4	[**no**] **service instance** *id* {**Ethernet** [*service-name*}
	Example:
	`Router(config-if)# service instance 101 ethernet`
		Creates a service instance on an interface and sets the device to the **config-if-srv** configuration mode.

	Command or Action	Purpose

Step 5	**`encapsulation dot1q`** ***`vlan id`***
	Example:
	`Router(config-if-srv)# encapsulation dot1q 5`
		Defines the matching criteria to map ingress dot1q frames on an interface to the appropriate service instance.
		Note Use the **encapsulation dot1q default** command to configure the default service instance on a port. Use the **encapsulation dot1q untagged** command to map the untagged Ethernet frames on an ingress interface to a service instance.

Step 6	`mac access- group aclname in`
	Example:
	`me7600-5(config-if-srv)# mac access-group test-l2-acl in`
		Applies a L2 ACL on the selected EVC.
		Note L2 ACL displays only positive permit and deny counts.

Step 7	`exit`	Exits the configuration mode.

Examples

You can view the ACL counters for an EVC as shown in the following example:

```
LLB-India-7#sh ethernet service instance id 1 int gig3/0/0 detail
Service Instance ID: 1
L2 ACL (inbound): l2acl                         <=====
Associated Interface: GigabitEthernet3/0/0
Associated EVC: test
L2protocol drop
CE-Vlans:
Interface Dot1q Tunnel Ethertype: 0x8100
State: Up
L2 ACL permit count: 0                              <=====
L2 ACL deny count: 0                                <=====
EFP Statistics:
   Pkts In   Bytes In   Pkts Out  Bytes Out
         0          0          0          0
```

Configuring Broadcast Storm Control on Switchports and Ports with Ethernet Virtual Connections

A traffic storm occurs when data packets flood the LAN, creates excessive traffic, and degrades network performance.

The traffic storm control level is a percentage of the total available bandwidth of the port. Each port has a single traffic storm control level that is used for all types of traffic (broadcast, multicast, and unicast).

In LAN cards, traffic storm control (also called traffic suppression), performs the following tasks:

- Monitors the incoming traffic levels over a 1-second traffic storm control interval.
- Compares the traffic level with the configured traffic storm control level during the interval time.

When the ingress traffic (for which the traffic storm control is enabled) reaches the configured level on the port, the traffic storm control terminates the traffic till the traffic storm control interval ends.

Detecting a Broadcast Storm

Broadcast storms can be prevented by configuring a network to block broadcast traffic. The mechanism to detect and control such storm event(s) is referred to as storm control or broadcast suppression.You can detect a broadcast storm when the following occurs:

- The port receives multicast and broadcast traffic in excess of the configured bandwidth value.
- The value in the 'TotalSuppDiscards' counter increments.This value is displayed when you use the **show interface gigabitEthernet <s***lot/port***> counters storm-control** command.

Traffic Storm Control on ES20 Switchports

For more information on traffic storm control on ES20 switchports, refer to the *Cisco 7600 Series Cisco IOS Software Configuration Guide, 12.2SR* located at the following URL:

http://www.cisco.com/en/US/docs/routers/7600/ios/12.2SR/configuration/guide/swcg.html

Restrictions and Usage Guidelines

- The Traffic Storm Control feature is not supported on:
  - Layer 3 interfaces
  - Unicast traffic
- Traffic storm control is supportedon layer 2 interfaces.
- Storm Control cannot be applied on an ES20 port configured as a switchport in a dot1q-tunnel mode.
- Rx SPAN is not supported on an ES20 port with switchports and Storm Control configuration.
- You can configure a maximum of 40% link rate (or 4Gbps) on a 10-gigabyte port.
- If you have configured Storm Control on an ES20 switchport, you cannot apply port access control lists (ACLs).
- Storm control feature is not supported onthe following line cards and also on other line cards based on the same ASIC:
  - WS-X6248-RJ-45
  - WS-X6348-RJ-45
  - WS-X6148-RJ-45
  - WS-X6148-RJ21
  - WS-X6248-RJ21
  - WS-X6196-RJ21
  - WS-X6148X2-RJ-45
  - WS-X6548-RJ-45
  - WS-X6548-RJ21

Traffic Storm Control on ES20 with EVCs

Traffic Storm Control is supported for an EVC-configured port at the interface level.

In Cisco IOS Release 15.0 (1)S and later releases, the following enhancements are covered as part of this feature on the 67xx, 6196, ES20 and ES+ line cards:

- Support is extended to port channel interfaces.
- When a storm is detected and the storm traffic exceeds the accepted threshold, the affected interface moves to error disable state. The traffic threshold is calculated as a percentage of the total bandwidth of the port (%BW). You can use the error disable detection and the recovery feature or the **shut/no shut** command to re-enable the port on the affected interface.
- An SNMP trap can be sent, when the storm is detected.

Restrictions and Usage Guidelines

- The Traffic Storm Control feature is supported on ports with EVC on an ES20 card.
- Storm control is configured per-interface level and not per EVC.
- Unicast traffic is not supported on port channel interfaces.
- We do not recommend using 0 or 100 to configure the level for broadcast or multicast traffic.
- Untagged frames can be subjected to storm control by having a service instance which marks all untagged frames. Once such a service instance is created, these frames behave like any storm control on any other EVC.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. s**nmp-server enable traps storm-control trap-rat**e *trap rate*
4. **interface** *type/ slot/port*
5. **storm-control** {{broadcast | multicast } level *level* | action {shutdown | trap}}
6. **end**

DETAILED STEPS

	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	**`snmp-server enable traps storm-control trap-rate`** *`trap rate`*
	Example:
	`Router(config)# snmp-server enable traps storm-control trap-rate 2`
		(Optional) Enables SNMP storm control trap parameters. The trap-rate range is 0 to 1000 traps per minute. However, the number of traps generated for storm control cannot exceed six per minute (by design).

Step 4	**`interface`** *`type slot/bay/port`*
	Example:
	`Router(config)# interface gigabitEthernet 1/0/18`
	`or`
	`Router(config)# interface port-channel 1`
		Selects an interface to configure.

Step 5	**`storm-control {{broadcast | multicast} level`** ***`level`*** **`| action {shutdown | trap}}`**
	Example:
	`Router(config-if)# storm-control broadcast level 50`
	`Router(config-if)# storm-control action shutdown`
		Sets the broadcast and multicast suppression level for traffic storm control on the interface. Enables an action for traffic storm control on a port channel, such as, shuts down a port channel or sends an SNMP trap. However, broadcast or multicast level supression must be enabled before setting the action on the interface.
		Note A suppression level of 100% means no suppression will occur and 0% suppression means no traffic of the suppressed type will be allowed.
		Use the **no** form of the command to disable storm control for broadcast or multicast traffic or to disable the specified storm-control action, on the selected interface.
		Note Unicast level traffic suppression is not supported on port channel interfaces.

Step 6	**`end`**	Exits the configuration mode.

Example

This is a sample output of traffic storm control configured at the physical interface.

```
interface GigabitEthernet9/0/3
 no ip address
 mls qos trust dscp
 storm-control broadcast level 10.00
 storm-control multicast level 10.00
 storm-control action shutdown
 storm-control action trap
end
```

This is a sample output of traffic storm control configured on ports with EVCs.

```
Router# show run interface gig1/18
Building configuration...

Current configuration : 380 bytes
!
interface GigabitEthernet1/18
 no ip address
 storm-control broadcast level 23.45
 storm-control multicast level 23.45
 service instance 1 ethernet
  encapsulation default
 !
 service instance 2 ethernet
  encapsulation untagged
 !
 service instance 100 ethernet
  encapsulation dot1q 100
 !
 service instance 120 ethernet
  encapsulation dot1q 120
   bridge-domain 200

This is a sample output of traffic storm control configured on a port channel.

Router# show run interface port-channe1
Building configuration...

Current configuration : 141 bytes
!
interface Port-channel1
 no ip address
 storm-control broadcast level 75.00
 storm-control action shutdown
 storm-control action trap
end
```

Verification

Use the **show interfaces** *interface counters* **storm-control** command to display the total suppression percentage of packets for the broadcast, multicast and unicast storm control traffic on all interfaces or on a specified interface. The storm control shutdown on an interface depends on the 'TotalSuppDiscards' counter (displayed in the example). This counter increments when a traffic storm occurs.

```
Router# show interfaces counters storm-control

Port         UcastSupp %     McastSupp %     BcastSupp %  TotalSuppDiscards
Gi1/1             100.00          100.00          100.00                  0
Gi1/2             100.00          100.00          100.00                  0
Gi1/3             100.00          100.00          100.00                  0
Gi1/4             100.00          100.00          100.00                  0
Gi1/5             100.00          100.00          100.00                  0
Gi1/6             100.00          100.00          100.00                  0
Gi1/7             100.00           20.00           20.00         2943374677
Gi1/8             100.00          100.00          100.00                  0
Gi1/9             100.00          100.00          100.00                  0
Gi1/10            100.00          100.00          100.00                  0
Gi1/11            100.00          100.00          100.00                  0
Gi1/12            100.00          100.00          100.00                  0
Gi1/13            100.00          100.00          100.00                  0
Gi1/14            100.00          100.00          100.00                  0
Gi1/15            100.00          100.00          100.00                  0
Gi1/16            100.00          100.00          100.00                  0
Gi1/17            100.00          100.00          100.00                  0
Gi1/18            100.00          100.00          100.00          434529474
Gi1/19            100.00          100.00          100.00                  0
Gi1/20            100.00          100.00          100.00                  0
Gi1/21            100.00          100.00          100.00                  0

Port         UcastSupp %     McastSupp %     BcastSupp %  TotalSuppDiscards
Gi1/22            100.00          100.00          100.00          499018427
```

```
Gi1/23            100.00          100.00          100.00                  0
Gi1/24            100.00          100.00          100.00                  0
Gi1/25            100.00          100.00          100.00                  0
Gi1/26            100.00          100.00          100.00                  0
Gi1/27            100.00          100.00          100.00                  0
Gi1/28            100.00          100.00          100.00                  0
Gi1/29            100.00          100.00          100.00                  0
Gi1/30            100.00          100.00          100.00                  0
Gi1/31            100.00          100.00          100.00                  0
Gi1/32            100.00          100.00          100.00                  0
Gi1/33            100.00          100.00          100.00                  0
Gi1/34            100.00          100.00          100.00                  0
Gi1/35            100.00          100.00          100.00                  0
Gi1/36            100.00          100.00          100.00                  0
Gi1/37            100.00          100.00          100.00                  0
Gi1/38            100.00          100.00          100.00                  0
Gi1/39            100.00          100.00          100.00                  0
Gi1/40            100.00          100.00          100.00                  0

Router# show interfaces gig1/18 counters storm-control

Port         UcastSupp %     McastSupp %     BcastSupp %  TotalSuppDiscards
Gi1/18            100.00          100.00          100.00          434529474
```

Troubleshooting

Contact TAC for troubleshooting scenarios.

Configuring Asymmetric Carrier-Delay

During redundant link deployments where the remote network element is enabled, a link or port may be displayed as **UP** before the port or link is ready to forward data. This anomaly leads to traffic loss during switchover as **UP** events are notified faster than the required routing protocol convergence time. With existing conventional carrier delay, both **UP** and **DOWN** events are notified within equal time that might not be feasible in certain network deployments. Asymmetric Carrier-Delays ensure stable topologies compared to conventional Carrier-Delay implementation.

Table 2-17 lists the differences between the conventional Carrier-Delay and Asymmetric Carrier-Delay implementations.

Table 2-17 Conventional Carrier-Delay versus Asymmetric Carrier-Delay

Conventional Carrier -Delay implementation	Asymmetric Carrier-Delay implementation
You can configure Carrier-Delay on a main physical interface.	You can configure Asymmetric Carrier-Delay on a main physical interface.
The acceptable limit to configure carrier delay can be as low as 0. If you configure 0, it will be treated as 1 msec.	The acceptable limit to configure Asymmetric Carrier-Delay: • **UP** time is 4 seconds and above • **DOWN** time is 11 msec

Conventional Carrier -Delay implementation	Asymmetric Carrier-Delay implementation
You can configure a single delay value used by both **Up** and **Down** events.	You can configure separate delay values for each **DOWN** and **UP** timers.
Traffic losses and timer optimization issues due to single configurable delay values for both **UP** and **Down** events.	Optimal timer configurations are achieved due to separate for timer values for **UP** and **Down** events.

Restrictions and Usage Guidelines

- The acceptable limit to configure Carrier-Delay **DOWN** time is eleven milliseconds and above. By default, Carrier-Delay is configured to 10 milliseconds during a card bootup. However, even if you configure a value less than 11msec, there will not be any impact on the carrier delay.
- As the Fast Link feature and Carrier-Delay features are mutually exclusive, Fast Link feature is enabled by default.
- If you configure Carrier-Delay values, Fast Link feature is disabled on a line card.
- Though the Fast Link feature is configured by default in the card, the Carrier-Delay feature overwrites the Fast Link feature when configured.
- If you have not configured the Carrier-Delay values, Fast link feature values are utilized for **DOWN** event notification.

SUMMARY STEPS

1. enable
2. **configure terminal**
3. interface *type/ slot/port*
4. carrier-delay [{up | down} [*seconds*]{msec| sec}]
5. end

DETAILED STEPS

	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	`config # interface` *type/ slot/port* **Example:** `P19_C7609-S(config)#int gig8/0/14`	Selects the maininterface to configure.

	Command or Action	Purpose

Step 4	`carrier-delay [{up | down}`
	`[seconds]{msec| sec}]`
	Example:
	`P19_C7609-S(config-if)#carrier-delay up 8`
	`P19_C7609-S(config-if)#carrier-delay down 5`
		Configures the Asymmetric Carrier-Delay up or down value in milliseconds or seconds.

Step 5	`end`	Exits the configuration mode.

Verification

You can use the **show run** command to display the Carrier-Delay configurations on an ES20 physical interface.

```
interface GigabitEthernet8/0/14
no ip address
carrier-delay msec 0
carrier-delay up 10
carrier-delay down 5
shutdown
```

Configuring MST on EVC Bridge Domain

The Multiple Spanning Tree (MST) on EVC Bridge Domain feature enables MST on EVC interfaces. It complements the H-VPLS N-PE Redundancy for QinQ and MPLS Access feature released in Cisco IOS Release 12.2(33)SRC. For more information on this feature, see http://www.cisco.com/en/US/docs/ios/mpls/configuration/guide/mp_hvpls_npe_red.html.

This section describes how to configure MST on EVC Bridge Domain. It contains the following topics:

- Overview of MST and STP, page 2-167
- Overview of MST on EVC Bridge Domain, page 2-168
- Restrictions and Usage Guidelines, page 2-168
- Examples, page 2-171

Overview of MST and STP

Spanning Tree Protocol (STP) is a Layer 2 link-management protocol that provides path redundancy while preventing undesirable loops in the network. For a Layer 2 Ethernet network to function properly, only one active path can exist between any two stations. STP operation is transparent to end stations, which cannot detect whether they are connected to a single LAN segment or a switched LAN of multiple segments.

Cisco 7600 series routers use STP (the IEEE 802.1D bridge protocol) on all VLANs. By default, a single instance of STP runs on each configured VLAN (provided you do not manually disable STP). You can enable and disable STP on a per-VLAN basis.

MST maps multiple VLANs into a spanning tree instance, with each instance having a spanning tree topology independent of other spanning tree instances. This architecture provides multiple forwarding paths for data traffic, enables load balancing, and reduces the number of spanning tree instances required to support a large number of VLANs. MST improves the fault tolerance of the network because a failure in one instance (forwarding path) does not affect other instances (forwarding paths).

For routers to participate in MST instances, you must consistently configure the routers with the same MST configuration information. A collection of interconnected routers that have the same MST configuration comprises an MST region. For two or more routers to be in the same MST region, they must have the same VLAN-to-instance mapping, the same configuration revision number, and the same MST name.

The MST configuration controls the MST region to which each router belongs. The configuration includes the name of the region, the revision number, and the MST VLAN-to-instance assignment map.

A region can have one or multiple members with the same MST configuration; each member must be capable of processing RSTP bridge protocol data units (BPDUs). There is no limit to the number of MST regions in a network, but each region can support up to 65 spanning tree instances. Instances can be identified by any number in the range from 0 to 4094. You can assign a VLAN to only one spanning tree instance at a time.

For additional information on STP and MST on the Cisco 7600 series routers, see *Configuring STP and MST* at:

http://www.cisco.com/en/US/docs/routers/7600/ios/15S/configuration/guide/spantree.html

Overview of MST on EVC Bridge Domain

The MST on EVC Bride-Domain feature uses VLAN IDs for service-instance-to-MST-instance mapping. EVC service instances with the same VLAN ID (the outer VLAN IDs in the QinQ case) as the one in another MST instance will be mapped to that MST instance.

EVC service instances can have encapsulations with a single tag as well as double tags. In case of double tag encapsulations, the outer VLAN ID shall be used for the MST instance mapping, and the inner VLAN ID is ignored.

A single VLAN per EVC is needed for the mapping with the MST instance. The following service instances without any VLAN ID or with multiple outer VLAN IDs are not supported:

- Untagged (encapsulation untagged)
- Priority-tagged (encapsulation priority-tagged)
- Default (encapsulation default)
- Multiple outer tags (encapsulation dot1q 200 to 400 second-dot1q 300)

Restrictions and Usage Guidelines

Follow these restrictions and usage guidelines while configuring MST on EVC bridge domain:

- Cisco IOS Release 15.1(1)S supports EVC port-channels.
- Main interface where the EFP is configured must be up and running with MSTP as the selected Spanning Tree Mode (PVST and Rapid-PVST are not supported).
- The SPT PortFast feature is not supported with EFPs.
- The co-existence of REP and mLACP with MST on the same port is not supported.

- Any action performed on VPORT (which represents a particular VLAN in a physical port) affects the bridge domain and other services.
- This feature cannot co-exist with Ethernet Bridging on FR/ATM that support only PVST.
- Supports 64 MSTs and one CIST (common and internal spanning tree).
- Supports one MST region.
- Scales to 32000 EFP.
- Service instances without any VLAN ID in the encapsulation are not supported, because a unique VLAN ID is required to map an EVC to an MST instance.
- Supports EFPs with unambiguous outer VLAN tag (that is, no range, list on outer VLAN, neither default nor untagged).
- ES20 and ES+ line cards support this feature.
- Removing dot1q encapsulation removes the EVC from MST.
- Changing the VLAN (outer encapsulation VLAN of EVC) mapping to a different MST instance will move the EVC port to the new MST instance.
- Changing an EVC service instance to a VLAN that is not defined in MST 1 will result in mapping of EVC port to MST 0.
- The peer router of the EVC port must also be running MST.
- MST is supported only on EVC BD. EVCs without BD configuration will not participate in MST.
- When an MST is configured on the outer VLAN, you can configure any number of service instances with the same outer VLAN as shown in the following configuration example.

```
nPE1#sh run int gi12/5
Building configuration...

Current configuration : 373 bytes
!
interface GigabitEthernet12/5
 description connected to CE1
 no ip address
 service instance 100 ethernet
  encapsulation dot1q 100 second-dot1q 1
  bridge-domain 100
 !
 service instance 101 ethernet
  encapsulation dot1q 100 second-dot1q 2
  bridge-domain 101
 !
 service instance 102 ethernet
  encapsulation dot1q 100 second-dot1q 120-140
  bridge-domain 102
 !
end


nPE1#sh run int gi12/6
Building configuration...

Current configuration : 373 bytes
!
interface GigabitEthernet12/6
 description connected to CE1
 no ip address
 service instance 100 ethernet
  encapsulation dot1q 100 second-dot1q 1
  bridge-domain 100
 !
 service instance 101 ethernet
  encapsulation dot1q 100 second-dot1q 2
  bridge-domain 101
 !
 service instance 102 ethernet
  encapsulation dot1q 100 second-dot1q 120-140
  bridge-domain 102
 !
end

nPE1#sh span vlan 100

MST0
  Spanning tree enabled protocol mstp
  Root ID    Priority    32768
             Address     0018.742f.3b80
             Cost        0
             Port        2821 (GigabitEthernet12/5)
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32768  (priority 32768 sys-id-ext 0)
             Address     001a.303c.3400
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Gi12/5              Root FWD 20000     128.2821 P2p
Gi12/6              Altn BLK 20000     128.2822 P2p

nPE1#
```

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot/subslot/port* or **interface tengigabitethernet** *slot/subslot/port*
4. **service instance** *id* **Ethernet** [*service-name*]
5. **encapsulation dot1q** *vlan-id*
6. [**no**] **bridge-domain** *bridge-id*

DETAILED STEPS

	Command	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	Purpose
Step 3	**interface gigabitethernet** *slot/subslot/port* or **interface tengigabitethernet** *slot/subslot/port* **Example:** `Router(config)# interface gigabitethernet 4/1/1`	Specifies the gigabit ethernet or the ten gigabit ethernet interface to configure. • *slot/subslot/port*—Specifies the location of the interface.
Step 4	[**no**] **service instance** *id* **Ethernet** [*service-name*] **Example:** `Router(config-if)# service instance 101 ethernet`	Creates a service instance (EVC instance) on an interface and sets the device into the config-if-srv submode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Router(config-if-srv)# encapsulation dot1q 13`	Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.
Step 6	[no] **bridge-domain** *bridge-id* **Example:** `Router(config-if-srv)# bridge-domain 12`	Binds the service instance to a bridge domain instance where *bridge-id* is the identifier for the bridge domain instance.

Examples

In this example, two interfaces participate in MST instance 0, the default instance to which all VLANs are mapped:

```
Router# enable
Router# configure terminal
Router(config)# interface g4/1/1
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 2
Router(config-if-srv)# bridge-domain 100
Router(config-if-srv)# interface g4/3
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 2
Router(config-if-srv)# bridge-domain 100
Router(config-if-srv)# end
```

Verification

Use this command to verify the configuration:

```
Router# show spanning-tree vlan 2

MST0
  Spanning tree enabled protocol mstp
  Root ID    Priority    32768
             Address     0009.e91a.bc40
             This bridge is the root
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32768  (priority 32768 sys-id-ext 0)
             Address     0009.e91a.bc40
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Gi4/1              Desg FWD 20000     128.1537 P2p
Gi4/3              Back BLK 20000     128.1540 P2p
```

In this example, interface gi4/1/1 and interface gi4/1/3 are connected back-to-back. Each has a service instance (EFP) attached to it. The EFP on both interfaces has an encapsulation VLAN ID of 2. Changing the VLAN ID from 2 to 8 in the encapsulation directive for the EFP on interface gi4/1/1 stops the MSTP from running in the MST instance to which the old VLAN is mapped and starts the MSTP in the MST instance to which the new VLAN is mapped:

```
Router(config-if)# interface g4/1/1
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encap dot1q 8
Router(config-if-srv)# end
```

Use this command to verify the configuration:

```
Router# show spanning-tree vlan 2

MST1
  Spanning tree enabled protocol mstp
  Root ID    Priority    32769
             Address     0009.e91a.bc40
             This bridge is the root
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32769  (priority 32768 sys-id-ext 1)
             Address     0009.e91a.bc40
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Gi4/3             Desg FWD 20000     128.1540 P2p

Router# show spanning-tree vlan 8

MST2
  Spanning tree enabled protocol mstp
  Root ID    Priority    32770
             Address     0009.e91a.bc40
             This bridge is the root
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32770  (priority 32768 sys-id-ext 2)
             Address     0009.e91a.bc40
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Gi4/1              Desg FWD 20000     128.1537 P2p
```

In this example, interface gi4/1/3 (with an EFP that has an outer encapsulation VLAN ID of 2 and a bridge domain of 100) receives a new service:

```
Router# enable
```

```
Router# configure terminal
Router(config)# interface g4/1/3
Router((config-if)# service instance 2 ethernet
Router((config-if-srv)# encap dot1q 2 second-dot1q 100
Router((config-if-srv)# bridge-domain 200
```

Now there are two EFPs configured on interface gi4/1/3 and both have the same outer VLAN 2.

```
interface GigabitEthernet4/3
     no ip address
    service instance 1 ethernet
    encapsulation dot1q 2
    bridge-domain 100
 !
 service instance 2 ethernet
  encapsulation dot1q 2 second-dot1q 100
   bridge-domain 200
```

The preceding configuration does not affect the MSTP operation on the interface; there is no state change for interface gi4/1/3 in the MST instance it belongs to.

```
Router# show spanning-tree mst 1

##### MST1    vlans mapped:   2
Bridge        address 0009.e91a.bc40  priority      32769 (32768 sysid 1)
Root          this switch for MST1

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Gi4/3          Desg FWD 20000     128.1540 P2p
```

This example shows MST on port channels:

```
Router# show spanning-tree mst 1
##### MST1 vlans mapped: 3
Bridge address 000a.f331.8e80 priority 32769 (32768 sysid 1)
Root address 0001.6441.68c0 priority 32769 (32768 sysid 1)
port Po5 cost 20000 rem hops 18

Interface Role Sts Cost Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Gi2/0/0 Desg FWD 20000 128.257 P2p
Po5 Root FWD 10000 128.3329 P2p
Po6 Altn BLK 10000 128.3330 P2p

Router# show spanning-tree vlan 3

MST1
Spanning tree enabled protocol mstp
Root ID Priority 32769
Address 0001.6441.68c0
Cost 20000
Port 3329 (Port-channel5)
Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec

Bridge ID Priority 32769 (priority 32768 sys-id-ext 1)
Address 000a.f331.8e80
Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec

Interface Role Sts Cost Prio.Nbr Type
------------------ ---- --- --------- -------- --------------------------------
Gi2/0/0 Desg FWD 20000 128.257 P2p
Po5 Root FWD 10000 128.3329 P2p
Po6 Altn BLK 10000 128.3330 P2p
```

DHCP Snooping with Option-82 on EVC

DHCP snooping determines whether traffic sources are trusted or untrusted. An untrusted source may initiate traffic attacks or other hostile actions. To prevent such attacks, DHCP snooping filters messages traffic from untrusted sources.

To do this, DHCP snooping dynamically builds and maintains the DHCP snooping database using information extracted from intercepted DHCP messages. The database contains an entry for each untrusted host with a leased IP address if the host is associated with a VLAN that has DHCP snooping enabled. The database does not contain entries for hosts connected through trusted interfaces.

Each entry in the DHCP snooping database includes the MAC address of the host, the leased IP address, the lease time, the binding type, and the VLAN number and interface information associated with the host.

Additionally, the DHCP Snooping with Option-82 feature can centrally manage the IP address assignments for a large number of subscribers. When this feature is enabled on the router, a subscriber device is identified by the router port through which it connects to the network (in addition to its MAC address). Multiple hosts on the subscriber LAN can be connected to the same port on the access router and are uniquely identified.

However, EVCs require additional information. If each EVC on an interface is mapped to a single VPN, it would be possible to use the internal VLAN to identify the path for reply packets. However, because multiple EVCs with different encapsulations can map to the same VPN, it is necessary to use the actual EVC encapsulation to distinguish between EVCs.

The DHCP Snooping with Option-82 on EVC feature allows the user to provide this additional information required for EVC-enabled interfaces. This information is inserted into the option 82 and is also stored in the binding table for retrieval by other services.

Use the **ip dhcp snooping information option allow-untrusted** command to enable the switch to accept incoming DHCP snooping packets with option 82 information from the edge switch. DHCP option 82 data insertion is enabled by default. Accepting incoming DHCP snooping packets with option 82 information from the edge switch is disabled by default.

Use the **ip dhcp relay information option subscriber-id** command to configure a subscriber string for an EVC that can be inserted into the option 82 field along with other information when relaying the DHCP packets to the server. The server can parse the option 82 information to match the subscriber string and act accordingly. The subscriber string configured for an EVC is not stored in the binding table and is only used when sending DHCP packets to the server by inserting into the option 82 field.

For additional information on DHCP Snooping and Option-82 on the Cisco 7600 router, see Configuring DHCP Snooping at
http://www.cisco.com/en/US/docs/routers/7600/ios/15S/configuration/guide/snoodhcp.html.

Restrictions and Usage Guidelines

Follow these restrictions and usage guidelines while you configure DHCP Snooping with Option-82:

- An EVC with multiple encapsulations is not supported.
- These EVCs are supported on the same interface and bridge-domain:
  - dot1q encapsulation
  - QinQ encapsulation
  - Untagged encapsulation
- 4000 EVCs are supported per port.

- 32000 EVCs are supported per router.
- Multiple EVCs are supported on same port, all having the same or different bridge domains.
- Multiple EVCs are supported on different ports, all having the same or different bridge domains.
- With Cisco IOS Release 15.1(1)S, DHCP snooping with Option 82 is supported on EVC port-channels.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot/port* or **interface tengigabitethernet** *slot/port* or **interface port-channel** *number*
4. **[no] ip address**
5. **negotiation** {**forced** | **auto**}
6. **service instance** *id* **Ethernet** [*service-name*]
7. **encapsulation dot1q** *vlan-id*
8. **rewrite ingress tag** {**push {dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {dot1q *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} **symmetric**
9. **ip dhcp relay information option subscriber-id** *value*
10. [**no**] **bridge-domain** *bridge-id*

DETAILED STEPS

	Command	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`** *`slot/subslot/port`*`[.`*`subinterface-number`*`]`
	or
	`interface tengigabitethernet` *`slot/subslot/port`*`[.`*`subinterface-number`*`]`
	or
	`interface port-channel` *`number`*
	Example:
	`Router(config)# interface gigabitethernet 4/1`
		Specifies the gigabit ethernet or the ten gigabit ethernet or the port-channel interface to configure.

	Command	Purpose
Step 4	**no ip address** **Example:** `Router# Router(config-if)# no ip address`	Removes an IP address or disables IP processing.

Step 5	**negotiation** {**forced** | **auto**}
	Example:
	`Router(config-if)# negotiation auto`
		Enables advertisement of speed, duplex mode, and flow control on a gigabit ethernet interface.

Step 6	[**no**] **service instance** *id* **Ethernet** [*service-name*}
	Example:
	`Router(config-if)# service instance 101 ethernet`
		Creates a service instance (an instantiation of an EVC) on an interface, and sets the device into the config-if-srv submode.

Step 7	**encapsulation dot1q** *vlan-id* **Example:** `Router(config-if-srv)# encapsulation dot1q 13`	Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.
Step 8	**rewrite ingress tag** {**push {dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {dot1q *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} **symmetric** **Example:** `Router(config-if-srv)# rewrite ingress tag push dot1q 20 symmetric`	Specifies the tag manipulation to be performed on the frame ingress to the service instance.
Step 9	**ip dhcp relay information option subscriber-id** *value* **Example:** `Router(config)# ip dhcp relay information option subscriber-id 123`	Configures a subscriber string that uniquely identifies the interface from where the DHCP packets originate.
Step 10	[**no**] **bridge-domain** *bridge-id* **Example:** `Router(config-if-srv)# bridge-domain 12`	Binds the service instance to a bridge domain instance where *bridge-id* is the identifier for the bridge domain instance.

Example

This example shows a typical configuration on the relay agent and the server. This is a configuration on the relay agent:

```
Router# enable
Router# configure terminal
Router(config)# interface GigabitEthernet8/1
Router(config-if)# no ip address
Router(config-if)# negotiation auto
Router(config-if)# service instance 2 ethernet
Router(config-if-srv)# encapsulation dot1q 2
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
  ip dhcp relay information option subscriber-id 11
Router(config-if-srv)# bridge-domain 100

Router(config)# interface Vlan100
Router(config-if)# ip address 10.0.0.1 255.255.255.0
Router(config-if)# ip helper-address global 20.0.0.2
Router(config-if)# ip helper-address 20.0.0.2


Router(config)# interface GigabitEthernet 2/1
Router(config-if)# ip dhcp snooping packets
Router(config-if)# ip address 20.0.0.1 255.255.255.0
Router(config-if)# negotiation auto
!
```

This is the configuration on the server:

```
:
Router# enable
Router# configure terminal
Router(config)# interface GigabitEthernet 1/1
Router(config-if)# ip address 20.0.0.2 255.255.255.0
Router(config-if)# negotiation auto
Router(config-if)# end

Router(config)# ip dhcp pool pool1
Router(dhcp-config)# network 10.0.0.0 255.255.0.0
   lease 2
Router(dhcp-config)# update arp
   class C1
      address range 10.0.0.2 10.0.0.10
   class C2
      address range 10.0.0.11 10.0.0.20
!
Router(config)# ip dhcp pool pool2
Router(config)# network 11.0.0.0 255.255.0.0 lease 2
!
Router(config)# ip dhcp pool pool3
   vrf vrf1
Router(config)# ip dhcp use vrf remote
Router(config)# network 10.0.0.0 255.255.255.0 lease 0 0 2
!
!
ip dhcp class C1 <-----------Class C1 maps to the subcriber-id string aabb11.
   relay agent information
      relay-information hex 00000000000000000000000000000006616162623131 mask
fffffffffffffffffffffffffffffff0000000000000
!
ip dhcp class C2
   relay agent information
      relay-information hex 00000000000000000000000000000006313162626161 mask
fffffffffffffffffffffffffffffff0000000000000

******************************************************************************************
```

Verification

Use these commands to verify operation.

Command	Purpose
Router# **show ip dhcp snooping**	Displays all VLANs (both primary and secondary) that have DHCP snooping enabled.
Router# **show ip dhcp snooping binding**	Checks the DHCP snooping database.
Router# **show ethernet service instance**	Displays the ethernet customer service instances.

Configuring MAC Address Security for EVC Bridge-Domain

Cisco 7600 series routers currently support port security on a per-port basis. For more information, see *Configuring Port Security* at:
http://www.cisco.com/en/US/docs/routers/7600/ios/12.2SR/configuration/guide/port_sec.html

The Media Access Control (MAC) Address Security for EVC Bridge-Domain feature addresses port security with EVCs by providing the capability to control and filter MAC address learning behavior at the granularity of a per-EFP basis. For instance, when a violation requires a shutdown, only the customer assigned to a given EFP is affected rather than all customers using the port.

Port Security and the MAC Address Security for EVC Bridge-Domain feature operate independently of each other. From the point of view of MAC Security, a secured port is not secure. From the point of view of Port Security, a secured EFP is not secure.

Restrictions and Usage Guidelines

- System-wide, the following limits apply to the total configured whitelist and learned MAC addresses:
  - Total number of MAC addresses supported on MAC Security is limited to 32,000.
  - Total number of MAC addresses supported on MAC Security, per bridge-domain, is limited to 10,000.
  - Total number of MAC addresses supported on MAC Security, per EFP, is limited to 1000.
- You can configure or remove the various MAC security elements irrespective of whether MAC security is enabled on the EFP. However these configurations will become operational only after MAC security is enabled.
- Upon enabling the MAC Address Security for EVC Bridge-Domain feature, existing MAC address table entries on the EFP are removed.
- Upon disabling the MAC Address Security for EVC Bridge-Domain feature, existing MAC address table entries on the EFP are removed.
- The MAC Address Security for EVC Bridge-Domain feature can be configured on an EFP only if the EFP is a member of a bridge domain.
- If you disassociate the EFP from the BD, the MAC security feature is completely removed.
- For port-channel, this configuration is propagated to all member links in the port-channel. Consistent with the already implemented bridge domain EVC port-channel functionality, packets on a secured EFP is received on any member link, but all the egress packets are sent out to one of the selected member link.

Configuring MAC Address Security for EVC Bridge-Domain

These sections describe how to configure port security:

- Configuring MAC Address Security for EVC Bridge-Domain, page 2-179
- Enabling MAC Address Security for EVC Bridge-Domain, page 2-179
- Disabling MAC Address Security for EVC Bridge-Domains on an EFP, page 2-180
- Configuring Whitelisted MAC Address on an EFP, page 2-181
- Configuring Sticky MAC Addresses on an EFP, page 2-183
- Configuring Secure MAC Address Aging on an EFP, page 2-185
- Configuring MAC Address Limiting on an EFP, page 2-188
- Configuring MAC Address Limiting on a Bridge-Domain, page 2-189
- Configuring Violation Response on an EFP, page 2-190

Enabling MAC Address Security for EVC Bridge-Domain

This section describes how to enable MAC address security for EVC bridge-domain.

SUMMARY STEPS

1. enable
2. **configure terminal**
3. **interface gigabitethernet** *slot/subslot/port* or **interface tengigabitethernet** *slot/subslot/port* **or interface port-channel** *number*
4. **service instance** *id* **Ethernet** [*service-name*]
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. mac security

DETAILED STEPS

	Command	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	Purpose
Step 3	**interface gigabitethernet** *slot*/subslot/*port* or **interface tengigabitethernet** slot/subslot/port **Example:** `Router(config)# interface gigabitethernet 4/1/0`	Specifies the Gigabit Ethernet or the Ten Gigabit Ethernet interface to configure, where: • *slot/subslot/port*—Specifies the location of the interface.
Step 4	**service instance** id **Ethernet** [service-name] **Example:** `Router(config-if)# service instance 101 ethernet`	Creates a service instance (an instance of an EVC) on an interface and sets the device into the config-if-srv submode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Router(config-if-srv)# encapsulation dot1q 13`	Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.
Step 6	**bridge-domain** *bridge-id* **Example:** `Router(config-if-srv)# bridge-domain 12`	Binds the service instance to a bridge domain instance where *bridge-id* is the identifier for the bridge domain instance.
Step 7	**mac security** or **no mac security** **Example:** `Router(config-if-srv)# mac security`	Enables MAC Security on the EFP.

Examples

This example shows how to enable MAC address security for EVC bridge-domains.

```
Router# enable
Router# configure terminal
Router(config)# interface GigabitEthernet 4/1/0
Router(config-if)# service instance 10 ethernet
Router(config-srv)# encapsulation dot1q 20
Router(config-if-srv)# bridge-domain 100
Router(config-if-srv)# mac security
```

Disabling MAC Address Security for EVC Bridge-Domains on an EFP

This section describes how to disable MAC address security for EVC bridge-domains.

SUMMARY STEPS

1. enable
2. **configure terminal**
3. ***interface gigabitethernet*** *slot/subslot/port or* ***interface tengigabitethernet*** *slot/subslot/port*

4. **service instance id Ethernet** [*service-name*]
5. no mac security

DETAILED STEPS

	Command	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`** *`slot`*`/subslot/`*`port`*
	`or`
	`interface tengigabitethernet` `slot/subslot/port`
	Example:
	`Router(config)# interface gigabitethernet 4/1/0`
		Specifies the Gigabit Ethernet or the Ten Gigabit Ethernet interface to configure, where:
		• *slot/subslot/port*—Specifies the location of the interface.

Step 4	**`service instance`** `id` **`Ethernet`** `[service-name]`
	Example:
	`Router(config-if)# service instance 101 ethernet`
		Creates a service instance (an instance of an EVC) on an interface and sets the device into the config-if-srv submode.

Step 5	**`no mac security`**
	Example:
	`Router(config-if-srv)# no mac security`
		Disables MAC Security on the EFP.

Examples

This example shows how to disable MAC address security for EVC bridge-domains.

```
Router# enable
Router# configure terminal
Router(config)# interface GigabitEthernet 4/1/0
Router(config-if)# service instance 10 ethernet
Router(config-if-srv)# no mac security
```

Configuring Whitelisted MAC Address on an EFP

This section describes how to configure whitelisted MAC addresses on an EFP which is a member of the bridge-domains.

SUMMARY STEPS

1. enable
2. **configure terminal**
3. **interface gigabitethernet** *slot/subslot/port* or **interface tengigabitethernet** *slot/subslot/port* **or interface port-channel** *number*
4. **service instance** *id* **Ethernet** [*service-name*]
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. **mac security address permit** *mac address*
8. mac security

DETAILED STEPS

	Command	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`** *`slot/`*`subslot`*`/port`*
	or
	`interface tengigabitethernet` *`slot/`*`subslot`*`/port`*
	`or`
	`interface port-channel` *`number`*
	`Example:`
	`Router(config)# interface gigabitethernet 4/1/0`
		Specifies the Gigabit Ethernet or the Ten Gigabit Ethernet interface to configure, where:
		• *slot/subslot/port*—Specifies the location of the interface.

Step 4	**`service instance`** `id` **`Ethernet`** `[service-name]`
	Example:
	`Router(config-if)# service instance 101 ethernet`
		Creates a service instance (an instance of an EVC) on an interface and sets the device into the config-if-srv submode.

Step 5	**`encapsulation dot1q`** *`vlan-id`*
	Example:
	`Router(config-if-srv)# encapsulation dot1q 13`
		Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.

	Command	Purpose
Step 6	**bridge-domain** *bridge-id* **Example:** `Router(config-if-srv)# bridge-domain 12`	Binds the service instance to a bridge domain instance where *bridge-id* is the identifier for the bridge domain instance.
Step 7	**mac security address permit** *mac address* **Example:** `Router(config-if-srv)# mac security address permit 0000.1111.2222`	Adds the specified MAC Address as a whitelist (permitted) MAC Address for the EFP.
Step 8	**mac security** **Example:** `Router(config-if-srv)# mac security`	Enables MAC Security on the EFP.

Examples

This example shows how to configure whitelisted MAC addresses on an EFP which is a member of a bridge-domain.

```
Router# enable
Router# configure terminal
Router(config)# interface GigabitEthernet 4/1/0
Router(config-if)# service instance 10 ethernet
Router(config-srv)# encapsulation dot1q 20
Router(config-if-srv)# bridge-domain 100
Router(config-if-srv)# mac security address permit 0000.1111.2222
Router(config-if-srv)# mac security
```

Configuring Sticky MAC Addresses on an EFP

MAC addresses learned dynamically on the EFP after mac security sticky is configured are retained during a link-down condition. Stickly Mac is shown in the MAC table as static addresses. However, you should copy the running config details to retain the mac address details.

This section describes how to configure sticky MAC addresses on an EFP.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot/subslot/port* or **interface tengigabitethernet** *slot/subslot/port* **or interface port-channel** *number*
4. **service instance** *id* **Ethernet** [*service-name*]
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. **mac security sticky**
8. **mac security**
9. **no mac security**

DETAILED STEPS

	Command	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`** *`slot`*`/subslot/`*`port`*
	or
	`interface tengigabitethernet` *`slot`*`/subslot/`*`port`*
	`interface port-channel` *`number`*
	Example:
	`Router(config)# interface gigabitethernet 4/1/0`
		Specifies the Gigabit Ethernet or the Ten Gigabit Ethernet interface to configure, where:
		- *slot/subslot/port*—Specifies the location of the interface.

Step 4	**`service instance`** `id` **`Ethernet`** `[service-name]`
	Example:
	`Router(config-if)# service instance 101 ethernet`
		Creates a service instance (an instance of an EVC) on an interface and sets the device into the config-if-srv submode.

Step 5	**`encapsulation dot1q`** *`vlan-id`*
	Example:
	`Router(config-if-srv)# encapsulation dot1q 13`
		Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.

Step 6	**`bridge-domain`** *`bridge-id`*
	Example:
	`Router(config-if-srv)# bridge-domain 12`
		Binds the service instance to a bridge domain instance where *bridge-id* is the identifier for the bridge domain instance.

Step 7	**`mac security sticky`**
	Example:
	`Router(config-if-srv)# mac security sticky`
		Enables Sticky feature causing all dynamic secure MAC addresses to become sticky MAC addresses. Any new MAC address learnt from then on becomes sticky.
		Note To retain the sticky MAC addresses across reloads, ensure that you saved the running-config to startup-config.

	Command	Purpose
Step 8	**mac security** **Example:** `Router(config-if-srv)# mac security`	Enables MAC Security on the EFP.
Step 9	**no mac security** **Example:** `Router(config-if-srv)# no mac security`	Disables MAC Security on the EFP.

Examples

This example configures sticky MAC addresses on an EFP.

```
Router# enable
Router# configure terminal
Router(config)# interface GigabitEthernet 4/1/0
Router(config-if)# service instance 10 ethernet
Router(config-if-srv)# encapsulation dot1q 20
Router(config-if-srv)# bridge-domain 100
Router(config-if-srv)# mac security sticky
Router(config-if-srv)# mac security
```

Configuring Secure MAC Address Aging on an EFP

This section shows how to configure aging of secured MAC addresses on MAC Security. Secured MAC addresses are not subject to the normal aging of MAC table entries in the system. If aging is not configured, secured MAC addresses are never aged out.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** *slot/subslot/port* or **interface tengigabitethernet** *slot/subslot/port* **or interface port-channel** *number*
4. **service instance** *id* **Ethernet** [*service-name*]
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. **mac security aging time** *m* [**inactivity**]
8. **mac security aging static**
9. **mac security aging sticky**
10. **mac security**
11. **no mac security**

DETAILED STEPS

	Command	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`** *`slot`*`/subslot/`*`port`*
	or
	`interface tengigabitethernet` *`slot`*`/subslot/`*`port`*
	or
	`interface port-channel` *`number`*
	Example:
	`Router(config)# interface gigabitethernet 4/1/0`
		Specifies the Gigabit Ethernet or the Ten Gigabit Ethernet interface to configure, where:
		- *slot/subslot/port*—Specifies the location of the interface.

Step 4	**`service instance`** `id` **`Ethernet`** `[service-name]`
	Example:
	`Router(config-if)# service instance 101 ethernet`
		Creates a service instance (an instance of an EVC) on an interface and sets the device into the config-if-srv submode.

Step 5	**`encapsulation dot1q`** *`vlan-id`*
	Example:
	`Router(config-if-srv)# encapsulation dot1q 13`
		Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.

Step 6	**`bridge-domain`** *`bridge-id`*
	Example:
	`Router(config-if-srv)# bridge-domain 12`
		Binds the service instance to a bridge domain instance where *bridge-id* is the identifier for the bridge domain instance.

Step 7	**`mac security aging time`** *`m`* `[`**`inactivity`**`]`
	Example:
	`Router(config-if-srv)# mac security aging time 200`
		Sets the aging time for secure addresses to *m* minutes (range is 0-1440). The optional **inactivity** keyword specifies that the aging out of addresses is based on inactivity of the sending hosts (as opposed to absolute aging).

Step 8	**`mac security aging static`**	Applies aging controls to statically configured addresses.
Step 9	**`mac security aging sticky`**	Applies aging controls to sticky addresses.

	Command	Purpose
Step 10	**mac security** **Example:** `Router(config-if-srv)# mac security`	Enables MAC Security on the EFP. Stickly Mac will be shown in the MAC table as static addresses.
Step 11	**no mac security** **Example:** `Router(config-if-srv)# no mac security`	Disables the MAC Security on the EFP.

Examples

This examples shows how to configure the aging time for secure addresses to 10 minutes.

```
Router# enable
Router# configure terminal
Router(config)# interface GigabitEthernet 4/1/0
Router(config-if)# service instance 10 ethernet
Router(config-if-srv)# encapsulation dot1q 20
Router(config-if-srv)# bridge-domain 100
Router(config-if-srv)# mac security aging time 10
Router(config-if-srv)# mac security
```

This examples shows a configuration where the aging out of addresses is based on inactivity of the sending hosts (as opposed to the aging time of 10 minutes).

```
Router# enable
Router# configure terminal
Router(config)# interface GigabitEthernet 4/1/0
Router(config-if)# service instance 10 ethernet
Router(config-if-srv)# encapsulation dot1q 20
Router(config-if-srv)# bridge-domain 100
Router(config-if-srv)# mac security aging time 10 inactivity
Router(config-if-srv)# mac security
```

The **mac security aging time** command only ages out secure addresses that are learned. To enable aging out of whitelist or sticky addresses when the **mac security aging time** command is configured, use the **mac security aging static** command (applies aging controls to statically configured addresses) or the **mac security aging sticky** command (applies aging controls to persistent, that is, sticky, addresses). The configuration below shows an example of applying aging to a sticky address.

```
Router# enable
Router# configure terminal
Router(config)# interface GigabitEthernet 4/1/0
Router(config-if)# service instance 10 ethernet
Router(config-if-srv)# encapsulation dot1q 10
Router(config-if-srv)# bridge-domain 100
Router(config-if-srv)# mac security
Router(config-if-srv)# mac security sticky
Router(config-if-srv)# mac security aging sticky
Router(config-if-srv)# mac security aging time 100
```

Configuring MAC Address Limiting on an EFP

This section describes how to configure an upper limit for the number of secured MAC addresses allowed on an EFP. This includes addresses added as part of a whitelist, as well as dynamically learned MAC addresses. If the upper limit is decreased, one or more learned MAC entries may be removed. The default limit is 1.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** ***slot/subslot/port*** **or interface tengigabitethernet** ***slot/subslot/port*** **or interface port-chanel** ***number***
4. **service instance** *id* **Ethernet** [*service-name*]
5. **encapsulation dot1q** *vlan-id double tagged*
6. **bridge-domain** *bridge-id*
7. **mac security maximum addresses** *n*
8. **mac security**

DETAILED STEPS

	Command	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** *slot*/subslot/*port* or **interface tengigabitethernet** *slot*/subslot/*port* or **interface port-channel** *number* `Router(config)# interface gigabitethernet 4/1/0`	Specifies the Gigabit Ethernet or the Ten Gigabit Ethernet interface to configure, where: • *slot/subslot/port*—Specifies the location of the interface.
Step 4	**service instance** id **Ethernet** [service-name] **Example:** `Router(config-if)# service instance 101 ethernet`	Creates a service instance (an instance of an EVC) on an interface and sets the device into the config-if-srv submode.

	Command	Purpose
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Router(config-if-srv)# encapsulation dot1q 13`	Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance. The acceptable
Step 6	**bridge-domain** *bridge-id* **Example:** `Router(config-if-srv)# bridge-domain 12`	Binds the service instance to a bridge-domain instance where *bridge-id* is the identifier for the bridge-domain instance.
Step 7	**mac security maximum addresses** *n* **Example:** `Router(config-if-srv)# mac security maximum addresses 10`	Sets (or changes) the maximum number of secure addresses permitted on the EFP to the integer value *n*. The acceptable range secure addresses is 1-1024.
Step 8	**mac security** **Example:** `Router(config-if-srv)# mac security`	Enables MAC Security on the EFP.

Examples

This example configures an upper limit of 10 for the number of secured MAC addresses allowed on an EFP.

```
Router# enable
Router# configure terminal
Router(config)# interface GigabitEthernet 4/1/0
Router(config-if)# service instance 10 ethernet
Router(config-if-srv)# encapsulation dot1q 20
Router(config-if-srv)# bridge-domain 100
Router(config-if-srv)# mac security maximum addresses 10
Router(config-if-srv)# mac security
```

Configuring MAC Address Limiting on a Bridge-Domain

This section describes how to configure an upper limit for the number of secured MAC addresses that resides on the bridge domain.

SUMMARY STEPS

1. enable
2. **configure terminal**
3. **bridge-domain** *vlan-id* [**access** | **dot1q** [*tag*] | **dot1q-tunnel**] [**broadcast**] [**ignore-bpdu-pid**] [**pvst-tlv** *CE-vlan*] [**increment**] [**lan-fcs**] [**split-horizon**]
4. **mac limit maximum addresses** [*n*]

DETAILED STEPS

	Command	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	bridge-domain *bridge-id* **Example:** `Router(config)# bridge-domain 12`	Configures a bridge domain instance with *bridge-id* as the identifier. This command allows you to globally configure the features within the bridge domain instance.
Step 4	**mac limit maximum addresses** [*n*] **Example:** `Router(config-bdomain)# mac limit maximum addresses 1000`	Sets the maximum number of secured MAC address. The maximum default value is 10240.

Examples

This example configures an upper limit of 10240 for the number of secured MAC addresses.

```
Router# enable
Router# configure terminal
Router(config)# bridge-domain 100
Router(config-if-srv)# mac limit maximum address 10240
```

Configuring Violation Response on an EFP

This section describes how to specify the expected behavior of the device when an attempt to dynamically learn a MAC address fails because of a violation of the configured MAC security policy on the EFP. The default violation behavior is EFP shutdown.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** ***slot/subslot/port*** **or interface tengigabitethernet** ***slot/subslot/port*** **or interface port-channel** ***number***
4. **service instance** *id* **Ethernet** [*service-name*]
5. **encapsulation dot1q** *vlan-id*
6. **bridge-domain** *bridge-id*
7. **[no] mac security violation restrict** or [no] **mac security violation protect**

8. **mac security**

DETAILED STEPS

	Command	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** *slot*/subslot/*port* or **interface tengigabitethernet** slot/subslot/port **Example:** `Router(config)# interface gigabitethernet 4/1/0`	Specifies the Gigabit Ethernet or the Ten Gigabit Ethernet interface to configure, where: - *slot/subslot/port*—Specifies the location of the interface.
Step 4	**service instance** id **Ethernet** [service-name] **Example:** `Router(config-if)# service instance 101 ethernet`	Creates a service instance (an instance of an EVC) on an interface and sets the device into the config-if-srv submode.
Step 5	**encapsulation dot1q** *vlan-id* **Example:** `Router(config-if-srv)# encapsulation dot1q 13`	Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.
Step 6	**bridge-domain** *bridge-id* **Example:** `Router(config-if-srv)# bridge-domain 12`	Binds the service instance to a bridge domain instance where *bridge-id* is the identifier for the bridge domain instance.

	Command	Purpose
Step 7	**mac security violation restrict** or **mac security violation protect** **Example:** `Router(config-if-srv)# mac security violation restrict`	Sets the violation mode to one of restrict or protect. The **no** version of this command sets the violation response back to default(default is shutdown). In a Restrict scenario the packets are dropped and an error message is displayed about the log warning level; in the Protect scenario, the packets are silently dropped and no messages are displayed.
Step 8	**mac security** **Example:** `Router(config-if-srv)# mac security`	Enables MAC Security on the EFP.

Examples

This example configures a restrict violation response on EFP.

```
Router# enable
Router# configure terminal
Router(config)# interface GigabitEthernet 4/1/0
Router(config-if)# service instance 10 ethernet
Router(config-if-srv)# encapsulation dot1q 20
Router(config-if-srv)# bridge-domain 100
Router(config-if-srv)# mac security violation restrict
Router(config-if-srv)# mac security
```

Troubleshooting

Table 2-18 provides troubleshooting solutions for the MAC Security feature.

Table 2-18 Troubleshooting Scenarios for MAC Security feature

Problem	Solution
MAC security errors on the RP	Use the **debug ethern serv instance id** *id* **interface** *int* mac *sec errors* and **debug ethern serv instance id** *id* **interface** *int mac table errors* commands. Share the output with TAC for further investigation.
MAC security errors on the SP	Use the **debug ethernet service instance mac security** *errors* and **debug ethernet service instance mac table** *errors* commands to troubleshoot mac security issues on the RP.

Problem	Solution
EFP is disabled and is unable to automatically recover from error disable state	Use the configuration command **errdisable recovery cause mac-security** *interval* or **clear ethernet service instance id** *id* interface *interface-name* **errdisable** to re-enable the EFP.
Mac security aging timer is inactive	When **mac security aging time inactivity** is configured, the hardware mac table aging timer for the EFP VLAN is set with the configuration command **mac address-table aging-time** *time* [*vlan* <*vlan id*>] command. To resolve aging timer inactivity, re-set the aging time to the default value of 300 seconds.

Configuring Static MAC on Ethernet Flow Point and Pseudowire

Static MAC on Ethernet Flow Point (EFP) and Pseudowire (PW) provides the functionality to configure static unicast or multicast MAC address on EFP and PW. A MAC address can be statically added on an EFP under port channel. This feature provides the functionality to:

- Avoid dynamically learning the traffic in both the directions.
- Configure MAC address for Service Instance (SI) and PW.
- Limit the scope of the data traffic flood by creating multicast groups.
- Implement security by explicitly enabling a single MAC address.
- Resolve the problem of MAC address aging out as the dynamic learning is disabled.
- Optimize L2 table performance by limiting the table size.
- Configure static MAC on EFPs on port channels.
- Configure fully meshed pseudowire network between core facing routers and place them under single multicast group.

Restrictions and Usage Guidelines

When configuring static MAC on EFP and PW for the Cisco 7600 routers, follow these guidelines and restrictions:

- You cannot configure unicast static MAC address and MAC security on the same EFP simultaneously. For multicast addresses, static MAC and MAC security can be simultaneously supported under EFP.
- No support for static MAC on PWs on C-MAC Bridge-domain.
- Static MACs are related to a L2 Bridge-domain table, so only the bridged services are supported.
- When static MAC is configured on VPLS PW, and core-facing interface fails resulting in egress interface to move to available interface, the traffic may be delayed.
- Static MAC configuration is supported only on EVC bridge-domain interfaces and VFI pseudowires.
- Static Mac configuration on EFP is supported on ES+ and ESM20 line cards
- Static Mac configuration on VFI PW is supported on ES+, ESM20 and SIP 400 line cards.
- Number of MACs per PW (unicast and multicast) is limited to 1024.

- Number of MACs per Bridge-domain or VFI (unicast and multicast) is limited to 1024.

Number of MACs per system (unicast and multicast) is limited to 1024.This section describes how to configure Static MAC on EFP and PW for the Cisco 7600 router. You need to configure MPLS on core-facing router before configuring static MAC on PW. The information about configuring MPLS on core-facing router is included as a separate section.

Configuring Static MAC over EFP for the Cisco 7600 Router

This section describes how to configure static MAC over EFP or SIs.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet** ***slot/port*** **or interface tengigabitethernet** ***slot/port***
4. **service instance** *id* **Ethernet** [*service-name*]
5. **encapsulation dot1q | untagged | default** *vlan-id*
6. **bridge-domain** *bridge-id*
7. **mac static address mac_address [auto-learn | disable-snooping]**
8. **mac static address** *mac_address*
9. **exit**

DETAILED STEPS

	Command	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 slot/port or interface tengigabitethernet slot/port`**
	Example:
	`Router(config)# Interface GigabitEthernet2/0/0`
		Specifies the Gigabit Ethernet or the Ten Gigabit Ethernet interface to configure, where slot/port specifies the location of the interface.

	Command	Purpose

Step 4	**encapsulation dot1q** {any | vlan-id [vlan-id[vlain-id]]} **second-dot1q** {any | vlan-id[vlan-id[vlan-id]]}
	Example:
	`Router(config-if-srv)# encapsulation dot1q 100 second-dot1q 200`
		Configuring the encapsulation. Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.

Step 5	**bridge-domain** *bridge-id* **Example:** `Router(config-if-srv)# bridge-domain 12`	Configuring the bridge domain. Binds the service instance to a bridge domain instance where *bridge-id* is the identifier for the bridge domain instance.
Step 6	**mac static address** *mac_address* [**auto-learn** | **disable-snooping**] **Example:** `Router(config-if-srv)# mac static address 0002.1122.0010`	Configuring the static mac address for service instance. The option: • auto-learn is used for unicast static MAC address only. This option is not available for multicast static mac address. • disable-snooping is used for multicast static MAC address. This option disables IGMP snooping on the multicast MAC address. The mac_address is in hexadecimal format.
Step 7	**exit** **Example:** `Router(config-if-srv)# exit`	Exits interface configuration mode.

Examples

This example shows how to configure static MAC over EFP or SIs:

```
Router# enable
Router# configure terminal
Router(config-if)# service instance 10 ethernet
Router(config-srv)# encapsulation dot1q 20
Router(config-if-srv)# bridge-domain 100
Router(config-if-srv)# mac static address 0002.1122.0010
Router(config-if-srv)# mac static address 0100.5e00.1111 disable-snooping
Router(config-if-srv)# mac static address 0002.1122.0011 auto-learn
Router(config-if-srv)# mac static address 0100.5e00.1112
Router(config-if-srv)# mac static address 0002.1122.0012 auto-learn
Router(config-if-srv)# mac static address 0100.5e00.1113 disable-snooping
Router(config-if-srv)# exit
```

Configuring MPLS on Core-Facing Interface

You need to configure MPLS on the core-facing router before configuring static MAC over pseudowire. This section describes how to configure MPLS on the core-facing router interface.

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **interface gigabitethernet** *slot/subslot/por***t**
4. **ip address** *ip_Address mask*
5. **mpls ip**
6. **mpls label protocol ldp**
7. **exit**
8. **interface loopback** *Loopback_Id*
9. **ip address** *loopback_address mask*
10. **exit**
11. **mpls ldp** *router-id* **loopback** *loopback_Id* **force**
12. **router ospf** *ospf_Id*
13. **network** *loopback_network wildcard_mask* **area** ***0***
14. **exit**

DETAILED STEPS

	Command	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** *slot/subslot/port* **Example:** `Router(config)# interface gigabitethernet 3/0/0`	Specifies the Gigabit Ethernet interface to configure, where: *slot/subslot/port*—Specifies the location of the interface.
Step 4	**ip address** *ip_Address mask* **Example:** `Router(config-if)# ip address 10.192.0.2 255.255.0.0`	Configures ip address for the interface.
Step 5	**mpls ip** **Example:** `Router(config-if)# mpls ip`	Enables MPLS.

	Command	Purpose
Step 6	**mpls label protocol ldp** **Example:** **Router(config-if)# mpls label protocol ldp**	Configures the mpls parameters.
Step 7	**exit** **Example:** Router(config-if)# exit	Exits interface configuration mode.
Step 8	**interface loopback** loopback_Id **Example:** Router(config)# interface loopback 0	Creates a loopback with the specified loopback_Id.
Step 9	**ip address** loopback_address mask **Example:** Router(config-if)# ip address 1.1.1.1 mask 255.255.255.255	Creates an IP address for the loopback.
Step 10	**exit** **Example:** Router(config)# exit	Exits interface configuration mode.
Step 11	**mpls ldp** router-id **loopback** *loopback_Id* **force** **Example:** Router(config)# mpls ldp router-id loopback 0 force	Configures loopback address as router-id.
Step 12	**router ospf** *ospf_Id* **Example:** Router(config)# router ospf 50	Enables OSPF router configuration mode.
Step 13	**network** loopback_network wildcard_mask **area 0** **Example:** Router(config)# network 192.168.1.1 255.255.255.225 area 0	Defines an interface on which OSPF runs and define the area ID for that interface.
Step 14	**exit** **Example:** outer(config)# exit	Exits the interface configuration mode.

Configuring Static MAC over Pseudowire for the Cisco 7600 Router

This section describes how to configure static MAC over pseudowire.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2 vfi** *vfi_Id* **manual**
4. **vpn id** *vpn_id*
5. **bridge-domain** *bd_number* **vlan**
6. **neighbor** *ip_address* **encapsulation mpls**
7. **mac static address** *mac_address*
8. **exit**
9. **Interface vlan** *vlan_Id*
10. **xconnect vfi** *vfi_Id*
11. **no shutdown**
12. **exit**

DETAILED STEPS

	Command	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	**l2 vfi** *vfi_name* **manual** **Example:** `Router(config-vfi)# l2 vfi smac_vfi manual`	Creates a VFI and enters L2 VFI configuration mode.
Step 4	**vpn id** *vpn_id* **Example:** `Router(config-vfi)# vpn id 30`	Configure the VPN Identifier.

	Command	Purpose
Step 5	**bridge-domain** *bd_number* **vlan** **Example:** `Router(vfi-config)# bridge-domain 40 vlan`	Configures the bridge domain.
Step 6	**neighbor** *ip_address* **encapsulation mpls** **Example:** `Router(vfi-config)# neighbor 192.168.1.1 encapsulation mpls`	Configures the remote peering router-id and tunnel encapsulation type.
Step 7	**mac static address** *mac_address* [**auto-learn** | **disable-snooping**] **Example:** `Router(config-vfi-neighbor)# mac static address 2222.1111.1000`	Configures the unicast and/or multicast static MAC address to the interface. MAC address is in hexadecimal format. Configuring the static mac address for service instance. The option: • auto-learn is used for unicast static MAC address only. This option is not available for multicast static mac address. • disable-snooping is used for multicast static MAC address. This option disables IGMP snooping on the multicast MAC address.
Step 8	**exit** **Example:** `Router(config)# exit`	Exits the VFI configuration mode.
Step 9	**Interface vlan** ***vlan_Id*** **Example:** `Router(config)# interface vlan 40`	Creates an interface VLAN, where the VLAN Id should be same as the bd_number configured in step 5.

	Command	Purpose
Step 10	**xconnect vfi** *VFI_Id* **Example:** `Router(config-if)# xconnect vfi smac_vfi`	Binds the Ethernet or VLAN port to the L2 VFI.
Step 11	**exit** **Example:** `Router(config-if)# exit`	Exits the interface configuration mode.

Examples

This example shows how to configure static MAC over pseudowire.

```
Router# enable
Router# configure terminal
Router(config)# interface GigabitEthernet 4/1/0
Router(config)# l2 vfi foo-core manual
Router(config-vfi)# vpn id 100
Router(config-vfi)# bridge-domain 10 vlan
Router(config-vfi)# neighbor 11.0.0.1 encapsulation mpls
Router(config-vfi-neighbor)# mac static address 0002.1122.0010 auto-learn
Router(config-vfi-neighbor)# mac static address 0100.5e00.1111
Router(config-vfi-neighbor)# mac static address 0002.1122.0011
Router(config-vfi-neighbor)# mac static address 0100.5e00.1112 disable-snooping
Router(config-vfi-neighbor)# mac static address 0002.1122.0012 auto-learn
Router(config-vfi-neighbor)# mac static address 0100.5e00.1113 disable-snooping
Router(config-vfi-neighbor)# interface vlan 10
Router(config-if)# xconnect vfi foo-core
Router(config-vfi)# exit
Router(config)# exit
```

Verification

- You can use the **show bridge-domain** *domain_Id* **mac static address** command to verify the configuration:

```
Bridge-Domain ID : 10
Static MAC count : System : 8, bridge-domain : 8 Port Address Action
vfi foo-core neighbor 1.1.1.1 100 0000.0200.1112
vfi foo-core neighbor 1.1.1.1 100 0000.1111.1001 auto-learn
vfi foo-core neighbor 1.1.1.1 100 0100.5e11.1002
vfi foo-core neighbor 1.1.1.1 100 0100.5e11.1003 disable-snooping
Gi2/0/0 ServInst 2
0000.1111.1003
Gi2/0/0 ServInst 2
0000.1111.1004 auto-learn
Po500 ServInst 1
0000.0000.0777
Po500 ServInst 1 0100.5e00.1111 disable-snooping
```

- You can use the **show ethernet service instance id** *si_Id* **interface** *interface* **mac static address** command to verify the configuration:

```
Router#Router# show ethernet service instance id 1 interface Gi 2/0/0 mac static
address
Bridge domain ID : 10
Port static MAC count : 2
Port Address Action
Gi2/0/0 ServInst 1 0000.1111.1001
Gi2/0/0 ServInst 1 0000.1111.1002 auto-learn
```

- You can use the **show vfi** { **name** *vfi_name*> | **neighbor** *peer_ip_address* **vcid** *id* } **mac static address** command to verify the configuration:

```
Router#show vfi neighbor 1.1.1.1 vcid 100 mac static address
Bridge domain ID : 10
Port Address Action
vfi foo-core neighbor 1.1.1.1 100 0000.0200.1112
vfi foo-core neighbor 1.1.1.1 100 0000.1111.1001
vfi foo-core neighbor 1.1.1.1 100 0000.1111.1002 auto-learn
vfi foo-core neighbor 1.1.1.1 100 0100.5e11.1002
```

Troubleshooting

Table 2-19 provides the troubleshooting solutions for the Static MAC binding feature.

Table 2-19 Troubleshooting Static MAC binding feature

Problem	Solution
Pseudowire (PW) state changes	Complete these steps: **1.** If a PW is down, flush all the static MAC addresses configured within the PW. **2.** If the PW is up, re-install all the static MAC addresses configured within the PW. **3.** If there is a PW change in egress due to load-balancing or FRR, update all static MAC addresses configured within the PW in the HW MAC table to use the new egress information.
MAC address is not installed or deleted from the MAC address table Data is not synchronized with the standby supervisor EFP or PW is disabled	Use the **debug mac static** [*event* | *error* | *ha* | *issu*] command to confirm if the MAC address (configured through static mac over EFP/PW feature) is installed or deleted from the mac address table and if the data is synchronized to the standby supervisor. Share the output with TAC for further investigation.

Configuring Resilient Ethernet Protocol

Resilient Ethernet Protocol (REP) is a Cisco proprietary protocol that provides an alternative to Spanning Tree Protocol (STP) to support L2 resiliency, and fast switchover with Ethernet networks. REP is a segment protocol that integrates easily into existing Carrier Ethernet networks. It allows network architects to limit the scope of STP domains. A REP segment is a connected chain of ports configured with a segment ID. Each segment consists of standard (non-edge) segment ports, and two user-configured edge ports. REP is supported on Layer 2 trunk interfaces and EVC ports.

REP provides functionality to:

- Control network loops
- Handle link failures
- Improve convergence time
- Control a group of ports connected in a segment
- Ensure that the segment does not create any bridging loops
- Respond to link failures within the segment
- Provide a basis for constructing more complex networks
- Support VLAN load balancing at service instance level
- Extend the network resiliency across Cisco IP Next-Generation Network (NGN) Carrier Ethernet Design
- Notify the STP about potential topology changes, allowing interoperability with Spanning Tree
- Control data traffic

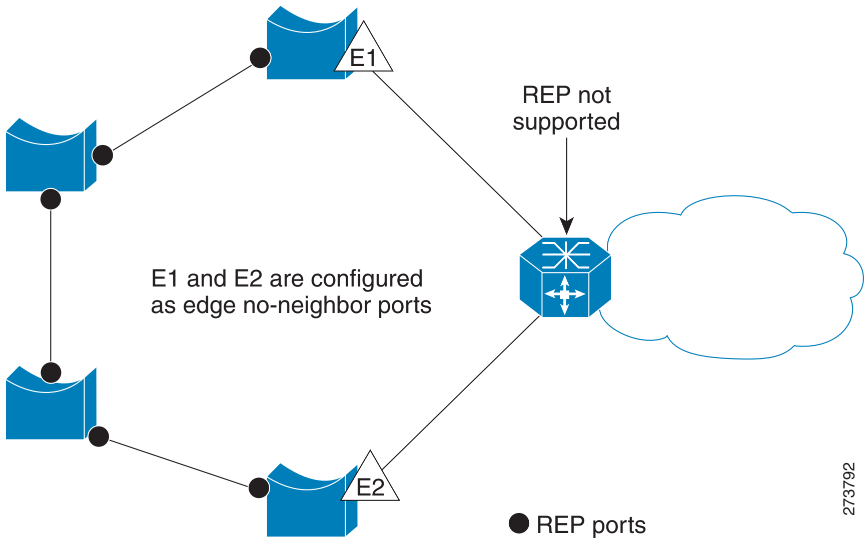
The ports on a C7600 platform are classified into three different types: switchports, routed ports, and EVC ports. By default, a port is a routed port. REP is not supported on routed ports. You need to configure a port to a switchport or EVC port to configure REP on it. A port that is configured with one or more service instances is called an EVC port.

REP is a distributed and secure protocol and does not rely on a master node controlling the status of the ring. Hence, the failures can be detected locally either through loss of signal (LOS) or loss of neighbor adjacency. Any REP port can initiate a switchover after acquiring the secure key to unblock the alternate port. An REP segment is a chain of ports connected to each other and configured with the same segment ID. Each end of a segment terminates on an edge switch. The port where the segment terminates is called the edge port.

REP Edge No-Neighbor

Effective from Cisco IOS release 15.1(01)S, a new functionality provides capability to configure the non-rep switch facing ports as edge no-neighbor ports. These ports inherit the properties of edge ports, and overcome the limitation of not being able to converge quickly during a failure.

Figure 2-7 Edge No-Neighbor Ports



In access ring topologies, the neighboring switch might not support REP, as shown in Figure 4-2. In this case, you can configure the non-REP facing ports (E1 and E2) as edge no-neighbor ports. These ports inherit all the properties of edge ports. You can configure these no-neighbor ports as any other edge port and also enable the ports to send STP or REP topology change notices to the aggregation switch. In this case the STP Topology Change Notice (TCN) that is sent is a Multiple Spanning-Tree (MST) STP message.

These sections describes how to configure REP for the Cisco 7600 router:

- Configuring REP over Ethernet Virtual Circuit, page 2-203
- Configuring Resilient Ethernet Protocol Configurable Timers, page 2-220

Configuring REP over Ethernet Virtual Circuit

The REP over Ethernet Virtual Circuit (EVC) feature allows you to configure and manage ports on the interface, and not per service instance and can be used to block and unblock encapsulated vlans at service level.

An EVC port can have multiple service instances. Each service instance corresponds to a unique Event Flow Processor (EFP). By default, REP is disabled on all ports. REP can be configured only REP can selectively block or forward data traffic on particular VLANs. For EVC, the VLAN Id refers to the outer tag of the dot1q encapsulation that is configured on a service instance. REP is supported on a bridge-domain service. If **ethernet vlan color-block all** command is configured, REP is supported on connect and cross connect services.

Restrictions and Usage Guidelines

When configuring REP over EVC for the Cisco 7600 router, follow these guidelines and restrictions:

- REP is not supported on service instances configured with encapsulation, untagged, or default type.

- Cisco recommends that you begin by configuring one port and then configure the contiguous ports to minimize the number of segments and the number of blocked ports.
- REP is not supported on mLACP.
- If more than two ports in a segment fail when no external neighbors are configured, one port goes into a forwarding state to maintain connectivity during configuration.
- REP ports must be Layer 2 trunk ports or EVC ports.
- To avoid misconfiguration, you must configure all trunk ports in the segment with the same set of allowed VLANs.
- Since REP blocks all VLANs until another REP interface sends a message to unblock it, you might loose connectivity to the port. This happens if you enable REP in a Telnet session that accesses the EVC port through the same interface.
- You cannot execute REP and STP/MST, or REP and Flex Links on the same segment or interface.
- You cannot run REP and STP/MST, or REP and Flex Links on the same segment or interface.
- If you connect an STP network to the REP segment, ensure that the connection is at the segment edge. An STP connection that is not at the edge causes a bridging loop because STP does not run on REP segments. All STP BPDUs are dropped at REP interfaces.
- On a router if REP is enabled on two ports for a segment, both ports must either be a regular segment ports or edge ports. REP ports follow these rules on a router:
  - If only one port is configured in a segment, the port should be an edge port.
  - If two ports belong to the same segment, both ports must be edge ports or the regular segment ports.
  - If two ports belong to the same segment and one is configured as an edge port and other as a regular segment port, the edge port is treated as a regular segment port.
  - There can be only two edge ports in a segment, if there are two edge routers in a segment, each router can have only one edge port. All the other ports on the edge router function as normal ports.
- REP interfaces come up in a blocked state and remains in a blocked state until notified that it is safe to unblock.
- REP relays all LSL PDUs in untagged frames on the native VLAN andonly HFL packets are relayed on the admin vlan. However, BPA messages are untagged as well.
- REP ports cannot be configured as:
  - SPAN destination port
  - Private VLAN port
  - Tunnel port
  - Access port
- REP is supported on EtherChannels, but not on an individual port that belongs to an EtherChannel. It is supported on Swichports and EVC port-channels. REP is implemented on port-channels instead of its individual member links.
- In case of double VLAN tagged frame, REP is implemented only on the outer VLAN tag.
- When an edge no-neighbor is configured on a router, configuring and unconfiguring an edge port is not allowed.

Configuring REP over EVC for the Cisco 7600 Router

This section describes how to configure REP over EVC for the Cisco 7600 router:

- Configuring REP over EVC using Cross connect for the Cisco 7600 Router, page 2-205
- Configuring REP over EVC using connect for the Cisco 7600 Router, page 2-208
- Configuring REP over EVC using bridge domain for the Cisco 7600 Router, page 2-213

Configuring REP over EVC using Cross connect for the Cisco 7600 Router

This section describes how to configure REP over EVC using cross connect at the global configuration level.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitEthernet** *slot/subslot/port*
4. **ether vlan color-block all**
5. **service instance** *id* {**ethernet** [*service-name*}
6. **encapsulation dot1q** *vlan_id*
7. **rewrite ingress tag**
8. **xconnect** *loopback_ip vc_id* encapsulation mpls
9. **rep segment** *segment_id* **[edge [no-neighbor][primary]] [preferred]**
10. **exit**

DETAILED STEPS

	Command	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`** *`slot`*`/subslot/`*`port`*
	Example:
	`Router(config)# interface gigabitethernet 4/0/5`
		Specifies the Gigabit Ethernet interface to configure, where:
		• *slot/subslot/port*—Specifies the location of the interface.

	Command	Purpose

Step 4	`ether vlan color-block all`
	Example:
	`Router(config-if)# ether vlan color-block all`
		Configures REP to block xconnect type of service instances.

Step 5	**`service instance`** `id` **`ethernet`** `[service-name]`
	Example:
	`Router(config-if)# service instance 101 ethernet`
		Creates a service instance (an instance of an EVC) on an interface and sets the device into the config-if-srv submode.

Step 6	**`encapsulation dot1q`** `{any | vlan-id[vlan-id[-vlan-id]]}`
	Example:
	`Router(config-if-srv)# encapsulation dot1q 100 second-dot1q 200`
		Configures the encapsulation. Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.

Step 7	**`rewrite ingress tag`**
	Example:
	`Router(config-if-srv)# rewrite ingress tag dot1q single symmetric`
		Specifies the tag manipulation that is to be performed on the frame ingress to the service instance.

Step 8	**`xconnect`** `loopback_id vc_id` **`encapsulation mpls`**
	Example:
	`Router(config-if-srv)# xconnect 10.0.0.2 999 encapsulation mpls`
		Configures forwarding mechanism on a service instance. Ensure that the MPLS connectivity is up.

	Command	Purpose
Step 9	**rep segment** *segment_id* **[edge [no-neighbor][primary]] [preferred]** **Example:** `Router(config-if)# rep segment 3 edge`	Configures the REP over EVC. The segment ID range is from 1 to 1024. **Note** You must configure two edge ports, including one primary edge port for each segment. These optional keywords are available. • Enter **edge** to configure the port as an edge port. Entering **edge** without the **primary** keyword configures the port as the secondary edge port. Each segment has only two edge ports. • On an edge port, enter **primary** to configure the port as the primary edge port, the port on which you can configure VLAN load balancing. **Note** Although each segment can have only one primary edge port, if you configure edge ports on two different switches and enter the **primary** keyword on both switches, the configuration is allowed. However, REP selects only one of these ports as the segment primary edge port. You can identify the primary edge port for a segment by using the **show rep topology** privileged EXEC command. • On an edge port, use the **no-neighbor** keyword to configure the segment edge with no external rep neighbor. • Enter **preferred** to indicate that the port is the preferred alternate port, or the preferred port for VLAN load balancing. Configuring a port as preferred does not guarantee that it becomes the alternate port; it merely gives it a slight edge among equal contenders. The alternate port is usually a previously failed port.
Step 10	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.

Examples

This example shows how to configure REP over EVC using cross connect.

```
Router# enable
Router# configure terminal
Router(config)# interface GigabitEthernet 5/3
Router(config-if-srv)# rep segment 3 edge
Router(config-if)# service instance 10 ethernet
Router(config-srv)# encapsulation dot1q 20
Router(config-if-srv)# rewrite ingress tag pop 2 sym
Router(config-if-srv)# xconnect 10.0.0.2 999 encapsulation MPLS
Router(config-if-srv)# exit
```

Configuring REP over EVC using connect for the Cisco 7600 Router

This section describes how to configure REP over EVC using connect at global configuration level.

SUMMARY STEPS

1. enable
2. **configure terminal**
3. ***interface type*** *slot/subslot/port*
4. **ether vlan color-block all**
5. **service instance** *id* **{Ethernet [***service-name***}**
6. **encapsulation dot1q** ***vlan_id***
7. **rewrite ingress tag** {**push {dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} [**symmetric**]
8. exit
9. **rep segment** *segment_id* **[edge [no-neighbor] [primary]] [preferred]**
10. exit
11. ***interface type*** *slot/subslot/port*
12. **ether vlan color-block all**
13. **service instance** *id* **{Ethernet [***service-name***}**
14. **encapsulation dot1q** ***vlan_id***
15. **rewrite ingress tag** {**push {dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} [**symmetric**]
16. exit
17. **rep segment** *segment_id* [**edge [no-neighbor] [primary]] [preferred]**
18. exit
19. **connect** <*connect_name*> <*interface*> <*service_instance_id*> <*interface*> <*service_instance_id*>

DETAILED STEPS

	Command	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** *slot/subslot/port* **Example:** `Router(config)# interface gigabitethernet 3/0/0`	Specifies the Gigabit Ethernet interface to configure, where: • *slot/subslot/port*—Specifies the location of the interface.
Step 4	**ether vlan color-block all** **Example:** `Router(config-if)# Ether vlan color-block all`	Configures REP to block connect type of service instances.
Step 5	**service instance** id **Ethernet** [service-name] **Example:** `Router(config-if)# service instance 10 ethernet`	Creates a service instance (an instance of an EVC) on an interface and sets the device into the config-if-srv submode.

Step 6	**encapsulation dot1q** {any | vlan-id[vlan-id[-vlain-id]]} second-dot1q {any | vlan-id[vlan-id[-vlan-id]]}
	Example:
	`Router(config-if-srv)# encapsulation dot1q 10`
		Configures the encapsulation. Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.

	Command	Purpose
Step 7	**rewrite ingress tag** {**push {dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} [**symmetric**] **tag pop** id **symmetric** **Example:** `Router(config-if-srv)# rewrite ingress tag pop 1 symmetric`	Specifies the tag manipulation that is to be performed on the frame ingress to the service instance.

Step 8	`exit`
	Example:
	`Router(config-if-srv)# exit`
		Exits service instance mode.

	Command	Purpose
Step 9	**rep segment** *segment_id* **[edge [no-neighbor][primary]] [preferred]** **Example:** `Router(config-if)# rep segment 2 edge primary`	Configures REP over EVC. The segment ID range is from 1 to 1024. **Note** You must configure two edge ports, including one primary edge port for each segment. These optional keywords are available. • Enter **edge** to configure the port as an edge port. Entering **edge** without the **primary** keyword configures the port as the secondary edge port. Each segment has only two edge ports. • On an edge port, enter **primary** to configure the port as the primary edge port, the port on which you can configure VLAN load balancing. **Note** Although each segment can have only one primary edge port, if you configure edge ports on two different switches and enter the **primary** keyword on both switches, the configuration is allowed. However, REP selects only one of these ports as the segment primary edge port. You can identify the primary edge port for a segment by using the **show rep topology** privileged EXEC command. • On an edge port, use the **no-neighbor** keyword to configure the segment edge with no external rep neighbor. • Enter **preferred** to indicate that the port is the preferred alternate port, or the preferred port for VLAN load balancing. Configuring a port as preferred does not guarantee that it becomes the alternate port; it merely gives it a slight edge among equal contenders. The alternate port is usually a previously failed port.

Step 10	`exit`
	Example:
	`Router(config-if)# exit`
		Exits interface configuration mode.

Step 11	**interface gigabitethernet** *slot/subslot/port* **Example:** `Router(config)# interface gigabitethernet 4/0/0`	Specifies the Gigabit Ethernet interface to configure, where: *slot/subslot/port*—Specifies the location of the interface.

	Command	Purpose

Step 12	`ether vlan color-block all`
	Example:
	`Router(config-if)# Ether vlan color-block all`
		Configures REP to block connect type of service instances.

Step 13	**service instance** id **Ethernet** [service-name] **Example:** `Router(config-if)# service instance 102 ethernet`	Creates a service instance (an instance of an EVC) on an interface and sets the device into the config-if-srv submode.

Step 14	***encapsulation dot1q*** *{any | vlan-id[vlan-id[-vlain-id]]}* ***second-dot1q*** *{any | vlan-id[vlan-id[-vlan-id]]}*
	Example:
	`Router(config-if-srv)# encapsulation dot1q 100 second dot1q 200`
		Configures the encapsulation. Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.

Step 15	**rewrite ingress tag** {**push {dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} [**symmetric**] **tag pop** id **symmetric** **Example:** `Router(config-if-srv)# rewrite ingress tag push dot1q 20`	Specifies the tag manipulation that is to be performed on the frame ingress to the service instance.

Step 16	`exit`
	Example:
	`Router(config-if-srv)# exit`
		Exits service instance mode.

Step 17	**rep segment** *segment_id* [**edge [no-neighbor][primary]] [preferred]** **Example:** `Router(config-if)# rep segment 2 edge primary`	Configures REP over EVC.

	Command	Purpose

Step 18	`exit`
	Example:
	`Router(config-if)# exit`
		Exits interface configuration mode.

Step 19	**`connect`** *`<connect_name> <interface> <service_instance_id> <interface> <service_instance_id>`*
	Example:
	`outer(config)#` **`connect test gigabitEthernet 3/0/0 10 gigabitEthernet 4/0/0 20`**
		Configures local connect between the two service instances of two different interfaces.

Examples

This example shows how to configure REP over EVC using connect.

```
Router# enable
Router# configure terminal
Router(config)# interface gigabitEthernet 3/0/0
Router(config-if)# ether vlan color-block all
Router(config-if)# service instance 10 ethernet
Router(config-if-srv)# encapsulation dot1q 10
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
Router(config-if-srv)# exit
Router(config-if)# rep segment 2 edge primary
Router(config-if)# exit
Router(config)# interface gigabitEthernet 4/0/0
Router(config-if)# service instance 20 ethernet
Router(config-if-srv)# encapsulation dot1q 20
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
Router(config-if-srv)# exit
Router(config-if)# rep segment 2 edge
Router(config-if)# exit
Router(config)#connect test gigabitEthernet 3/0/0 10 gigabitEthernet 4/0/0 20
Router(config-connection)#end
```

Configuring REP over EVC using bridge domain for the Cisco 7600 Router

This section describes how to configure REP over EVC using bridge domain at service instance level.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. ***i*nterface type** *slot/subslot/port*
4. **service instance** *id* **{Ethernet** [*service-name*}
5. **encapsulation dot1q** *vlan_id*

6. **rewrite ingress tag** {**push** {**dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} {**symmetric**}
7. **bridge-domain** ***bd_Id***
8. **exit**
9. **rep segment** *segment_id* [**edge** [**no-neighbor**] [**primary**]] [**preferred**]
10. **exit**

DETAILED STEPS

	Command	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`** *`slot`*`/subslot/`*`port`*
	Example:
	`Router(config)# interface gigabitethernet 3/0/0`
		Specifies the Gigabit Ethernet interface to configure, where:
		slot/subslot/port—Specifies the location of the interface.

Step 4	**`service instance`** `id` **`Ethernet`** `[service-name]`
	Example:
	`Router(config-if)# service instance 101 ethernet`
		Creates a service instance (an instance of an EVC) on an interface and sets the device into the config-if-srv submode.

Step 5	**`encapsulation dot1q`** `{any | vlan-id[vlan-id[-vlan-id]]} second-dot1q {any | vlan-id[vlan-id[-vlan-id]]}`
	Example:
	`Router(config-if-srv)# encapsulation dot1q 100 second dot1q 200`
		Configures the encapsulation. Defines the matching criteria to be used in order to map ingress dot1q frames on an interface to the appropriate service instance.

	Command	Purpose

Step 6	**rewrite ingress tag** {**push** {**dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} {**symmetric**}
	Example:
	`Router(config-if-srv)# rewrite ingress tag push dot1q 20`
		Specifies the tag manipulation that is to be performed on the frame ingress to the service instance.

Step 7	`bridge-domain bd_Id`
	Example:
	`Router(config-if-srv)# bridge-domain 10`
		Configures bridge-domain to add another VLAN tag of type bridge-domain to the incoming packet.

Step 8	`exit`
	Example:
	`Router(config-if-srv)# exit`
		Exits service instance mode.

	Command	Purpose
Step 9	**rep segment** *segment_id* **[edge [no-neighbor] [primary]] [preferred]** **Example:** `Router(config-if)# rep segment 2 edge primary`	Configures REP over EVC. The segment ID range is from 1 to 1024. **Note** You must configure two edge ports, including one primary edge port for each segment. These optional keywords are available. • Enter **edge** to configure the port as an edge port. Entering **edge** without the **primary** keyword configures the port as the secondary edge port. Each segment has only two edge ports. • On an edge port, enter **primary** to configure the port as the primary edge port, the port on which you can configure VLAN load balancing. **Note** Although each segment can have only one primary edge port, if you configure edge ports on two different switches and enter the **primary** keyword on both switches, the configuration is allowed. However, REP selects only one of these ports as the segment primary edge port. You can identify the primary edge port for a segment by using the **show rep topology** privileged EXEC command. • On an edge port, use the **no-neighbor** keyword to configure the segment edge with no external rep neighbor. • Enter **preferred** to indicate that the port is the preferred alternate port, or the preferred port for VLAN load balancing. Configuring a port as preferred does not guarantee that it becomes the alternate port; it merely gives it a slight edge among equal contenders. The alternate port is usually a previously failed port.
Step 10	**exit** **Example:** `Router(config-if)# end`	Exits global configuration mode.

Examples

This example shows how to configure REP over EVC using bridge domain.

```
Router# enable
Router# configure terminal
Router(config)# interface gigabitEthernet 3/0/0
Router(config-if)# service instance 10 ethernet
Router(config-if-srv)# encapsulation dot1q 10
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
Router(config-if-srv)# bridge-domain 100
Router(config-if-srv)# exit
Router(config-if)# rep segment 2 edge
Router(config-if)# end
```

This example shows how to configure REP with the edge **no-neighbor** keyword.

```
Router# enable
Router# configure terminal
Router(config)# interface gigabitEthernet 7/1
Router(config-if)# switchport
Router(config-if)# switchport mode trunk
Router(config-if)# rep segment 1 edge no-neighbor primary
Router(config-if)# end
```

Verification

You can use the **show rep topology, show rep topology detail** and **show interface rep** commands to verify the REP over EVC configuration. This information is displayed as sample output:

- Specific EVCs if an EVC ID is specified.
- All the EVCs on an interface if an interface is specified.
- The detailed option provides additional information about the EVC. This can be given on RP and LC consoles to determine custom ethertype configured under a physical port.

Example of **show rep topology** command:

```
Router#show rep topology
REP Segment 3
BridgeName PortName Edge Role
---------------- ---------- ---- ----
Router Gi4/0/0 Pri Open
REP-ALPHA Gi2/12 Open
REP-ALPHA Fa3/1 Open
REP-BETA Fa1/1 Open
REP-BETA Gi6/1 Open
Router Gi3/4 Sec Alt
--
```

Example of **show rep topology detail** command.

```
Router#show rep topology segment 3 detail
REP Segment 3
Router, Gi4/0/0 (Primary Edge)
Open Port, all vlans forwarding
Bridge MAC: 0015.fa66.ff80
Port Number: 0301
Port Priority: 000
Neighbor Number: 1 / [-6]
REP-ALPHA, Gi2/12 (Intermediate)
Open Port, all vlans forwarding
Bridge MAC: 0005.7495.cd00
Port Number: 010C
Port Priority: 000
Neighbor Number: 2 / [-5]
REP-ALPHA, Fa3/1 (Intermediate)
Open Port, all vlans forwarding
Bridge MAC: 0005.7495.cd00
Port Number: 0201
Port Priority: 000
Neighbor Number: 3 / [-4]
REP-BETA, Fa1/1 (Intermediate)
Open Port, all vlans forwarding
Bridge MAC: 0005.7495.c900
```

```
Port Number: 001
Port Priority: 000
Neighbor Number: 4 / [-3]
REP-BETA, Gi6/1 (Intermediate)
Open Port, all vlans forwarding
Bridge MAC: 0005.7495.c900
Port Number: 0501
Port Priority: 000
Neighbor Number: 5 / [-2]
Router, Gi3/4 (Secondary Edge)
Alternate Port, some vlans blocked
Bridge MAC: 0015.fa66.ff80
Port Number: 0204
Port Priority: 010
Neighbor Number: 6 / [-1]
```

Example of **show interface <> rep** command:

```
Router#show interface gig4/0/0 rep detail
GigabitEthernet4/0/0 REP enabled
Segment-id: 3 (Primary Edge)
PortID: 03010015FA66FF80
Preferred flag: No
Operational Link Status: TWO_WAY
Current Key: 02040015FA66FF804050
Port Role: Open
Blocked VLAN: <empty>
Admin-vlan: 1
Preempt Delay Timer: disabled
Configured Load-balancing Block Port: none
Configured Load-balancing Block VLAN: none
STCN Propagate to: none
LSL PDU rx: 999, tx: 652
HFL PDU rx: 0, tx: 0
BPA TLV rx: 500, tx: 4
BPA (STCN, LSL) TLV rx: 0, tx: 0
BPA (STCN, HFL) TLV rx: 0, tx: 0
EPA-ELECTION TLV rx: 6, tx: 5
EPA-COMMAND TLV rx: 0, tx: 0
EPA-INFO TLV rx: 135, tx: 136
```

Show outputs for REP with Edge No-Neighbor keyword

Example of **show rep topology** command with REP edge **no-neighbor** keyword:

```
Router#show rep topology
REP Segment 3
BridgeName       PortName   Edge  Role
---------------- ---------- ----  ----
sw8-ts8-51       Gi0/2      Pri*  Open
sw9-ts11-50      Gi1/0/4          Open
sw9-ts11-50      Gi1/0/2          Open
sw1-ts11-45      Gi0/2            Alt
sw1-ts11-45      Po1              Open
sw8-ts8-51       Gi0/1      Sec*  Open
--
```

Example of **show rep topology detail** command with REP edge **no-neighbor** keyword:

```
Router#show rep topology segment 3 detail
REP Segment 3
Router, Gi4/0/0 (Primary Edge No-Neighbor)
Open Port, all vlans forwarding
Bridge MAC: 0015.fa66.ff80
```

```
    Port Number: 0301
    Port Priority: 000
    Neighbor Number: 1 / [-6]
    REP-ALPHA, Gi2/12 (Intermediate)
    Open Port, all vlans forwarding
    Bridge MAC: 0005.7495.cd00
    Port Number: 010C
    Port Priority: 000
    Neighbor Number: 2 / [-5]
    REP-ALPHA, Fa3/1 (Intermediate)
    Open Port, all vlans forwarding
    Bridge MAC: 0005.7495.cd00
    Port Number: 0201
    Port Priority: 000
    Neighbor Number: 3 / [-4]
    REP-BETA, Fa1/1 (Intermediate)
    Open Port, all vlans forwarding
    Bridge MAC: 0005.7495.c900
    Port Number: 001
    Port Priority: 000
    Neighbor Number: 4 / [-3]
    REP-BETA, Gi6/1 (Intermediate)
    Open Port, all vlans forwarding
    Bridge MAC: 0005.7495.c900
    Port Number: 0501
    Port Priority: 000
    Neighbor Number: 5 / [-2]
    Router, Gi3/4 (Secondary Edge)
    Alternate Port, some vlans blocked
    Bridge MAC: 0015.fa66.ff80
    Port Number: 0204
    Port Priority: 010
Neighbor Number: 6 / [-1]
```

Example of **show interface <> rep** command with REP edge **no-neighbor** keyword:

```
Router#show interface gig4/0/0 rep detail
GigabitEthernet4/0/0 REP enabled
Segment-id: 3 (Primary Edge No-Neighbor)
PortID: 03010015FA66FF80
Preferred flag: No
Operational Link Status: TWO_WAY
Current Key: 02040015FA66FF804050
Port Role: Open
Blocked VLAN: <empty>
Admin-vlan: 1
Preempt Delay Timer: disabled
Configured Load-balancing Block Port: none
Configured Load-balancing Block VLAN: none
STCN Propagate to: none
LSL PDU rx: 999, tx: 652
HFL PDU rx: 0, tx: 0
BPA TLV rx: 500, tx: 4
BPA (STCN, LSL) TLV rx: 0, tx: 0
BPA (STCN, HFL) TLV rx: 0, tx: 0
EPA-ELECTION TLV rx: 6, tx: 5
EPA-COMMAND TLV rx: 0, tx: 0
EPA-INFO TLV rx: 135, tx: 136
```

Configuring Resilient Ethernet Protocol Configurable Timers

The REP Configurable Timer (REP Fast Hellos) feature provides a fast re-convergence in a ring topology with higher timer granularity and quicker failure detection on the remote side. The feature also supports improved convergence of REP segments having nodes with copper based SFP's, where the link detection time varies between 300ms to 700 ms.

With the REP Link Status Layer (LSL) ageout timer configuration, the failure detection time can be configured between a range of 120 millisecond to 10,000 millisecond, in multiples of 40ms. The result of this configuration is that, even if the copper pull takes about 700 ms to notify the remote end about the failure, the REP Configurable Timers process will detect it much earlier and takes subsequent action for the failure recovery with in 200 ms.

Restrictions and Usage Guidelines

Follow these guidelines and restrictions:

- The LSL Age Out Timer configuration is available on switchports, EVC, L2 Port-channel and Port-channel EVC interfaces.
- The SUP 720, RSP 720, RSP 10G supervisors and the ES20, ES40, and LAN line cards support the REP Configurable Timers configuration.
- While configuring REP configurable timers, we recommend you shut the port, configure REP and only then use the no shut command. This prevents the REP from flapping and generating large number of internal messages.
- If incompatible switches are neighbors, configure the correct LSL Age Out value first. In some scenarios, you might not get the expected convergence range.
- In order to inter-operate with switches running old IOS versions, the default LSL Age Out time is set to 5 seconds, default LSL retries is 5, and the hello packet is sent every one second.
- Except for the LSL Age Out time, all the other timer values are retained. For example, the EPA (End Port Advertisement) hello timer continues to be 4 seconds, as it is not required to send EPA PDUs at a higher frequency.
- While configuring REP configurable timers, we recommend you configure the REP LSL number of retries first and then configure the REP LSL age out timer value.
- Effective from Cisco IOS release 15.1(2)S:
  - The REP Configurable Timers feature is SSO compliant for RSP720, RSP10G (endor) and SUP720 supervisors.
  - The REP Configurable Timers feature on SSO is not supported with SUP32 supervisor.
  - The REP LSL Age Out value can be configured as low as 1520 ms (approximately 500 ms * 3) for HA systems as this prevents traffic loss.
- The REP Configurable Timers feature is supported only on Cisco 7600 S-chassis.

Configuring REP Configurable Timers for the Cisco 7600 Router

This section describes how to configure the LSL age out timer and the LSL number of retries on a Cisco 7600 router:

- Configuring the REP Link Status Layer Retries, page 2-221
- Configuring the REP Link Status Layer Age Out Timer, page 2-225

Configuring the REP Link Status Layer Retries

This section describes how to configure REP link status layer number of retries at interface configuration level.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface type** *slot/subslot/port*
4. **rep segment** *segment_id* [**edge** [**no-neighbor**] [**primary**]] [**preferred**]
5. **rep lsl-retries** <*no-of-retries*>
6. **end**

DETAILED STEPS

	Command	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`*`/subslot/`*`port`*
	Example:
	`Router(config)# interface gigabitethernet 2/0/3`
		Specifies the Gigabit Ethernet, Ten Gigabit Ethernet and Port Channel interfaces to configure, where:
		- *slot/subslot/port*—Specifies the location of the interface.

	Command	Purpose
Step 4	**rep segment** *segment_id* **[edge [no-neighbor] [primary]] [preferred]** **Example:** `Router(config-if)# rep segment 2 edge primary`	Configures the REP. The segment ID range is from 1 to 1024. **Note** You must configure two edge ports, including one primary edge port for each segment. These optional keywords are available. • Enter **edge** to configure the port as an edge port. Entering **edge** without the **primary** keyword configures the port as the secondary edge port. Each segment has only two edge ports. • On an edge port, enter **primary** to configure the port as the primary edge port, the port on which you can configure VLAN load balancing. **Note** Although each segment can have only one primary edge port, if you configure edge ports on two different switches and enter the **primary** keyword on both switches, the configuration is allowed. However, REP selects only one of these ports as the segment primary edge port. You can identify the primary edge port for a segment by using the **show rep topology** privileged EXEC command. • On an edge port, use the **no-neighbor** keyword to configure the segment edge with no external rep neighbor. • Enter **preferred** to indicate that the port is the preferred alternate port, or the preferred port for VLAN load balancing. Configuring a port as preferred does not guarantee that it becomes the alternate port; it merely gives it a slight edge among equal contenders. The alternate port is usually a previously failed port.
Step 5	**rep lsl-retries** *<no-of-retries>* **Example:** `Router(config-if)#` **`rep lsl-retries`** `4`	Configures the number of retries before the REP link is disabled. The acceptable range of retries is 3-10. The default LSL number of retries is 5.

Step 6	**end**
	
	Example:
	`Router(config-if)# end`
		Exits configuration mode.

Examples

This example shows how to configure REP link status layer number of retries.

```
Router# enable
Router# configure terminal
Router(config)# interface gigabitethernet 2/0/3
Router(config-if)# rep segment 2 edge primary
Router(config-if)# rep lsl-retries 4
Router(config-if)# end
```

Configuring the REP Link Status Layer Age Out Timer

This section describes how to configure the REP Link Status Layer Age Out Timer at interface configuration level.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface type** *slot/subslot/port*
4. **rep segment** *segment_id* [**edge** [**no-neighbor**] [**primary**]] [**preferred**]
5. **rep lsl-age-timer** <*lsl-age-timer*>
6. **end**

DETAILED STEPS

	Command	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*/subslot/*port* **Example:** `Router(config)# interface gigabitethernet 5/0/2`	Specifies the Gigabit Ethernet, Ten Gigabit Ethernet and Port Channel interfaces to configure, where: • *slot/subslot/port*—Specifies the location of the interface.

	Command	Purpose
Step 4	**rep segment** *segment_id* **[edge [no-neighbor] [primary]] [preferred]** **Example:** `Router(config-if)# rep segment 1 edge primary`	Configures the REP. The segment ID range is from 1 to 1024. **Note** You must configure two edge ports, including one primary edge port for each segment. These optional keywords are available. • Enter **edge** to configure the port as an edge port. Entering **edge** without the **primary** keyword configures the port as the secondary edge port. Each segment has only two edge ports. • On an edge port, enter **primary** to configure the port as the primary edge port, the port on which you can configure VLAN load balancing. **Note** Although each segment can have only one primary edge port, if you configure edge ports on two different switches and enter the **primary** keyword on both switches, the configuration is allowed. However, REP selects only one of these ports as the segment primary edge port. You can identify the primary edge port for a segment by using the **show rep topology** privileged EXEC command. • On an edge port, use the **no-neighbor** keyword to configure the segment edge with no external rep neighbor. • Enter **preferred** to indicate that the port is the preferred alternate port, or the preferred port for VLAN load balancing. Configuring a port as preferred does not guarantee that it becomes the alternate port; it merely gives it a slight edge among equal contenders. The alternate port is usually a previously failed port.
Step 5	rep lsl-age-timer <*lsl-age-timer*> **Example:** `Router(config-if)# rep lsl-age-timer 2000`	Configures REP link status layer age out timer value. The acceptable range of *lsl-age-timer* is between 120ms and 10000ms, in multiples of 40ms. The default LSL Age Out time is 5 seconds.

Step 6	**end**
	Example:
	`Router(config-if)# end`
		Exits configuration mode.

Examples

This example shows how to configure REP link status layer number of retries.

```
Router# enable
Router# configure terminal
```

```
Router(config)# interface gigabitethernet 2/0/3
Router(config-if)# rep segment 2 edge primary
Router(config-if)# rep lsl-retries 4
Router(config-if)# end
```

Configuring the REP Link Status Layer Age Out Timer

This section describes how to configure the REP Link Status Layer Age Out Timer at interface configuration level.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface type** *slot/subslot/port*
4. **rep segment** *segment_id* [**edge** [**primary**]] [*preferred*]
5. **rep lsl-age-timer** <*lsl-age-timer*>
6. **end**

DETAILED STEPS

	Command	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*/subslot/*port* **Example:** `Router(config)# interface gigabitethernet 5/0/2`	Specifies the Gigabit Ethernet, Ten Gigabit Ethernet and Port Channel interfaces to configure, where: • *slot/subslot/port*—Specifies the location of the interface.

	Command	Purpose
Step 4	**rep segment** *segment_id* **[edge [primary]] [preferred]** **Example:** `Router(config-if)# rep segment 1 edge primary`	Configures the REP. The segment ID range is from 1 to 1024. **Note** You must configure two edge ports, including one primary edge port for each segment. These optional keywords are available. • Enter **edge** to configure the port as an edge port. Entering **edge** without the **primary** keyword configures the port as the secondary edge port. Each segment has only two edge ports. • On an edge port, enter **primary** to configure the port as the primary edge port, the port on which you can configure VLAN load balancing. **Note** Although each segment can have only one primary edge port, if you configure edge ports on two different switches and enter the **primary** keyword on both switches, the configuration is allowed. However, REP selects only one of these ports as the segment primary edge port. You can identify the primary edge port for a segment by entering the **show rep topology** privileged EXEC command. • Enter **preferred** to indicate that the port is the preferred alternate port or the preferred port for VLAN load balancing. Configuring a port as preferred does not guarantee that it becomes the alternate port; it merely gives it a slight edge among equal contenders. The alternate port is usually a previously failed port.
Step 5	rep lsl-age-timer <*lsl-age-timer*> **Example:** `Router(config-if)# rep lsl-age-timer 2000`	Configures REP link status layer age out timer value. The acceptable range of *lsl-age-timer* is between 120ms and 10000ms, in multiples of 40ms. The default LSL Age Out time is 5 seconds.

Step 6	**end**
	Example:
	`Router(config-if)# end`
		Exits configuration mode.

Examples

This example shows how to configure REP link status layer ageout timer value.

```
Router# enable
Router# configure terminal
Router(config)# interface GigabitEthernet 5/0/2
Router(config-if)# rep segment 1 edge primary
Router(config-if)# rep lsl-age-timer 2000
Router(config-if)# end
```

Verification

Use the **show interfaces** <*interface name*> **rep detail** command to view the configured LSL number of retries and the LSL Age Out timer values.

```
7600-1#show interfaces GigabitEthernet11/0/1  rep detail
GigabitEthernet11/0/1   REP enabled
Segment-id: 10 (Segment)
PortID: 0A010009B6D8F700
Preferred flag: No
Operational Link Status: NO_NEIGHBOR
Current Key: 0A010009B6D8F700EEA1
Port Role: Fail No Ext Neighbor
Blocked VLAN: 1-4094
Admin-vlan: 1
Preempt Delay Timer: disabled
LSL Ageout Timer: 120 ms
LSL Ageout Retries: 3
Configured Load-balancing Block Port: none
Configured Load-balancing Block VLAN: none
STCN Propagate to: none
LSL PDU rx: 0, tx: 175
HFL PDU rx: 0, tx: 0
BPA TLV rx: 0, tx: 0
BPA (STCN, LSL) TLV rx: 0, tx: 0
BPA (STCN, HFL) TLV rx: 0, tx: 0
EPA-ELECTION TLV rx: 0, tx: 0
EPA-COMMAND TLV rx: 0, tx: 0
EPA-INFO TLV rx: 0, tx: 0
```

Troubleshooting

Table 2-20 provides troubleshooting solutions for the REP feature.

Table 2-20 Troubleshooting Reverse REP feature

Problem	Solution
RL2GP configuration issues	Use the **show spanning-tree pseudo-information** [*id* [*configuration* | *interface*]] and **debug spanning-tree pseudo-information** commands to trace the configuration sequence of the R-L2GP commands and the messages between the route and switch processor. Share the output with TAC for further investigation.
Disabled STP or MST instances	Use the **show spanning-tree** [*active* | *detail* | *interface*] command to verify the state of the STP or MST. Share the output with TAC for further investigation.
spanning-tree pseudo-information transmit command is rejected	Verify if : • All MST instances within the pseudo-information are configured within the MST global configuration. • MSTI 0 (IST) is configured within the pseudo-information.
Cannot configure MST	Re-configure MSTE and ensure that priority, MAC address and cost are the same on both the network processor engines.

Problem	Solution
System loops	Re-configure all the 64 VLAN instances per RL2GP within a Pseudo ID.
Configuration is rejected when the MST region ID is modified.	As IOS supports only single region MST, remove the multiple MSTregion IDs that have been configured and configure only a single MST ID.

Configuring CFM over EFP Interface with cross connect

Ethernet Connectivity Fault Management (CFM) is an end-to-end per-service-instance Ethernet layer OAM protocol that includes proactive connectivity monitoring, fault verification, and fault isolation. Currently, Ethernet CFM supports Up facing and Down facing Maintenance Endpoints (MEPs). For information on Ethernet Connectivity Fault Management, see http://www.cisco.com/en/US/docs/ios/12_2sr/12_2sra/feature/guide/srethcfm.html

The CFM over EFP Interface with xconnect feature allows you to:

- Forward continuity check messages (CCM) towards the core over cross connect pseudowires.
- Receive CFM messages from the core.
- Forward CFM messages to the access side (after Continuity Check Database [CCDB] based on maintenance point [MP] filtering rules).

Restrictions and Usage Guidelines

When configuring CFM over EFP Interface with cross connect, follow these restrictions and usage guidelines:

- The following line cards are supported:
  - ES20 line cards
  - ES+ line cards
- Only a single down-facing MEP is allowed on the L2VFI.
- As the number of PEs in a VPLS instance scale up, the number of CFM CC messages processed increases. Accordingly, the configuration of the down-facing MEP on L2VFI for large fully meshed PW topologies should be considered for only premium valued networks.
- In the design of CFM domains, the maintenance level of an Down-facing MEP on the L2VFI interface must be lower than the level from the AC.
- Up MEP, Down MEP, and MIPs are supported.

Configuring CFM over EFP with xconnect for the Cisco 7600 Router

This section describes how to configure REP over EVC for the Cisco 7600 router:

- Configuring CFM over EFP Interface with Cross Connect—Basic Configuration, page 2-229
- Configuring CFM over EFP Interface with Cross Connect—Single Tag VLAN Cross Connect, page 2-231
- Configuring CFM over EFP Interface with Cross Connect—Double Tag VLAN Cross Connect, page 2-233

- Configuring CFM over EFP Interface with Cross Connect—Selective QinQ Cross Connect, page 2-235
- Configuring CFM over EFP Interface with Cross Connect—Port-Based Cross Connect Tunnel, page 2-237
- Configuring CFM over EFP Interface with Cross Connect—Port Channel-Based Cross Connect Tunnel, page 2-239

Configuring CFM over EFP Interface with Cross Connect—Basic Configuration

This section describes how to configure CFM over EFP Interface with cross connect.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **pseudowire-class** *[pw-class-name]*
4. **encapsulation mpls**
5. **exit**
6. **interface gigabitethernet** *slot/port* or **interface tengigabitethernet** *slot/port*
7. **service instance** *id* **{Ethernet [***service-name***}**
8. **encapsulation untagged, dot1q** ***vlan_id***
9. **xconnect** *peer-ip-address vc-id* {**encapsulation** {**l2tpv3** [**manual**] | **mpls** [**manual**]} | *pw-class pw-class-name* }[*pw-class pw-class-name*] [**sequencing** {**transmit** | **receive** | both}]
10. cfm mep domain *domain-name* [up | down] mpid *mpid-value* [cos *cos-value*]
11. exit

DETAILED STEPS

	Command	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	`pseudowire-class [pw-class-name]`
	Example:
	`Router(config)# pseudowire-class vlan-xconnect`
		Specifies the name of a Layer 2 pseudowire class and enter pseudowire class configuration mode.

	Command	Purpose
Step 4	encapsulation mpls **Example:** Router(config-if)# encapsulation mpls	Specifies that Multiprotocol Label Switching (MPLS) is used as the data encapsulation method for tunneling Layer 2 traffic over the pseudowire.
Step 5	*exit* **Example:** Router(config-if-srv)# exit	Exits the pseudowire class configuration mode.
Step 6	interface gigabitethernet slot/port or interface tengigabitethernet slot/port **Example:** Router(config-if-srv)# interface Gi2/0/2	Specifies the Gigabit Ethernet or the Ten Gigabit Ethernet interface to configure.
Step 7	service instance id ethernet [service-name] **Example:** Router(config-if-srv)# service instance 101 ethernet	Creates a service instance (an instantiation of an EVC) on an interface and sets the device into the config-if-srv submode.
Step 8	***encapsulation dot1q*** *{any | vlan-id[vlan-id[-vlan-id]]}* ***second-dot1q*** *{any | vlan-id[vlan-id[-vlan-id]]}* **Example:** Router(config-if-srv)# encapsulation dot1q 100 second dot1q 200	Configures the encapsulation. Defines the matching criteria that maps the ingress dot1q, QinQ, or untagged frames on an interface for the appropriate service instance.
Step 9	xconnect peer-ip-address vc-id {encapsulation {l2tpv3 [manual] | mpls [manual]} | pw-class pw-class-name }[pw-class pw-class-name] [sequencing {transmit | receive | both}] **Example:** Router(config-if-srv)# xconnect 10.0.3.201 123 pw-class vlan-xconnect	Binds an attachment circuit to a pseudowire, and configures an Any Transport over MPLS (AToM) static pseudowire.

	Command	Purpose

Step 10	`cfm mep domain domain-name [up | down] mpid mpid-value [cos cos-value]`
	Example:
	`Router(config-if-srv)# cfm mep down mpid 100 domain Core`
		Configures a maintenance endpoint (MEP) for a domain.

Step 11	**`exit`**
	Example:
	`Router(config-if-srv)# exit`
		Exits the interface configuration mode.

Examples

This example shows how to configure CFM over EVC using cross connect.

```
PE3#conf terminal
Enter configuration commands, one per line.  End with CNTL/Z.
PE3(config)#ethernet cfm domain L6 level 6
PE3(config-ecfm)# service s256 evc 256
PE3(config-ecfm-srv)#  continuity-check
PE3(config-ecfm-srv)#end

PE3(config)#int ten 2/0/0
PE3(config-if)#no ip address
PE3(config-if)# service instance 256 ethernet 256
PE3(config-if-srv)#  encapsulation dot1q 256
PE3(config-if-srv)#  xconnect 1.1.1.1 1 encapsulation mpls
PE3(cfg-if-ether-vc-xconn)#  cfm mep domain L6 mpid 256
PE3(config-if-srv-ecfm-mep)#end
PE3#
PE3(config)#ethernet cfm domain L2 level 2
PE3(config-ecfm)# service s256 evc 256 direction down
PE3(config-ecfm-srv)#  continuity-check
PE3(config-ecfm-srv)#end
PE3#
PE3(config)#int ten 2/0/0
PE3(config-if)#no ip address
PE3(config-if)# service instance 256 ethernet 256
PE3(config-if-srv)#  encapsulation dot1q 256
PE3(config-if-srv)#  xconnect 1.1.1.1 1 encapsulation mpls
PE3(cfg-if-ether-vc-xconn)#  cfm mep domain L6 mpid 256
PE3(config-if-srv-ecfm-mep)#end
PE3#
```

Configuring CFM over EFP Interface with Cross Connect—Single Tag VLAN Cross Connect

This section describes how to configure CFM over EFP Interface with Single Tag VLAN cross connect.

SUMMARY STEPS

1. enable
2. **configure terminal**
3. ***interface type*** *slot/subslot/port* ***or interface tengigabitethernet*** *slot/port*

4. **service instance** *id* **{Ethernet [***service-name***}**
5. **encapsulation untagged dot1q {any |** ***vlan-id*[*vlan-id*[*vlan-id*]]} second-dot1q {any |*vlan-id*[*vlan-id*[*vlan-id*]]}**
6. **rewrite ingress tag** {push {dot1q *vlan-id* | dot1q *vlan-id* second-dot1q *vlan-id* | dot1ad *vlan-id* dot1q *vlan-id*} | pop {1 | 2} | translate {1-to-1 {dot1q *vlan-id* | dot1ad *vlan-id*}| 2-to-1 dot1q *vlan-id* | dot1ad *vlan-id*}| 1-to-2 {dot1q *vlan-id* second-dot1q *vlan-id* | dot1ad *vlan-id* dot1q *vlan-id*} | 2-to-2 {dot1q *vlan-id* second-dot1q *vlan-id* | dot1ad *vlan-id* dot1q *vlan-id*}} [symmetric]
7. **xconnect** ***peer-ip*-address** ***vc-id*** **{encapsulation {l2tpv3 [manual] | mpls [manual]} | pw-class** ***pw-class-name*** **}[*pw-class pw-class-name*] [sequencing {transmit | receive | both}]**
8. cfm mep domain *domain-name* [up | down] mpid *mpid-value* [cos *cos-value*]

DETAILED STEPS

	Command	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 the global configuration mode.
Step 3	**interface gigabitethernet** *slot*/subslot/*port* **Example:** Router(config)# interface Gi2/0/2	Specifies the Gigabit Ethernet interface to configure, where: *slot/subslot/port*—Specifies the location of the interface.
Step 4	**service instance** id **Ethernet** [service-name] **Example:** Router(config-if)# service instance 101 ethernet	Creates a service instance (an instantiation of an EVC) on an interface and sets the device into the config-if-srv submode.
Step 5	***encapsulation dot1q*** *{any | vlan-id[vlan-id[-vlan-id]]}* ***second-dot1q*** *{any | vlan-id[vlan-id[-vlan-id]]}* **Example:** Router(config-if-srv)# encapsulation dot1q 100 second dot1q 100	Configures the encapsulation. Defines the matching criteria that maps the ingress dot1q, QinQ, or untagged frames on an interface for the appropriate service instance.

	Command	Purpose
Step 6	**rewrite ingress tag** {**push** {**dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} [**symmetric**] **Example:** `Router(config-if-srv)# rewrite dot1q single symmetric`	Specifies the tag manipulation that is to be performed on the frame ingress to the service instance.
Step 7	xconnect peer-ip-address vc-id {encapsulation {l2tpv3 [manual] | mpls [manual]} | pw-class pw-class-name }[pw-class pw-class-name] [sequencing {transmit | receive | both}] **Example:** `Router(config)# xconnect 10.0.3.201 123 pw-class vlan-xconnect`	Binds an attachment circuit to a pseudowire, and configures an Any Transport over MPLS (AToM) static pseudowire.
Step 8	**cfm mep domain domain-name [up | down] mpid mpid-value [cos cos-value]** **Example:** `Router# cfm mep up mpid 100 domain Core`	Configures a maintenance endpoint (MEP) for a domain.

Examples

This example shows how to configure CFM over EFP Interface with Single Tag VLAN cross connect:

```
PE3(config)#ethernet cfm domain L2 level 2
PE3(config-ecfm)# service s256 evc 256 direction down
PE3(config-ecfm-srv)#  continuity-check
PE3(config-ecfm-srv)#end
PE3#
PE3(config)#int ten 2/0/0
PE3(config-if)#no ip address
PE3(config-if)# service instance 256 ethernet 256
PE3(config-if-srv)#  encapsulation dot1q 256
PE3(config-if-srv)#  xconnect 1.1.1.1 1 encapsulation mpls
PE3(cfg-if-ether-vc-xconn)#  cfm mep domain L6 mpid 256
PE3(config-if-srv-ecfm-mep)#end
PE3#
```

Configuring CFM over EFP Interface with Cross Connect—Double Tag VLAN Cross Connect

This section describes how to configure CFM over EFP Interface with Double Tag VLAN cross connect.

SUMMARY STEPS

1. enable

2. **configure terminal**
3. ***interface type*** *slot/subslot/port*
4. **service instance** *id* **{Ethernet** [*service-name*}
5. **encapsulation untagged dot1q {any |** ***vlan-id***[***vlan-id***[***vlan-id***]**} second-dot1q {any** |***vlan-id***[***vlan-id***[***vlan-id***]]**}**
6. **rewrite ingress tag** {**push {dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} [**symmetric**]
7. xconnect *peer-ip-address vc-id* {encapsulation {l2tpv3 [manual] | mpls [manual]} | *pw-class pw-class-name* }[*pw-class pw-class-name*] [sequencing {transmit | receive | both}]
8. cfm mep domain *domain-name* [up | down] mpid *mpid-value* [cos *cos-value*]
9. exit

DETAILED STEPS

	Command	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`** *`slot`*`/subslot/`*`port`*
	Example:
	`Router(config)# interface Gi2/0/2`
		Specifies the Gigabit Ethernet interface to configure, where:
		slot/subslot/port—Specifies the location of the interface.

Step 4	**`service instance`** `id` **`Ethernet`** `[service-name]`
	Example:
	`Router(config-if)# service instance 100 ethernet`
		Creates a service instance (an instance of an EVC) on an interface and sets the device into the config-if-srv submode.

Step 5	***`encapsulation untagged dot1q`*** *`{any | vlan-id[vlan-id[-vlan-id]]}`* ***`second-dot1q`*** *`{any | vlan-id[vlan-id[-vlan-id]]}`*
	Example:
	`Router(config-if-srv)# encapsulation dot1q 100 second-dot1q 200`
		Configures the encapsulation. Defines the matching criteria that maps the ingress dot1q, QinQ, or untagged frames on an interface for the appropriate service instance.

	Command	Purpose
Step 6	**rewrite ingress tag** {**push** {**dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} [**symmetric**] **Example:** Router(config-if-srv)# rewrite dot1q double symmetric	Specifies the tag manipulation that is to be performed on the frame ingress to the service instance.
Step 7	xconnect peer-ip-address vc-id {encapsulation {l2tpv3 [manual] | mpls [manual]} | pw-class pw-class-name }[pw-class pw-class-name] [sequencing {transmit | receive | both}] **Example:** Router(config)# xconnect 1.1.1.1 100 pw-class vlan-xconnect	Binds an attachment circuit to a pseudowire, and configures an Any Transport over MPLS (AToM) static pseudowire.
Step 8	**cfm mep domain domain-name [up | down] mpid mpid-value [cos cos-value]** **Example:** Router# cfm mep down mpid 100 domain Core	Configures a maintenance endpoint (MEP) for a domain.

Examples

This example shows how to configure CFM over EFP Interface with Double Tag VLAN cross connect:

```
PE3(config)#ethernet cfm domain L2 level 2
PE3(config-ecfm)# service s256 evc 256 direction down
PE3(config-ecfm-srv)#  continuity-check
PE3(config-ecfm-srv)#end
PE3#
PE3(config)#int ten 2/0/0
PE3(config-if)#no ip address
PE3(config-if)# service instance 256 ethernet 256
PE3(config-if-srv)#  encapsulation dot1q 256 second-dot1q 257
PE3(config-if-srv)#  xconnect 1.1.1.1 1 encapsulation mpls
PE3(cfg-if-ether-vc-xconn)#  cfm mep domain L6 mpid 256
PE3(config-if-srv-ecfm-mep)#end
PE3#
```

Configuring CFM over EFP Interface with Cross Connect—Selective QinQ Cross Connect

This section describes how to configure CFM over EFP Interface with Selective QinQ cross connect.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface type** *slot/subslot/port*
4. **exit**
5. **service instance** *id* **{Ethernet [***service-name***}**
6. **encapsulation untagged dot1q {any |** ***vlan-id*****[*****vlan-id*****[*****vlan-id*****]} second-dot1q {any |*****vlan-id*****[*****vlan-id*****[*****vlan-id*****]]}**
7. **xconnect** *peer-ip-address vc-id* {**encapsulation** {l2tpv3 [manual] | mpls [manual]} | *pw-class pw-class-name* }[*pw-class pw-class-name*] [sequencing {transmit | receive | both}]
8. **cfm mep domain** *domain-name* [up | down] mpid mpid-value [cos *cos-value*]
9. **exit**

DETAILED STEPS

	Command	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** *slot*/subslot/*port* **Example:** `Router(config)# interface Gi2/0/2`	Specifies the Gigabit Ethernet interface to configure, where: *slot/subslot/port*—Specifies the location of the interface.
Step 4	**service instance** id **Ethernet** [service-name] **Example:** `Router(config-if)# service instance 101 ethernet`	Creates a service instance (an instance of an EVC) on an interface and sets the device into the config-if-srv submode.

Step 5	***encapsulation untagged dot1q*** *{any | vlan-id[vlan-id[-vlan-id]]}* ***second-dot1q*** *{any | vlan-id[vlan-id[-vlan-id]]}*
	Example:
	`Router(config-if-srv)# encapsulation default`
		Configures the encapsulation. Defines the matching criteria that maps the ingress dot1q, QinQ, or untagged frames on an interface for the appropriate service instance.

	Command	Purpose

Step 6	`xconnect peer-ip-address vc-id {encapsulation {l2tpv3 [manual] | mpls [manual]} | pw-class pw-class-name }[pw-class pw-class-name] [sequencing {transmit | receive | both}]`
	Example:
	`Router(config)# xconnect 10.0.3.201 123 pw-class vlan-xconnect`
		Binds an attachment circuit to a pseudowire, and configures an Any Transport over MPLS (AToM) static pseudowire.

Step 7	**`cfm mep domain domain-name [up | down] mpid mpid-value [cos cos-value]`**
	Example:
	`Router# cfm mep down mpid 100 domain Core`
		Configures a maintenance endpoint (MEP) for a domain.

Examples

This example shows how to configure CFM over EFP Interface with Selective QinQ cross connect:

```
PE3(config)#ethernet cfm domain L2 level 2
PE3(config-ecfm)# service s256 evc 256 direction down
PE3(config-ecfm-srv)#  continuity-check
PE3(config-ecfm-srv)#end
PE3#
PE3(config)#int ten 2/0/0
PE3(config-if)#no ip address
PE3(config-if)# service instance 256 ethernet 256
PE3(config-if-srv)#  encapsulation dot1q 256 second-dot1q 257 cos 7
PE3(config-if-srv)#  xconnect 1.1.1.1 1 encapsulation mpls
PE3(cfg-if-ether-vc-xconn)#  cfm mep domain L6 mpid 256
PE3(config-if-srv-ecfm-mep)#end
PE3#
```

Configuring CFM over EFP Interface with Cross Connect—Port-Based Cross Connect Tunnel

This section describes how to configure CFM over EFP Interface with Port-Based cross connect Tunnel.

SUMMARY STEPS

1. enable
2. **configure terminal**
3. ***interface type*** *slot/subslot/port*
4. **service instance** *id* **{Ethernet [***service-name***}**
5. **encapsulation untagged dot1q {any |** ***vlan-id*[*vlan-id*[*vlan-id*]}** **second-dot1q {any |*vlan-id*[*vlan-id*[*vlan-id*]]}**
6. xconnect *peer-ip-address vc-id* {encapsulation {l2tpv3 [manual] | mpls [manual]} | *pw-class pw-class-name* }[*pw-class pw-class-name*] [sequencing {transmit | receive | both}]
7. cfm mep domain *domain-name* [up | down] mpid *mpid-value* [cos *cos-value*]
8. exit

DETAILED STEPS

	Command	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** *slot*/subslot/*port* **Example:** `Router(config)# interface Gi2/0/2`	Specifies the Gigabit Ethernet interface to configure, where: *slot/subslot/port*—Specifies the location of the interface.
Step 4	**service instance** id **Ethernet** [service-name] **Example:** `Router(config-if)# service instance 100 ethernet`	Creates a service instance (an instance of an EVC) on an interface and sets the device into the config-if-srv submode.

Step 5	***encapsulation untagged dot1q*** *{any | vlan-id[vlan-id[-vlan-id]]}* ***second-dot1q*** *{any | vlan-id[vlan-id[-vlan-id]]}*
	Example:
	`Router(config-if-srv)# encapsulation dot1q 10-20, 30, 50-60`
		Configures the encapsulation. Defines the matching criteria that maps the ingress dot1q, QinQ, or untagged frames on an interface for the appropriate service instance.

Step 6	xconnect peer-ip-address vc-id {encapsulation {l2tpv3 [manual] | mpls [manual]} | pw-class pw-class-name}[pw-class pw-class-name] [sequencing {transmit | receive | both}] **Example:** `Router(config)# xconnect 1.1.1.1 100 pw-class vlan-xconnect`	Binds an attachment circuit to a pseudowire, and configures an Any Transport over MPLS (AToM) static pseudowire.
Step 7	**cfm mep domain domain-name [up | down] mpid mpid-value [cos cos-value]** **Example:** `Router# cfm mep up mpid 100 domain Core`	Configures a maintenance endpoint (MEP) for a domain.

Examples

This example shows how to configure CFM over EFP Interface with Port-Based cross connect Tunnel:

```
PE3(config)#ethernet cfm domain L2 level 2
PE3(config-ecfm)# service s256 evc 256 direction down
PE3(config-ecfm-srv)#  continuity-check
PE3(config-ecfm-srv)#end
PE3#
PE3(config)#int ten 2/0/0
PE3(config-if)#no ip address
PE3(config-if)# service instance 256 ethernet 256
PE3(config-if-srv)#  encapsulation dot1q 256
PE3(config-if-srv)#  xconnect 1.1.1.1 1 encapsulation mpls
PE3(cfg-if-ether-vc-xconn)#  cfm mep domain L6 mpid 256
PE3(config-if-srv-ecfm-mep)#end
PE3#
```

Configuring CFM over EFP Interface with Cross Connect—Port Channel-Based Cross Connect Tunnel

This section describes how to configure CFM over EFP Interface with Port Channel-Based cross connect Tunnel.

SUMMARY STEPS

1. enable
2. **configure terminal**
3. ***interface type*** *slot/subslot/port*
4. **service instance** *id* **{Ethernet [***service-name***}**
5. **encapsulation untagged dot1q {any |** ***vlan-id*****[*****vlan-id*****[*****vlan-id*****]} second-dot1q {any |*****vlan-id*****[*****vlan-id*****[*****vlan-id*****]]}**
6. **rewrite ingress tag** {**push {dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} [**symmetric**]
7. xconnect *peer-ip-address vc-id* {encapsulation {l2tpv3 [manual] | mpls [manual]} | *pw-class pw-class-name* }[*pw-class pw-class-name*] [sequencing {transmit | receive | both}]
8. cfm mep domain *domain-name* [up | down] mpid *mpid-value* [cos *cos-value*]
9. exit

DETAILED STEPS

	Command	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** *slot*/subslot/*port* **Example:** `Router(config)# interface Port-channel 1`	Specifies the Gigabit Ethernet interface to configure, where: *slot/subslot/port*—Specifies the location of the interface.
Step 4	**service instance** id **Ethernet** [service-name] **Example:** `Router(config-if)# service instance 101 ethernet`	Creates a service instance (an instance of an EVC) on an interface and sets the device into the config-if-srv submode.

Step 5	***encapsulation untagged dot1q*** *{any | vlan-id[vlan-id[-vlan-id]]}* ***second-dot1q*** *{any | vlan-id[vlan-id[-vlan-id]]}*
	Example:
	`Router(config-if-srv)# encapsulation dot1q 20 second-dot1q 30`
		Configures the encapsulation. Defines the matching criteria that maps the ingress dot1q, QinQ, or untagged frames on an interface for the appropriate service instance.

Step 6	**rewrite ingress tag** {**push {dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} [**symmetric**] **Example:** `Router(config-if-srv)# rewrite ingress tag pop 2 symmetric`	Specifies the tag manipulation that is to be performed on the frame ingress to the service instance.

	Command	Purpose

Step 7	`xconnect peer-ip-address vc-id {encapsulation {l2tpv3 [manual] | mpls [manual]} | pw-class pw-class-name }[pw-class pw-class-name] [sequencing {transmit | receive | both}]`
	Example:
	`Router(config)# xconnect 1.1.1.1 100 pw-class vlan-xconnect`
		Binds an attachment circuit to a pseudowire, and configures an Any Transport over MPLS (AToM) static pseudowire.

Step 8	**`cfm mep domain domain-name [up | down] mpid mpid-value [cos cos-value]`**
	Example:
	`Router# cfm mep up mpid 100 domain Core`
		Configures a maintenance endpoint (MEP) for a domain.

Examples

This example shows how to configure CFM over EFP Interface with Port Channel-Based cross connect Tunnel:

```
PE3(config)#ethernet cfm domain L2 level 2
PE3(config-ecfm)# service s256 evc 256 direction down
PE3(config-ecfm-srv)#  continuity-check
PE3(config-ecfm-srv)#end
PE3#
PE3(config)#int port-20
PE3(config-if)#no ip address
PE3(config-if)# service instance 256 ethernet 256
PE3(config-if-srv)#  encapsulation dot1q 256
PE3(config-if-srv)#  xconnect 1.1.1.1 1 encapsulation mpls
PE3(cfg-if-ether-vc-xconn)#  cfm mep domain L6 mpid 256
PE3(config-if-srv-ecfm-mep)#end
```

Verification

- Use the following commands to verify a configuration:

You can use the **show ethernet cfm ma remote** commands to verify the CFM over EVC configuration. This command shows the basic configuration information for CFM.

```
Router-30-PE1#show ethernet cfm ma local
Local MEPs:
--------------------------------------------------------------------------------
MPID Domain Name                                 Lvl   MacAddress     Type  CC
     Domain Id                                   Dir   Port           Id
     MA Name                                           SrvcInst
     EVC name
--------------------------------------------------------------------------------
1    L6                                          6     000a.f393.56d0 XCON  Y
     L6                                          Down    Te2/0/0        N/A
     bbb                                               1
     bbb
3    L5                                          5     0007.8478.4410 XCON  Y
     L5                                          Up    Te2/0/0        N/A
     bbb                                               1
     bbb

Total Local MEPs: 2
```

```
Local MIPs:
* = MIP Manually Configured
--------------------------------------------------------------------------------
 Level Port           MacAddress     SrvcInst   Type    Id
--------------------------------------------------------------------------------
 7     Te2/0/0        0007.8478.4410 1          XCON    N/A

Total Local MIPs: 1
```

- Use the **show ethernet cfm ma remote** to verify the MEP configuration:

```
Router-30-PE1#show ethernet cfm ma remote
--------------------------------------------------------------------------------
MPID  Domain Name                                 MacAddress          IfSt  PtSt
 Lvl  Domain ID                                   Ingress
 RDI  MA Name                                     Type Id             SrvcInst
      EVC Name                                                        Age
--------------------------------------------------------------------------------
4     L5                                          000a.f393.56d0      Up    Up
 5    L5                                          Te2/0/0:(2.2.2.2, 1)
 -    bbb                                         XCON N/A            1
      bbb                                                             9s
2     L6                                          000a.f393.56d0      Up    Up
 6    L6                                          Te2/0/0:(2.2.2.2, 1)
 -    bbb                                         XCON N/A            1
      bbb                                                             1s

Total Remote MEPs: 2
```

- Use the **show ethernet cfm mpdb** command to verify the catalouge of CC with MIP in intermediate routers.

```
PE2#show ethernet cfm mpdb
* = Can Ping/Traceroute to MEP
--------------------------------------------------------------------------------
MPID  Domain Name                                 MacAddress          Version
Lvl   Domain ID                                   Ingress
Expd  MA Name                                     Type Id             SrvcInst
      EVC Name                                                        Age
--------------------------------------------------------------------------------
600 * L6                                          0021.d8ca.d7d0      IEEE-CFM
6     L6                                          Te2/1:(2.2.2.2, 1)
-     s1                                          XCON N/A            1
      1                                                               2s
700   L7                                          001f.cab7.fd01      IEEE-CFM
7     L7                                          Te2/1:(2.2.2.2, 1)
-     s1                                          XCON N/A            1
      1                                                               3s

Total Remote MEPs: 2
```

- Use the **show mpls l2 vc 1 detail** commaned to show detailed configuration information:

```
PE1#sh mpls l2 vc 1 deta
Local interface: Te8/0/1 up, line protocol up, Eth VLAN 200 up
  Interworking type is Ethernet
  Destination address: 3.3.3.3, VC ID: 1, VC status: up
    Output interface: Te8/0/0, imposed label stack {21}
    Preferred path: not configured
    Default path: active
    Next hop: 20.1.1.2
  Create time: 21:13:27, last status change time: 02:55:33
  Signaling protocol: LDP, peer 3.3.3.3:0 up
    Targeted Hello: 2.2.2.2(LDP Id) -> 3.3.3.3, LDP is UP
    Status TLV support (local/remote)   : enabled/supported
      LDP route watch                   : enabled
      Label/status state machine        : established, LruRru
      Last local dataplane   status rcvd: No fault
      Last local SSS circuit status rcvd: No fault
      Last local SSS circuit status sent: No fault
      Last local  LDP TLV    status sent: No fault
      Last remote LDP TLV    status rcvd: No fault
      Last remote LDP ADJ    status rcvd: No fault
    MPLS VC labels: local 21, remote 21
    Group ID: local 0, remote 0
    MTU: local 1500, remote 1500
    Remote interface description:
  Sequencing: receive disabled, send disabled
  Control Word: On (configured: autosense)
  VC statistics:
    transit packet totals: receive 37, send 1067452272
    transit byte totals:   receive 4181, send 72586757556
    transit packet drops:  receive 0, seq error 0, send 0
```

- Use **show mpls forwarding-table** command to verify the cross connect VC:

```
PE1#show mpls forwarding-table
Local      Outgoing   Prefix           Bytes Label   Outgoing   Next Hop
Label      Label      or Tunnel Id     Switched      interface
17         Pop Label  3.3.3.3/32       23038746624   Te8/0/0    20.1.1.2
21         No Label   l2ckt(1)         4181          Te8/0/1    point2point
```

- Use **show ethernet cfm error** command to view the error report:

```
PE2#show ethernet cfm error
--------------------------------------------------------------------------------
MPID Domain Id                                   Mac Address     Type   Id  Lvl
     MAName                                      Reason                 Age
--------------------------------------------------------------------------------
  -  L3                                          001d.45fe.ca81  BD-V   200  3
     s2                                          Receive AIS            8s
PE2#
```

Troubleshooting CFM Features

Table 2-21 provides troubleshooting solutions for the CFM features.

Table 2-21 Troubleshooting Scenarios for CFM features

Problem	Solution
When you configure CFM, the message "Match registers are not available" is displayed.	Use the **show platform mrm info** command on the SP console to verify the match registers. Based on the derived output, perform these tasks: 1. Check the hardware limitations on the affected ports. 2. Enable CFM across the system to allow co-existence with other protocols. 3. Ensure that no CFM traffic is present in any supervisor or ports. 4. Configure STP mode to Multiple Spanning Tree (MST) and re-enable CFM or disable CFM completely. For more information on match registers, see *Ethernet Connectivity Fault Management* at http://www.cisco.com/en/US/docs/ios/12_2sr/12_2sra/feature/guide/srethcfm.html. CFM uses two match registers to identify the control packet type and each VLAN spanning tree also uses a match register to identify its control packet type. For both protocols to work on the same system, each line card should support three match registers, and at least one supporting only a 44 bit MAC match.
CFM configuration errors	CFM configuration error occurs when when a MEP receives a continuity check with an overlapping MPID. To verify the source of the error, use the command **show ethernet cfm errors configuration** or **show ethernet cfm errors**.
CFM ping and traceroute result is "not found"	Complete these steps: 1. Use **show run ethernet cfm** to view all CFM global configurations. 2. Use **show ethernet cfm location main** to view local MEPs and their CCM statistics 3. Use **show ethernet cfm peer meps** command to View CFM CCM received from Peer MEPs. 4. Use **trace ethernet cfm** command to start a CFM trace.
CFM connectivity is down and issues at the maintenance domain levels	Use the **ping ethernet {mac-address | mpid id | multicast} domain domain-name { vlan vlan-id | port | evc evc-name }** or t**raceroute ethernet {mac-address | mpid id } domain domain-name { vlan vlan-id | port | evc evc-name }** commands to verify ethernet CFM connectivity. Share the output with TAC for further investigation.

Problem	Solution

Loop trap error	Use the **show ethernet cfm error** command to check for Loop Trap errors as shown here:
	`CE(config-if)#do sh ethernet cfm err`
	`--------------------------------------------------`
	`-----------------------------`
	`Level Vlan MPID Remote MAC     Reason`
	`Service ID`
	`--------------------------------------------------`
	`-----------------------------`
	`5    711  550  1001.1001.1001 Loop Trap Error`
	`OUT`
	`PE#sh ethernet cfm err`
	`--------------------------------------------------`
	`-----------------------------`
	`Level Vlan MPID Remote MAC     Reason`
	`Service ID`
	`--------------------------------------------------`
	`-----------------------------`
	`5    711  550  1001.1001.1001 Loop Trap Error`
	`OUT`

Module has insufficient match registers	Complete these steps: **1.** Verify and confirm if a unsupported line card is inserted into the router. **2.** If yes, perform a OIR to remove the unsupported line card.
CFM is deactivated	Complete these steps: **1.** Check if all the line cards have free match reagisters. **2.** Check if CFM is activated on supervisor cards. CFM is not supported on supervisor cards that has two match registers. In this scenario, CFM is automatically disabled on the SUP ports and enabled on rest of the line cards.

Configuring Reverse Layer 2 Gateway Ports for the Cisco 7600 Router

Layer 2 Gateway Ports (L2GP) is a proposed IEEE standard (802.1ah) to address the issues that arise when two independent bridged domains are connected redundantly through an arbitrary number of links. Layer 2 Gateway Ports define how the forwarding gateways are selected so that only redundant ports are blocked, and there are no temporary loops. The transitions can be at least as fast as STP L2GP resolves the transient loop problem during the re-convergence as it does not require cooperation from the outside domain.

Reverse L2GP (R-L2GP) is a variation of L2GP. In case of R-L2GP, the pseudo information of the R-L2GP is transmitted by NPEs, instead of uPEs. R-L2GP provides a mechanism to send out static preconfigured BPDUs on each ring access port of nPEs to stimulate a per-access ring instantiation of the protocol. In order for this to work, the pair of nPEs are programmed to send out BPDUs on the access ring ports in such a way that they appear to be either:

- The root bridge itself (the bridge with the lowest bridge id/priority).
- The bridge with the second lowest bridge ID/priority, and with a 0 cost path to the root.

Using R-L2GP, you can configure static BPDUs instead of dynamic configuration.

Effective with Release 15.1(3)S, multiple Bridge Protocol Data Unit (BPDU) pseudowires (PW) are supported for the R-L2GP protocol. Note that only one BPDU PW is supported for Multiple Spanning Tree (MST) protocol.

Multiple BPDU PWs are created on the NPE when it participates in multiple layer 2 domains. Multiple BPDU PWs are required to propagate Topology Change Notices (TCNs) over BPDU PW within an layer 2 ring. For instance, if any topology change is observed in a particular layer 2 domain, the TCN should be propagated across the MPLS backbone to only that layer 2 domain and not to the other layer 2 domains in the MPLS backbone.

For R-L2GP, the maximum number of BPDU PWs supported on Cisco 7600 platform is ten.

Restrictions and Usage Guidelines

When configuring Reverse L2GP for the Cisco 7600 router, follow these guidelines and restrictions:

- Because VLAN ID is required for EVC service instance to MST instance mapping, EVC service instances without any VLAN ID in the encapsulation are not supported. This includes:
  - Untagged encapsulation
  - Priority-tagged encapsulation
  - Default encapsulation
- Service instances with multiple outer tags are not supported.
- MST and R-L2GP can co-exist on the same router as long as MST ports do not create parallel links to Reverse L2GP link in a L2 domain.
- Multiple BPDU PW is supported only on R-L2GP. MST and R-L2GP cannot co-exist with multiple BPDU PW.
- R-L2GP does not provide any automatic detection or recovery mechanisms for BPDU data.
- The R-L2GP pseudo information should be the same on the local and remote PE. For instance, mac-address, remote ID, local ID, and so on.
- Before you attach an R-L2GP instance to a port, you should configure the MST instance 0 within the R-L2GP.
- You can remove an R-L2GP instance only when the total number of BPDU PW in the system is one.
- Under the R-L2GP configuration (pseudo-information), the configuration for MST instance 0 should be the same on both N-PEs.
- If you are creating multiple BPDU PWs in the PE, then each BPDU should have separate R-L2GP instance, and they cannot share the same R-L2GP instance.
- We recommended that you configure multiple BPDU PW on the trunk VLAN 1.
- Cisco IOS Release 15.1(1)S supports EVC port-channels.

Configuring Reverse L2GP for the Cisco 7600 Router

To enable R-L2GP on a port, you need to:

- Configure MST
- Configure R-L2GP instance
- Attach R-L2GP instance to a port
- Configure VPLS BPDU pseudowire

Configuration of MST must be done before configuring RL2GP and attaching it to a port. For MST configuration, you need to configure:

- Provider Bridge Mode
- Hello Time
- Name
- Revision
- MSTI information (VLAN mapping, bridge priority, port priority, and cost)
- Priority Vector information (bridge ID, port ID, Root Bridge ID)

Since the R-L2GP configuration is bundled with the MSTI configuration, the above parameters can be recycled from the MSTI and MST region (currently only one MST region is supported on IOS) configurations.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spanning-tree mode mst**
4. **spanning-tree mst configuration**
5. **[no] name** *name*
6. **[no] revision** *version*
7. **[no] instance** *vlan*
8. **exit**
9. **spanning-tree pseudo-information transmit** *identifier*
10. **remote-id** *id*
11. **mst** *instance* **root mac-address** *mac*
12. **mst** *instance_id range* **root priority** *priority*
13. **mst** *instance* **root** *priority* **mac**
14. **mst** *instance* **cost** *cost*
15. **exit**
16. **interface gigabitethernet** *slot/port* or **interface tengigabitethernet** *slot/port*
17. **spanning-tree pseudo-information transmit** *identifier*
18. **exit**
19. **l2 vfi** *vfi name* **manual**
20. **vpn id** *vpn_id*
21. **forward permit l2protocol all**
22. **neighbor** *ip-address vc-id* **encapsulation mpls**
23. **exit**
24. **interface vlan** *vlanid*
25. **xconnect vfi** *vfi_name*
26. **exit**

DETAILED STEPS

	Command	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	**`spanning-tree mode mst`**
	Example:
	`Router(config)# spanning-tree mode mst`
		Enters the MST mode.

Step 4	**`spanning-tree mst configuration`**
	Example:
	`Router(config)# spanning-tree mst configuration`
		Enters MST-configuration submode.

Step 5	**`[no] name`** *`name`*
	Example:
	`Router(config-mst)# name Cisco`
		Sets the name of a Multiple Spanning Tree (MST) region.

Step 6	**`[no] revision`** *`version`*
	Example:
	`Router(config-mst)# revision 5`
		Sets the revision number for the Multiple Spanning Tree (802.1s) (MST) configuration.

Step 7	**`[no] instance`** *`vlan`*
	Example:
	`Router(config-mst)#instance 2 vlan 30-35`
		Maps the MST instance to a range of VLANs.

Step 8	**`exit`**
	Example:
	`Router(config-mst)#exit`
		Exits MST configuration mode.

Step 9	**`spanning-tree pseudo-information transmit`** *`identifier`*
	Example:
	`Router(config)#spanning-tree pseudo-information transmit 20`
		Configures the R-L2GP instance.

	Command	Purpose
Step 10	**remote-id** *id* **Example:** Router(config-pseudo)# remote-id 30	Configures the remote R-L2GP instance ID that pairs with the specified R-L2GP instance ID.
Step 11	**mst** *instance* **root mac-address** *mac* **Example:** Router(config-pseudo)#mst 1 root mac-address 2.3.4	Adds the MST instance list to R-L2GP instance and configures R-L2GP root bridge MAC address for MST instance (or multiple MST instances).
Step 12	**mst** *instance_id range* **root priority** *priority* **Example:** Router(config-pseudo)#mst 1-10 root priority 8192	Adds the MST instance list to RL2GP instance and configures the R-L2GP bridge priority (in multiples of 4096) for instances.
Step 13	**mst** *instance* **root** *priority mac* **Example:** Router(config-pseudo)#mst 1 root 8192 A.A.A	Adds the MST instances to RL2GP instances and configures the MAC address and priority for MST instances.
Step 14	**mst** *instance* **cost** *cost* **Example:** Router(config-pseudo)#mst 1-2 cost 2000	Adds the MST instance list to R-L2GP instance and configures R-L2GP path cost for MST instance (or multiple MST instances).
Step 15	**exit** **Example:** Router(config-pseudo)# exit	Exits the MST configuration mode.
Step 16	**interface gigabitethernet** *slot/port* **or** **interface tengigabitethernet** *slot/port* **Example:** Router(config)# interface gigabitethernet 4/1	Specifies the Gigabit Ethernet or the Ten Gigabit Ethernet interface to configure. • slot/port—Specifies the location of the interface.

	Command	Purpose
Step 17	**spanning-tree pseudo-information transmit** *identifier* **Example:** `Router(config-if)#spanning-tree pseudo-information transmit 10`	Attaches the R-L2GP instance to a port.
Step 18	**exit** **Example:** `Router(config)# exit`	Exits the interface configuration mode.
Step 19	**l2 vfi** *vfi_name* **manual** **Example:** `Router(config)# l2 vfi vfitest1 manual`	Creates a Layer 2 VFI and enters the Layer 2 VFI manual configuration submode.
Step 20	**vpn id** *vpn_id* **Example:** `Router(config-vfi)# vpn id 303`	Sets or updates a Virtual Private Network (VPN) ID on a VPN routing and forwarding (VRF) instance.
Step 21	**forward permit l2protocol all** **Example:** `Router(config-vfi)# forward permit l2protocol all`	Defines the VPLS pseudowire that is used to transport bridge protocol data unit (BPDU) information between two network provider edge (N-PE) routers.
Step 22	**neighbor** *ip-address* [*vc-id*] **encapsulation mpls** **Example:** `Router(config-vfi)# neighbor 10.10.10.10 encapsulation mpls`	Specifies the routers that should form a point-to-point Layer 2 virtual forwarding interface (VFI) connection.
Step 23	**exit** **Example:** `Router(config-vfi)# exit`	Exits the VFI manual configuration mode.
Step 24	**interface** *vlan-id* **Example:** `Router(config)# interface vlan 1`	Creates a dynamic Switch Virtual Interface (SVI).

	Command	Purpose
Step 25	**xconnect vfi** *vfi name* **Example:** `Router(config-if)# xconnect vfi vfitest1`	Specifies the Layer 2 VFI you bind to the VLAN port.
Step 26	**exit**	Exits the global configuration mode.

Configuration Examples

This example shows how to configure R-L2GP and a single BPDU PW for the Cisco 7600 router.

```
Router# configure terminal
Router(config)# spanning-tree mode mst
Router(config)# spanning-tree mst configuration
Router(config-mst)# name cisco
Router(config-mst)# revision 1
Router(config-mst)# instance 1 vlan 100
Router(config-mst)# instance 2 vlan 101-103
Router(config-mst)# instance 3 vlan 104
Router(config-mst)# exit
Router(config)# spanning-tree pseudo-information transmit 30
Router(config-pseudo)# remote-id 40
Router(config-pseudo)# mst 0 root mac-address 0000.0000.0001
Router(config-pseudo)# mst 1 root mac-address 0000.0000.0002
Router(config-pseudo)# mst 2 root mac-address 0000.0000.0003
Router(config-pseudo)# exit
Router(config)# interface Gi8/0/0
Router(config-if)# switchport
Router(config-if)# switchport mode trunk
Router(config-if)# no shutdown
Router(config-if)# spanning-tree pseudo-information transmit 30
Router(config-if)# exit
Router(config)# l2 vfi vfi_test1 manual
Router(config-vfi)# vpn id 300
Router(config-vfi)# forward permit l2protocol all
Router(config-vfi)# neighbor 1.1.1.1 encapsulation mpls
Router(config-vfi)# exit
Router(config)# interface vlan 1
Router(config-if)# xconnect vfi vfi_test1
Router(config-if)# no shutdown
Router(config-if)# end
```

This example shows how to configure MST before you configure R-L2GP instance.

```
PE1# configure terminal
PE1(config)# spanning-tree mode mst
PE1(config)# spanning-tree mst configuration
PE1(config-mst)# name cisco
PE1(config-mst)# revision 1
PE1(config-mst)# instance 1 vlan 100
PE1(config-mst)# instance 2 vlan 101-103
PE1(config-mst)# exit
```

Note The CE routers should have the same configuration as shown above.

This example shows how to configure an R-L2GP instance 30 for the first layer 2 domain connected to the interface 8/0/0. Similarly, in the remote PE2, configure instance as 40 and remote-ID as 30 with the same root mac and priority.

```
PE1(config)# spanning-tree pseudo-information transmit 30
PE1(config-pseudo)# remote-id 40
PE1(config-pseudo)# mst 0 root mac-address 0000.0000.0001
PE1(config-pseudo)# mst 1 root mac-address 0000.0000.0002
PE1(config-pseudo)# mst 2 root mac-address 0000.0000.0003
PE1(config-pseudo)# exit
PE1(config)# interface Gi8/0/0
PE1(config-if)# switchport
PE1(config-if)# switchport mode trunk
PE1(config-if)# no shutdown
PE1(config-if)# spanning-tree pseudo-information transmit 30
PE1(config-if)# exit
```

This example shows how to configure an R-L2GP instance 10 for the second layer 2 domain connected to the interface 7/0/0. Similarly, in the remote PE3, configure instance as 10 and remote-ID as 20 with the same root mac and priority.

```
PE1# configure terminal
PE1(config)# spanning-tree pseudo-information transmit 20
PE1(config-pseudo)# remote-id 10
PE1(config-pseudo)# mst 0 root mac-address 0000.0000.0001
PE1(config-pseudo)# mst 1 root mac-address 0000.0000.0002
PE1(config-pseudo)# mst 2 root mac-address 0000.0000.0003
PE1(config-pseudo)# exit
PE1(config)# interface Gi7/0/0
PE1(config-if)# switchport
PE1(config-if)# switchport mode trunk
PE1(config-if)# no shutdown
PE1(config-if)# spanning-tree pseudo-information transmit 20
PE1(config-if)# exit
```

This example shows how to configure multiple BPDU PWs.

```
PE1(config)# l2 vfi vfi_mpdu manual
PE1(config-vfi)# vpn id 1
PE1(config-vfi)# forward permit l2protocol all
PE1(config-vfi)# neighbor 1.1.1.1 1 encapsulation mpls ===> for the first L2 domain
PE1(config-vfi)# neighbor 2.2.2.2 1 encapsulation mpls ====>for the second L2 domain
PE1(config-vfi)# exit
PE1(config)# interface vlan 1
PE1(config-if)# xconnect vfi vfi_mpdu
PE1(config-if)# no shutdown
PE1(config-if)# end
```

Verification

Use the **show spanning-tree pseudo-information** command to check the R-L2GP global configuration and instances:

```
Router# show spanning-tree pseudo-information
Pseudo id 5, type transmit:
remote_id 5
mst_region_id 0, port_count 1, update_flag 0x0
mrecord 0x4ACE3BD8, mrec_count 2:
msti 0: root_id 4096.0000.9c6d.2ec0, root_cost 0, update_flag 0x0
msti 1: root_id 4097.0000.9c6d.2ec0, root_cost 0, update_flag 0x0
Pseudo interfaces:
GigabitEthernet1/6
Pseudo id 10, type transmit:
```

```
remote_id 10
mst_region_id 0, port_count 1, update_flag 0x0
mrecord 0x4ACE3B9C, mrec_count 2:
msti 0: root_id 24576.0018.7415.c980, root_cost 0, update_flag 0x0
msti 1: root_id 24577.0018.7415.c980, root_cost 0, update_flag 0x0
Pseudo interfaces:
GigabitEthernet2/0/0
```

Use the **show spanning-tree mst** command to check MSTI instances:

```
Router# show span mst
##### MST0 vlans mapped: 1-99,101-199,201-249,251-4094
Bridge address 0018.7415.c980 priority 24576 (24576 sysid 0)
Root this switch for the CIST
Operational hello time 2 , forward delay 15, max age 20, txholdcount 6
Configured hello time 2 , forward delay 15, max age 20, max hops 20
Interface Role Sts Cost Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Gi1/5 Desg FWD 20000 128.5 P2p R-L2GP
Gi5/0/0 Desg FWD 20000 128.1025 P2p R-L2GP
PW 2.2.2.2:160 Desg FWD 200 128.1055 P2p R-L2GP
##### MST1 vlans mapped: 100,250
Bridge address 0018.7415.c980 priority 24577 (24576 sysid 1)
Root this switch for MST1
Interface Role Sts Cost Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Gi1/5 Desg FWD 20000 128.5 P2p R-L2GP
Gi5/0/0 Desg FWD 20000 128.1025 P2p R-L2GP
PW 2.2.2.2:160 Desg FWD 200 128.1055 P2p R-L2GP
```

Use the **show spanning-tree** command to check STP and some MSTI instances:

```
Router# show spanning-tree detail
Port 1025 (GigabitEthernet5/0/0) of MST0 is designated forwarding
Port path cost 20000, Port priority 128, Port Identifier 128.1025.
Designated root has priority 24576, address 0018.7415.c980
Designated bridge has priority 24576, address 0018.7415.c980
Designated port id is 128.1025, designated path cost 0
Timers: message age 0, forward delay 0, hold 0
Number of transitions to forwarding state: 1
Link type is point-to-point by default, Internal
Pseudo-info (id 10) is running
BPDU: sent 119967, received 239360
hd-uut2#sho span mst interface gig5/0/0
GigabitEthernet5/0/0 of MST0 is designated forwarding
Edge port: no (default) port guard : none (default)
Link type: point-to-point (auto) bpdu filter: disable (default)
Boundary : internal bpdu guard : disable (default)
Pseudo-info (id 10) is running
Bpdus sent 120090, received 239606
Instance Role Sts Cost Prio.Nbr Vlans mapped
-------- ---- --- --------- -------- -------------------------------
0 Desg FWD 20000 128.1025 1-99,101-199,201-249,251-4094
1 Desg FWD 20000 128.1025 100,250
2 Desg FWD 20000 128.1025 200
```

Troubleshooting Tips

Table 2-22 Troubleshooting Reverse L2GP feature

Problem	Solution
RL2GP configuration issues	Use the **show spanning-tree pseudo-information** [*id* [*configuration* | *interface*]] and **debug spanning-tree pseudo-information** commands to trace the configuration sequence of the R-L2GP commands and the messages between the route and switch processor. Share the output with TAC for further investigation.
Disabled STP or MST instances	Use the **show spanning-tree** [*active* | *detail* | *interface*] command to verify the state of the STP or MST. Share the output with TAC for further investigation.
spanning-tree pseudo-information transmit command is rejected	Verify if : • All MST instances within the pseudo-information are configured within the MST global configuration. • MSTI 0 (IST) is configured within the pseudo-information.
Cannot configure MST	Re-configure MSTE and ensure that priority, MAC address and cost are the same on both the network processor engines.
System loops	Re-configure all the 64 VLAN instances per RL2GP within a Pseudo ID.
Configuration is rejected when the MST region ID is modified.	As IOS supports only single region MST, remove the multiple MSTregion IDs that have been configured and configure only a single MST ID.

Configuring Private Host Switch Virtual Interface (VLAN and VPLS)

The Private Hosts feature allows automatic insertion of Router Switch Virtual Interface (SVI) MAC into the private host configuration. Private hosts track the L2 port that a server is connected to, and limits undesired traffic through MAC-layer access control lists (ACLs). Hosts can carry multiple traffic types via trunk ports, remain isolated from each other, and still communicate to a common server. Private hosts work at the Layer 2 interface level.

From 12.2(33)SRD4 onwards, this feature redirects broadcast and unicast traffic from isolated ports over VPLS Virtual Circuit. You can add a VPLS enabled VLAN (cross-connect configured in a VLAN) in the Private Host Vlan-list along with regular VLAN and SVI.

Private Host limits VPLS support for only one VLAN. You cannot add another VPLAS VLAN if a VPLS VLAN exists in the Private Host VLAN-list. Similary , if any VLAN in the Vlan-list has cross-connect configured, you cannot configure another cross-connect on another VLAN in the Vlan-list.

Port Classification

You can classify the Ports into the following groups:

- Isolated ports: The hosts which need to be isolated will be directly or indirectly connected through DSLAMs to this type of ports. The unicast traffic received on these ports should be always destined towards specified upstream devices

- Promiscuous ports: The ports facing the core network or devices like BRAS and multicast servers are called promiscuous ports. These ports can allow any unicast or broadcast traffic received from upstream devices.

  Private hosts traffic is treated as Layer 2 traffic and routing needs an external router to be configured. Instead of configuring a server MAC address into Private Hosts, you must configure the router MAC address. This feature adds the SVIs into the Private Host configuration, eliminating the need for the external router. For more information on Private Hosts, see *Cisco 7600 Series Cisco IOS Software Configuration Guide, 12.2SR* at http://www.cisco.com/en/US/docs/routers/7600/ios/12.2SR/configuration/guide/pacl.html

Requirements and Restrictions

When you configure a Private Host SVI, follow these requirements and restrictions:

- Many VLANs can be associated with Private host feature.
- However only one of those VLANs can have cross-connect configured for Private host with VPLS.
- Other VLANs in the router can have cross-connect, but they cannot be associated with Private host feature.
- You cannot restrict Private Host SVIs to a configured subset of VLANs. If you want a subset of VLANs to use SVI's, you must ensure there are no SVIs on the VLANs that are not to be routed.
- This feature is not supported on hybrid systems.
- This feature installs protocol independent Protocol-Independent MAC Access Control Lists (PACLs) and enables MAC classification on the VLAN. As a result features likeRouting ACLs (RACLs) do not work with it.
- This feature is supported only in PFC-3BXL or cards higher than the PFC-3BX L configuration.
- This feature is not supported on EARL6 or below.
- Private Host limits VPLS support for only one VLAN. If the Private Host VLAN-list already has a VPLS VLAN (VLAN with cross-connect), addition of another VPLS VLAN is blocked. Similary, if cross-connect is configured in any VLAN in the VLAN-list, you cannot configure cross-connect on another VLAN in the VLAN-list.

To configure the private hosts SVI feature, perform the following steps in the global configuration mode:

	Command	Purpose
Step 1	`Router(config)# [no] private-hosts`	This command is used enable or disable private hosts feature on a Cisco 7600 device globally. A [no] form of the command disables the private hosts feature globally. This command is in disabled mode by default .
Step 2	`Router(config)# [no] private-hosts mac-list <mac-list name> <mac address >`	This command is used to populate the MAC address list. A [no] form of the command is used to delete MAC address from the list. The list itself is deleted after the deletion of last MAC address .

	Command	Purpose
Step 3	`Router(config)#` **`[no] private-hosts vlan-list`** `<VLAN-ID>`	This command is used to provide list of VLANs that need to be isolated. A [no] form will remove the given VLANs from the isolated VLAN list. **Note** This VLAN -list is also used to program the promiscuous devices' MAC addresses.
Step 4	`Router(config)#` **`[no] private-hosts promiscuous`** `<mac-list name> [vlan-list <VLAN IDs>]`	This command is used to provide list of promiscuous MAC addresses and optional VLAN-list on which these devices might exist. If the VLAN-list is not given, the VLAN list is taken from the global isolated VLAN- list configured. This command can be executed multiple times with different MAC-list and vlan-list combination.

Verifying the Private Hosts SVI configuration

Use the following show commands to verify the Private Hosts SVI (Interface VLAN) configuration:

Command	Purpose
`Router#` ***`show private-hosts configuration`***	Displays the global private hosts configuration.
`Router# show private-hosts access-lists`	Displays the private hosts related access lists.
`Router# show private-hosts interface configuration`	Displays the ports on which the feature is enabled with the configured mode.
`Router# show private-hosts mac-list`	Displays the configured mac-lists and their members.
`Router-sp#` `show private-hosts vlans`	Displays the private host related VPLS enabled vlans.
`Router-sp#` `show private-hosts index`	Displays the redirect index of the private hosts.

Sample Configuration For Private Hosts VPLS Configuration

The following example shows the Private Hosts VPLS configuration:

```
Router(config)# pseudowire-class mpls
Router(config-pw-class)# encapsulation mpls
Router(config)# l2 vfi 200 manual
Router(config-vfi)# vpn id 200
Router(config-vfi)# neighbor 2.2.2.2 pw-class mpls
Router(config)# private-hosts vlan-list 200-202,204-205
Router(config)# private-hosts promiscuous maclist-1
Router(config)# private-hosts promiscuous maclist-2
Router(config)# private-hosts mac-list maclist-1 0000.1111.9991
Router(config)# private-hosts mac-list maclist-2 0000.1111.9992
Router(config)# private-hosts layer3
Router(config)# private-hosts
Router(config)# int vlan 200
Router(config-if)# no shutdown
Router(config-if)# xconnect vfi 200
! Router(config-if)# interface GigabitEthernet3/2
```

```
Router(config-if)# switchport
Router(config-if)# switchport trunk encapsulation dot1q
Router(config-if)# switchport trunk allowed vlan 200-205
Router(config-if)# switchport mode trunk
Router(config-if)# private-hosts mode isolated
PE17_C7606# show private-hosts ?
access-lists Show the private hosts related access lists
configuration Show private hosts global configuration
interface Show private hosts interface related configuration
mac-list Show the mac lists and their members
PE17_C7606# show private-hosts configuration
Private hosts enabled. BR INDEX 1
Layer-3 switching on Private Hosts is enabled
All mandatory configurations configured
Privated hosts vlans lists:
100
Private promiscuous MAC configuration:
A '*' mark behind the mac list indicates non-existent mac-list
--------------------------------------------------------------------------------
MAC-list VLAN list
--------------------------------------------------------------------------------
server_list *** Uses the isolated vlans ***
--------------------------------------------------------------------------------
VLAN intf MAC addr VLAN list
--------------------------------------------------------------------------------
PE17_C7606#
```

Sample configuration for Private Hosts Interface Vlan configuration

The following example shows a typical configuration of the private hosts SVI (Interface VLAN) feature:

```
Router(config)# private-hosts vlan-list 200-202,204-205
Router(config)# private-hosts promiscuous maclist-1
Router(config)# private-hosts promiscuous maclist-2
Router(config)# private-hosts mac-list maclist-1 0000.1111.9991
Router(config)# private-hosts mac-list maclist-2 0000.1111.9992
Router(config)# private-hosts layer3
Router(config)# private-hosts
Router(config)# interface GigabitEthernet3/1
Router(config-if)# switchport
Router(config-if)# switchport access vlan 201
Router(config-if)# switchport mode access
Router(config-if)# private-hosts mode promiscuous
Router(config-if)# interface GigabitEthernet3/2
Router(config-if)# switchport
Router(config-if)# switchport trunk encapsulation dot1q
Router(config-if)# switchport trunk allowed vlan 200-205
Router(config-if)# switchport mode trunk
Router(config-if)# private-hosts mode isolated
```

Configuring Multicast Features

This section provides information about configuring ES20 line card-specific multicast features.

Configuring IGMP/PIM Snooping for VPLS Pseudowire on 7600-ESM-2X10GE and 7600-ESM-20X1GE

The Internet Group Management Protocol (IGMP)/Protocol Independent Multicast (PIM) Snooping for VPLS Pseudowire on 7600-ESM-2X10GE and 7600-ESM-20X1GE feature provides the ability to send Layer 2 multicast frames from customer equipment (CE) in a VPLS virtual forwarding instance (VFI) or from a multipoint bridging VLAN only to those remote peer CEs that have sent an IGMP request to join the multicast group.

IGMP)/PIM Snooping for VPLS Pseudowire on 7600-ESM-2X10GE and 7600-ESM-20X1GE manages multicast traffic at Layer 2 by configuring Layer 2 LAN ports dynamically to forward multicast traffic only to those ports that want to receive it. In VPLS or multipoint bridging, IGMP snooping can be set up on per- VLAN or per-VFI basis to build the membership tree because each of the remote legs of a VLAN or VFI can be identified with a virtual port and VLAN ID.

Note If a Layer2 pseudowire is configured on an interface VLAN instead of the phisycal interface and the pseudowire traffic on Layer2 reaches the destination (core facing) through an ES20 line card, the PIM and IGMP traffic crossing the pseudowire is switched to the Route Processor (software switched) resulting in high CPU utilization. To enable hardware switching for the PIM and IGMP traffic, disable IGMP snooping.

Restrictions and Usage Guidelines

When configuring the IGMP/PIM Snooping for VPLS Pseudowire on 7600-ESM-2X10GE and 7600-ESM-20X1GE feature, follow these restrictions and usage guidelines:

- IGMP/PIM snooping is enabled by default on the bridge-domain VLAN (use the **no ip igmp snooping** command to disable default behavior).
- When IGMP snooping is globally enabled, it enables IGMP snooping on all the existing VLAN interfaces. When IGMP snooping is globally disabled, it disables IGMP snooping on all the existing VLAN interfaces.
- System support for 32,000 IGMP groups with no line card-specific limitation.
- Supports MultiPoint Bridging over Ethernet on 7600-ESM-2X10GE and 7600-ESM-20X1GE
- Supports Virtual Private LAN Service (VPLS)
- Use the **show ip igmp snooping** privileged EXEC command to verify your IGMP settings.
- IGMP snooping only works when there is no tunneling operation (there should not be any VLAN tags in the packet when it is put on the bridge-domain VLAN).

SUMMARY STEPS

1. enable
2. **configure terminal**
3. interface vlan *vlanid*

4. no ip address *ip-address mask* [secondary]
5. ip igmp snooping
6. ipv6 mld snooping
7. xconnect vfi *vfi name*

DETAILED STEPS

	Command	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 vlan`** *`vlanid`*
	Example:
	`Router(config)#` **`interface vlan`** *`12`*
		Creates a unique VLAN ID number and enters subinterface configuration mode.

Step 4	`no ip address` *`ip-address mask`* `[secondary]`
	Example:
	`Router(config)# no ip address`
		Disables IP processing and enters interface configuration mode.

Step 5	`ip igmp snooping`
	Example:
	`Router(config-if)# ip igmp snooping`
		Enables IGMP snooping. To disable IGMP snooping, use the **no** form of this command.

Step 6	**`ipv6 mld snooping`**
	Example:
	`Router(config)# ipv6 mld snooping`
		Enables Multicast Listener Discovery version 2 (MLDv2) snooping globally. To disable the MLDv2 snooping globally, use the **no** form of this command.

Step 7	**`xconnect vfi`** *`vfi name`*
	Example:
	`Router(config-if)# xconnect vfi vfi16`
		Specifies the Layer 2 VFI that you are binding to the VLAN port.

Example

This is a VLAN configuration.

```
Router(config)# interface Vlan700
Router(config)# no ip address
Router(config-if)# ip igmp snooping
Router(config-if)# ipv6 mld snooping
Router(config-if)# xconnect vfi vfi700
```

Verification

Use the **show ip igmp interface vlan** command to verify a configuration.

Configuring Link State Tracking (LST)

When a link failure occurs on a REP and MST segment, the associated protocols handle the link failure event. However, if the primary link to the switch is enabled even though the corresponding uplink ports on the switch are disabled, the REP and MST protocol is unaware of backbone side, and does not trigger a failover. The router continues to receive the traffic from the access side and then drops it discreetly due to lack of backbone connectivity. Link state tracking provides a solution to this problem by allowing the uplink interfaces to bind the link status to the down link ports. Uplink state tracking is configured such that when a set of uplink ports are disabled, other ports linked through CLI commands are disabled as well. The state of all the downlink interfaces are error-disabled only when all the upstream interfaces are disabled.

The LST triggers REP/MST re-convergence on the access side depending on the state of the core-facing interface. The link state of the core facing interface and the access facing interface are bound by link state tracking group.

LST facilitates:

  - Enabling and disabling of link state group tracking.
  - Removal of downstream interfaces from a link state group.
  - Performing shut/no shut on error disabled interface.

Restrictions and Usage Guidelines

Follow these restrictions and usage guidelines when you configure the LST:

- Ensure that the management interfaces are not part of a link state group.
- REP port cannot be configured as uplink port.
- LST does not allow any interface, upstream or downstream, to be part of more than one link state group.
- You can configure a maximum of 10 link state groups.
- When you configure LST for the first time, you must add upstream interfaces to the link state group before adding downstream, otherwise the state of the downlink interfaces are error-disabled.
- The configurable interfaces are physical (both routed and switch port), port-channel, sub-interface and VLAN.
- Upstream interfaces are required to be among:
  - L3 interface(physical or portchannel)
  - SVI
- Downstream interfaces are required to be among:
  - L2 interface
  - L2 Port-channel
  - EVC

Configuring Link-State Tracking

Perform the following tasks to configure a LST.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **link state track** *number*
4. **interface** *slot/port*
5. **link state group** *[number] {***upstream** | **downstream***}*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** Example: `Router> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** Example: `Router# configure terminal`	Enters global configuration mode.
Step 3	**link state track** *number* Example: `Router(config)# link state track 1`	Creates a link-state group, and enables LST. The acceptable range is 1-10; the default value is 1.
Step 4	**interface** *slot/port* Example: `Router(config)# interface gigabitethernet 2/1`	Configures an interface.

Step 5	**link state group** *[number] {***upstream** | **downstream***}*
	Example:
	`Router(config-if)# link state group 1 upstream`
		Specifies a link-state group and configures the interface as either an upstream or downstream interface in the group.The group number can be 1 to 10; the default value is 1.

Step 6	**end**
	Example:
	`Router(config-if)# end`
		Exits the CLI to privileged EXEC mode.

This example shows how to create a link-state group and configure the interfaces:

```
Router# configure terminal
Router(config)# link state track 1
Router(config)# interface gigabitethernet3/1
Router(config-if)# link state group 1 upstream
Router(config-if)# interface gigabitethernet3/3
Router(config-if)# link state group 1 upstream
Router(config-if)# interface gigabitethernet3/5
Router(config-if)# link state group 1 downstream
Router(config-if)# interface gigabitethernet3/7
Router(config-if)# link state group 1 downstream
Router(config-if)# end
```

Verification

Use the **show link state group** command to display the link-state group information.

```
Router> show link state group 1
Link State Group: 1 Status: Enabled, Down
```

Use the **show link state group detail** command to display detailed information about the group.

```
Router> show link state group detail
(Up):Interface up (Dwn):Interface Down (Dis):Interface disabled
Link State Group: 1 Status: Enabled, Down
Upstream Interfaces : Gi3/5(Dwn) Gi3/6(Dwn)
Downstream Interfaces : Gi3/1(Dis) Gi3/2(Dis) Gi3/3(Dis) Gi3/4(Dis)
Link State Group: 2 Status: Enabled, Down
Upstream Interfaces : Gi3/15(Dwn) Gi3/16(Dwn) Gi3/17(Dwn)
Downstream Interfaces : Gi3/11(Dis) Gi3/12(Dis) Gi3/13(Dis) Gi3/14(Dis)
(Up):Interface up (Dwn):Interface Down (Dis):Interface disabled
```

Troubleshooting the Link State Tracking

Table 2-23 lists the troubleshooting issues while configuring LST:

Table 2-23 Troubleshooting LST Issues

Problem	Solution
The downstream interface is in error-disabled state even though the upstream interfaces are up.	Use the **show interfaces** <interface> **status err-disabled** command to check why the interface is in such state. Use the **show errdisable recovery** command to view information about the error-disable recovery timer.

Configuring Multicast VLAN Registration

Multicast VLAN Registration (MVR) is used to deploy multicast traffic across an Ethernet ring-based service-provider network. For example, the broadcast of multiple television channels over a service-provider network.

MVR performs the following:

- Identifies the MVR IP multicast streams and their associated IP multicast groups in the Layer 2 forwarding table.
- Intercepts the IGMP messages.
- Allows a subscriber on a port to subscribe and unsubscribe to a multicast stream on the multicast VLAN.
- Allows a single multicast VLAN to be shared in the network while subscribers remain in separate VLANs.

- Provides the ability to continuously send multicast streams in the multicast VLAN and isolate the streams from the subscriber VLANs for bandwidth and security reasons.
- Modifies the Layer 2 forwarding table to include or remove the subscriber as a receiver of the multicast stream, even though the receivers might be in a different VLAN from the source. This forwarding behavior selectively allows traffic to cross between different VLANs.

The router forwards multicast data for MVR IP multicast streams only to MVR ports on which hosts have joined, either by IGMP reports or by MVR static configuration. The router forwards IGMP reports received from MVR hosts only to the source (uplink) port. This eliminates using unnecessary bandwidth on MVR data port links.

Note Only layer 2 ports participate in MVR. You must configure ports as MVR receiver ports. Only one MVR multicast VLAN per router is allowed.

During MVR, subscriber ports subscribe and unsubscribe multicast streams by sending out IGMP join and leave messages. These messages can originate from an IGMP version-2-compatible host with an Ethernet connection. Although MVR operates on the underlying mechanism of IGMP snooping, the two features operate independent of each other. However, if IGMP snooping and MVR are both enabled, MVR reacts only to join and leave messages from multicast groups configured under MVR. Join and leave messages from all other multicast groups are managed by IGMP snooping.

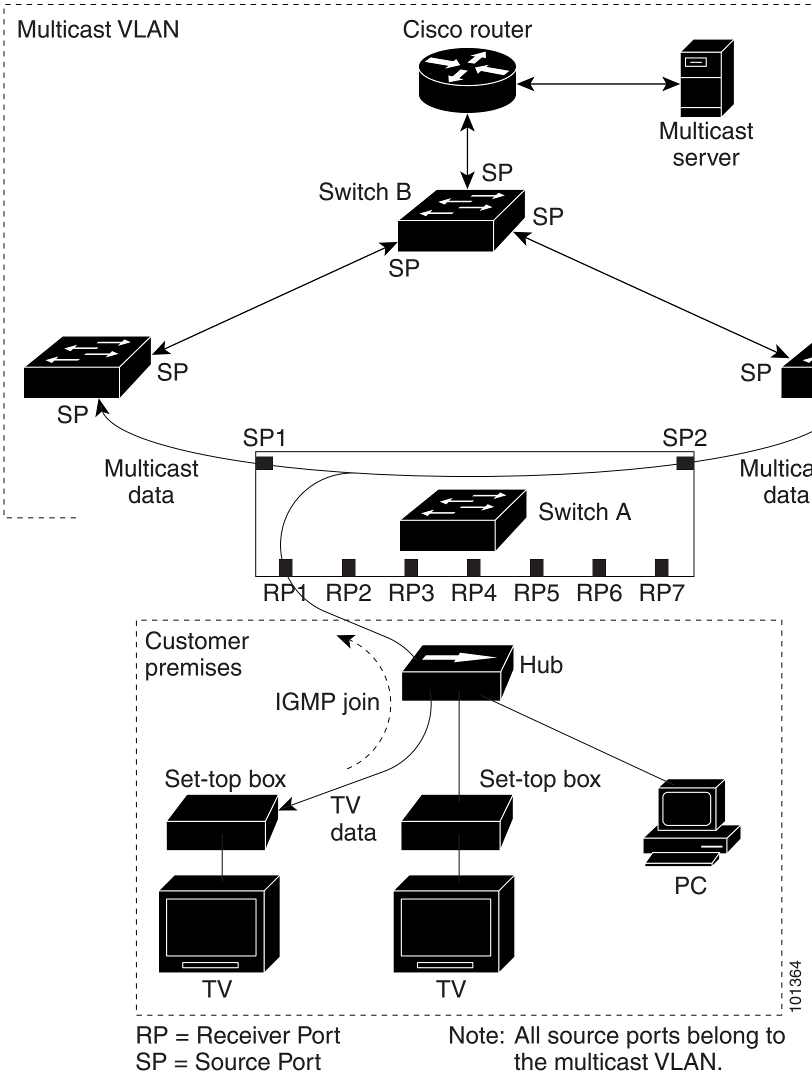
Using MVR in a Multicast Television Application

In a multicast television application, a PC or a television with a set-top box receives the multicast stream. Multiple set-top boxes or PCs can be connected to one subscriber port, which is a switch port configured as an MVR receiver port. Figure 2-8 illustrates this configuration.

The MVR feature in a multicast television application functions in this sequence:

- DHCP assigns an IP address to the set-top box or the PC. When a subscriber selects a channel, the set-top box or PC sends an IGMP report to Switch A to join the appropriate multicast. If the IGMP report matches one of the configured IP multicast group addresses, the Source Port (SP) CPU modifies the hardware address table to include this receiver port and VLAN as a forwarding destination of the specified multicast stream when it is received from the multicast VLAN. Uplink ports that send and receive multicast data to and from the multicast VLAN are called MVR source ports.

Figure 2-8 Multicast VLAN Registration



- When a subscriber changes channels or switches off the television, the set-top box sends an IGMP leave message to the multicast stream. The SP CPU sends a MAC-based general query through the receiver port VLAN. If there is another set-top box in the VLAN still subscribing to this group, that set-top box must respond within the maximum response time specified in the query. If the CPU does not receive a response, it eliminates the receiver port as a forwarding destination for this group.
- Unless the Immediate Leave feature is enabled, when the router receives an IGMP leave message from a subscriber on a receiver port, it sends out an IGMP query on that port and waits for IGMP group membership reports. If no reports are received in a configured time period, the receiver port is removed from multicast group membership. With the Immediate Leave feature enabled, an IGMP query is not sent from the receiver port on which the IGMP leave was received. As soon as the leave message is received, the receiver port is removed from multicast group membership, which speeds up leave latency. Enable the Immediate Leave feature only on receiver ports to which a single receiver device is connected.
- MVR eliminates the need to duplicate television-channel multicast traffic for subscribers in each VLAN. Multicast traffic for all channels is only sent around the VLAN trunk once—only on the multicast VLAN. The IGMP leave and join messages are in the VLAN to which the subscriber port

is assigned. These messages dynamically register for streams of multicast traffic in the multicast VLAN on the layer 3 device, Switch B. The access layer switch, Switch A, modifies the forwarding behavior to allow the traffic to be forwarded from the multicast VLAN to the subscriber port in a different VLAN, selectively allowing traffic to cross between two VLANs.

- IGMP reports are sent to the same IP multicast group address as the multicast data. The Switch A CPU must capture all IGMP join and leave messages from receiver ports and forward them to the multicast VLAN of the source (uplink) port.

Configuring MVR

For information on configuring and troubleshooting the MVR, see:
http://www.cisco.com/en/US/docs/routers/7600/ios/15S/configuration/guide/snooigmp.html

Configuring Layer 3 and Layer 4 Features

This section provides information about configuring Layer 3 and Layer 4 features on the Cisco 7600 Series ES20 line card on the Cisco 7600 series router. It includes the following topic:

- Configuring Layer 3 and Layer 4 Access Control List on a Service Instance, page 2-265

For more information about the commands used in this chapter, see the *Cisco IOS Release 12.2 SR Command References* at
http://www.cisco.com/en/US/docs/routers/7600/ios/12.2SR/configuration/guide/swcg.html.

Configuring Layer 3 and Layer 4 Access Control List on a Service Instance

An Access Control List (ACL) is a series of rules with ACEs (Access Control Elements). These ACEs define the network traffic profile, and permit or deny network traffic to Layer 3 or 4. You can use sequential ACL rules to filter network traffic. Each rule contains a filter element based on source address, destination address, protocol, and ports. For more information on the supported criteria, refer Table 2-24.

Layer 3 and Layer 4 ACL on Service Instance feature permits you to configure ACLs within an Ethernet Virtual Circuit (EVC) on the Cisco 7600 Series ES20 line cards. For more detailed information on the commands used this section, see *Cisco IOS Security Configuration Guide, Release 12.2SR* at

http://www.cisco.com/en/US/docs/ios/sec_secure_connectivity/configuration/guide/12_2sr/sec_secure_connectivity_12_2sr_book.html

Table 2-24 maps the Layer 3 and 4 ACL criterion.

Table 2-24 Supported Layers versus Criterion

Layer	Based on
Layer 3 and Layer 4	• Source • Destination IP address • Differentiated Services Code Point • Precedence • TOS (Type of Service) • Protocols • Fragmentation
Layer 4	• Source and Destination Ports

Restrictions and Usage Guidelines

When configuring the Layer 3 and Layer 4 ACLs on a Cisco 7600 Series ES20 line cards, follow these restrictions and usage guidelines:

- L3 and L4 ACLs are supported only in ingress.
- You cannot simultaneously apply L2 ACL or L3/L4 ACLs on an EVC. You can either apply a L2 ACL, or a L3/L4 ACL within an EVC.
- L3 and L4 ACLs are not supported on EVCs in port-channels.
- IPv6 ACLs are not supported.
- Per ACE counters are not supported.
- You can apply a maximum of 4000 unique ACLs.
- You can configure a maximum of 8000 ACEs in a ES20 line card.
- In a L3 or L4 ACLs, if you apply the ACL name or number without actually creating the ACL, all the packets are permitted. However, in L2 ACLs, if you apply the ACL name,the packets are dropped.
- For **eq** and **neq** L4 operators, a maximum of 10 ports are used to relay the parameters. However, you can apply the ACLs only on the first port.
- Though the ACEs contain many rules based on which network traffic is filtered, only the criterion listed in Table 2-24 are supported.

SUMMARY STEPS TO CREATE A STANDARD L3 or L4 ACL

1. **enable**
2. **configure terminal**
3. **ip access-list** ***standard*** **{*access-list-number* | *access-list-name*}**
4. permit {source *[source-wildcard]* | *any*}
5. exit

DETAILED STEPS TO CREATE A STANDARD L3 or L4 ACL

	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 access-list`** ***`standard`*** **{*`access-list-number`*|*`access-list-name`*}**
	Example:
	`Router(config)#ip access-list standard test`
		Creates a standard ACL on the selected interface. The acceptable range for a standard ACL list is 1-99 and an expanded range of 1300-1999.

Step 4	**`permit|deny {source [source-wildcard]	any}`** **Example:** `Router(config-standard-nacl)#permit udp any host 1.2.3.4`	Denies or allows the packets on the selected interface.
Step 5	`exit`	Exits the configuration mode.

SUMMARY STEPS TO CREATE A NON UDP or TCP EXTENDED LIST

1. **enable**
2. **configure terminal**
3. **ip access-list** *extended* {*access-list-number*|*access-list-name*}
4. [*sequence-number*] **deny|permit** *protocol source source-wildcard destination destination-wildcard* [[**precedence** *precedence*] [**tos** *tos*] | **dscp** *dscp*] [**fragments**]
5. Repeat Step 4 as necessary
6. end
7. **show ip access-lists** *{name|number}*

DETAILED STEPS TO CREATE CREATE A NON UDP or TCP EXTENDED LIST

	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 access-list`** ***`extended {access-list-number|access-list-name}`***
	Example:
	`Router(config)#ip access-list extended test`
		Creates a standard or extended ACL on the selected interface. The acceptable range for a extended ACL is 100-199 and the expanded range is 2000-2699.

Step 4	*`[sequence-number]`* `deny|permit` *`protocol source source-wildcard destination destination-wildcard`* `[[precedence` *`precedence`*`] [tos` *`tos`*`]	dscp` *`dscp`*`] [fragments]` **Example:** `Router(config-ext-nacl)# deny ip any any`	(Optional) Specifies a **Deny** statement in the selected IP access list mode. • This access list uses a **deny** statement first, but a **permit** statement could appear first, depending on the order of statements you need. • Use the **no** *sequence-number* form of this command to delete an entry.
Step 5	`Repeat Step 4 as necessary`	Allows you to revise the access list.

Step 6	`end`
	Example:
	`Router(config-ext-nacl)# end`
		(Optional) Exits the configuration mode and returns to the privileged EXEC mode.

Step 7	**`show ip access-lists`** ***`{name|number}`***
	Example:
	`Router# show ip access-lists test`
		(Optional) Displays the contents of the IP access list and allows you to review the output to verify the inclusion of the new access list entry.

SUMMARY STEPS TO CREATE A TCP and UDP BASED EXTENDED LIST

1. enable
2. configure terminal
3. **ip access-list extended** *access-list-name*
4. [*sequence-number*] **{permit|deny}** {**tcp|udp}** *source source-wildcard* [*operator [port]] destination destination-wildcard [operator [port*]] [**precedence** *precedence*] [**tos** *tos*] [*fragments*]
5. Repeat Step 4 as necessary, adding statements by sequence number where you planned. Use the **no** *sequence-number* command to delete an entry.

6. end
7. **show ip access-lists** *{name|number}*

DETAILED STEPS TO CREATE A A TCP and UDP BASED EXTENDED LIST

	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 access-list extended** *access-list-name* **Example:** `Router(config)#ip access-list extended test`	Creates a standard IP access list or an extended ACL on the selected interface. The acceptable range for a extended ACL is 100-199 and the expanded range is 2000-2699.
Step 4	*[sequence-number]* **{permit|deny} {tcp|udp}** *source source-wildcard [operator [port]] destination destination-wildcard [operator [port]]* [**precedence** *precedence]* [**tos** *tos]* [**fragments**] **Example:** `Router(config-ext-nacl)#permit tcp any any`	Specifies a **permit** statement in named IP access list mode. • This access list uses a **permit** statement first, but a **deny** statement could appear first, depending on the order of statements you need. • Use the TCP or the UDP command syntax of the permit command.
Step 5	Repeat Step 4 as necessary, adding statements by sequence number where you planned. Use the no sequence-number command to delete an entry.	Allows you to revise the access list.

Step 6	end
	Example:
	`Router(config-ext-nacl)# end`
		(Optional) Exits the configuration mode and returns to privileged EXEC mode.

Step 7	**show ip access-lists** ***{name|number}***
	Example:
	`Router# show ip access-lists test`
		(Optional) Displays the contents of the IP access list and allows you to review the output to verify that the access list includes the new entry.

SUMMARY STEPS TO APPLY A L3 or L4 ACL

1. **enable**
2. **configure terminal**
3. ***interface gigabitethernet*** type/ slot/port

4. **service instance** *id* **Ethernet**
5. **ip access-group** {*list name|number}* **in**
6. exit

DETAILED STEPS TO APPLY A L3 or L4 ACL

	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`** `type/ slot/port [subinterface-number]`
	Example:
	`Router(config)#interface gigabitethernet 4/0/0`
		Specifies the Gigabit Ethernet or the Ten Gigabit Ethernet interface to configure, where:
		- *slot/subslot/port*—Specifies the location of the interface.

Step 4	**`service instance`** *`id`* `{`**`Ethernet`** `[`*`service-name`*`}`
	Example:
	`Router(config-if)#service instance 101 ethernet`
		Creates a service instance on an interface and sets the device to the **config-if-srv** configuration mode.

Step 5	**`ip access-group`** ***`{list name|number} in`***
	Example:
	`Router(config-if-srv)#ip access-group test in`
		Applies a ACL on the selected EVC.
		Note The ACL displays only positive permit and deny counts.

Step 6	`exit`	Exits the configuration mode.

Examples

You can view the L3/L4 ACL counters for an EVC as shown in the following example:

```
7600bb# sh access-list test
Extended IP access list test
    10 permit udp any host 1.2.3.4 eq 5

7600bb#sh run int gi 13/0/5
Building configuration...

Current configuration : 186 bytes
!
interface GigabitEthernet13/0/5
 no ip address
 no mls qos trust
 service instance 1 ethernet
```

```
  encapsulation dot1q 100
  ip access-group test in
  bridge-domain 100
 !
end
7600bb# show ethernet service instance interface gigabitEthernet 13/0/5 detail
Service Instance ID: 1
Associated Interface: GigabitEthernet13/0/5
Associated EVC:
L2protocol drop
CE-Vlans:
Encapsulation: dot1q 100 vlan protocol type 0x8100
Interface Dot1q Tunnel Ethertype: 0x8100
State: Up
EFP Statistics:
   Pkts In   Bytes In   Pkts Out  Bytes Out
         0          0          0          0
SACL permit in count: 0
SACL deny in count: 0
EFP Microblocks:
****************
Microblock type: Bridge-domain
Bridge-domain: 100
7600bb#show ethernet service instance interface gigabitEthernet 13/0/5 stats
Port maximum number of service instances: 8000
Service Instance 1, Interface GigabitEthernet13/0/5
   Pkts In   Bytes In   Pkts Out  Bytes Out
         0          0          0          0
SACL permit in count: 0
SACL deny in count: 0
```

VRF aware IPv6 tunnel

The current IPv6 tunneling feature on c7600 does not support Virtual Routing and Forwarding (VRF) awareness. The forwarding table lookups for IPv6 overlay addresses and IPv4 transport addresses are performed in the global routing tables. This feature extends the tunneling support for IPv6 overlay addresses in VRF.

These scenarios explain the VRF aware IPv6 tunnel function:

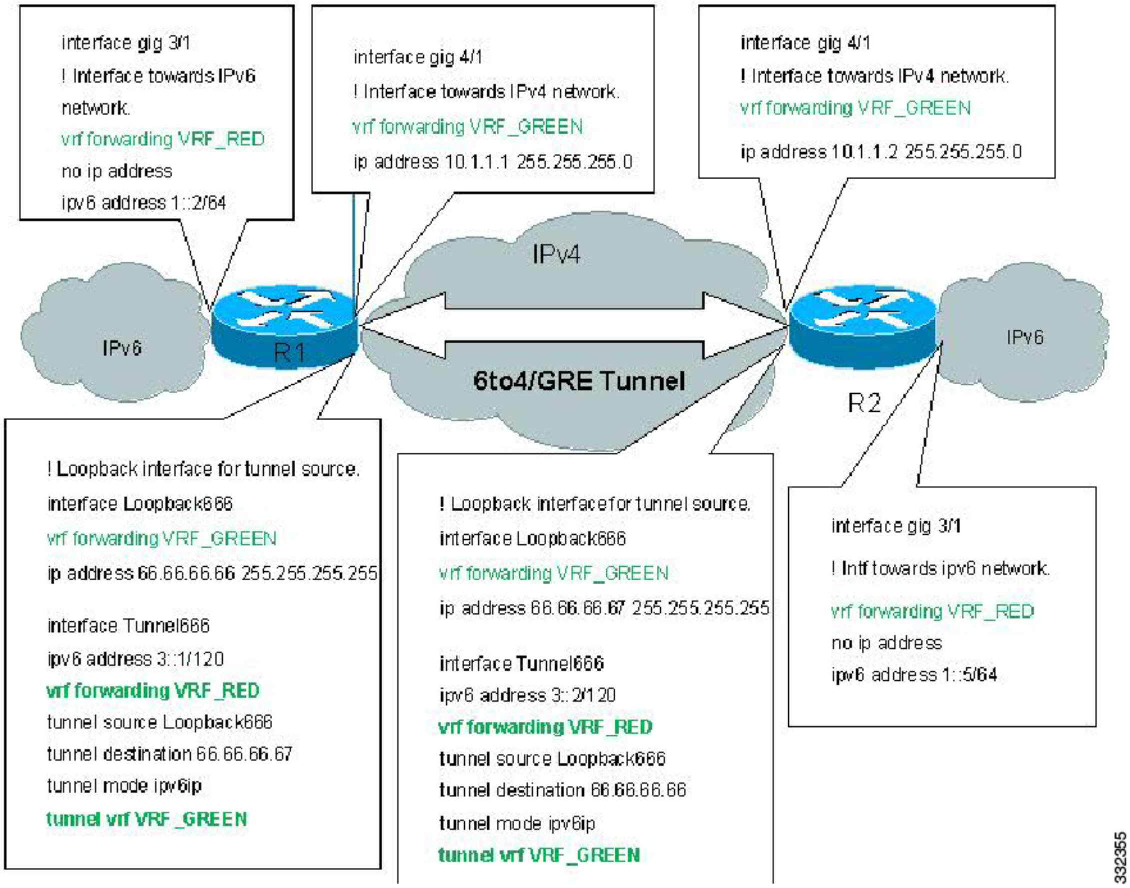
- IPv6 overlay address in VRF and IPv4 transport address in Global routing table (RT).
- IPv6 overlay address in VRF and IPv4 transport address in VRF.

Figure 2-9 illustrates the topology for the IPv6 overlay address in VRF, and the IPv4 transport address in VRF.

Figure 2-9 Topology for VRF aware IPv6 Tunnel

The VRF Aware IPv6 over IPv4 Tunnel can have any line card towards the core facing side.

.



```
interface gig 3/1
! Interface towards IPv6
network.
vrf forwarding VRF_RED
no ip address
ipv6 address 1::2/64
```

```
interface gig 4/1
! Interface towards IPv4 network.
vrf forwarding VRF_GREEN
ip address 10.1.1.1 255.255.255.0
```

```
interface gig 4/1
! Interface towards IPv4 network.
vrf forwarding VRF_GREEN
ip address 10.1.1.2 255.255.255.0
```

```
! Loopback interface for tunnel source.
interface Loopback666
vrf forwarding VRF_GREEN
ip address 66.66.66.66 255.255.255.255

interface Tunnel666
ipv6 address 3::1/120
vrf forwarding VRF_RED
tunnel source Loopback666
tunnel destination 66.66.66.67
tunnel mode ipv6ip
tunnel vrf VRF_GREEN
```

```
! Loopback interface for tunnel source.
interface Loopback666
vrf forwarding VRF_GREEN
ip address 66.66.66.67 255.255.255.255

interface Tunnel666
ipv6 address 3::2/120
vrf forwarding VRF_RED
tunnel source Loopback666
tunnel destination 66.66.66.66
tunnel mode ipv6ip
tunnel vrf VRF_GREEN
```

```
interface gig 3/1
! Intf towards ipv6 network.
vrf forwarding VRF_RED
no ip address
ipv6 address 1::5/64
```

Restrictions for VRF aware IPv6 tunnels

Following restrictions apply to the VRF aware IPv6 tunnels feature:

- This feature supports the IPv6IP and 6to4 tunnels mode.
- Due to EARL limitation, the same source tunnels across VRFs are not supported.
- The tunnel source and the tunnel destination should be in the same VRF instance.
- The tunnel IPv4 transport addresses and the physical interface where the tunnel traffic exits, should be in the same VRF instance.
- The incoming IPv6 interface and the tunnel should be in the same VRF instance.
- This feature does not support IPv6IP auto-tunnels and ISATAP.

Configuring VRF aware IPv6 tunnel

The following sections describe how to configure VRF aware IPv6 tunnel on c7600:

- Configure IPv6 overlay addresses in VRF and IPv4 transport addresses in Global RT, page 2-273

- Configure IPv6 overlay addresses in VRF and IPv4 transport addresses in VRF, page 2-278

Configure IPv6 overlay addresses in VRF and IPv4 transport addresses in Global RT

Complete the following steps to configure IPv6 overlay addresses in VRF and IPv4 transport addresses in Global RT:

SUMMARY STEPS

Step 1 **enable**

Step 2 **configure terminal**

Step 3 **ipv6 unicast-routing**

Step 4 **mls ipv6 vrf**

Step 5 **vrf definition** *vrf name*

Step 6 **rd** {*ASN:nn* | *IP address: nn*}

Step 7 **route-target [import | export | both]**{*ASN:nn* | *IP address: nn*}

Step 8 **address-family ipv6**

Step 9 **exit**

Step 10 **address-family ipv4**

Step 11 **exit**

Step 12 **exit**

Step 13 **interface gigabitethernet** *slot/subslot/port*

Step 14 **vrf forwarding** *vrf name*

Step 15 **ipv6 address** {*ipv6-address/prefix-length* | *prefix-name sub-bits/prefix-length*}

Step 16 **exit**

Step 17 **interface gigabitethernet** *slot/subslot/port*

Step 18 **ip address** *ip-address*

Step 19 **exit**

Step 20 **interface loopback** *interface-number*

Step 21 **ip address** *ip-address*

Step 22 **exit**

Step 23 **interface tunnel** *tunnel-number*

Step 24 **vrf forwarding** *vrf name*

Step 25 **ipv6 address** {*ipv6-address/prefix-length* | *prefix-name sub-bits/prefix-length*}

Step 26 **tunnel source {ip-address | interface-type** *interface-number*}

Step 27 **tunnel destination** *{hostname* | *ip-address* | *ipv6-address}*

Step 28 **tunnel mode ipv6ip**

Step 29 **end**

DETAILED STEPS

	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	**ipv6 unicast-routing** **Example:** `Router(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**mls ipv6 vrf** **Example:** `Router(config)# mls ipv6 vrf`	Enables IPv6 globally in a VRF instance.
Step 5	**vrf definition** *vrf name* **Example:** `Router(config)# vrf definition VRF_RED`	Configures a VRF instance and enters the VRF configuration mode.

Step 6	**rd** {*ASN:nn* | *IP address: nn*}
	Example:
	`Router(config-vrf)# rd 1:1`
		Specifies a route distinguisher (RD).
		• *ASN:nn:* Specifies an autonomous system number and an arbitrary number.
		• *IP address: nn:* Specifies an IP address and an arbitrary number.

Step 7	**route-target [import | export | both]{***ASN:nn* | *IP address: nn***}**
	Example:
	`Router(config-vrf)#route-target export 1:1`
	`Router(config-vrf)#route-target import 1:1`
		Creates a route-target extended community for a VRF instance. Route target extended community attributes are used to identify a set of sites and VRF instances that can receive routes with a configured route target.
		• **import:** Imports routing information from the target VPN extended community.
		• **export**: Exports routing information to the target VPN extended community.
		• **both**: Imports both import and export routing information to the target VPN extended community.
		• *ASN:nn:* Specifies an autonomous system number and an arbitrary number.
		• *IP address: nn:* Specifies an IP address and an arbitrary number.

Step 8	**address-family ipv6** **Example:** `Router#(config-vrf)#address-family ipv6`	Selects an address family type for a VRF table and enters VRF address family configuration mode. This command configures the separate route-target policies for IPv6.

	Command or Action	Purpose
Step 9	**exit** **Example:** `Router#(config-vrf-af)#exit`	Exits the address family configuration mode.
Step 10	**address-family ipv4** **Example:** `Router#(config-vrf)#address -family ipv4`	Selects an address family type for a VRF table and enters VRF address family configuration mode. This command configures the separate route-target policies for IPv4.
Step 11	**exit** **Example:** `Router# (config-vrf-af)#exit`	Exits the address family configuration mode.
Step 12	**exit** **Example:** `Router#(config-vrf)#exit`	Exits the VRF configuration mode.
Step 13	**interface gigabitethernet** *slot/subslot/port* **Example:** `Router(config)# interface gigabitethernet 3/0/1`	Enters the interface configuration mode and specifies the Gigabit interface to configure. • *slot/subslot/port* —Specifies the location of the interface. **Note** This command configures the interface towards the IPv6 network.
Step 14	**vrf forwarding** *vrf name* **Example:** `Router(config-if)#vrf forwarding VRF_RED`	Associates a VRF instance with an interface or a subinterface.

Step 15	**ipv6 address** {*ipv6-address*|*prefix-length* | *prefix-name sub-bits* |*prefix-length*}
	Example:
	`Router (config-if)# ipv6 address 1::2/64`
		Specifies the IPv6 address assigned to the interface and enables IPv6 processing on the interface.

Step 16	**exit** **Example:** `Router (config-if)#exit`	Exits interface configuration mode.

	Command or Action	Purpose

Step 17	**`interface gigabitethernet`** *`slot/subslot/port`*
	Example:
	`Router(config)# interface gigabitethernet 4/0/1`
		Enters the interface configuration mode and specifies the Gigabit interface to configure.
		• *slot/subslot/port* —Specifies the location of the interface.
		Note This command configures the interface towards the IPv4 network.

Step 18	**`ip address`** *`ip-address`*
	Example:
	`Router(config-if)#ip address 10.1.1.1 255.255.255.0`
		Assigns an IP address and subnet mask to the interface.

Step 19	**`exit`**
	Example:
	`Router(config-if)# exit`
		Exits interface configuration mode.

Step 20	**`interface loopback`** *`interface-number`*
	Example:
	`Router(config)# interface Loopback 666`
		Enters interface configuration mode and names the new loopback interface.
		Note This command configures a loopback interface for the tunnel source

Step 21	**`ip address`** *`ip-address`*
	Example:
	`Router(config-if)#ip address 66.66.66.66 255.255.255.255`
		Assigns an IP address and subnet mask to the loopback interface.

Step 22	**`exit`**
	Example:
	`Router(config-if)# exit`
		Exits interface configuration mode.

Step 23	**`interface tunnel`** *`tunnel-number`*
	Example:
	`Router(config)# interface tunnel 666`
		Specifies a tunnel interface and enters the interface configuration mode.
		Note This command configures the IPv6 tunneling over IPv4 Transport.

Step 24	**`vrf forwarding`** *`vrf name`*
	Example:
	`Router# (config-if)#vrf forwarding VRF_RED`
		Associates a VRF instance with an interface or a subinterface.
		Note This command specifies the VRF instance to which the tunnel belongs, that is, the VRF instance used for IPv6 overlay address lookup.

	Command or Action	Purpose

Step 25	**ipv6 address** {*ipv6-address/prefix-length* | *prefix-name sub-bits/prefix-length*}
	Example:
	`Router(config-if)# ipv6 address 3::1/120`
		Specifies the IPv6 address assigned to the interface and enables IPv6 processing on the interface.

Step 26	**tunnel source {ip-address | interface-type interface-number}**
	Example:
	`Router(config-if)# tunnel source loopback 666`
		Specifies the source interface type and number for the tunnel interface.

Step 27	**tunnel destination** {*host-name* | *ip-address* | *ipv6-address*}
	Example:
	`Router(config-if)# tunnel destination 10.66.66.1`
		Specifies the destination address for a tunnel interface.

Step 28	**tunnel mode ipv6ip [6rd | 6to4 | auto-tunnel | isatap]** **Example:** `Router(config-if)# tunnel mode ipv6ip`	Configures a static IPv6 tunnel interface.

Step 29	**end**
	Example:
	`Router(config-if)# end`
		Ends the current configuration session.

Configuration Example

This example shows how to configure the IPv6 overlay addresses in VRF, and the IPv4 transport addresses in the Global Routing Table:

```
Router# enable
Router# configure terminal
Router(config)# ipv6 unicast-routing
Router(config)# mls ipv6 vrf
Router(config)# vrf definition VRF_RED
Router(config-vrf)# rd 1:1
Router(config-vrf)# route-target export 1:1
Router(config-vrf)# route-target import 1:1
Router(config-vrf)# address-family ipv6
Router(config-vrf-af)# exit
Router(config-vrf)# address-family ipv4
Router(config-vrf)# (config-vrf-af)# exit
Router(config-vrf)# exit
Router(config)# interface gigabitethernet 3/0/1
Router(config-if)# vrf forwarding VRF_RED
Router(config-if)# ipv6 address 1::2/64
Router(config-if)# exit
Router(config)# interface gigabitethernet 4/0/1
Router(config-if)# ip address 10.1.1.1 255.255.255.0
Router(config-if)# exit
Router(config)# interface Loopback 666
Router(config-if)# ip address 66.66.66.66 255.255.255.255
Router(config-if)# exit
Router(config)# interface tunnel 666
Router(config-if)# vrf forwarding VRF_RED
Router(config-if)# ipv6 address 3::1/120
Router(config-if)# tunnel source loopback 666
Router(config-if)# tunnel destination 10.66.66.1
Router(config-if)# tunnel mode ipv6ip
Router(config-if)# end
```

Configure IPv6 overlay addresses in VRF and IPv4 transport addresses in VRF

Complete the following steps to configure IPv6 overlay addresses in VRF, and IPv4 transport addresses in VRF:

SUMMARY STEPS

Step 1 **enable**

Step 2 **configure terminal**

Step 3 **ipv6 unicast-routing**

Step 4 **mls ipv6 vrf**

Step 5 **vrf definition** *vrf name 1*

Step 6 **rd** *{ASN:nn* | *IP address: nn}*

Step 7 **route-target [import | export | both]** *{ASN:nn* | *IP address: nn}*

Step 8 **address-family ipv6**

Step 9 **exit**

Step 10 **address-family ipv4**

Step 11 **exit**

Step 12 **exit**

Step 13 **vrf definition** *vrf name 2*

Step 14 **rd** *{ASN:nn* | *IP address: nn}*

Step 15 **route-target [import | export | both]** *{ASN:nn* | *IP address: nn}*

Step 16 **address-family ipv4**

Step 17 **exit**

Step 18 **exit**

Step 19 **interface gigabitethernet** *slot/subslot/port*

Step 20 **vrf forwarding** *vrf name 1*

Step 21 **ipv6 address** {*ipv6-address/prefix-length* | *prefix-name sub-bits/prefix-length*}

Step 22 **exit**

Step 23 **interface gigabitethernet** *slot/subslot/port*

Step 24 **vrf forwarding** *vrf name 2*

Step 25 **ip address** *ip-address*

Step 26 **exit**

Step 27 **interface loopback** *interface-number*

Step 28 **vrf forwarding** *vrf name 2*

Step 29 **ip address** *ip-address*

Step 30 **exit**

Step 31 **interface tunnel** *tunnel-number*

Step 32 **vrf forwarding** *vrf name 1*

Step 33 **ipv6 address** {*ipv6-address/prefix-length* | *prefix-name sub-bits/prefix-length*}

Step 34 **tunnel source** {**ip-address** | **interface-type** *interface-number*}

Step 35 **tunnel destination** *{hostname* | *ip-address* | *ipv6-address}*

Step 36 **tunnel mode ipv6ip**

Step 37 **tunnel vrf** *vrf name 2*

Step 38 **end**

DETAILED STEPS

	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	**ipv6 unicast-routing** **Example:** `Router(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**mls ipv6 vrf** **Example:** `Router(config)# mls ipv6 vrf`	Enables IPv6 globally in a VRF instance.
Step 5	**vrf definition** *vrf name 1* **Example:** `Router(config)# vrf definition VRF_RED`	Configures a VRF instance and enters the VRF configuration mode.

Step 6	**rd** {*ASN:nn* | *IP address: nn*}
	Example:
	`Router(config-vrf)# rd 1:1`
		Specifies a route distinguisher (RD).
		• *ASN:nn:* Specifies an autonomous system number and an arbitrary number.
		• *IP address: nn:* Specifies an IP address and an arbitrary number.

Step 7	**route-target [import | export | both]{***ASN:nn* | *IP address: nn***}**
	Example:
	`Router(config-vrf)#route-target export 1:1`
	`Router(config-vrf)#route-target import 1:1`
		Creates a route-target extended community for a VRF instance. Route target extended community attributes are used to identify a set of sites and VRF instances that can receive routes with a configured route target.
		• **import:** Imports routing information from the target VPN extended community.
		• **export**: Exports routing information to the target VPN extended community.
		• **both**: Imports both import and export routing information to the target VPN extended community.
		• *ASN:nn:* Specifies an autonomous system number and an arbitrary number.
		• *IP address: nn:* Specifies an IP address and an arbitrary number.

Step 8	**address-family ipv6** **Example:** `Router(config-vrf)#address-family ipv6`	Select san address family type for a VRF table and enters VRF address family configuration mode. This command configures the separate route-target policies for IPv6.

	Command or Action	Purpose
Step 9	**exit** **Example:** `Router(config-vrf-af)#exit`	Exits the address family configuration mode.
Step 10	**address-family ipv4** **Example:** `Router(config-vrf)#address-family ipv4`	Selects an address family type for a VRF table and enters VRF address family configuration mode. This command configures the separate route-target policies for IPv4.
Step 11	**exit** **Example:** `Router(config-vrf-af)#exit`	Exits the address family configuration mode.
Step 12	**exit** **Example:** `Router(config-vrf)#exit`	Exits the VRF configuration mode.
Step 13	**vrf definition** *vrf name 2* **Example:** `Router(config)# vrf definition VRF_GREEN`	Configures a VRF instance and enters the VRF configuration mode.

Step 14	**rd** {*ASN:nn* | *IP address: nn*}
	Example:
	`Router(config-vrf)# rd 1:1`
		Specifies a route distinguisher (RD).
		• *ASN:nn:* Specifies an autonomous system number and an arbitrary number.
		• *IP address: nn:* Specifies an IP address and an arbitrary number.

Step 15	**route-target [import | export | both]{***ASN:nn* | *IP address: nn***}**
	Example:
	`Router(config-vrf)#route-target export 1:1`
	`Router(config-vrf)#route-target import 1:1`
		Creates a route-target extended community for a VRF instance. Route target extended community attributes are used to identify a set of sites and VRF instances that can receive routes with a configured route target.
		• **import:** Imports routing information from the target VPN extended community.
		• **export**: Exports routing information to the target VPN extended community.
		• **both**: Imports both import and export routing information to the target VPN extended community.
		• *ASN:nn:* Specifies an autonomous system number and an arbitrary number.
		• *IP address: nn:* Specifies an IP address and an arbitrary number.

Step 16	**address-family ipv4** **Example:** `Router(config-vrf)#address-family ipv4`	Selects an address family type for a VRF table and enters VRF address family configuration mode. This command configures the separate route-target policies for IPv4.

	Command or Action	Purpose

Step 17	**`exit`**
	Example:
	`Router (config-vrf-af)#exit`
		Exits the address family configuration mode.

Step 18	**`exit`**
	Example:
	`Router(config-vrf)#exit`
		Exits the VRF configuration mode.

Step 19	**`interface gigabitethernet`** *`slot/subslot/port`*
	Example:
	`Router(config)# interface gigabitethernet 3/0/1`
		Enters the interface configuration mode and specifies the Gigabit interface to configure.
		• *slot/subslot/port* —Specifies the location of the interface.
		Note This command configures the interface towards the IPv6 network.

Step 20	**`vrf forwarding`** *`vrf name 1`*
	Example:
	`Router(config-if)#vrf forwarding VRF_RED`
		Associates a VRF instance with an interface or a subinterface.

Step 21	**`ipv6 address`** `{`*`ipv6-address`*`|`*`prefix-length`* `|` *`prefix-name sub-bits`* `|`*`prefix-length`*`}`
	Example:
	`Router(config-if)# ipv6 address 1::2/64`
		Specifies the IPv6 address assigned to the interface and enables IPv6 processing on the interface.

Step 22	**exit** **Example:** `Router# (config-if)# exit`	Exits interface configuration mode.

Step 23	**`interface gigabitethernet`** *`slot/subslot/port`*
	Example:
	`Router(config)# interface gigabitethernet 4/0/1`
		Enters the interface configuration mode and specifies the Gigabit interface to configure.
		• *slot/subslot/port* —Specifies the location of the interface.
		Note This command configures the interface towards the IPv4 network.

Step 24	**`vrf forwarding`** *`vrf name 2`*
	Example:
	`Router(config-if)#vrf forwarding VRF_GREEN`
		Associates a VRF instance with an interface or a subinterface.

	Command or Action	Purpose
Step 25	**ip address** *ip-address* **Example:** `Router(config-if)#ip address 10.1.1.1 255.255.255.0`	Assigns an IP address and subnet mask to the interface.
Step 26	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 27	**interface loopback** *interface-number* **Example:** `Router(config)# interface Loopback 666`	Enters interface configuration mode and names the new loopback interface. **Note** This command configures a loopback interface for the tunnel source
Step 28	**vrf forwarding** *vrf name 2* **Example:** `Router(config-if)#vrf forwarding VRF_GREEN`	Associates a VRF instance with an interface or a subinterface.
Step 29	**ip address** *ip-address* **Example:** `Router(config-if)#ip address 66.66.66.66 255.255.255.255`	Assigns an IP address and subnet mask to the loopback interface.
Step 30	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 31	**interface tunnel** *tunnel-number* **Example:** `Router(config)# interface tunnel 666`	Specifies a tunnel interface and enters the interface configuration mode. **Note** This command configures the IPv6 tunneling over IPv4 Transport.
Step 32	**vrf forwarding** *vrf name 1* **Example:** `Router(config-if)#vrf forwarding VRF_RED`	Associates a VRF instance with an interface or a subinterface. **Note** This command specifies the VRF instance to which the tunnel belongs, that is, the VRF instance used for IPv6 overlay address lookup.

	Command or Action	Purpose

Step 33	**ipv6 address** {*ipv6-address/prefix-length* | *prefix-name sub-bits/prefix-length*}
	Example:
	`Router(config-if)# ipv6 address 3::1/120`
		Specifies the IPv6 address assigned to the interface and enables IPv6 processing on the interface.

Step 34	**tunnel source {ip-address | interface-type interface-number}**
	Example:
	`Router(config-if)# tunnel source loopback 666`
		Specifies the source interface type and number for the tunnel interface.

Step 35	**tunnel destination** {*host-name* | *ip-address* | *ipv6-address*}
	Example:
	`Router(config-if)# tunnel destination 10.66.66.1`
		Specifies the destination address for a tunnel interface.

Step 36	**tunnel mode ipv6ip** **Example:** `Router(config-if)# tunnel mode ipv6ip`	Configures a static IPv6 tunnel interface.
Step 37	**tunnel vrf** *vrf name 2* **Example:** `Router(config-if)# tunnel vrf VRF_GREEN`	Configures a VRF instance with a specific tunnel destination, interface or a subinterface. **Note** This command specifies the VRF instance used for tunnel IPv4 transport address lookup, that is, the tunnel source and the tunnel destination.

Step 38	**end**
	Example:
	`Router(config-if)# end`
		Ends the current configuration session.

Configuration Example

This example shows how to configure the IPv6 overlay addresses in VRF, and the IPv4 transport addresses in VRF:

```
Router# enable
Router# configure terminal
Router(config)# ipv6 unicast-routing
Router(config)# mls ipv6 vrf
Router(config)# vrf definition VRF_RED
Router(config-vrf)# rd 1:1
Router(config-vrf)# route-target export 1:1
Router(config-vrf)# route-target import 1:1
Router(config-vrf)# address-family ipv6
Router(config-vrf-af)# exit
Router(config-vrf)# address-family ipv4
Router(config-vrf-af)# exit
Router(config-vrf)# exit
Router(config)# vrf definition VRF_GREEN
Router(config-vrf)# rd 1:1
Router(config-vrf)# route-target export 1:1
Router(config-vrf)# route-target import 1:1
Router(config-vrf)# address-family ipv4
Router(config-vrf-af)# exit
Router(config-vrf)# exit
Router(config)# interface gigabitethernet 3/0/1
Router(config-if)# vrf forwarding VRF_RED
Router(config-if)# ipv6 address 1::2/64
Router(config-if)# exit
Router(config)# interface gigabitethernet 4/0/1
Router(config-if)# vrf forwarding VRF_GREEN
Router(config-if)# ip address 10.1.1.1 255.255.255.0
Router(config-if)# exit
Router(config)# interface Loopback 666
Router(config-if)# vrf forwarding VRF_GREEN
Router(config-if)# ip address 66.66.66.66 255.255.255.255
Router(config-if)# exit
Router(config)# interface tunnel 666
Router(config-if)# vrf forwarding VRF_RED
Router(config-if)# ipv6 address 3::1/120
Router(config-if)# tunnel source loopback 666
Router(config-if)# tunnel destination 10.66.66.1
Router(config-if)# tunnel mode ipv6ip
Router(config-if)# tunnel vrf VRF_GREEN
Router(config-if)# end
```

Verifying the Configuration

Use these commands to verify the configuration of VRF aware IPv6 tunnel on c7600:

```
Router# show vrf vrf-red
    Name                             Default RD          Protocols   Interfaces
    vrf-red                          100:1               ipv4,ipv6   Tu666

Router# show interface tunnel 666
    Tunnel666 is up, line protocol is up
      Hardware is Tunnel
      Internet address is 80.1.1.1/24
      MTU 17916 bytes, BW 100 Kbit/sec, DLY 50000 usec,
         reliability 255/255, txload 1/255, rxload 1/255
      Encapsulation TUNNEL, loopback not set
      Keepalive not set
      Tunnel source 66.66.66.66 (Loopback666), destination 66.66.66.65
       Tunnel Subblocks:
          src-track:
             Tunnel666 source tracking subblock associated with Loopback666
              Set of tunnels with source Loopback666, 1 member (includes iterators), on
    interface <OK>
      Tunnel protocol/transport GRE/IP
        Key disabled, sequencing disabled
        Checksumming of packets disabled
      Tunnel TTL 255, Fast tunneling enabled
      Tunnel transport MTU 1476 bytes
      Tunnel transmit bandwidth 8000 (kbps)
      Tunnel receive bandwidth 8000 (kbps)
      Last input 00:07:00, output 00:02:39, output hang never
      Last clearing of "show interface" counters 00:07:19
      Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
```

```
Queueing strategy: fifo
Output queue: 0/0 (size/max)
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
L2 Switched: ucast: 0 pkt, 0 bytes - mcast: 0 pkt, 0 bytes
L3 in Switched: ucast: 0 pkt, 0 bytes - mcast: 0 pkt, 0 bytes mcast
L3 out Switched: ucast: 0 pkt, 0 bytes mcast: 0 pkt, 0 bytes
   20 packets input, 1944 bytes, 0 no buffer
   Received 0 broadcasts (0 IP multicasts)
   0 runts, 0 giants, 0 throttles
   0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
   26 packets output, 2504 bytes, 0 underruns
   0 output errors, 0 collisions, 0 interface resets
   0 unknown protocol drops
   0 output buffer failures, 0 output buffers swapped out
```

Troubleshooting Tips

For troubleshooting information, contact Cisco Technical Assistance Center (TAC) at:

http://www.cisco.com/en/US/support/tsd_cisco_worldwide_contacts.html

IPv6 Policy Based Routing

IPv6 policy-based routing (PBR) provides a flexible mechanism to route packets and define policy for the traffic flows. It extends and complements the existing mechanisms provided by routing protocols. PBR also provides a basic packet-marking capability.

PBR performs the following tasks:

- Classifies traffic based on extended access list criteria. It provides access to lists and then establishes the match criteria.
- Sets IPv6 precedence bits and enables the network to differentiate classes of service.
- Routes packets to specific traffic-engineered paths. You can route the packets to allow a specific quality of service (QoS) through the network.

The Cisco 7600 Series Router implements this feature using the Earl7 forwarding engines capability to classify traffic through an Access Control List (ACL) Ternary Content Addressable Memory (TCAM) lookup. The ACL TCAM lookup classifies traffic based on the combination of a variety of Layer 3 and Layer 4 traffic parameters. Once classified, the ACL TCAM drives results for matching flows. The Feature Manager (FM) component converts the route map policy configured on an interface into a series of values, masks and results (VMRs) and programs these in the ACL TCAM.

Policy Based Routing

All packets received on a PBR-enabled interface are passed through enhanced packet filters known as route maps. Route maps are composed of statements that are marked as *permit* or *deny*, and they are interpreted in these ways:

- If a packet matches all match statements for a route map that is marked as *permit*, the router subjects the packet to PBR using the set statements.
- If the packet matches any match statements for a route map that is marked as *deny*, the router does not subject the packet to PBR and forwards it normally.

- If the statement is marked as permit and the packets do not match any route map statements, the router sends the packets back through the normal forwarding channels and performs destination-based routing.

Packet Matching

The IPv6 PBR match criterion for a sequence is specified through a combination of IPv6 access-lists and packet length operations. Match statements are evaluated first by the criteria specified in the **match ipv6 address** command and then by criteria specified in the **match length** command. Therefore, if both an ACL and a length statement are used, a packet is first subjected to an ACL match. Only packets that pass the ACL match are subjected to the length match. Finally, only packets that pass both the ACL and the length statement are policy routed.

Packet Forwarding Using Set Statements

PBR for IPv6 packet forwarding is controlled using a number of set statements in the PBR route map. Listed below are the forwarding actions in order of decreasing priority, and the manner in which these options are reflected in the result from the VMRs programmed in the ACL TCAM. When more than one kind of packet forwarding action is specified in a sequence, the one with the highest priority is chosen.

Table 2-25 Packet Forwarding Set Statements

Set Statement	Notes
set vrf *vrf name*	Specifies the VPN Routing and Forwarding (VRF) instance to which the packet should be sent, based on packet attributes. By default the VRF that a packet is forwarded on is the same as the VRF that receives the packet.
set ipv6 next-hop *next-hop ipv6 address*	Specifies the next hop for the packet. The next hop must be present in the Routing Information Base (RIB); it must be directly connected, and it must be a global IPv6 address. If the next hop is invalid, the set statement is ignored.
set interface *next-hop interface*	Specifies the next hop interface for the packet. A packet is forwarded out of a specified interface. An entry for the packet destination address must exist in the IPv6 RIB, and the specified output interface must be in the path set. If the interface is invalid, the set statement is ignored.
set ipv6 default next-hop *default next-hop ipv6 address*	Specifies the connected next hop for the packet if the usual forwarding method fails to produce the default result. It must be a global IPv6 address. This set statement is used only when there is no explicit entry for the packet destination in the IPv6 RIB.
set default interface *default next-hop interface*	Specifies the default next-hop interface, from which the matching packets are forwarded if the usual forwarding method fails to produce a result. This set statement is used only when there is no explicit entry for the packet destination in the IPv6 RIB.

Restrictions for IPv6 PBR

Following restrictions apply to the IPv6 PBR:

- Match length is not supported in the hardware, and the PBR is applied to the software.
- Packet marking actions are not supported in the hardware, and packets requiring marking due to PBR are punted to the software.
- Set interface is supported in the hardware only for the serial interface. Other interfaces are supported on the software.
- Packets containing an IPv6 hop-by-hop header need to be examined by the router and are punted to the software. Such packets are subjected to PBR in the software.
- PBR policies using access-lists matching on IPv6 flow label, DSCP value and extension headers such as, routing, mobility, destination headers cannot be fully classified in the hardware, and are punted to the software after partial classification.
- It is not possible to completely classify traffic in hardware, when access-lists matching on non compressible addresses are used. In such cases, the PBR is applied to the software.
- On Tycho based systems, fragment packets that require matching on layer 4 protocol are punted to the software .
- IPv6 PBR on SVI interfaces is applied to the software, and hardware provides only partial classification.
- IPv6 PBR when applied to hardware will also be applied on packets destined to a router address.
- A set next-hop action where the next-hop is at the other end of a tunnel is not supported in the hardware.
- For set interface and set default interface, the interface should be a point-to-point one.
- PBR is not applied to multicast traffic and the traffic destined to link local addresses.
- When there is no traffic flow, the TCAM entry does not change from punt to policy-route.

Configuring IPv6 PBR

To configure, verify and troubleshoot the IPv6 PBR, see: Configuring IPv6 PBR.

Configuring MPLS Features

This section describes the MPLS features that have ES20 line card-specific configuration guidelines.

This section includes the following topics:

- Configuring Any Transport over MPLS, page 2-289
- Configuring MPLS Traffic Engineering Class-Based Tunnel Selection, page 2-293
- Configuring Virtual Private LAN Service, page 2-302

Configuring Any Transport over MPLS

Any Transport over MPLS (AToM) transports Layer 2 packets over a Multiprotocol Label Switching (MPLS) backbone. AToM uses a directed Label Distribution Protocol (LDP) session between edge routers for setting up and maintaining connections. Forwarding occurs through the use of two levels of labels, switching between the edge routers. The external label (tunnel label) routes the packet over the MPLS backbone to the egress Provider Edge (PE) at the ingress PE. The VC label is a demuxing label that determines the connection at the tunnel endpoint (the particular egress interface on the egress PE as well as the virtual path identifier [VPI]/virtual channel identifier [VCI] value for an ATM Adaptation Layer 5 [AAL5] protocol data unit [PDU], the data-link connection identifier [DLCI] value for a Frame Relay PDU, or the virtual LAN [VLAN] identifier for an Ethernet frame).

Scalable EoMPLS on 7600-ESM-2X10GE and 7600-ESM-20X1GE

The Scalable EoMPLS on 7600-ESM-2X10GE and 7600-ESM-20X1GE feature improves EoMPLS scalability on the Cisco 7600 router. With Scalable EoMPLS, the CE-facing line card performs all EoMPLS imposition and disposition label processing. From the core-side line card perspective, the AToM packets in and out of the router appear as generic MPLS frames.

Restrictions and Usage Guidelines

When configuring the Scalable EoMPLS on 7600-ESM-2X10GE and 7600-ESM-20X1GE feature, follow these restrictions and usage guidelines:

- Scalable EoMPLS is supported with EVCs (Ethernet Virtual Circuits). An EVC is an end-to-end representation of a single instance of a layer 2 service being offered by a provider to a customer.
- Scalable EoMPLS is supported as a mapped service for the QinQ termination
- Service Instances supported- 16, 000 per line card (32, 000 per Cisco 7600 series router)
- VC type 4 and VC type 5 are supported.
- Control word operation is supported.
- For ingress policing, only the drop action and the accept action for the **police** command are supported.
- Ingress CoS marking is not supported.
- For QoS marking, mapping of the incoming VLAN dot1q p-bits to the outgoing MPLS EXP bits is supported.
- For QoS marking, mapping of the incoming MPLS EXP bits to the outgoing VLAN dot1q p-bits is supported.
- For QoS shaping, egress pseudowire shaping is supported. Matching is based on the MPLS EXP bits.
- The Dot1q Transparency for EoMPLS feature is supported.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. ***interface gigabitethernet*** *slot/subslot/port[.subinterface-number] or* ***interface tengigabitethernet*** *slot/subslot/port[.subinterface-number]*

4. **[no] service instance** *id* **{Ethernet** [*service-name*}
5. **encapsulation dot1q** *vlad id*
6. **rewrite ingress tag** {**push {dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} [**symmetric**]
7. **xconnect** *peer-id vc-id* **encapsulation mpls**

DETAILED STEPS

	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`** `slot/subslot/port[.subinterface-number]`
	`or`
	`interface tengigabitethernet` `slot/subslot/port[.subinterface-number]`
	Example:
	`Router(config)# interface gigabitethernet 4/0/0`
		Specifies the Gigabit Ethernet or the Ten Gigabit Ethernet interface to configure, where:
		• *slot/subslot/port*—Specifies the location of the interface.
		• *subinterface-number*—(Optional) Specifies a secondary interface (subinterface) number.

Step 4	`[`**`no`**`]` **`service instance`** *`id`* `{`**`Ethernet`** `[`*`service-name`*`}`
	Example:
	`Router(config-if)# service instance 101 ethernet`
		Creates a service instance (an instantiation of an EVC) on an interface and sets the device into the config-if-srv submode.

Step 5	**`encapsulation dot1q`** ***`vlan id`***
	Example:
	`Router(config-if-srv)# encapsulation dot1q 5`
		Defines the matching criteria to map ingress dot1q frames on an interface to the appropriate service instance.
		Note Use the **encapsulation dot1q default** command to configure the default service instance on a port. Use the **encapsulation dot1q untagged** command to map untagged Ethernet frames on an ingress interface to a service instance.

	Command or Action	Purpose
Step 6	**rewrite ingress tag** {**push {dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} [**symmetric**] **Example:** `Router(config-if-srv)# rewrite ingress tag dot1q single symmetric`	Specifies the tag manipulation that is to be performed on the frame ingress to the service instance.
Step 7	**xconnect** peer-id vc-id **encapsulation mpls** **Example:** `Router(config-if-srv)# xconnect 10.0.0.1 123 encapsulation mpls`	Configures scalable EoMPLS on a service instance. On the ingress side, after proper encapsulation manipulations, a packet is tunneled in an EoMPLS VC and transmitted on the core.

Examples

The following is an example of a basic configuration:

This is the customer-facing port at router 1.

```
Router(config)# interface TenGigabitEthernet1/0/1
Router(config-if)# service instance 100 ethernet
Router(config-if-srv)# encapsulation dot1q 100
Router(config-if-srv)# rewrite ingress tag translate 1-to-2 dot1q 5 second-dot1q 5
symmetric
Router(config-if-srv)# xconnect 2.2.2.2 100 encapsulation mpls
```

This is the global configuration at router 1.

```
Router(config)# interface loopback1
Router(config-if)# ip address 1.1.1.1 255.255.255.255

!MPLS core facing port
Router(config-if)# ip address 20.1.1.1 255.255.255.0
Router(config-if)# mpls label protocol ldp
Router(config-if)# mpls ip
```

This is the customer-facing port at router 2.

```
Router(config)# interface TenGigabitEthernet2/0/2
Router(config-if)# service instance 100 ethernet
Router(config-if-srv)# encapsulation dot1q 100
Router(config-if-srv)# rewrite ingress tag translate 1-to-2 dot1q 5 second-dot1q 5
symmetric
Router(config-if-srv)# xconnect 1.1.1.1 100 encapsulation mpls
```

This is the global configuration at router 2.

```
Router(config)# interface loopback1
Router(config-if)# ip address 2.2.2.2 255.255.255.255
```

!MPLS core facing port

```
Router(config-if)# ip address 20.1.1.2 255.255.255.0
Router(config-if)# mpls label protocol ldp
Router(config-if)# mpls ip
```

The following is an example of single tag VLAN configuration for tunneling a single VLAN service instance.

! Customer facing port

```
Router(config)# interface TenGigabitEthernet2/0/2
Router(config-if)# service instance 100 ethernet
Router(config-if-srv)# encapsulation dot1q 100
Router(config-if-srv)# rewrite ingress tag translate 1-to-2 dot1q 5 second-dot1q 5
symmetric
Router(config-if-srv)# xconnect 1.1.1.1 100 encapsulation mpls
```

The following is an example of double tag VLAN configuration for tunneling double tag VLAN frames.

! Customer facing port

```
Router(config)# interface TenGigabitEthernet2/0/2
Router(config-if)# service instance 100 ethernet
Router(config-if-srv)# encapsulation dot1q 100 second-dot1q 200
Router(config-if-srv)# rewrite ingress tag translate 2-to-2 dot1q 5 second-dot1q 5
symmetric
Router(config-if-srv)# xconnect 1.1.1.1 100 encapsulation mpls
```

The following is an example of a selective QinQ cross connect configuration.

! Customer facing port

```
Router(config)# interface TenGigabitEthernet2/0/2
Router(config-if)# service instance 100 ethernet
Router(config-if-srv)# encapsulation dot1q 10-20, 30, 50-60
Router(config-if-srv)# xconnect 1.1.1.1 100 encapsulation mpls
```

The following is an example of a port-based cross connect tunnel configuration that tunnels all incoming packets to the remote peer.

```
!All tag and non-tag packets aggregation
Router(config)# interface TenGigabitEthernet2/0/2
Router(config-if)# service instance 100 ethernet
Router(config-if-srv)# encapsulation default
Router(config-if-srv)# xconnect 1.1.1.1 100 encapsulation mpls

!All non-tag packets aggregation
Router(config)# interface TenGigabitEthernet2/0/2
Router(config-if)# service instance 100 ethernet
Router(config-if-srv)# encapsulation untagged
Router(config-if-srv)# xconnect 1.1.1.1 100 encapsulation mpls
```

Verification

Use the following commands can be used to verify operation.

.

Command	Purpose
Router# **show ethernet service evc** [**id** *evc-id* | **interface** *interface-id*] [**detail**]	Displays information pertaining to a specific EVC if an EVC ID is specified, or pertaining to all EVCs on an interface if an interface is specified. The **detailed** option provides additional information on the EVC.
Router# **show ethernet service instance** [**id** *instance-id* **interface** *interface-id* | **interface** *interface-id*] [**detail**]	Displays information about one or more service instances. If a service instance ID and interface are specified, only data pertaining to that particular service instance is displayed. If only an interface ID is specified, displays data for all service instances s on the given interface.
Router# **show ethernet service interface** [*interface-id*] [**detail**]	Displays information in the Port Data Block (PDB).
Router# **show mpls l2 vc** *min VC ID max VC ID* **detail**	Displays detailed information related to the virtual connection (VC).
Router# **show mpls l2transport vc**	Displays the state of VCs.
Router# **show mpls forwarding** (Output should have the label entry l2ckt)	Displays the contents of the Multiprotocol Label Switching (MPLS) Label Forwarding Information Base (LFIB).

Configuring MPLS Traffic Engineering Class-Based Tunnel Selection

Multiprotocol Label Switching (MPLS) Traffic Engineering (TE) Class-Based Tunnel Selection (CBTS) enables you to dynamically route and forward traffic with different class of service (CoS) values onto different TE tunnels between the same tunnel headend and the same tailend. The TE tunnels can be regular TE tunnels or DiffServ-aware TE (DS-TE) tunnels.

The set of TE (or DS-TE) tunnels from the same headend to the same tailend that you configure to carry different CoS values is referred to as a “tunnel bundle.” Tunnels are “bundled” by creating a master tunnel and then attaching member tunnels to the master tunnel. After configuration, CBTS dynamically routes and forwards each packet into the tunnel that meets the following requirements:

- Is configured to carry the CoS of the packet
- Has the right tailend for the destination of the packet

Because CBTS offers dynamic routing over DS-TE tunnels and requires minimum configuration, it greatly eases deployment of DS-TE in large-scale networks.

CBTS can distribute all CoS values on eight different tunnels.

CBTS also allows the TE tunnels of a tunnel bundle to exit headend routers through different interfaces.

CBTS configuration involves performing the following tasks:

- Creating multiple (DS-) TE tunnels with the same headend and tailend and indicating on each of these tunnels which CoSs are to be transported on the tunnel.
- Creating a master tunnel, attaching the member tunnels to it, and making the master tunnel visible for routing.

Restrictions and Usage Guidelines

When configuring MPLS Traffic Engineering Class-Based Tunnel Selection (CBTS), follow these restrictions and usage guidelines:

- CBTS has the following prerequisites:
  - MPLS enabled on all tunnel interfaces
  - Cisco Express Forwarding (CEF) or distributed CEF (dCEF) enabled in general configuration mode
- CBTS has the following restrictions:
  - For a given destination, all CoS values are carried in tunnels terminating at the same tailend. Either all CoS values are carried in tunnels or no values are carried in tunnels. In other words, for a given destination, you cannot map some CoS values in a DS-TE tunnel and other CoS values in a Shortest Path First (SPF) Label Distribution Protocol (LDP) or SPF IP path.
  - No LSP is established for the master tunnel and regular traffic engineering attributes (bandwidth, path option, fast reroute) are irrelevant on a master tunnel. TE attributes (bandwidth, bandwidth pool, preemption, priorities, path options, and so on) are configured completely independently for each tunnel.
  - CBTS does not allow load-balancing of a given EXP value in multiple tunnels. If two or more tunnels are configured to carry a given experimental (EXP) value, CBTS picks one of these tunnels to carry this EXP value.
  - CBTS supports aggregate control of bumping (that is, it is possible to define default tunnels to be used if other tunnels go down. However, CBTS does not allow control of bumping if the default tunnel goes down. CBTS does not support finer-grain control of bumping. For example, if the voice tunnel goes down, redirect voice to T2, but if video goes down, redirect to T3.
  - The operation of CBTS is not supported with Any Transport over MPLS (AToM), MPLS TE Automesh, or label-controlled (LC) ATM.

Creating Multiple MPLS Member TE or DS-TE Tunnels with the Same Headend and the Same Tailend

Perform the following task to create multiple MPLS member TE or DS-TE tunnels with the same headend and same tailend and to configure EXP values to be carried by each of these tunnels. The procedure begins in global configuration mode.

SUMMARY STEPS

1. **interface tunnel** *number*
2. **ip unnumbered** *type number*
3. **tunnel destination** {*hostname* | *ip-address*}
4. **tunnel mode mpls traffic-eng**
5. **tunnel mpls traffic-eng bandwidth** [**sub-pool** | **global**] *bandwidth*
6. **tunnel mpls traffic-eng exp** [*list-of-exp-values*] [**default**]
7. **exit**

DETAILED STEPS

	Command	Purpose
Step 1	**interface tunnel** *number* **Example:** Router(config)# **interface tunnel 7**	Configures a tunnel interface type and enters interface configuration mode. • *number*—Number of the tunnel interface that you want to create or configure.
Step 2	**ip unnumbered** *type number* **Example:** Router(config-if)# ip unnumbered loopback0	Enables IP processing on an interface without assigning an explicit IP address to the interface. • *type*—Type of another interface on which the router has an assigned IP address. • *number*—Number of another interface on which the router has an assigned IP address. It cannot be another unnumbered interface.
Step 3	**tunnel destination** {*hostname* | *ip-address*} **Example:** Router(config-if)# **tunnel destination 10.5.5.5**	Specifies the destination of the tunnel for this path option. • *hostname*—Name of the host destination. • *ip-address*—IP address of the host destination expressed in four-part, dotted decimal notation.
Step 4	**tunnel mode mpls traffic-eng** **Example:** Router(config-if)# **tunnel mode mpls traffic-eng**	Sets the mode of a tunnel to MPLS for TE.
Step 5	**tunnel mpls traffic-eng bandwidth** [**sub-pool** | **global**] *bandwidth* **Example:** Router(config-if)# **tunnel mpls traffic-eng bandwidth 100**	Configures the bandwidth for the MPLS TE tunnel. If automatic bandwidth is configured for the tunnel, use the **tunnel mpls traffic-eng bandwidth** command to configure the initial tunnel bandwidth, which is adjusted by the auto-bandwidth mechanism. • **sub-pool**—(Optional) Indicates a subpool tunnel. • **global**—(Optional) Indicates a global pool tunnel. Entering this keyword is not necessary, because all tunnels are global pool in the absence of the **sub-pool** keyword. But if users of pre-DiffServ-aware Traffic Engineering (DS-TE) images enter this keyword, it is accepted. • *bandwidth*—Bandwidth, in kilobytes per second, set aside for the MPLS traffic engineering tunnel. Range is between 1 and 4294967295. **Note** You can configure any existing **mpls traffic-eng** command on these TE or DS-TE tunnels.

	Command	Purpose
Step 6	**tunnel mpls traffic-eng exp** [*list-of-exp-values*] [**default**] **Example:** `Router(config-if)# tunnel mpls traffic-eng exp 9`	Specifies an EXP value or values for an MPLS TE tunnel. • *list-of-exp-values*—EXP value or values that are are to be carried by the specified tunnel. Values range from 0 to 7. • **default**—The specified tunnel is to carry all EXP values that are: – Not explicitly allocated to another tunnel – Allocated to a tunnel that is currently down
Step 7	**exit** **Example:** `Router(config-if)#`	Exits to global configuration mode.
Repeat Step 1 through Step 7 on the same headend router to create additional tunnels from this headend to the same tailend.		

Creating a Master Tunnel, Attaching Member Tunnels, and Making the Master Tunnel Visible

Perform the followings task to create a master tunnel, attach member tunnels to it, and make the master tunnel visible for routing. The procedure begins in global configuration mode.

Summary Steps

1. **interface tunnel** *number*
2. **ip unnumbered** *type number*
3. **tunnel destination** {*hostname* | *ip-address*}
4. **tunnel mode mpls traffic-eng exp-bundle master**
5. **tunnel mode mpls traffic-eng exp-bundle member** *tunnel-id*
6. **tunnel mpls traffic-eng autoroute announce**
7. **tunnel mpls traffic-eng autoroute metric** {**absolute** | **relative**} *value*

DETAILED STEPS

	Command	Purpose
Step 1	**interface tunnel** *number* **Example:** `Router(config)# interface tunnel 7`	Configures a tunnel interface type and enters interface configuration mode. • *number*—Number of the tunnel interface that you want to create or configure.
Step 2	**ip unnumbered** *type number* **Example:** `Router(config-if)# ip unnumbered loopback0`	Enables IP processing on an interface without assigning an explicit IP address to the interface. • *type*—Type of another interface on which the router has an assigned IP address. • *number*—Number of another interface on which the router has an assigned IP address. It cannot be another unnumbered interface.

Step 3	**tunnel destination** {*hostname* | *ip-address*}
	Example:
	`Router(config-if)# tunnel destination 10.5.5.5`
		Specifies the destination of the tunnel for this path option.
		• *hostname*—Name of the host destination.
		• *ip-address*—IP address of the host destination expressed in four-part, dotted decimal notation.

Step 4	**tunnel mode mpls traffic-eng exp-bundle master** **Example:** `Router(config-if)# tunnel mode mpls traffic-eng exp-bundle master`	Specifies this is the master tunnel for the CBTS configuration.
Step 5	**tunnel mode mpls traffic-eng exp-bundle member** *tunnel-id* **Example:** `Router(config-if)# tunnel mode mpls traffic-eng exp-bundle member Tunnel20000`	Attaches a member tunnel to the master tunnel. • *tunnel-id*—Number of the tunnel interface to be attached to the master tunnel. Repeat this command for each member tunnel.

	Command	Purpose
Step 6	**tunnel mpls traffic-eng autoroute announce** **Example:** Router(config-if)# **tunnel mpls traffic-eng autoroute announce**	Specifies that the Interior Gateway Protocol (IGP) should use the tunnel (if the tunnel is up) in its enhanced shortest path first (SPF) calculation.
Step 7	**tunnel mpls traffic-eng autoroute metric** {**absolute** | **relative**} *value* **Example:** Router(config-if)# **tunnel mpls traffic-eng autoroute metric relative -1**	(Optional) Specifies the MPLS TE tunnel metric that the IGP-enhanced SPF calculation uses. • **absolute**—Indicates the absolute metric mode; you can enter a positive metric value. • **relative**—Indicates the relative metric mode; you can enter a positive, negative, or zero value. • *value*—Metric that the IGP enhanced SPF calculation uses. The relative value can be from -10 to 10. **Note** Even though the value for a relative metric can be from -10 to +10, configuring a tunnel metric with a negative value is considered a misconfiguration. If the metric to the tunnel tailend appears to be 4 from the routing table, then the cost to the tunnel tailend router is actually 3 because 1 is added to the cost for getting to the loopback address. In this instance, the lowest value that you can configure for the relative metric is -3.

Note Alternatively, static routing could be used instead of autoroute to make the TE or DS-TE tunnels visible for routing.

EXAMPLE

The following example shows how to configure Multiprotocol Label Switching (MPLS) Traffic Engineering (TE) Class-Based Tunnel Selection (CBTS). Tunnel1, Tunnel2, and Tunnel3 are member tunnels, and Tunnel4 is the master tunnel.

```
Router(config)# interface Tunnel1
Router(config-if)# ip unnumbered loopback0
Router(config-if)# interface destination 24.1.1.1
Router(config-if)# tunnel mode mpls traffic-eng
Router(config-if)# tunnel mpls traffic-eng bandwidth sub-pool 30000
Router(config-if)# tunnel mpls traffic-eng exp 5

Router(config)# interface Tunnel2
Router(config-if)# ip unnumbered loopback0
Router(config-if)# interface destination 24.1.1.1
Router(config-if)# tunnel mode mpls traffic-eng
Router(config-if)# tunnel mpls traffic-eng bandwidth 50000
Router(config-if)# tunnel mpls traffic-eng exp 3 4

Router(config)# interface Tunnel3
Router(config-if)# ip unnumbered loopback0
Router(config-if)# interface destination 24.1.1.1
Router(config-if)# tunnel mode mpls traffic-eng
Router(config-if)# tunnel mpls traffic-eng bandwidth 10000
Router(config-if)# tunnel mpls traffic-eng exp default

Router(config)# interface Tunnel4
Router(config-if)# interface destination 24.1.1.1
Router(config-if)# tunnel mpls traffic-eng exp-bundle master
Router(config-if)# tunnel mpls traffic-eng exp-bundle member Tunnel1
Router(config-if)# tunnel mpls traffic-eng exp-bundle member Tunnel2
Router(config-if)# tunnel mpls traffic-eng exp-bundle member Tunnel3
Router(config-if)# tunnel mpls traffic-eng autoroute enable
```

Verifying That the MPLS TE or DS-TE Tunnels Are Operating and Announced to the IGP

The following **show** commands can be used to verify that the MPLS TE or DS-TE tunnels are operating and announced to the IGP. The commands are all entered in privileged EXEC configuration mode.

Command	Purpose
show mpls traffic-eng topology {*A.B.*C.*D* | **igp-id** {**isis** *nsap-address* | **ospf** *A.B.*C.*D*} [**brief**]}	Shows the MPLS traffic engineering global topology as currently known at this node. • *A.B*.C.*D*—Specifies the node by the IP address (router identifier to interface address). • **igp-id**—Specifies the node by IGP router identifier. • **isis** *nsap-address*—Specifies the node by router identification (*nsap-address*) if you are using Integrated Intermediate System-to-Intermediate System (IS-IS). • **ospf** *A.B*.C.*D*—Specifies the node by router identifier if you are using Open Shortest Path First (OSPF). • **brief**—Provides a less-detailed version of the topology.
show mpls traffic-eng exp	Displays EXP mapping.
show ip cef [*type number*] [**detail**]	Displays entries in the forwarding information base (FIB) or displays a summary of the FIB. • *type number* —Identifies the interface type and number for which to display FIB entries. • **detail**—Displays detailed FIB entry information.

Command	Purpose
show mpls forwarding-table [*network* {*mask* | *length*} [**detail**]]	Displays the contents of the MPLS label forwarding information base (LFIB). • *network*—Identifies the destination network number. • *mask*—Identifies the network mask to be used with the specified network. • *length*—Identifies the number of bits in the destination mask. • **detail**—Displays information in long form (includes length of encapsulation, length of MAC string, maximum transmission unit [MTU], and all labels).
show mpls traffic-eng autoroute	Displays tunnels that are announced to the Interior Gateway Protocol (IGP).

The **show mpls traffic-eng topology** command output displays the MPLS TE global topology:

```
Router# show mpls traffic-eng topology 10.0.0.1

IGP Id: 10.0.0.1, MPLS TE Id:10.0.0.1 Router Node  (ospf 10  area 0) id 1
link[0]: Broadcast, DR: 180.0.1.2, nbr_node_id:6, gen:18
      frag_id 0, Intf Address:180.0.1.1
      TE metric:1, IGP metric:1, attribute_flags:0x0
      SRLGs: None
      physical_bw: 100000 (kbps), max_reservable_bw_global: 1000 (kbps)
      max_reservable_bw_sub: 0 (kbps)
                             Global Pool       Sub Pool
           Total Allocated   Reservable        Reservable
           BW (kbps)         BW (kbps)         BW (kbps)
               ---------------   -----------       ----------
        bw[0]:            0          1000                0
        bw[1]:            0          1000                0
        bw[2]:            0          1000                0
        bw[3]:            0          1000                0
        bw[4]:            0          1000                0
        bw[5]:            0          1000                0
        bw[6]:            0          1000                0
        bw[7]:          100           900                0

link[1]: Broadcast, DR: 180.0.2.2, nbr_node_id:7, gen:19
      frag_id 1, Intf Address:180.0.2.1
      TE metric:1, IGP metric:1, attribute_flags:0x0
      SRLGs: None
      physical_bw: 100000 (kbps), max_reservable_bw_global: 1000 (kbps)
          max_reservable_bw_sub: 0 (kbps)
                                 Global Pool       Sub Pool
               Total Allocated   Reservable        Reservable
               BW (kbps)         BW (kbps)         BW (kbps)
           ---------------   -----------       ----------
    bw[0]:            0          1000                0
    bw[1]:            0          1000                0
    bw[2]:            0          1000                0
    bw[3]:            0          1000                0
    bw[4]:            0          1000                0
    bw[5]:            0          1000                0
    bw[6]:            0          1000                0
    bw[7]:            0          1000                0
```

The **show mpls traffic-eng exp** command output displays EXP mapping information about a tunnel:

```
Router# show mpls traffic-eng exp

Destination: 10.0.0.9
    Master:Tunnel10Status: IP

    Members: StatusConf EXPActual EXP
    Tunnel1UP/ACTIVE55
    Tunnel2UP/ACTIVEdefault0 1 2 3 4 6 7
    Tunnel3UP/INACTIVE(T)2
    Tunnel4DOWN3
    Tunnel5UP/ACTIVE(NE)

(T)=Tailend is different to master
(NE)=There is no exp value configured on this tunnel.
```

The **show ip cef detail** command output displays detailed FIB entry information for a tunnel:

```
Router# show ip cef tunnel1 detail

IP CEF with switching (Table Version 46), flags=0x0
31 routes, 0 reresolve, 0 unresolved (0 old, 0 new), peak 2
2 instant recursive resolutions, 0 used background process
8 load sharing elements, 8 references
6 in-place/0 aborted modifications
34696 bytes allocated to the FIB table data structures
universal per-destination load sharing algorithm, id 9EDD49E1
1(0) CEF resets
    Resolution Timer: Exponential (currently 1s, peak 1s)
    Tree summary:
    8-8-8-8 stride pattern
    short mask protection disabled
31 leaves, 23 nodes using 26428 bytes
Table epoch: 0 (31 entries at this epoch)
Adjacency Table has 13 adjacencies
10.0.0.9/32, version 45, epoch 0, per-destination sharing
0 packets, 0 bytes
tag information set, all rewrites inherited
local tag: tunnel head
via 0.0.0.0, Tunnel1, 0 dependencies
traffic share 1
next hop 0.0.0.0, Tunnel1
valid adjacency
tag rewrite with Tu1, point2point, tags imposed {12304}
0 packets, 0 bytes switched through the prefix
tmstats: external 0 packets, 0 bytes
internal 0 packets, 0 bytes
```

The **show mpls forwarding-table detail** command output displays detailed information from the MPLS LFIB:

```
    Router# show mpls forwarding 10.0.0.9 detail

    Local  Outgoing    Prefix            Bytes tag  Outgoing   Next Hop
    tag    tag or VC   or Tunnel Id      switched   interface
Tun hd Untagged    10.0.0.9/32       0          Tu1        point2point
    MAC/Encaps=14/18, MRU=1500, Tag Stack{12304}, via Fa6/0
    00027D884000000ED70178A88847 03010000
    No output feature configured
Per-exp selection: 1
Untagged    10.0.0.9/32       0          Tu2        point2point
    MAC/Encaps=14/18, MRU=1500, Tag Stack{12305}, via Fa6/1
    00027D884001000ED70178A98847 03011000
    No output feature configured
Per-exp selection: 2  3
Untagged    10.0.0.9/32       0          Tu3        point2point
    MAC/Encaps=14/18, MRU=1500, Tag Stack{12306}, via Fa6/1
    00027D884001000ED70178A98847 03012000
    No output feature configured
Per-exp selection: 4  5
Untagged    10.0.0.9/32       0          Tu4        point2point
    MAC/Encaps=14/18, MRU=1500, Tag Stack{12307}, via Fa6/1
    00027D884001000ED70178A98847 03013000
    No output feature configured
Per-exp selection: 0  6  7
```

The **show mpls traffic-eng autoroute** command output displays tunnels that are announced to the Interior Gateway Protocol (IGP).

```
Router# show mpls traffic-eng autoroute

MPLS TE autorouting enabled
destination 10.0.0.9, area ospf 10  area 0, has 4 tunnels
Tunnel1     (load balancing metric 20000000, nexthop 10.0.0.9)
(flags: Announce)
Tunnel2     (load balancing metric 20000000, nexthop 10.0.0.9)
(flags: Announce)
Tunnel3     (load balancing metric 20000000, nexthop 10.0.0.9)
(flags: Announce)
Tunnel4     (load balancing metric 20000000, nexthop 10.0.0.9)
(flags: Announce)
```

Configuring Virtual Private LAN Service

Virtual Private LAN Service (VPLS) enables geographically separate LAN segments to be interconnected as a single bridged domain over a packet switched network, such as IP, MPLS, or a hybrid of both.

VPLS solves the network reconfiguration problems at the customer equipment (CE) that is associated with Layer 2 Virtual Private Network (L2VPN) implementations. The current Cisco IOS software L2VPN implementation builds a point-to-point connection to interconnect the two attachment VCs of two peering customer sites. To communicate directly among all sites of an L2VPN network, a distinct emulated VC needs to be created between each pair of peering attachment VCs.

For example, when two sites of the same L2VPN network are connected to the same PE, you must establish two separate emulated VCs towards a given remote site, instead of sharing a common emulated VC between these two sites. For an L2VPN customer who uses the service provider backbone to interconnect its LAN segments, the current implementation effectively turns its multiaccess broadcast network into a fully meshed point-to-point network, which requires extensive reconfiguration on the existing CE devices.

VPLS is a multipoint L2VPN architecture that connects two or more customer devices using EoMPLS bridging techniques. VPLS with EoMPLS uses an MPLS-based provider core, where the PE routers have to cooperate to forward customer Ethernet traffic for a given VPLS instance in the core.

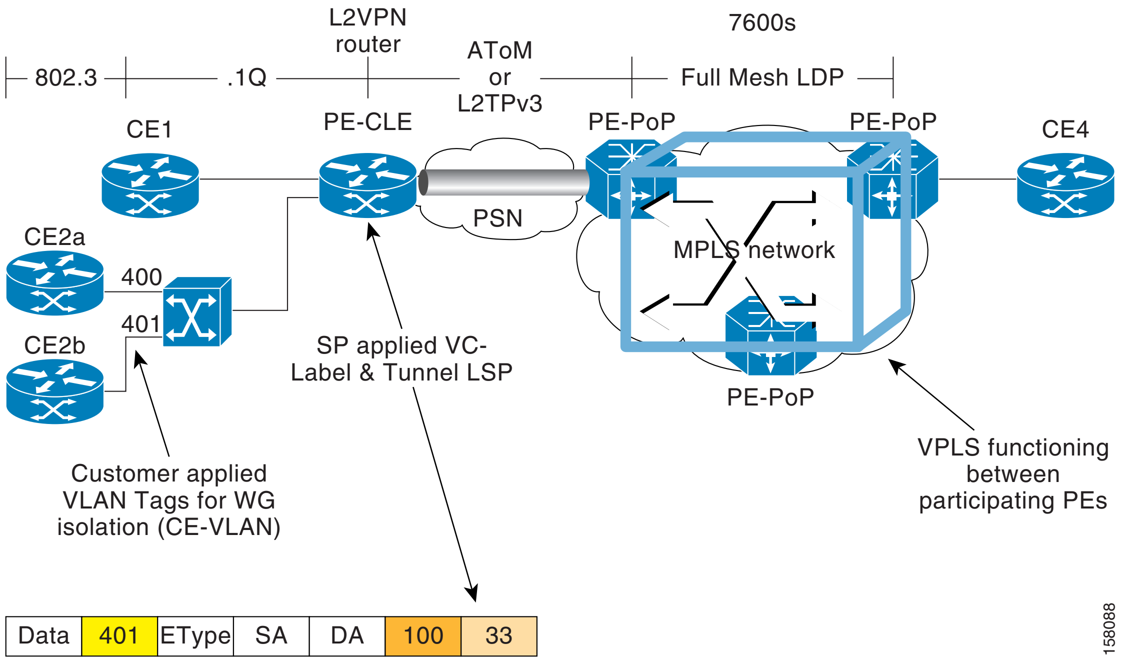
VPLS uses the provider core to join multiple attachment circuits together to simulate a virtual bridge that connects the multiple attachment circuits together. From a customer point of view, there is no topology for VPLS. All of the CE devices appear to connect to a logical bridge emulated by the provider core.

Hierarchical Virtual Private LAN Service (H-VPLS) with MPLS to the Edge

In a flat or nonhierarchical VPLS configuration, a full mesh of pseudowires (PWs) is needed between all PE nodes. A *pseudowire* defines a VLAN and its corresponding pseudoport.

Hierarchical Virtual Private LAN Service (H-VPLS) reduces both signaling and replication overhead by using a combination of full-mesh and hub-and-spoke configurations. Hub-and-spoke configurations operate with split horizon to allow packets to be switched between PWs, which effectively reduce the number of PWs between PEs.

Figure 2-10 H-VPLS with MPLS to the Edge Network



In the H-VPLS with MPLS to the edge architecture, Ethernet Access Islands (EAIs) work in combination with a VPLS core network, with MPLS as the underlying transport mechanism. EAIs operate like standard Ethernet networks. In Figure 2-10, devices CE1, CE2a and CE2b reside in an EAI. Traffic from any CE devices within the EAI is switched locally within the EAI by the user-facing provider edge (UPE) device along the computed spanning-tree path. Each UPE device is connected to one or more network-facing provider edge (NPE) devices using PWs. The traffic local to the UPE is not forwarded to any network-facing provider edge (NPE) devices.

VPLS Configuration Guidelines

When configuring VPLS on a Cisco 7600 Series Ethernet Services 20G line card, consider the following guidelines:

- The Cisco 7600 Series Ethernet Services 20G line card supports up to 4000 VPLS domains per Cisco 7600 series router.
- The Cisco 7600 Series Ethernet Services 20G line card supports up to 60 VPLS peers per domain per Cisco 7600 series router.

Note From Cisco IOS release12.2(33) SRD onwards, the Cisco 7600 Series Ethernet Services 20G line card supports up to 110 VPLS peers per domain per Cisco 7600 series router.

- The Cisco 7600 Series Ethernet Services 20G line card supports up to 30,000 pseudowires, used in any combination of domains and peers up to the 4000-domain or 60-peer maximums. For example, support of up to 4000 domains with 7 peers, or up to 60 peers in 500 domains. However, although 30, 000 vcs are supported, these are not the recommended system operating limits.
- When configuring VPLS on a Cisco 7600 Series Ethernet Services 20G line card, consider the following guidelines:
  - H-VPLS with QinQ edge—Requires a Cisco 7600 Series Ethernet Services 20G line card in the uplink, and any LAN port or Cisco 7600 Series Ethernet Services 20G line card on the downlink.
- H-VPLS with MPLS edge requires either an optical service module, Cisco 7600 SIP-600, Cisco 7600 SIP-400, or Cisco 7600 Series Ethernet Services 20G line cards in both the downlink (facing UPE) and uplink (MPLS core).
- The Cisco 7600 Series Ethernet Services 20G line cards provide Transparent LAN Services (TLS) and Ethernet Virtual Connection Services (EVCS).
- The Cisco 7600 Series Ethernet Services 20G line cards support the following VPLS features:
  - H-VPLS with MPLS edge
  - H-VPLS with QinQ edge
  - VPLS with point-to-multipoint EoMPLS and fully-meshed PE configuration
- For information about configuring VPLS on the Cisco 7600 Series Ethernet Services 20G line cards, consider the guidelines in this document:

  http://www.cisco.com/en/US/docs/interfaces_modules/shared_port_adapters/configuration/7600series/76cfgeth.html
- VPLS is supported only with intelligent cards as core facing cards. If both intelligent and non-intelligent cards are used as core facing cards, then route change events can cause VPLS VC to go over non-intelligent cards, and under such conditions, VPLS may not work as expected.

Configuring SVI-Based IP/Routed Interworking

Interworking interconnects two heterogeneous Attachment Circuits (ACs).

The types of interworking functions used depends on the following criteria:

- AC types used.
- Data type relayed.
- Level of functions supported by the router.

The interworking functions currently supported are:

- Bridged Interworking —Used with the Layer 2 (L2) packets regardless of the Layer 3 (L3) contents. The Internet Service Provider (ISP) do not participate in the routing.
- Ethernet Interworking— Used with a Ethernet (Port) over Multiprotocol Label Switching (MPLS) pseudowire for bridged interworking.

- Routed Interworking—Used to relay L3 packets. There are different routed interworking for each protocol type. Internet Protocol (IP) Interworking or Switched Virtual Interface (SVI)/VLAN-based IP/Routed interworking uses IP over MPLS (Multiprotocol Label Switching) pseudowires to relay data.

As Ethernet-to-Ethernet or PPP-to-PPP supports Layer 2 transport over MPLS and IP, L2VPN Interworking augments this functionality and connects disparate attachment circuits. Interworking function translates data between the different Layer 2 encapsulations. ES 20 line card supports IP routed interworking, also known as IP interworking.

Note If a Layer2 pseudowire is configured on an interface VLAN instead of the phisycal interface and the pseudowire traffic on Layer2 reaches the destination (core facing) through an ES20 line card, the PIM and IGMP traffic crossing the pseudowire is switched to the Route Processor (software switched) resulting in high CPU utilization. To enable hardware switching for the PIM and IGMP traffic, disable IGMP snooping.

In a Routed mode AToM (Any Transport over MPLS) interworking:

- Address Resolution Protocol (ARP) is terminated on the ethernet PE.
- When the payload is sent to core, there is no ethernet header and cannot relay to the PE using Frame Relay or ATM.
- ATM or FR side does not handle L2 encapsulation.

When you execute the interworking {**ethernet** | **ip**} command in pseudowire- class configuration mode:

- The attachment circuits are locally terminated.
- The ethernet keyword removes the Ethernet frames from the attachment circuit, and relays them over the pseudowire.
- Ethernet end-to-end transmission is assumed.
- Nonethernet attachment circuit frames are dropped.
- The VLAN tag is removed, and the untagged Ethernet frame is retained.

Restrictions and Usage Guidelines

- ES20 supports only SVI- based interworking.
- ES20 does not support ATM/FR/PPP.

SUMMARY STEPS TO CONFIGURE Ethernet VLAN to ATM AAL5

1. **enable**
2. **configure terminal**
3. **pseudowire-class** *pw-class-name*
4. encapsulation mpls
5. interworking ip
6. **vlan** *vlan-id*
7. **interface vlan** *vlan-id*
8. no ip address

9. **xconnect** *destination-id vc-id* **pw-class** *pw-class-name*
10. exit

DETAILED STEPS TO CONFIGURE Ethernet VLAN to ATM AAL5

	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	`pseudowire-class` *`pw-class-name`*
	Example:
	`Router (config)# pseudowire-class abc`
		Defines the pseudowire class instance in the pseudowire class.

Step 4	`encapsulation mpls`
	Example:
	`Router# (config-pw-class)# encapsulation mpls`
		Defines the encapsulation in the pseudowire class.

Step 5	`interworking ip`
	Example:
	`Router# (config-pw-class)# interworking ip`
		Defines the interworking type in the pseudowire class.

Step 6	`vlan` *`vlan-id`*
	Example:
	`Router# vlan 119`
		Sets a VLAN instance on the router mode.

Step 7	`interface vlan` *`vlan-id`*
	Example:
	`Router (config)# interface vlan 119`
		Creates a VLAN interface in the configuration mode.

Step 8	`no shut`
	Example:
	`Router (config-if)# no shut`
		Releases the VLAN interface.

	Command or Action	Purpose

Step 9	`xconnect` *`destination-id vc-id`* `pw-class` *`pw-class-name`*
	Example:
	`Router (config-if)# xconnect 12.205.2.2 119 pw-class abc`
		Creates the pseudowire VC to the remote end.

Step 10	`exit`	Exits the configuration mode.

Example

Use the following commands to configure Ethernet VLAN to ATM AAL5 on an SVI interface:

```
pseudowire-class pwc-mpls-ip
encapsulation mpls
interworking ip
!
vlan 119
!
interface Vlan119
 no ip address
 xconnect 12.205.2.2 119 pw-class pwc-mpls-ip
```

Verification

Use the **show mpls l2 vc or show mpls l2 vc** *<vc-id>* **detail** command to verify the status of the VC or the pseudowire.

Resetting a Cisco 7600 Series Ethernet Services 20G Line Card

To reset an ES20 line card, use the following command in privileged EXEC configuration mode:

Command	Purpose
Router# **hw-module module** *slot* **reset**	Turns power off and on to the ES20 line card in the specified slot, where: • *slot*—Specifies the chassis slot number where the ES20 line card is installed.

SFP-GE-T Support

The SFP-GE-T on the Cisco 7600-ES20-GE line card supports speeds of 10 Mbps, 100 Mbps, and 1000 Mbps. Speed is not autonegotiated; you must configure it using the speed command. Only full-duplex mode is supported.

Note Auto negotiation of duplex mode is not supported, hence, manually configure the remote port for full-duplex mode.

You can configure each Ethernet interface independently using any combination of 10 Mbps, 100 Mbps, or 1000 Mbps.

To set the interface speed, use the following command in the interface configuration mode.

Command	Purpose
Router (config-if)# **speed** **Example:** Router(config-if)# **speed 10**	Configures the interface speed. Accepted values are: • 10 for 10 Mbps operation • 100 for 100 Mbps operation • 1000 for 1000 Mbps operation

CHAPTER 3

Configuring QoS on the Cisco 7600 Series Ethernet Services 20G Line Card

This chapter provides information about configuring Quality of Service (QoS) on the Cisco 7600 Series Ethernet Services 20G (ES20) line card on the Cisco 7600 series router.

Before referring to any other QoS documentation for the platform or in the Cisco IOS software, use this chapter to determine ES20 line card-specific QoS feature support and configuration guidelines.

For additional details about QoS concepts and features in Cisco IOS Release 12.2, you can refer to the *Cisco IOS Quality of Service Solutions Configuration Guide, Release 12.2SR*, at:

http://www.cisco.com/en/US/docs/ios/qos/configuration/guide/12_2sr/qos_12_2sr_book.html

This chapter includes the following sections:

- QoS Functions in the Cisco 7600 Series Ethernet Services 20G Line Card, page 3-2
- Configuring QoS Features Using MQC, page 3-4
  - Dual Rate Three Color (2R3C) Ingress Policer on Service Instances, page 3-11
- EVCS QoS Support, page 3-5
- Configuring Qos Over an EVC Group, page 3-6
- Configuring Policing, page 3-21
- Configuring Marking, page 3-29
- Configuring Shaping, page 3-34
  - Configuring Shaping on the ES20 Main Interface, page 3-37
- Configuring PFC QoS on a Cisco 7600 Series Ethernet Services 20G Line Card, page 3-47
- Configuring Hierarchical QoS, page 3-48
  - Configuring Bandwidth Remaining Ratio (BRR) - Utilizing Unused Bandwidth Support, page 3-57
- EVC Configuration Examples, page 3-54
- Troubleshooting QoS Features in a ES20 Line Card, page 3-65

QoS Functions in the Cisco 7600 Series Ethernet Services 20G Line Card

Note Main interface refers to the maininterface and switchport unless specified otherwise.
If the supported feature is not mentioned, then the feature is not supported in the particular interface.

The following sections describe ingress and egress QoS functions.

Ingress QoS Functions in a Cisco 7600 Series Ethernet Services 20G Line Card

The following paragraphs describe ingress QoS support on the ES20 line card.

Ingress Trust

Trust is a port assignment instructing the port to trust (leave) existing priorities as they are on incoming frames or to rewrite the priorities back to zero.

A packet can arrive at an interface with a priority value already present in the packets header. The router needs to determine if the priority setting was set by a valid application or network device according to pre defined rules or if it was set by a user hoping to get better service.

The router has to decide whether to honor the priority value or change it to another value. How the router makes this determination is by using the port “trust” setting.

The ES20 line card always trusts Differentiated Services Code Point (DSCP) by default. Maininterfaces and subinterfaces use the ingress trust functionality.

Ingress Queue Scheduling

The ES20 line card does not support ingress queue scheduling.

Ingress Classification

After a frame is queued, the frame is passed on for classification. Classification entails using a traffic descriptor to categorize a packet within a specific group to define that packet and make it accessible for QoS handling on the network. Using packet classification, you can partition network traffic into multiple priority levels or classes of service.

Traffic is classified to determine whether it should be:

- Marked for further processing.
- Policed to rate limit specific traffic types.

The ES20 line card supports ingress classification. For information on configuring classification, see “Configuring Classification” section on page 3-16.

Ingress Policing

Policing provides a means to limit the amount of bandwidth that traffic traveling through a given port, or a collection of ports in a VLAN, can use. Policing works by defining an amount of data that the router is willing to send or receive in kilobytes per second.

When policing is configured, it limits the flow of data through the router by dropping or marking down the QoS value traffic that is out-of-profiles. Policing allows the router to limit the rate of specific types to a level lower than what they might get otherwise based only the interface bandwidth.

The ES20 line card supports ingress policing. For information on configuring policing, see "Configuring Policing" section on page 3-21.

Ingress Marking

After it has been classified, traffic can be marked. Marking is a way to selectively modify the classification bits in a packet to identify traffic within the network. Other interfaces can then match traffic based on the markings.

The ES20 line card supports ingress marking. For information on configuring marking, see "Configuring Marking" section on page 3-29.

Ingress Shaping

The ES20 line card does not support ingress shaping.

Egress QoS Functions in a Cisco 7600 Series Ethernet Services 20G Line Card

The following sections describe QoS functions on the ES20 line card egress ports.

Egress Classification

Classification entails using a traffic descriptor to categorize a packet within a specific group to define that packet and make it accessible for QoS handling on the network. Using packet classification, you can partition network traffic into multiple priority levels or classes of service.

Traffic is classified to determine whether it should be:

- Marked for further processing.
- Queued to rate limit specific traffic types.

The ES20 line card supports egress classification. For information on configuring classification, see "Configuring Classification" section on page 3-16.

Egress Policing

The ES20 line card supports egress port policing only when paired with the **priority** command.

Egress Marking

After traffic has been classified, the router can mark it. You use marking to selectively modify the classification bits in the packet to differentiate packets based on the designated markings.

The ES20 line card supports egress port marking on service instances, maininterfaces, and subinterfaces. For information on configuring marking, see "Configuring Marking" section on page 3-29.

Egress Shaping

Traffic shaping allows you to control the traffic going out an interface in order to match its flow to the speed of the remote target interface and to ensure that the traffic conforms to policies contracted for it. You can use shaping to meet downstream requirements, thereby eliminating bottlenecks in topologies with data-rate mismatches.

The ES20 line card supports shaping on egress port, maininterfaces, subinterfaces, and service instances. For information on configuring shaping, see "Configuring Shaping" section on page 3-34.

Egress Queue Scheduling

The egress line card uses congestion avoidance to help prevent congestion and keep its buffers from overflowing.

The ES20 line card supports Class-based Weighted Fair Queuing (CBWFQ), Low Latency Queueing (LLQ), and Weighted Random Early Detection (WRED).

Restrictions

Follow these restrctions and guidelines when you configure QoS in a ES20 line card:

- A policy-map output on a route processor is linecard specific because all the policy-map statistics are exported from the linecard to the route processor. In a ES20 linecard egress scenario, an active Ternary Content Addressable Memory (TCAM) entry requires a queueing ASIC to relay the packet out of an egress interface. This is because the queue id is mapped to an egress port number programmed in a queuing ASIC through which packets are relayed out. However, a packet is dropped if a valid queue id is not allocated, and the TCAM entry is active. Hence classification happens only for classes with a valid queuing action in an egress direction.
- You can configure a minimum bandwidth of 128kbps when using the **bandwidth** command.

Configuring QoS Features Using MQC

The Modular QoS CLI (MQC) is a CLI structure that allows users to create traffic policies and attach these policies to interfaces. A traffic policy contains a traffic class and one or more QoS features. A traffic class is used to select traffic, while the QoS features in the traffic policy determine how to treat the classified traffic.

To configure QoS features using the Modular QoS CLI on the ES20 line card, complete the following basic steps:

Step 1 Define a traffic class using the **class-map** command.

Step 2 Create a traffic policy by associating the traffic class with one or more QoS features (using the **policy-map** command).

Step 3 Attach the traffic policy to the interface using the **service-policy** command.

For a complete discussion about MQC, refer to the “Modular Quality of Service Command-Line Interface” section of the *Cisco IOS Quality of Service Solutions Configuration Guide, Release 12.2* publication at:

http://www.cisco.com/en/US/docs/ios/qos/configuration/guide/12_2sr/qos_12_2sr_book.html

EVCS QoS Support

Ethernet Virtual Connection Services (EVCS) uses the concepts of service instances and EVCs (Ethernet virtual circuits). A service instance is the instantiation of an EVC on a given port on a given router. An EVC is an end-to-end representation of a single instance of a Layer 2 service being offered by a provider to a customer. It embodies the different parameters on which the service is being offered.

Mapping Between Bay and Ports

The following table maps the bay and port information in a ES20 line card:

Table 3-1 Mapping Between Bay and Ports

Specifications	Port/Bay Information
ESM20 1 GIG Variant with 2 Xchip ASICs where lower number of ports are assigned to Bay 0 and higher number ports are mapped to Bay 1.	Ports 0 to 9 mapped to Bay 0 Ports 10 to 19 mapped to Bay 1
ESM20 10 GIG Variant	10 G Port 0 mapped to Bay 0 10 G Port 1 mapped to Bay 1

Restrictions and Usage Guidelines

When configuring QoS with EVCs on the ES20 line card, follow these restrictions and usage guidelines:

- Service instances use MQC.
- QoS supports 16 000 service instances per line card and 8000 per bay.
- HQoS supports 990 policy-maps per bay.
- Configuring EVcs or subinterfaces is not permissible with QoS on the main interface.
- Ingress shaping is not supported.
- For egress QoS, both hierarchical and flat policy-maps are supported.
- Before creating a service instance or a sub interface, remove any policy-maps on the maininterface.
- Only classes defined with **match vlan** and **vlan-inner** and **class-default** can exist in a parent policy.
- As service instance configurations change, you must remove and reapply the policy because the policy-map configuration may no longer be valid. However, changes on the main interface related to QoS are not cascaded to EVC QoS.
- Because the policy-map does not work to its optimum capacity due to 1% of the port bandwidth being reserved for control packets, ensure that the policy-map is configured to use only 99% of the port bandwidth.
- EVCs with default encapsulation support policy-maps similar to other supported EVCs.

- Layer 3 classification, Ingress and Egress Marking combinations are supported for the ES20 line card on the Cisco 7600 series router. For more information on the supported combinations, see "Layer 3 Ingress Marking Scenarios On a Service Instance".

EVC QoS support is as follows:

- For EVC QoS, classification is based on the following filters, which can be combined:
  - Inner VLAN tag
  - Outer VLAN tag
  - CoS
  - CoS Inner
  - Precedence
  - Dscp
  - Mpls exp (for egress classification only)
- For EVC QoS, marking is based on:
  - Copy or rewrite inner CoS
  - Copy or rewrite outer CoS
  - Copy inner to outer CoS or vice versa
  - Copy or rewrite on EXP
  - Rewrite precedence
  - Rewrite dscp
- For EVC QoS, actions supported include:
  - Marking, bandwidth, WRED aggregate, shape average, queue-limit, priority with police

For EVC QoS configuration examples, see "EVC Configuration Examples" section on page 3-54.

Restrictions and User Guidelines for EVC Port Channel

When configuring QoS on a EVC port channel, follow these restrictions and user guidelines:

- In a EVC port channel, the maximum number of restricted queues is 8000 per bay. Ensure that you do not exceed the 8000 limitation.
- Maximum number of member links that can be configured on a line card is 8.
- On a port channel, each EVC is replicated by the number of member links. Ensure that the total number of EVCs after replication per bay does not exceed 8000.
- EVC port channels do not support percentage values of bandwidth and policing.
- Egress IP Precedence and DSCP marking are supported for EVCs or for EVCs with port channels.

Configuring Qos Over an EVC Group

A Service group is a logical interface that helps you to group EVCs, and apply features on the aggregate logical entity. An EVC group helps you to configure a single QoS policy-map on different EVCs. You can globally configure service groups, and associate the group IDs with each members to add new members. For example, you can configure the *group ID* within each EVC to associate EVCs with a particular service group.

Each EVC belongs to only one service group at a time and the group must exist before any members (EVCs) can join the group. In addition to the policy on the service group of which it is a member, you can also apply a policy on each EVC. A policy map is rejected if you simultaneously apply the policy-map on a group and an EVC in the same direction.

In 12.2(33)SRE release, QoS is supported on the service groups, and you can add EVCs or EVCs over PC as members of a service-group. You can use this feature to apply a QoS policy (both in ingress and egress) within service groups having EVCs or EVCs over PortChannel as its members.

Restrictions and Usage Guidelines

When configuring Qos over a EVC group, follow these restrictions and usage guidelines:

- This feature is supported only on service instances.
- Each service instance can belong to only one service group.
- In EVCs, all members of a service group must reside within the same interface.
- In EVCs within a port-channel, all members of a service group must reside within the port-channel.
- You cannot individually assign a member of a service group to a load-balance link of a port-channel. You should assign the whole group to the link.
- An EVC group cannot have members that are assigned to multiple load-balance links of a port-channel.
- Ensure that you have configured a EVC group before adding a service instance to it.
- Service Group membership is extended only to EVCs (service instances).
- You can apply a Qos Policy on groups or individual EVCs that are part of groups.
- Ingress 1R2C policing is not supported on EVC groups.
- You cannot simultaneously apply QoS on a EVC and its group in the same direction.
- Service group supports hierarchical Qos and flat (class-default) policy-maps.
- All QoS parameters (Shape, Set, Bandwidth, BRR, Police (2r3c) , Set, Wred) within a class, applicable to an EVC, are also supported on a group.
- 2r3c policer is supported in a ingress policy-map on a group.

Table 3-2 lists the scalability values of EVC groups:

Table 3-2 Scalability Values for ES20 line cards

Scalability	Values
Service Groups per bay	2000
Service Groups per line card	4000
Number of EVCs per service group	4000

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **service-group** *ID number*

4. **service-policy** [*input*| *output*]
5. **sh service-group** *ID number*
6. **interface GigabitEthernet** *slot/bay/port*
7. **service instance** *name*
8. **group** *ID number*
9. **sh service-group** *ID number*
10. **end**

DETAILED STEPS

	Command	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode and enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**service-group** *ID number* **Example:** `Router(config)# service-group 1000`	Assigns a service group ID number. The acceptable range is 1-32768.
Step 4	**service-policy** *[input| output]* **Example:** `Router(config-service-group)# service-policy output`	Creates a service policy within the service group and attaches it to the ingress or egress of a service group.
Step 5	**sh service-group** ID number **Example:** `Router# sh service-group 1000`	Displays the service group for which an ID has been assigned or the members belonging to a service group.
Step 6	**interface GigabitEthernet** slot/bay/port **Example:** `Router(config)# int gi 2/0/0`	Identifies the interface to which the service group should be attached.
Step 7	**service instance** name **Example:** `Router(config-if)# service instance 1 eth`	Creates a service instance within the interface.
Step 8	**group** ID number **Example:** `Router(config-if-srv)# group 1000`	Adds the created group to the service instance.

	Command	Purpose
Step 9	**sh service-group** ID number **Example:** `Router# sh service-group 1000`	Displays the service group for which an ID has been assigned or the service instances belonging to a service group.

Step 10	**end**
	Example:
	`Router(config-if-srv)# end`
		Ends the command operations.

Example

The following example shows the creation of a service group and attaching the policy map to the egress:

```
Router(config)# service-g
Router(config)# service-group ?
  <1-32768>  Service Group ID Number
Router(config)# service-group 1000
Router(config-service-group)#ser
Router(config-service-group)# service-policy ?
input   Attach a policy-map to ingress of a service group
output  Attach a policy-map to egress of a service group
Router(config-service-group)# service-policy output <policy-map name>
```

The following example shows the creation of a service instance and adding a service group to the service instance:

```
Router(config)# int gi 2/0/0
Router(config-if)# ser
Router(config-if)# service in
Router(config-if)# service instance 1 eth
Router(config-if-srv)# en
Router(config-if-srv)# encapsulation d
Router(config-if-srv)# encapsulation do
Router(config-if-srv)# encapsulation dot1q 10
Router(config-if-srv)#?
Ethernet EFP configuration commands:
  bridge-domain   Bridge-domain
  default         Set a command to its defaults
  description     Service instance specific description
  encapsulation   Configure ethernet frame match criteria
  errdisable      Configure error disable
  ethernet        Configure ether lmi parameters
  exit            Exit from ETHER EFP configuration mode
  group           Join a service group
  ip              ip
  l2protocol      Configure l2 control protocol processing
  mac             Commands for MAC Address-based features
  no              Negate a command or set its defaults
  rewrite         Configure ethernet rewrite criteria
  service-policy  Attach a policy-map to an EFP
  shutdown        Take the Service Instance out of Service
  snmp            Modify SNMP service instance parameters
  xconnect        Xconnect commands
Router(config-if-srv)# gr
Router(config-if-srv)# group 1000
Router(config-if-srv)#
Router#
```

Execute the **sh run service-group 1** command to display the input and output policymaps applied on a service group:

```
Router#
Router#
Router#sh ser
Router# sh service
*Apr 24 01:17:21.904: %SYS-5-CONFIG_I: Configured from console by console
Router#sh service-g
Router#sh service-group 1000
Service Group 1000:
  Number of members:                        1
  State:                                    Up
  Interface:                                GigabitEthernet2/0/0
    Number of members:                      1
```

Dual Rate Three Color (2R3C) Ingress Policer on Service Instances

Dual Rate Three Color Policing (2R3C), elaborated in RFC 2698, lists the following characteristics of the data packets:

- Metered and colored as green, yellow or red.
- Marked for transmission drop or QoS marking.
- Replenished in token buffer mode.

Based on the new feature implementation in ES20 line cards, the policy meter operates in a color blind mode in flat policy-maps. You can configure dual rates policing using the **conform, exceed,** and **violate** actions in the ingress service instances on an ES20 line card. For more information on Dual rate Three Color policers, refer to the section "QoS: Color-Aware Policer" at the following URL:

http://www.cisco.com/en/US/docs/ios/12_0s/feature/guide/12s_cap.html

You can configure the policer in any classes supported for marking. You can use the **match vlan**, **match vlan inner, match QoS,** or **match CoS inner** filters to configure these classes. For more information on Policing in ES20 line cards, refer the section "Configuring Policing" section on page 3-21.

Restrictions and Usage Guidelines

When configuring 2R3C ingress service policy on a ES 20 line card, follow these restrictions and usage guidelines:

- You cannot configure 2R3C policer with the following characteristics:
  - A single rate two color (1R2C) policer in the same policy-map.
  - Packets marked for transmission drop.
  - Simultaneous execution of the **set** command operations in the same class.
- Maximum of 3000 policers are supported on an ES20 line card (for a total of 6000 per line card).
- If a 1R2C policer is configured, and actions other than **conform, transmit, exceed,** or **drop** are configured, or 1R2C is not as class-default, the microprocessor performs policing.
- If you do not configure the values for **cir** and **pir,** the CLI commands are rejected.
- 2R3C policer works with user-class and class-default values.
- 2R3C policer does not support **set- cos** command within the policer. **set- cos** command are part of policer action such as **set-cos transmit**.

- To set the DBUS COS, you can use **set-cos** action command instead of the **set mpls exp** .
- You cannot configure 2R3C and 1R2C policer within the same policy-map.
- You can configure a minimum policer value of 128 Kbps.
- A hierarchical qos policymap in ingress should comprise of police at parent policy and marking cos in child policy. Configuration of police in both parent and child policy maps is not supported.
- Flat 2R3C policer is supported.
- **conform, exceed,** and **violate** commands are used to mark all the policing actions.
- The following policing actions are supported in an ES20 line card:
  - **drop.**
  - **set-cos-inner-transmit (sets the outer tag Cos)**.
  - **set-cos-transmit (sets dbus cos no tag Cos)**.
  - **transmit**
  - **set-prec-transmit**
  - **set dscp-transmit**
  - **set mpls-exp-imposition-transmit**
- The following default policer actions are supported in an ES20 line card:
  - **transmit** for **conform**.
  - **drop** for **exceed and violate**.
- You can classify **cos-inner, vlan,** and **vlan-inner** policers.
- Ingress classification and policing is performed before an EVC rewrite.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map match-any** *class name*
4. **match cos** *name*
5. **match [ip] precedence** *precedence-value*
6. **policy-map** *policer*
7. **class** *class name*
8. **police** *cir bc pir be*
9. **conform-action [** *set-cos-inner-transmit* | *set-cos-transmit***]**
10. **exceed-action [s***et-cos-inner-transmit* | *set-cos-transmit]*
11. **violate-action [** *transmit* | *drop* **]**
12. **interface GigabitEthernet** *slot/bay/port*
13. **service instance** *ethernet interface*
14. **service policy input** *policy-map-name*
15. **end**

DETAILED STEPS

	Command	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	**`class-map match-any`** *`class name`*
	Example:
	`Router# configure class-map match any test`
		Configures the match criteria for a class-map to be successful match criteria for all packets.

Step 4	**`match cos`** *`name`*
	Example:
	`Router# (config-cmap)# match cos2`
		Matches a packet based on a Layer 2 class of service (CoS) name.

Step 5	**`Router (config-cmap)# match [ip] precendence`** *`precedence-value`*
	Example:
	`Router#(config-cpmap)# match ip precedence 3`
		Identifies IP precedence values as match criteria.

Step 6	**`Router (config-cmap)# policy-map`** *`policer`*
	Example:
	`Router (config-cmap)# policy-map 2r3c`
		Specifies the traffic policy name and also allows you to enter policy-map configuration mode (a prerequisite for enabling QoS features such as traffic policing or traffic shaping).

Step 7	**`Router (config-pmap)# class`** *`class name`*
	Example:
	`Router (config-pmap)# class test`
		Associates the traffic class with the traffic policy.
		For the class-name argument, you can specify the name of the class you created when you used the class-map command to create the traffic class.

	Command	Purpose
Step 8	**Router (config-pmap-c)# Police** *cir bc pir be* **Example:** Router (config-pmap-c)# police cir 1000000 bc 5000 pir 100000000 be 10000000	Specifies a maximum bandwidth usage by a traffic class through the use of a token bucket algorithm, where: • cir—Specifies the CIR (Committed Information Rate) at which the first token bucket is updated as a value in bits per second. The value is a number from 128000 to 10000000000. • bc-conform burst—Specifies the conform burst (bc) size in bytes used by the first token bucket for policing. The value is a number from 1000 to 31,250,000. • pir rate—Specifies the PIR (Peak Information Rate) at which the second token bucket is updated as a value in bits per second. The value is a number from 128000 to 10000000000. • be—Specifies the exceed burst value used by the second token bucket.
Step 9	**Router (config-pmap-c-police)# conform-action [***set-cos-inner-transmit | set-cos-transmit* **]** **Example:** Router (config-pmap-c-police)# conform-action set-cos-inner-transmit 4	Configures the traffic policing based on the options specified: • Inner-transmit. • Transmit.
Step 10	**Router (config-pmap-c-police)# exceed-action [** *set-cos-inner-transmit | set-cos-transmit* **]** **Example:** Router(config-pmap-c-police)# exceed action set-cos-transmit 3	Configures the traffic policing according to the options specified: • Inner-transmit. • Transmit.
Step 11	**Router (config-pmap-c-police)# violate-action [** *transmit | drop* **]** **Example:** Router (config-pmap-c-police)# violate action set-cos-transmit 3	Configures the traffic policing according to the options specified: • Transmit. • Drop.
Step 12	**Router (config)# interface GigabitEthernet** *slot/bay/port* **Example:** Router (config)# interface gig2/0/0	Identifies the interface to which the service policy should be attached.
Step 13	**Router (config-if)# service instance** *ethernet interface* **Example:** Router(config-if)# service instance 1 eth	Identifies the service instance of the interface to which the service policy is attached.

	Command	Purpose
Step 14	**Router (config-if-srv)# service policy input** *policy-map-name* **Example:** `Router(config-if-srv)# service policy input test`	Attaches the configured policy-map to the ethernet service instance.

Step 15	**end**
	Example:
	`Router(config-if-srv)# end`
		Ends the command operations.

Example

The following example shows the 2R3C configuration in a class and policy-map:

```
policy-map test
class cos2
police 1000000 pir 2000000 conform-action set-cos-transmit 3 exceed-action
set-cos-transmit 1 violate-action drop
```

The service policy should be attached to the service instance of the interface as follows:

```
interface gig2/0/0
service instance 1 eth
---- EVC configurations ---
service-policy input test
```

Ingress Policing on EVC Port Channel

The EVC port channel supports 1R0C and 2R3C policers. The classifications and police actions supported on regular EVCs are supported on an EVC port channel.

Restrictions and Usage Guidelines

When configuring ingress policing on EVC port channel, follow these restrictions and guidelines:

- 1R0C is supported only in class-default (classification), and not for service groups.
- 1R0C supports these policing actions on packets:
  - **transmit** for conformed packets
  - **drop** for exceed packets
- 1R0C does not support **set** action for conformed and exceeded packets.
- 1R0C supports flat and HQOS policy maps. HQOS policy maps support policing actions only for the parent class class-default and marking action in child classes.
- 2R3C is supported in these user class and class-default classification:
  - **match precedence**
  - **match dscp**
  - **match vlan-outer**

  - **match vlan-inner**
  - **match cos-outer**
  - **match cos-inner**
- 2R3C supports the following policing actions:
  - Conformed packets: **transmit, set-cos-transmit, set-cos-inner-transmit, set-prec-transmit, set-dscp-transmit, set-exp-transmit**
  - Exceed packets: **transmit, drop, set-cos-transmit, set-cos-inner-transmit, set-prec-transmit, set-dscp-transmit, set-exp-transmit**
  - Violate packets: **transmit, drop, set-prec-transmit, set-dscp-transmit, set-exp-transmit**
- 2R3C is supported for service groups.
- 2R3C supports only flat policy-maps.
- Each NPU complex supports a maximum of 3000 2R3C policers.
- The maximum value for **cir** is limited to 990 Mbps for a 1 Gb port and 9.9 Gbps for a 10 Gb port. For 2R3C policer, the same restriction applies to **pir.**

Configuring Ingress Policing on EVC Port Channel

Configuring policing on EVC port channel is same as that for configuring policing on a regular EVC. For configuration, refer the section "Configuring Policing" section on page 3-21.

Configuring Classification

Use the QoS classification features to select your network traffic and categorize it into classes for further QoS processing based on matching certain criteria. The default class, named "class-default," is the class to which traffic is directed for any traffic that does not match any of the selection criteria in the configured class-maps.

Restrictions and Usage Guidelines

When configuring traffic classes on an ES20 line card, follow these restrictions and usage guidelines:

- You can define up to 256 unique classes in a policy-map including class-default .
- A single class-map can contain up to 8 different **match** command statements.
- The ES20 line card does not support combining matches on QoS group, CoS, or input VLAN *with other types of matching criteria* (for example, access control lists [ACLs]) in the same class or policy-map. You cannot configure a **class c1** with **match CoS** and **class c2** with **match ip prec** in the same policy-map at same level.
- When configuring hierarchical QoS on the ES20 line card, if you configure matching on an input VLAN in a parent policy, then only matching on a QoS group is supported in the child policy.
- For ingress marking on EVCs, service instances support the **match vlan** command, **match vlan-inner** command, **match cos** command, **match precedence, match dscp** and **match cos-inner** commands.
- For egress marking on EVCs, service instances support the **match vlan** command, **match vlan-inner** command, **match cos** command, **match precedence**, **match dscp**, **match mpls experimental** and the **match cos-inner** commands.

- QoS classification on Egress ACL is not supported with the following TCP/UDP port parameters:
  - Port range
  - Port gt (greater than)
  - Port lt (lesser than)
  - Established (matching on established connections)

Table 3-3 provides information about which QoS classification features are supported for the ES20 line card on the Cisco 7600 series router. For more information about most of the commands documented in this table, refer to the *Cisco IOS Quality of Service Solutions Configuration Guide, Release 12.2SR*, at:

http://www.cisco.com/en/US/docs/ios/qos/configuration/guide/12_2sr/qos_12_2sr_book.html

Table 3-3 QoS Classification Feature Support

Feature (match command)	Ingress	Egress	Comments
Match on access list (ACL) number (**match access-group** command)	• Maininterface • Subinterface • Switchport	• Maininterface • Subinterface	Supports the following ACLs: • IPv4 and IPv6. • Protocols—ARP, RARP, ICMP, IGMP, UDP, MAC, MPLS, and TCP. • Source and destination ports. • TCP flags. • ToS.
Match on ACL name (**match access-group name** command)	• Maininterface • Subinterface • Switchport	• Maininterface • Subinterface	
Match on any packet (**match any** command)	• Maininterface • Subinterface • Switchport • EVC	• Maininterface • Subinterface • Switchport • EVC	
Match on ATM cell loss priority (CLP) (**match atm clp** command)	—	—	Not supported.
Match on class-map (**match class-map** command)	—	—	Not supported.
Match on Class of Service (CoS) (**match cos** command)	• EVC	• EVC • Switchport • Subinterfaces	Supported for switchport queueing and EVC interface. **Note** CoS classification is available through PFC QoS using MAC address ACLs.
Match on inner CoS (**match inner-cos command**)	• EVC	• EVC	

Table 3-3 QoS Classification Feature Support (continued)

Feature (match command)	Ingress	Egress	Comments
Match on Frame Relay discard eligibility (DE) bit (**match fr-de** command)	—	—	Not supported.
Match on Frame Relay data-link connection identifier (DLCI) (**match fr-dlci** command)	—	—	Not supported.
Match on input VLAN (**match input vlan** command—Matches the VLAN from an input interface.)	—	Maininterface	Supported for output interface only for software-based EoMPLS. **Note** The service policy is applied on the output interface to match the VLAN from the input interface. If you configure matching on input VLAN in a parent policy with hierarchical QoS, then only matching on QoS group is supported in the child policy.
Match on IP DSCP (**match ip dscp** command)	• Maininterface • Subinterface • Switchport • EVCs	• Maininterface • Subinterface • EVCs	
Match on IP precedence (**match ip precedence** command)	• Maininterface • Subinterface • Switchport • EVCs	• Maininterface • Subinterface • EVCs	
Match on IP Real-Time Protocol (RTP) (**match ip rtp** command)	—	—	Not supported.
Match on MAC address for an ACL name (**match mac address** command)	—	—	Not supported.
Match on destination MAC address (**match destination-address mac** command)	—	—	Not supported.
Match on source MAC address (**match source-address mac** command)	—	—	Not supported.
Match on MPLS experimental (EXP) bit (**match mpls experimental** command)	• Maininterface • Subinterface	• Maininterface • Subinterface • EVCs	Supported.
Match on Layer 3 packet length in IP header (**match packet length** command)	—	—	Not supported.

Table 3-3 QoS Classification Feature Support (continued)

Feature (match command)	Ingress	Egress	Comments
Match on QoS group (**match QoS-group** command)	—	Maininterface only for software-based EoMPLS configurations	Supported in software-based EoMPLS configurations only using hierarchical QoS, where the parent policy configures matching on input VLAN and the child policy configures matching on QoS group.
Match on protocol (**match protocol** command	• Maininterface • Subinterface • Switchport	• Maininterface • Subinterface	Supports matching on IP and IPv6.
Match on VLAN (**match vlan** command—Matches the outer VLAN of a Layer 2 802.1Q frame)	• EVC	• Switchport • EVC	Supported. • Outer VLAN ID matching for 802.1Q tagged frames on EVC.
Match on VLAN Inner (**match vlan inner** command—Matches the innermost VLAN of the 802.1Q tag in the Layer 2 frame)	• EVC	• EVC	Supported.
No match on specified criteria (**match not** command)	—	—	Not supported.

Configuration Tasks

To create a user-defined QoS traffic class, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	Router(config)# **class-map** [**match-all** | **match-any**] *class-name*	Creates a traffic class, where: • **match-all**—(Optional) Specifies that all match criteria in the class-map must be matched, using a logical AND of all matching statements defined within the class. This is the default. • **match-any**—(Optional) Specifies that one or more match criteria must match, using a logical OR of all matching statements defined within the class. • *class-name*—Specifies the user-defined name of the class. **Note** You can define up to 255 unique class-maps within a policy map as 1 class is always class-default.
Step 2	Router(config-cmap)# **match** *type*	Specifies the matching criterion to be applied to the traffic, where *type* represents one of the forms of the **match** command supported by the ES20 line card as shown in Table 3-3. **Note** A single class-map can contain up to 8 different **match** command statements.

Examples

This example shows how to configure a class-map named ipp5, and enter a match statement for IP precedence 5:

```
Router(config)# class-map ipp5
Router(config-cmap)# match ip precedence 5
Router(config-cmap)#
```

This is an example of configuring class matching on multiple match statements.

```
Router(config)# class-map match-any many (id 1047)
Router(config-cmap)# match ip  precedence 3
Router(config-cmap)# match access-group  100
Router(config-cmap)# match mpls experimental 5
```

This is an example of configuring class matching on named ACLS.

```
Router(config)# class-map match-all acl9 (id 1049)
Router(config-cmap)# match access-group name rock
```

This example shows a logical AND operation in a child policy with **match vlan** and **class-default** in a parent.

```
Router(config)# class-map match-all childAND
Router(config-cmap)# match cos 2-3
Router(config-cmap)# match cos inner 5 6
Router(config)# policy-map testchildAND
Router(config-pmap)# class childAND
Router(config-pmap-c)# shape average 100000000
Router(config)# policy-map parentAND
Router(config-pmap)# class vlan12
Router(config-pmap-c)# shape average 500000000
Router(config-pmap-c)# service-policy testchild
```

This example shows how to display class-map information for a specific class-map using the **show class-map** command:

```
Router# show class-map ipp5
Class Map match-all ipp5 (id 1)
Match ip precedence 5
```

This example shows how to display class-map information matching on extended ACLs using the **show class-map** command.

```
head# show class-map acl5
 Class Map match-all acl5 (id 1042)
   Match access-group  102
```

This example shows how to verify classification on a VLAN in the parent class of a HQoS policy.

```
head# show policy-map match
policy-map match4
  class child AND
   shape average 3000000

policy-map match2
  class child AND
    shape average 200000
```

Configuring Policing

This section describes information for configuring QoS traffic policing policies.

Restrictions and Usage Guidelines

The Cisco 7600 series routers support different forms of policing using the **police** command. See Table 3-4 to determine which policing features are supported by ES20 line card type.

When configuring ingress policing on main, subinterface, and VLAN, follow these restrictions and usage guidelines:

- The ES20 line card supports **conform-action** policing on input interfaces only.
- Egress policing command is executed when the priority is defined, and the policy-map is not rejected because we can add policing and priority. While you configure the LLQ values on a dynamically applied policy-map, ensure that you define police values followed by priority values.
- Egress policing command is executed when the priority is defined. The stand alone policing is not rejected because you can add priority values. When you configure the LLQ values on a dynamically applied policy-map, ensure that you define police values followed by **bc** and **be** priority values. You can configure these values and also define the queue limit.
- Use policing with priority queueing to limit the traffic rate in egress.When configuring policing paired with priority, the conform action is fixed to transmit while violate or exceed is fixed to drop. These action are not user configurable.
- When configuring one rate 2 color per EVC micro-flow policer:

- Up to 16,000 policers per line card are supported; only one per service instance configured within the class **class-default.**
- The conform-action or the exceed-action are statically defined and can not be altered.
- One form of policy configuration is support with optional marking in the child policy and policing in the parent policy.

Table 3-4 provides information about which policing features are supported for the ES20 line card on the Cisco 7600 series routers.

Table 3-4 QoS Policing Feature Support

Policing Feature (police command)	Supported Interfaces	ES20 Line Card
Policing by aggregate policer (**police aggregate** command)	• Maininterface • Subinterface	Supported in ingress only.
Policing by bandwidth using token bucket algorithm (**police** command)	• Maininterface • Subinterface • EVC (egress only)	For egress, must be paired with **priority** command.
Policing with 2-color marker (committed information rate [CIR] and peak information rate [PIR]) (**police (two rates)** command—**police cir pir** form)	• Maininterface • Subinterface	Supported in ingress only.
Policing by flow mask (**police flow mask** command)	• Maininterface • Subinterface	Supported in ingress only.
Policing by microflow (**police flow** command)	• Maininterface • Subinterface	Supported in ingress only.
One rate 2 color per EVC micro-flow policer (committed information rate [CIR]) (**police** command—**police cir** form)	• EVC	Supported in ingress only beneath class class-default.

Table 3-5 provides information about the interfaces and their supported actions on an ES20 line card.

Table 3-5 One-Rate Two Color Ingress Support

Supported Interfaces	Conform Actions	Exceed Actions
Maininterface	• transmit • set dscp transmit • set precedence transmit • set mpls imposition exp	• drop • policed dscp transmit
Subinterface	• transmit • set dscp transmit • set precedence transmit • set mpls imposition exp	• drop • policed dscp transmit

Table 3-5 One-Rate Two Color Ingress Support

Supported Interfaces	Conform Actions	Exceed Actions
EVC	transmit	drop **Note** EVC is supported only in class-default.
Switchport	• transmit • set dscp transmit • set precedence transmit • set mpls imposition exp	• drop • policed dscp transmit • set precdence • set dscp • set mpls imposition exp

Table 3-6 QoS Policing Action Support

Policing Action (set command)		ES20 Line Card
Drop the packet. (**drop** command)	• Maininterface • Subinterface • EVC	Supported—Ingress and egress.
Set the IP precedence and transmit. (**set-prec-transmit** command)	• Maininterface • Subinterface	Supported —Input interface only.
Set the IP DSCP and transmit. (**set-dscp-transmit** command)	• Maininterface • Subinterface	Supported—Input interface only.
Set the MPLS EXP bit (0–7) on imposition and transmit. (**set-mpls-experimental-imposition-transmit** command)	• Maininterface • Subinterface	Supported—Input interface only.
Set the MPLS EXP bit in the topmost label and transmit. (**set-mpls-experimental-topmost-transmit** command)	• Maininterface • Subinterface	Supported—Input interface only.
Transmit all packets without alteration. (**transmit** command)	• Maininterface • Subinterface • EVC	Supported—Ingress and egress.

Configuration Tasks

To create QoS traffic policies with policing, use the following commands beginning in global configuration mode:

Note When creating QoS traffic policies as shown below, you can perform only one of the commands shown in Step 3 through Step 7 for each policy. You can perform Step 4 or Step 5 or Step 6 or Step 7; do not attempt to perform Step 3 through Step 7 in the same policy.

	Command	Purpose
Step 1	Router(config)# **mls QoS**	Enables the QoS functionality on an interface.
Step 2	Router(config)# **policy-map** *policy-map-name*	Creates or modifies a traffic policy and enters policy-map configuration mode, where: • *policy-map-name*—Specifies the name of the traffic policy to configure. Names can be a maximum of 40 alphanumeric characters.
Step 3	Router (config-pmap)# **class** {*class-name* | **class-default**}	Specifies the name of the traffic class to which this policy applies and enters policy-map class configuration mode, where: • *class-name*—Specifies that the policy applies to a user-defined class name previously configured. • **class-default**—Specifies that the policy applies to the default traffic class.
Step 4	Router(config-pmap-c)# **police** *bps* [*burst-normal*] [*burst-max*] **conform-action** *action* **exceed-action** *action* **violate-action** *action*	Specifies a maximum bandwidth usage by a traffic class through the use of a token bucket algorithm, where: • *bps*—Specifies the average rate in bits per second. Valid values are 128000 to 1 Gigabyte or 10 Gigabyte. • *burst-normal*—(Optional) Specifies the normal burst size in bytes. Valid values are 1000 to 51200000. The default normal burst size is 1500 bytes. • *burst-max*—(Optional) Specifies the excess burst size in bytes. Valid values are 1000 to 51200000. • *action*—Specifies the policing command (as shown in Table 3-5) for the action to be applied to the corresponding conforming, exceeding, or violating traffic.
Or		

	Command	Purpose
	Router(config-pmap-c)# **police** {**cir** *cir*} [**bc** *conform-burst*] {**pir** *pir*} [**be** *peak-burst*] [**conform-action** *action* [**exceed-action** *action* [**violate-action** *action*]]]	Configures traffic policing using two rates, the committed information rate (CIR) and the peak information rate (PIR), where: • **cir** *cir*—Specifies the CIR at which the first token bucket is updated as a value in bits per second. The value is a number from 128000 to 10000000000. • **bc** *conform-burst*—(Optional) Specifies the conform burst (bc) size in bytes used by the first token bucket for policing. The value is a number from 1000 to 31,250,000. • **pir** *pir*—Specifies the PIR at which the second token bucket is updated as a value in bits per second. The value is a number from 128000 to 10000000000. • **be** *peak-burst*—(Optional) Specifies the peak burst (be) size in bytes used by the second token bucket for policing. The size varies according to the interface and platform in use. • *action*—(Optional) Specifies the policing command (as shown in Table 3-5) for the action to be applied to the corresponding conforming, exceeding, or violating traffic.
Or		
	Router(config-pmap-c)# **police flow** {*bits-per-second* [*normal-burst-bytes*] [*maximum-burst-bytes*] [**pir** *peak-rate-bps*]} | [**conform-action** *action*] [**exceed-action** *action*] [**violate-action** *action*]	Configures a microflow policer, where: • *bits-per-second*—Specifies the CIR in bits per second. Valid values are from 32,000 to 4,000,000,000 bits per second. • *normal-burst-bytes*—(Optional) Specifies the CIR token-bucket size. Valid values are from 1000 to 31,250,000 bytes. • *maximum-burst-bytes*—(Optional) Specifies the PIR token-bucket size. Valid values are from 1000 to 31,250,000 bytes. • **pir** *peak-rate-bps*—(Optional) Specifies the PIR in bits per second. Valid values are from 32,000 to 4,000,000,000 bits per second. • *action*—Specifies the policing command (as shown in Table 3-5) for the action to be applied to the corresponding conforming, exceeding, or violating traffic.
Or		

	Command	Purpose
	Router(config-pmap-c)# **police flow mask** {**dest-only** | **full-flow** | **src-only**} {*bits-per-second* [*normal-burst-bytes*] [*maximum-burst-bytes*]} [**conform-action** *action*] [**exceed-action** *action*]	Configures a flow mask to be used for policing, where: • **dest-only**—Specifies the destination-only flow mask. • **full-flow**—Specifies the full-flow mask. • **src-only**—Specifies the source-only flow mask. • *bits-per-second*—Specifies the CIR in bits per second. Valid values are from 32,000 to 4,000,000,000 bits per second. • *normal-burst-bytes*—(Optional) Specifies the CIR token-bucket size. Valid values are from 1000 to 31,250,000 bytes. • *maximum-burst-bytes*—(Optional) Specifies the PIR token-bucket size. Valid values are from 1000 to 31,250,000 bytes. • *action*—Specifies the policing command (as shown in Table 3-5) for the action to be applied to the corresponding conforming or exceeding traffic.
Or		
	Router(config-pmap-c)# **police aggregate** *name*	Specifies a previously defined aggregate policer name and configures the policy-map class to use the specified *name* of the aggregate policer.

Examples

This example shows a traffic policing configuration with the average rate at 128kbps, the normal burst size at 128,000 bytes, and the excess burst size at 4000 bytes:

```
Router(config)# class-map acgroup2
Router(config-cmap)# match access-group 2
Router(config-cmap)# exit
Router(config)# policy-map police
Router(config-pmap)# class acgroup2
police cir 128000 pir 10000000 conform-action transmit exceed-action drop 4 violate-action
drop set-cos-transmit 4 violate-action drop
Router(config-pmap-c)# exit
Router(config-pmap)# exit
Router(config)# interface gigethernet 1/0/0
Router(config-if)# service-policy input police
```

This example shows a policy that contains a one rate 2 color per EVC micro-flow policer with marking operations.

```
Router(config)# policy-map child
Router(config-pmap)# class cos1
Router(config-pmap-c)# set cos 5
Router(config-pmap)# class class-default
Router(config-pmap-c)# set cos 6

Router(config)# mls QoS
Router(config)# policy-map efppolicy
Router(config-pmap)# class class-default
Router(config-pmap-c)# police cir 200000000 bc 6250000 conform-action transmit
exceed-action drop
Router(config-if)# service-policy child
```

This example shows a policy that contains a one- rate zero color (1R0C) policer:

```
Router(config)#policy-map 1r0c
Router(config-pmap)#class class-default
Router(config-pmap-c)#police cir 1000000
Router(config-pmap-c-police)#end
```

This example shows a policy that contains a two- rate three color (2R3C) policer:

```
Router(config)#policy-map 2r3c
Router(config-pmap)#class cos1
Router(config-pmap-c)#police cir 1000000 pir 2000000 conform-action set-prec-transmit 3
exceed-action set-prec-transmit 4 violate-action drop
Router(config-pmap-c-police)#end
```

This example shows how to apply a policy-map on the EVC port channel:

```
Router(config)#int port-channel 100
Router(config-if)#service instance 1 ethernet
Router(config-if-srv)#encapsulation dot1q 100
Router(config-if-srv)#service-policy in 1r0c
```

Verification

Use the following commands to verify policing:

Command	Purpose
`Router# show policy-map`	Displays all configured policy-maps.
`Router# show policy-map policy-map-name`	Displays the user-specified policy-map.
`Router# show policy-map interface`	Displays statistics and configurations of all input and output policies that are attached to an interface.

This example shows how to display policing statistics using the **show policy-map interface** command in the EXEC mode.

```
sh policy-map int gig 4/0/5
GigabitEthernet4/0/5
Service-policy output: example
Counters last updated 00:00:00 ago
queue stats for all priority classes:
Queueing
queue limit 66 packets
(queue depth/total drops/no-buffer drops) 0/121416/0
(pkts output/bytes output) 428079/640406145

Class-map: prec1 (match-all)
497836 packets, 744762656 bytes
5 minute offered rate 18806000 bps, drop rate 828000 bps
Match: ip precedence 1
police:
```

```
cir 3000000 bps, bc 93750 bytes
conformed 324764 packets, 485846905 bytes; actions: transmit
exceeded 121416 packets, 181636879 bytes; actions: drop
conformed 12256000 bps, exceed 842000 bps
Priority: Strict, b/w exceed drops: 121416

Class-map: class-default (match-any)
1761338 packets, 1138941205 bytes
5 minute offered rate 27792000 bps, drop rate 26713000 bps
Match: any
Queueing
queue limit 66 packets
(queue depth/total drops/no-buffer drops) 70/824757/0
(pkts output/bytes output) 300574/420798000
shape (average) cir 2000000, bc 8000, be 8000
target shape rate 2000000
```

This is another example of displaying policing statistics using the **show policy-map interface** command; in this case the statistics are for a one rate 2 color per EVC micro-flow policer.

```
TenGigabitEthernet4/0/0: EFP 1
Service-policy input: policesmall
Counters last updated 00:00:00 ago
Class-map: class-default (match-any)
12353351 packets, 3627868092 bytes
5 minute offered rate 671206 bps, drop rate 543206 bps
Match: any
police:
cir 128000 bps, bc 4000 bytes
conformed 2096904 packets, 130008048 bytes; actions: transmit
exceeded 10256447 packets, 134308720 bytes; actions: drop
conformed 127000 bps, exceed 543206 bps
```

Attaching a QoS Traffic Policy to an Interface

Before a traffic policy can be enabled for a class of traffic, it must be configured on an interface. A traffic policy also can be attached to Ethernet subinterfaces, maininterfaces, and service instances.

Traffic policies can be applied for traffic coming into an interface (input), and for traffic leaving that interface (output).

Attaching a QoS Traffic Policy for an Input Interface

When you attach a traffic policy to an input interface, the policy is applied to traffic coming into that interface. To attach a traffic policy for an input interface, use the following command beginning in interface configuration mode:

Command	Purpose
Router(config-if)# **service-policy input** *policy-map-name*	Attaches a traffic policy to the input direction of an interface, where: • *policy-map-name*—Specifies the name of the traffic policy to configure.

Attaching a QoS Traffic Policy to an Output Interface

When you attach a traffic policy to an output interface, the policy is applied to traffic leaving that interface. To attach a traffic policy to an output interface, use the following command beginning in interface configuration mode:

Command	Purpose
Router(config-if)# **service-policy output** *policy-map-name*	Attaches a traffic policy to the output direction of an interface, where: • *policy-map-name*—Specifies the name of the traffic policy to configure.

Configuring Marking

After you have created your traffic classes, you can configure traffic policies to configure marking features to apply certain actions to the selected traffic in those classes.

In most cases, the purpose of a packet mark is identification. After a packet is marked, downstream devices identify traffic based on the marking and categorize the traffic according to network needs. This categorization occurs when the **match** commands in the traffic class are configured to identify the packets by the mark (for example, **match ip precedence**, **match ip dscp**, **match cos**, and so on). The traffic policy using this traffic class can then set the appropriate QoS features for the marked traffic.

In some cases, the markings can be used for purposes besides identification. Distributed WRED, for instance, can use the IP precedence, CoS, IP DSCP, or MPLS EXP values to detect and drop packets.

Restrictions and Usage Guidelines

When configuring class-based marking on an ES20 line card, follow these restrictions and usage guidelines:

- Ingress packet marking is supported on maininterfaces, subinterfaces, and service instances. You can configure packet marking and queueing actions in the same traffic policy.
- If an ingress service policy configures both class-based marking and marking as part of a policing action, then the marking using policing takes precedence over any class-based marking.
- The Scalable EoMPLS on 7600-ESM-2X10GE and 7600-ESM-20X1GE feature supports mapping of the incoming VLAN dot1q p-bits to the outgoing MPLS EXP bits. Only outer tag dot1q pbits mapping is supported.
- The Scalable EoMPLS on 7600-ESM-2X10GE and 7600-ESM-20X1GE feature supports mapping of the incoming MPLS EXP bits to the outgoing VLAN dot1q p-bits.
- MultiPoint Bridging over Ethernet on 7600-ESM-2X10GE and 7600-ESM-20X1GE supports **set cos** and **set cos inner** commands in ingress marking. However, from 12.2(33) SRE onwards, ingress marking also supports **set dscp** and **set precedence** commands.
- Set operations are not allowed at the parent level.
- Up to two marking combinations can occur with the exception that Layer 2 marking is not combined with Layer 3 marking operations.
- You should configure **no mls qos trust** value for ingress marking on the main and subinterfaces.

Table 3-7 and Table 3-8 provides information about the various Ingress and Egress Marking combinations supported for a ES20 line card on the Cisco 7600 series router.

For example, in Table 3-7, ✓ marked against DSCP and Cos (row 4 col 1) indicates that you can mark DSCP and Cos within a single class. Similarly, a **x** marked against MPLS Experimental and Precedence (row 5 col 3) indicates that you cannot use both within a single class.

Table 3-7 Layer 3 Egress Marking Scenarios For EVCs

	Cos	Cos Inner	Precedence	DSCP	EXP	Cos and Cos Inner
Cos	—	✓	✓	✓	✓	—
Cos_Inner	✓	—	x	x	✓	—
Precedence	✓	x	—	x	x	✓
DSCP	✓	x	x	—	x	✓
MPLS Experimental	✓	✓	x	x	—	✓
Cos and Cos Inner	—	—	✓	✓	✓	—

Table 3-8 Layer 3 Ingress Marking Scenarios On a Service Instance

	Cos	Cos Inner	Precedence	DSCP	EXP	Cos and Cos Inner
Cos	—	✓	✓	✓	x	—
Cos_Inner	✓	—	x	x	✓	x
Precedence	✓	x	—	—	✓	✓
DSCP	✓	x	—	—	✓	✓
MPLS Experimental	x	✓	✓	✓	✓	—
Cos and Cos Inner	—	—	✓	✓	—	—

The following example shows the sample ingress policy map for layer 3 marking support.

```
policymap policy1-in
class prec5
set cos 6
class cos_tos_all
set ip prec 4
class class-default
police cir 1000000
```

The following example shows Egress marking - port in set trust cos mode.

```
policymap policy1-eg-mark-BD
class prec5
set cos 5
set ip dscp af31
shape average 1000000
class cos_in
set cos cos-inner
shape average 2000000
```

```
class cos_exp_all
set cos 6
shape average 3000000
policymap policy-hqos-eg-mark-BD
class cosin_cos_dscp_all
set ip dscp af31
shape average 1000000
class class-default
shape average 2000000
service-policy policy1-eg-mark-BD
```

The following example shows the sample output of the **sh** command.

```
pacv-7609S#sh policy-map interface ten4/0/1 service instance 1
TenGigabitEthernet4/0/1: EFP 1
Service-policy output: policy-hqos-eg-mark-BD
Counters last updated 00:00:04 ago
Class-map: cosin_cos_dscp_all (match-all)
0 packets, 0 bytes
5 minute offered rate 0000 bps, drop rate 0000 bps
Match: cos  1
Match:  dscp 3
Match: cos inner  3
Queueing
queue limit 66 packets
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts output/bytes output) 0/0
QoS Set
dscp af31
Packets marked 0
shape (average) cir 1000000, bc 4000, be 4000
target shape rate 1000000
Class-map: class-default (match-any)
0 packets, 0 bytes
5 minute offered rate 0000 bps, drop rate 0000 bps
Match: any
Queueing
queue limit 66 packets
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts output/bytes output) 0/0
shape (average) cir 2000000, bc 8000, be 8000
target shape rate 2000000
Service-policy : policy1-eg-mark-BD
Counters last updated 00:00:04 ago
Class-map: prec5 (match-any)
0 packets, 0 bytes
5 minute offered rate 0000 bps, drop rate 0000 bps
Match: ip precedence 5
Queueing
queue limit 66 packets
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts output/bytes output) 0/0
QoS Set
cos 5
Packets marked 0
dscp af31
Packets marked 0
shape (average) cir 1000000, bc 4000, be 4000
target shape rate 1000000
Class-map: cos_in (match-all)
0 packets, 0 bytes
5 minute offered rate 0000 bps, drop rate 0000 bps
Match: cos inner  4
Queueing
```

```
Queue limit 66 packets
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts output/bytes output) 0/0
QoS Set
cos cos-inner
Packets marked 0
shape (average) cir 2000000, bc 8000, be 8000
target shape rate 2000000
Class-map: cos_exp_all (match-all)
0 packets, 0 bytes
5 minute offered rate 0000 bps, drop rate 0000 bps
Match: mpls experimental topmost 3
Match: cos  1
Queueing
queue limit 66 packets
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts output/bytes output) 0/0
QoS Set
cos 6
Packets marked 0
shape (average) cir 3000000, bc 12000, be 12000
target shape rate 3000000
Class-map: class-default (match-any)
0 packets, 0 bytes
5 minute offered rate 0000 bps, drop rate 0000 bps
Match: any
queue limit 66 packets
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts output/bytes output) 0/0
```

Table 3-9 Qos Class Based Marking Feature Support On A Main-Interface, Sub-Interface, and Serivce Instance

Marking Feature (set command)	Ingress	Egress	ES20 Line Card
Set ATM CLP bit (**set atm-clp** command—Mark the ATM cell loss priority bit with value of 1)	—	—	Not supported.
Set discard class (**set discard-class** command—Marks the packet with a discard class value for per-hop behavior)	—	—	Not supported.
Set Frame Relay DE bit (**set fr-de** command—Mark the Frame Relay discard eligibility bit with value of 1)	—	—	Not supported.
Set IP DSCP (**set ip dscp** command—Marks the IP differentiated services code point (DSCP) in the type of service (ToS) byte with a value from 0 to 63.)	• Subinterface • Maininterface • Switchport • EVC interface	• Subinterface • Maininterface • Switchport • EVC interface	Supported on ingress and egress.

Marking Feature (set command)	Ingress	Egress	ES20 Line Card
Set IP precedence (**set ip precedence** command—Marks the precedence value in the IP header with a value from 0 to 7.)	• Subinterface • Maininterface • Switchport • EVC	• Subinterface • Maininterface • Switchport • EVC	Supported on ingress and egress.
Set Layer 2 802.1Q CoS (**set cos** command—Marks the CoS value from 0 to 7 in an 802.1Q tagged frame.)	• EVC interface	• Subinterface • EVC interface • Switchport	• Subinterfaces supported on egress. • EVCs supported in ingress and egress.
Set Layer 2 802.1Q CoS (**set cos-inner** command—Marks the inner CoS field from 0 to 7 in a bridged frame.)	• EVC	• Subinterface • EVC interface	• Subinterfaces supported on output. • EVCs supported in ingress and egress.
Set Layer 2 802.1Q CoS (**set cos-inner cos** command—Copies out CoS to inner CoS.	• EVC	• Subinterface • EVC interface	• Subinterfaces supported on egress. • EVCs supported in ingress and egress.
Set Layer 2 802.1Q CoS **(set cos cos-inner** command)	• EVC	• Subinterface • EVC interface	• Subinterfaces supported on output. • EVCs supported in input and output direction.
Set MPLS experimental (EXP) bit on label imposition (**set mpls experimental imposition** command)	• Maininterface • Subinterface • EVC interface • Switchport	Not supported	Supported on ingress for a service instance. **Note** Marking of EXP in case of MPB results in marking of DBUS CoS. Marking of EXP in case of xconnect and connect results in the marking of DBUS Cos as well as the MPLS Tags added as part of xconnect/connect processing.
Set MPLS EXP topmost (**set mpls experimental topmost** command)	—	EVC interface	Supported for EVC in egress.
Set QoS group (**set QoS-group** command—Marks the packet with a QoS group association.)	—	—	Not supported.

Configuration Tasks

To configure a QoS traffic policy with class-based marking, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	Router(config)# **policy-map** *policy-map-name*	Creates or modifies a traffic policy and enters policy-map configuration mode, where: • *policy-map-name*—Specifies the name of the traffic policy to configure. Names can be a maximum of 40 alphanumeric characters.
Step 2	Router (config-pmap)# **class** {*class-name* | **class-default**}	Specifies the name of the traffic class to which this policy applies and enters policy-map class configuration mode, where: • *class-name*—Specifies that the policy applies to a user-defined class name previously configured. • **class-default**—Specifies that the policy applies to the default traffic class.
Step 3	Router(config-pmap-c)# **set** *type*	Specifies the marking action to be applied to the traffic, where *type* represents one of the forms of the **set** command supported by the ES20 line card as shown in Table 3-8.

Examples

This example shows the creation of a service policy called policy1. This service policy is associated to a previously defined classification policy through the use of the **class** command. This example assumes that a classification policy called class1 was previously configured.

```
Router(config)# policy-map policy1
Router(config-pmap)# class class1
Router(config-pmap-c)# set ip precedence 1
```

Verification

Use the following commands to verify marking:

Command	Purpose
`Router# show policy-map`	Displays all configured policy-maps.
`Router# show policy-map policy-map-name`	Displays the user-specified policy-map.
`Router# show policy-map interface`	Displays statistics and configurations of all input and output policies that are attached to an interface.

For more detailed information about configuring class-based marking features, refer to the *Class-Based Marking* document located at the following URL:

http://www.cisco.com/en/US/docs/ios/12_1t/12_1t5/feature/guide/cbpmark2.html

Configuring Shaping

This section describes information for configuring QoS traffic policies for shaping traffic.

Restrictions and Usage Guidelines

When configuring queueing features on an ES20 line card, follow these restrictions and usage guidelines:

- The Cisco 7600 series routers support different forms of queueing features. See Table 3-10 to determine which traffic shaping features are supported by the ES20 line card type.
- Use a hierarchical policy if you want to achieve minimum bandwidth guarantees using CBWFQ. First, configure a parent policy to shape to the total bandwidth required. Then, define a child policy using CBWFQ for the minimum bandwidth percentages.
- Traffic shaping is a queue with some CIR and excess information rate (EIR) value, such that CIR plus EIR is the shaped rate. Traffic shaping on an average cuts the flow of traffic from the queue at the configured shape rate over a mean period of time

For more detailed information about configuring congestion management features, refer to the *Cisco IOS Quality of Service Solutions Configuration Guide* document corresponding to your Cisco IOS software release.

Table 3-10 provides information about which QoS traffic shaping features are supported for the ES20 line card on the Cisco 7600 series router.

Table 3-10 QoS Traffic Shaping Feature Support

Traffic Shaping Feature (command)	ES20 Line Card
Adaptive shaping for Frame Relay (**shape adaptive** command)	Not supported.
Class-based shaping (**shape average**, **shape peak** commands)	Supports **shape average** only in egress.
Policy-map class shaping with adaptation to backward explicit congestion notification (BECN) (**shape adaptive** command)	Not supported.
Policy-map class shaping with reflection of forward explicit congestion notification (FECN) as BECN (**shape fecn-adapt** command)	Not supported.
Policy-map class shaping of peak rate of traffic by percentage of bandwidth (**shape peak percent** command)	Not supported.

Configuration Tasks

Use MQC to configure traffic shaping. Create a class-map using the **class-map** command and a policy-map using the **policy-map** command. Attach the class to the policy using the **class** command and then use the **shape** command to configure shaping for that class.

	Command	Purpose
Step 1	Router(config)# **class-map** [**match-all** | **match-any**] *class-name*	Creates a class-map to be used for matching packets to a class.
Step 2	Router(config-cmap)# **match** [**ip dscp** *ip-dscp-value* | **ip precedence** *ip-precedence-value* | **mpls experimental** *mpls-exp-value*]	Specifies a specific IP DSCP, IP precedence, or MPLS EXP value as a match criterion.
Step 3	Router(config)# **policy-map** *policy-name*	Specifies the name of the policy-map to configure.
Step 4	Router(config-pmap)# **class** *class-name*	Specifies the name of a predefined class included in the service policy.
Step 5	Router(config-pmap-c)# **shape** [**average**] *mean-rate* [[*burst-size*] [*excess-burst-size*]]	Specifies new values for traffic shaping. Maximum values are 128000 to 990 Mbps or 9.9Gb.

Examples

This example shows traffic shaping on a main interface; traffic leaving interface Gig1/0/0 is shaped at the rate of 10 Mbps:

```
Router(config)# class-map class-interface-all
Router(config-cmap)# match ip precedence 1
Router(config-cmap)# exit
Router(config)# policy-map dts-interface-all-action
Router(config-pmap)# class class-interface-all
Router(config-pmap-c)# shape average 10000000
Router(config-pmap-c)# exit
Router(config)# interface gig 1/0/0
Router(config-if)# service-policy output dts-interface-all-action
```

This is an example of an output shaping policy on a switchport interface that matches on a CoS value queuing defined in the classes.

```
Router(config)# policy-map switchport-cos-policy
Router(config-pmap)# class cos1
Router(config-pmap-c)# shape ave 100000000
```

Now the policy is applied in the egress direction on the main switchport.

```
Router(config)# interface TenGigabitEthernet9/0/0
Router(config-if)# switchport
Router(config-if)# switchport access vlan 2000
Router(config-if)# switchport mode access
Router(config-if)# mls QoS trust cos
Router(config-if)# service-policy output switchport-cos-policy
```

In this example, the flat policy-map is applied in the egress direction to a subinterface.

```
Router(config)# policy-map x
Router(config-pmap)# class prec5
Router(config-pmap-c)# police 100000000
Router(config-pmap-c)# priority
Router(config-pmap)# class class-default
Router(config-pmap-c)# shape average 100000000
```

In this example, the following policy-map is applied in the egress direction to a subinterface.

```
Router(config)# policy-map child2
Router(config-pmap)# class prec5
Router(config-pmap-c)# shape average 100000000
Router(config)# policy-map pcd
Router(config-pmap)# class class-default
Router(config-pmap-c)# shape average 300000000
Router(config-if)# service-policy child2
```

Verification

Use the following commands to verify traffic shaping:

Command	Purpose
`Router# show policy policy-name`	Displays the configuration of all classes composing the specified traffic policy.
`Router# show policy policy-name class class-name`	Displays the configuration of the specified class of the specified traffic policy.

Configuring Shaping on the ES20 Main Interface

A traffic policy can be nested within a QoS policy when the **service-policy** command is used in a policy-map class configuration mode. A traffic policy that contains a nested traffic policy is called a hierarchical traffic policy.

A hierarchical traffic policy contains a child and a parent policy. The **service-policy** command associates the previously defined child policy with the new traffic policy and the parent policy uses the preexisting traffic policy.

Hierarchical traffic policies are supported on subinterfaces, EVCs, and EVC port channels. When hierarchical traffic policies are used, you c an use a single traffic policy (with a child and a parent policy) to shape and prioritize traffic.

You can use this feature to configure a HQoS on an ES20 main interface where the child policy-map is attached to the class-default of a parent policy-map. For more information on hierarchical shaping, refer to the section "Configuring Hierarchical QoS with Tiered Policy-Maps" section on page 3-50

About Traffic Shaping

Traffic shaping:

- Controls the outbound traffic of an interface to match its flow to the speed of the remote target interface.
- Manages the access to the available bandwidth.
- Regulates the traffic flow to avoid congestion.
- Ensures that the traffic conforms to policies contracted for it.
- Shapes the traffic to eliminate the bottlenecks in topologies with data-rate mismatches.

Cisco IOS QoS uses policing and shaping to regulate data traffic. The rate-limiting features of the Committed Access Rate (CAR) and the Traffic Policing feature provides the functionality to police the traffic and the features of Generic Traffic Shaping (GTS), Class-Based Shaping, and Distributed Traffic

Shaping (DTS) provides the functionality to shape traffic.
For more information on Shaping, refer to the "HQoS Polices with Shape in Child Classes"section in the *Cisco IOS Quality of Serivce Configuration* Guide.

Cisco IOS QoS software uses two types of traffic shaping: GTS and Class-Based.

Their have:

- Similar traffic shaping methods using different CLIs.
- Different types of queues to contain and shape deferred traffic.
- Weighted fair queue to hold the delayed traffic for the deferred packets.

Restrictions and Usage Guidelines

Follow these restrictions and usage guidelines when you configure shaping on an ES20 main interface:

- When a child policy is added to the class-default, no other classes should exist on the parent level.
- When you apply a service policy on a main interface, ensure that the EVC or subinterface is not configured within the main interface.
- A minimum of 128 Kbps shape value and a maximum of 99% of interface bandwidth is supported.

Non example

In the following non example, the policy-map parent has a user-defined class **prec1** and a class **class-default** with a attached child policy.

Based on the following example, this feature does not support policies with:

- user-defined classes at the parent level.
- Class defaults with attached child policies.

```
Policy-map parent
Class prec1
Shape average 10000000
Class class-default
Shape average 100000000
Service-policy child

Policy-map child
Class prec0
Shape average 50000000
Class class-default
Shape average 50000000
```

Though this feature does not support the following scenario, you can apply the policies in the following example.

```
Policy-map parent
Class vlan12 (matches on Vlan 12)
Shape average 10000000
Class class-default
Shape average 20000000
Service-policy child

policy-map child
class cos0
shape average 5000000
class cos1
```

```
shape average 5000000
class class-default
shape average 20000000
```

Configuring the Child Policy

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** ***policy name***
4. **class class-default**
5. **class** *class name*
6. **shape** [average] *mean-rate*
7. **class class-defaul**t
8. **shape [average]** *mean-rate*
9. **exit**

DETAILED STEPS

	Command	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	**policy-map** *policy name* **Example:** `P19_C7609-S(config)# policy-map child`	Enters the policy name.

Step 4	**class class default**
	Example:
	`P19_C7609-S(config)#(config-pmap-c) class class default`
		Applies a class default value to the policy-map.

Step 5	**class** *class name*
	Example:
	`P19_C7609-S(config-pmap)# class test`
		Attaches a class map to the child policy-map.

	Command	Purpose
Step 6	**shape [average]** *mean-rate* **Example:** `P19_C7609-S(config-pmap-c)# shape average 50000000`	Sets the average shaping rate in bits per second for the child policy-map.

Step 7	**class class-default**
	Example:
	`P19_C7609-S(config-pmap-c)# class class-default`
		Specifies the name of the class, whose policy you want to create or change, and the default class (commonly known as the class-default class) before you configure its policy.

Step 8	**shape [average]** *mean-rate* **Example:** `P19_C7609-S(config-pmap-c)# shape average 50000000`	Sets the average shaping rate in bits per second for the child policy-map.
Step 9	**exit** **Example:** `Router (config-if)#`	Exits to global configuration mode.

Configuring the Parent Policy

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy name*
4. **class class-default**
5. **shape** [**average**] *mean-rate*
6. **service-policy** *policy-map name*
7. **interface** *type value*
8. **service-policy** [**output** class-default]
9. **exit**

DETAILED STEPS

	Command	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	**`policy-map`** *`policy-map`*
	Example:
	`P19_C7609-S(config-pmap-c)# policy-map parent`
		Creates a parent policy-map with a specified name.

Step 4	**`class class-default`**
	Example:
	`P19_C7609-S(config-pmap)# class class-default`
		Specifies the name of the class whose policy you want to create or change, and the default class (commonly known as the class-default class), before you configure its policy.

Step 5	**`shape [average]`** *`mean-rate`*
	Example:
	`P19_C7609-S(config-pmap-c)# shape average 100000000`
		Sets the average shaping value in bits per second to ensure a specific bandwidth.

Step 6	**`service- policy`** *`policy-map name`*
	Example:
	`P19_C7609-S(config-pmap-c)# service-policy child`
		Configures the hierarchical child policy.

Step 7	**interface** *type value* **Example:** `Router (config)# interface gig2/0/0`	Identifies the interface to which the service policy should be attached. The format of *value* is slot/subslot/port and ensure that you do not include a space between the *type* and *value* when you execute the command.
Step 8	**service-policy** [**output** class-default] **Example:** `Router(config-if)# service-policy output parent`	Attaches the configured parent policy-map to the Ethernet interface.

Step 9	**`exit`**
	Example:
	`Router (config-if)#`
		Exits to global configuration mode.

Verification

You can execute the **show run** command to view the policy-map attached to the interface.In the following example, the child policy classifies and prioritizes the traffic, and the parent policy shapes the traffic.

```
USA# show policy-map int gig 1/0/0
GigabitEthernet1/0/0
Service-policy output: parent
Counters last updated 00:00:00 ago
Class-map: class-default (match-any)
2 packets, 164 bytes
5 minute offered rate 0000 bps, drop rate 0000 bps
Match: any
Queueing
queue limit 3315 packets
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts output/bytes output) 2/164
shape (average) cir 100000000, bc 400000, be 400000
target shape rate 100000000
Service-policy : child
Counters last updated 00:00:00 ago
Class-map: prec0 (match-all)
0 packets, 0 bytes
5 minute offered rate 0000 bps, drop rate 0000 bps
Match: ip precedence 0
Queueing
queue limit 1655 packets
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts output/bytes output) 0/0
shape (average) cir 50000000, bc 200000, be 200000
target shape rate 50000000
Class-map: class-default (match-any)
2 packets, 164 bytes
5 minute offered rate 0000 bps, drop rate 0000 bps
Match: any
Queueing
queue limit 1655 packets
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts output/bytes output) 2/164
shape (average) cir 50000000, bc 200000, be 200000
target shape rate 50000000
end
```

Configuring QoS Queue Scheduling

This section describes ES20 line card-specific information for configuring QoS queue scheduling.

Restrictions and Usage Guidelines

When configuring queueing features on an ES20 line card, follow these restrictions and usage guidelines:

- The Cisco 7600 series routers support different forms of queueing features. See Table 3-11 to determine which queueing features are supported by ES20 line card type.
- You must use policing with priority queueing to limit the traffic rate.
- ES20 line card supports up to two LLQ queues per policy-maps with equal priority.

- If you receive any **exceed** errors on the line card, we recommend you to remove all the policy-maps from the erroneous port.
- Ensure that you configure the policy-maps to use 99% of the port bandwidth. Policy-map functions erroneously when it is configured to use the 100% bandwidth of the port because 1% of the port bandwidth is reserved for control packets.
- A maximum of 8000 queues are supported per bay.
- The maximum acceptable limit of the Excess Information Rate (EIR) limitation per port is 549 Gigabyte.
- Priority or low latency queue with bit rates (**priority <kbps>** command) is not supported.
- Priority or low latency queue with percent (**priority percent** command) is not supported.

For more detailed information about configuring congestion management features, refer to the *Cisco IOS Quality of Service Solutions Configuration Guide* document corresponding to your Cisco IOS software release.

Table 3-11 provides information about which QoS queueing features are supported for the ES20 line card on the Cisco 7600 series routers.

Table 3-11 QoS Congestion Management and Avoidance Feature Support

Congestion Management and Avoidance Feature (command)	ES20 Line Card
Aggregate Weighted Random Early Detection (WRED) (**random-detect aggregate**, **random-detect dscp (aggregate)**, and **random-detect precedence (aggregate)** commands)	Egress supported. **Note** WRED DSCP based aggregate is supported only on the main and subinterfaces of Layer 3. It is not supported on EVC interfaces.
Weighted RED (WRED)	Not supported.
Class-based Weighted Fair Queueing (CBWFQ) (**bandwidth**, **queue-limit** commands)	Egress supported.
Flow-based Queueing (fair queueing/WFQ) (**fair-queue** command)	Not supported.
Low Latency Queueing (LLQ)/Priority Queueing (**priority** command paired with **police** command)	Egress supported.
Random Early Detection (RED) (**random-detect** commands)	Not supported.

Configuring WRED

Note You can use a maximum of 4 random detect statements with a single class-map and total of 128 WRED statements per bay.

To configure a QoS WRED policy, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	Router(config)# **policy-map** *policy-name*	Specifies the name of the policy-map to configure.
Step 2	Router(config-pmap)# **class** *class-name*	Specifies the name of a predefined class included in the service policy.
Step 3	Router(config-pmap-c)# **shape average**	Shapes traffic to the indicated bit rate for the specified class.
Step 4	Router(config-pmap-c)# **random-detect aggregate**	Enables WRED.
Step 5	Router(config)# **interface** *interface-name*	Specifies the interface to which the policy-map will be applied.
Step 6	Router(config-if)# **service-policy** [**output** *policy-name*]	Attaches the specified policy-map to the interface.

Examples

This is an example of a WRED configuration.

```
Router(config)# policy-map wredtest
Router(config-pmap)# class cos5
Router(config-pmap-c)# shape average 200000000
Router(config-pmap-c)# random-detect precedence-based aggregate
Router(config-pmap-c)# random-detect precedence values 0 min 100 max 200 mark-prob 1
Router(config-pmap-c)# random-detect precedence values 1 min 300 max 500 mark-prob 1
Router(config-pmap-c)# random-detect precedence values 2 min 600 max 900 mark-prob 1
```

The default line can also be configured as follows:

```
Router(config-pmap-c)# random-detect precedence-based aggregate min 200 max 300 mark-prob
1
```

Min and max threshold values listed in the previous command signifies the number of 256 byte packet buffers.

Note Packets or datagrams or frames larger than 256 bytes consumes more packet buffers.

Note In a EVC, if WRED config with precedence is used, the CoS value is mapped to the Prec value.

Configuring CBWFQ

To configure a QoS CBWFQ policy, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	Router(config)# **policy-map** *policy-map-name*	Creates or modifies a traffic policy and enters policy-map configuration mode, where: • *policy-map-name*—Specifies the name of the traffic policy to configure. Names can be a maximum of 40 alphanumeric characters.
Step 2	Router (config-pmap)# **class** {*class-name* | **class-default**}	Specifies the name of the traffic class to which this policy applies and enters policy-map class configuration mode, where: • *class-name*—Specifies that the policy applies to a user-defined class name previously configured. • **class-default**—Specifies that the policy applies to the default traffic class.
Step 3	Router(config-pmap-c)# **bandwidth** {*bandwidth-kbps* | **percent** *percent*}	Specifies the amount of bandwidth, in kbps, or percentage of available bandwidth, to be assigned to the class. The minimum value to set is 128kbps.The amount of bandwidth configured should be large enough to also accommodate Layer 2 overhead.
Step 4	Router(config-pmap-c)# **queue-limit** *number-of-packets*	Specifies the maximum number of 256 byte packets that can be queued for the class. **Note** Packets or datagrams or frames larger than the 256 bytes consumes more packet buffers. For example a packet of size 500 bytes occupies 2 packet buffers. **Note** You cannot have more than 128 **queue- limit** commands per bay. **Note** You can use the **queue-limit** command with shape and priority queues. **Note** Cisco recommends that you use the default queue-limit tuned to the bandwidth rate configured in the queueing action. Use this step if the default is not satisfactory.

Examples

This example shows a service policy called policy1 that specifies the amount of bandwidth to allocate for traffic classes 1 and 2:

```
Router(config)# class-map class1
Router(config-cmap)# match ip dscp 30
Router(config-cmap)# exit

Router(config)# class-map class2
Router(config-cmap)# match ip dscp 10
Router(config-cmap)# exit

Router(config)# policy-map policy1
Router(config-pmap)# class class1
Router(config-pmap-c)# bandwidth 30000
Router(config-pmap-c)# exit
Router(config-pmap)# exit
Router(config)# class class2
Router(config-pmap-c)# bandwidth 20000
Router(config-pmap-c)# exit
Router(config-pmap)# exit
Router(config)#
```

```
Router(config)# interface gig 1/0/0
Router(config-if)# service-policy output policy1
Router(config-if)# exit
```

Use the following commands to verify CBWFQ:

Command	Purpose
Router# **show policy-map** *policy-map*	Displays the configuration of all classes that make up the specified policy-map.
Router# **show policy-map** *policy-map* **class** *class-name*	Displays the configuration of the specified class of the specified policy-map.
Router# **show policy-map interface** *interface-name*	Displays the configuration of all classes configured for all policy-maps on the specified interface.

Configuring LLQ

To configure LLQ, use the Modular QoS command-line interface. Define the class of traffic with the **class-map** command, create a policy-map that contains the **priority** command, and apply the policy to the appropriate interface with the **service-policy** command.

You can configure a maximum of 2 priority queue with policing on two different classes.

To configure a policy with LLQ and to assign the policy to an interface, perform the following tasks in global configuration mode:

	Command	Purpose
Step 1	Router(config)# **policy-map** *policy-name*	Specifies the name of the policy-map to configure.
Step 2	Router(config-pmap)# **class** *class-map-name*	Specifies the name of a predefined class included in the service policy.

	Command	Purpose
Step 3	Router(config-pmap-c)# **police** *bps* [*burst-normal*] [*burst-max*] **conform-action** *action* **exceed-action** *action* **violate-action** *action*	Specifies a maximum bandwidth usage by a traffic class through the use of a token bucket algorithm, where: • *bps*—Specifies the average rate in bits per second. Valid values are 128,000 to 200000000. • *burst-normal*—(Optional) Specifies the normal burst size in bytes. Valid values are 1000 to 31,250,000. The default normal burst size is 1500 bytes. • *burst-max*—(Optional) Specifies the excess burst size in bytes. Valid values are 1000 to 31,250,000. • *action*—Specifies the policing command (as shown in Table 3-5) for the corresponding conforming, exceeding, or violating traffic actions. Ensure that **confirm** *action* is always set to *transmit* and **exceed** and **violate** *action* is set to *drop*.
Step 4	`Router(config-pmap-c)# priority`	Gives strict priority to a class of traffic belonging to the policy-map.
Step 5	`Router(config)# interface interface-name`	Specifies the interface to which the policy-map will be applied.
Step 6	`Router(config-if)# service-policy output policy-name`	Attaches the specified policy-map to the interface.

Examples

The following example configures an output LLQ policy on a switchport interface that matches on a CoS value queuing defined in the classes.

```
Router(config)# policy-map switchport-llq-policy
Router(config-pmap)# class cos0
Router(config-pmap-c)# police 500000000
Router(config-pmap-c)# priority
```

Now the policy is applied to the interface.

```
Router(config)# interface TenGigabitEthernet9/0/0
Router(config-if)# switchport
Router(config-if)# switchport access vlan 2000
Router(config-if)# switchport mode access
Router(config-if)# mls QoS trust cos
Router(config-if)# service-policy output switchport-llq-policy
```

Configuring PFC QoS on a Cisco 7600 Series Ethernet Services 20G Line Card

The ES20 line card supports most of the same QoS features as those supported by the Policy Feature Card (PFC) on the Cisco 7600 series routers.

This section describes those QoS features that have ES20 line card-specific configuration guidelines. After you review the ES20 line card-specific guidelines described in this document, then refer to the *Cisco 7600 Series Cisco IOS Software Configuration Guide, 12.2SR* located at the following URL:

http://www.cisco.com/en/US/docs/routers/7600/ios/12.2SR/configuration/guide/swcg.html

PFC QoS on a Cisco 7600 Series Ethernet Services 20G Line Card Configuration Guidelines

- Output policing is not supported.
- PFC egress marking is not supported.
- Ingress marking and policing is performed with PFC Qos.

Configuring Hierarchical QoS

Table 3-12 provides information about where the hierarchical QoS features are supported.

Table 3-12 Hierarchical QoS Feature Support

Feature	Supported Interfaces	ES20 Line Card
Hierarchical QoS for EoMPLS VCs (parent policy)		Supported at egress only using **match iput vlan** command in parent policy.
Hierarchical QoS—Tiered policy-maps with parent policy using class-default only on the subinterface.	Subinterface Maininterface	Parent supports class-default only in egress direction.
Hierarchical QoS—Tiered policy-maps with parent policy in user-defined match vlan on the maininterface.	Main switchport interface	Parent supports match vlan only in egress direction. Hierarchical class-default is not supported.
Hierarchical QoS with Ethernet MPB (service instance)	EVC	Parent supports **class-default**, **match vlan**, and **match vlan inner** commands in egress direction.

Hybrid Policy Support

The following hybrid policy is not supported on an ES20 line card.

```
LLB#sh policy-map unsupp
Policy Map unsupp
Class voip
Average Rate Traffic Shaping
cir 3000000 (bps)
service-policy child1
Class video
Average Rate Traffic Shaping
cir 2000000 (bps)
service-policy child2
Class class-default
Average Rate Traffic Shaping
```

```
cir 5000000 (bps)
service-policy child3

LLB#sh policy-map child1
Policy Map child1
Class prec3
Average Rate Traffic Shaping
cir 1500000 (bps)
Class prec4
Average Rate Traffic Shaping
cir 1000000 (bps)
LLB#sh policy-map child2
Policy Map child2
Class prec3
Average Rate Traffic Shaping
cir 1000000 (bps)
Class prec4
Average Rate Traffic Shaping
cir 1500000 (bps)
LLB#sh policy-map child3
Policy Map child3
Class prec3
Average Rate Traffic Shaping
cir 1500000 (bps)
Class prec4
Average Rate Traffic Shaping
cir 1000000 (bps)
LLB# sh policy-map unsupp
Policy Map unsupp
Class voip
Average Rate Traffic Shaping
cir 3000000 (bps)
Class video
Average Rate Traffic Shaping
cir 2000000 (bps)
Class class-default
Average Rate Traffic Shaping
cir 5000000 (bps)
service-policy child3

LLB#sh policy-map child3
Policy Map child3
Class prec3
Average Rate Traffic Shaping
cir 1500000 (bps)
Class prec4
Average Rate Traffic Shaping
cir 1000000 (bps)
```

The following configuration is supported on the maininterfaces, switchport, and EVCs:

```
LLB#sh policy-map supp
Policy Map supp
Class voip
Average Rate Traffic Shaping
cir 3000000 (bps)
service-policy child1
Class video
Average Rate Traffic Shaping
cir 2000000 (bps)
service-policy child2
Class class-default
Average Rate Traffic Shaping
```

```
cir 5000000 (bps)

LLB#sh policy-map child1
Policy Map child1
Class prec3
Average Rate Traffic Shaping
cir 1500000 (bps)
Class prec4
Average Rate Traffic Shaping
cir 1000000 (bps)

LLB#sh policy-map child2
Policy Map child2
Class prec3
Average Rate Traffic Shaping
cir 1000000 (bps)
Class prec4
Average Rate Traffic Shaping
cir 1500000 (bps)

LLB#sh policy-map child3
Policy Map child3
Class prec3
Average Rate Traffic Shaping
cir 1500000 (bps)
Class prec4
Average Rate Traffic Shaping
cir 1000000 (bps)
```

Configuring Hierarchical QoS with Tiered Policy-Maps

Hierarchical QoS with tiered policy-maps is a configuration where the actions associated with a class contain a queuing action (such as shaping) and a nested service policy, which in itself is a policy-map with classes and actions. This hierarchy of the QoS policy-map is then translated into a corresponding hierarchy of queues.

Configuration Guidelines

When configuring hierarchical QoS with tiered policy-maps on an ES20 line card, follow these restrictions and usage guidelines:

- You can configure up to two levels of hierarchy within the policy-maps.
- The parent policy-map has the following restrictions on a main switchport interface:
  - Supports **match vlan** command in parent class and **match cos** command in the child class. You can configure shape or bandwidth queueing actions in any class (user-defined and class-default).
- When configuring hierarchical QoS for software-based EoMPLS, if you configure **match input vlan** in the parent policy, then you can only configure **match QoS-group** in the child policy.
- ES20 line card does not support a flat **match QoS-group** policy.
- In hierarchical QoS, the **set** command is not supported in the parent policy.
- In egress direction, the **set** command works if at least one queueing action is configured in the policy. This queueing action can also exist in the parent policy while child policies contain only marking.
- The child policy-map supports shape, bandwidth, priority, set operations, and WRED QoS features.

- With hierarchical QoS on a subinterface, the parent policy-map supports hierarchical QoS using only the **shape average** command as a queueing action in the default class (class-default) only.
- If you configure shaping at both the parent policy and the child policy, the traffic is shaped first according to the parameters defined in the parent policy, followed by the parameters of the child policy.
- Use a hierarchical policy if you want to achieve minimum bandwidth guarantees using CBWFQ with a map class. First, configure a parent policy to shape to the total bandwidth required. Then, define a child policy using CBWFQ for the minimum bandwidth percentages.
- ES20 supports class-default hQoS policy-maps. A user-defined class-map in the parent policy is not supported on subinterfaces. The following example shows an unsupported policy-map on a sub-interface:

```
pavement#sh policy-map unsup
Policy Map unsup
Class prec4 *** not supported
Average Rate Traffic Shaping
cir 300000000 (bps)
Class class-default
Average Rate Traffic Shaping
cir 200000000 (bps)
service-policy unsup2
pavement#sh policy-map unsup2
Policy Map unsup2
Class prec2
Average Rate Traffic Shaping
cir 50000000 (bps)
Class prec3
Average Rate Traffic Shaping
cir 100000000 (bps)
```

- You can configure hierarchical QoS up to the following limits, according to the current Cisco IOS software limits:
  - Up to 1024 class-maps
  - Up to 1024 policy-maps
  - Up to 256 unique classes in a policy-map including class-default

The following example shows configuration of hierarchical QoS that maps to two levels of hierarchical queues. The first-level policy (the parent policy) configures the aggregated data rate to be shaped to 1 Mbps for the class-default class. The second-level policy (the child policy) configures the traffic in User-A class for 40 percent of the bandwidth and traffic in User-B class for 20 percent of the bandwidth.

```
! Configure the class-maps with your matching criteria
!
Router(config)# class-map match-any User-A
Router(config-cmap)# match access-group A
Router(config-cmap)# exit
Router(config)# class-map match-any User-B
Router(config-cmap)# match access-group B
Router(config-cmap)# exit
!
```

Configure the parent policy for class-default to shape all traffic in that class and apply a second-level policy.

```
Router(config)# policy-map child
Router(config-pmap)# class User-A
Router(config-pmap-c)# bandwidth percent 40
Router(config-pmap-c)# exit
Router(config-pmap)# class User-B
Router(config-pmap-c)# bandwidth percent 20
Router(config-pmap-c)# exit
Router(config-pmap)# exit
!
Router(config)# policy-map parent
Router(config-pmap)# class class-default
Router(config-pmap-c)# shape 1000000
Router(config-pmap-c)# service-policy child
Router(config-pmap-c)# exit
Router(config-pmap)# exit
!
```

Configure the child policy to allocate different percentages of bandwidth by class.

```
!Apply the parent service policy to an output subinterface.
!
Router(config)# interface TenGigabitEthernet 2/0/0.1
Router(config-if)# encapsulation dot1q 11
Router(config-if)# service-policy output parent
```

The following example shows configuration of hierarchical QoS that maps to two levels of hierarchical queues, where the parent policy configures average traffic shaping rates on both user-defined classes as well as the class-default class, which is supported beginning in Cisco IOS Release 12.2(33)SRA. This configuration does not show the corresponding class-map configuration, which also is required to support these policy-maps.

Configure the parent policy for user-defined and class-default classes to shape traffic in those classes and apply a second-level policy.

Configure the child policy to allocate different percentages of bandwidth by class.

```
!
Router(config)# policy-map child-pm
Router(config-pmap)# class cos0
Router(config-pmap-c)# bandwidth percent 10
Router(config-pmap-c)# exit
Router(config-pmap)# class cos1
Router(config-pmap-c)# bandwidth percent 10
Router(config-pmap-c)# exit
Router(config-pmap)# exit
!
Router(config)# policy-map parent
Router(config-pmap)# class vlan100
Router(config-pmap-c)# shape average 1000000
Router(config-pmap-c)# service-policy child-pm
Router(config-pmap-c)# exit
Router(config-pmap)# class vlan200
Router(config-pmap-c)# shape average 1000000
Router(config-pmap-c)# service-policy child-pm
Router(config-pmap-c)# exit
Router(config-pmap)# class class-default
Router(config-pmap-c)# shape average 2000000
Router(config-pmap-c)# exit
Router(config-pmap)# exit
!
! Apply the parent service policy to an output interface.
!
Router(config)# interface TenGigabitEthernet 2/0/0
Router(config-if)# switchport
Router(config-if)# switchport access vlan 2000
Router(config-if)# switchport mode access
Router(config-if)# mls QoS trust cos
Router(config-if)# service-policy output switchport-cos-policy
Router(config-if)# service-policy output parent-pm
```

This example shows how to display information a class-default HQoS policy with child policy matching on various supported criteria.

```
Router# show policy-map parent2
  Policy Map parent2
    Class class-default
Router(config-pmap-c)# shape average 3000000000 12000000 12000000
      service-policy child
head# show policy-map child
  Policy Map child
    Class dscp3
shape average 100000000 400000 400000
Class prec5
     police cir 200000000 bc 6250000 be 6250000 conform-action transmit exceed-action drop
      priority
    Class acl2
shape average 50000000 200000 200000
    Class class-default
      bandwidth 10 (%)
```

Configuring Hierarchical QoS for EoMPLS VCs

The Hierarchical Quality of Service (HQoS) for EoMPLS VCs feature extends support for hierarchical, parent and child relationships in QoS policy-maps. This feature also provides EoMPLS per-VC QoS for point-to-point VCs.

The new feature adds the ability to match the virtual LAN (VLAN) IDs that were present on a packet when the packet was originally received by the router. It also supports the ability to match on a QoS group that is set to the same value of the IP precedence or 802.1P class of service (CoS) bits that are received on the incoming interface. This allows service providers to classify traffic easily for all or part of a particular EoMPLS network, as well as to preserve the customer's original differentiated services (DiffServ) QoS values.

In EoMPLS applications, the parent policy-map typically specifies the maximum or the minimum bandwidth for a group of specific VCs in an EoMPLS network. Then child policy-maps in the policy can implement a different bandwidth or perform other QoS operations (such as traffic shaping) on a subset of the selected VCs.

This feature enables service providers to provide more granular QoS services to their customers. It also gives service providers the ability to preserve customer IP precedence or CoS values in the network.

Hierarchical QoS with Service Instances

The Hierarchical Quality of Service (HQoS) with Service Instances extends support for hierarchical, parent and child relationships in QoS policy-maps. Both flat and hierarchical policies can be applied to service instances configured as Ethernet MPB, EoMPLS, local connect, or selective QinQ.

Configuration Guidelines

When configuring hierarchical QoS with Ethernet Service Instances on an ES20 line card, follow these restrictions and usage guidelines:

- You can classify on the inner VLAN tag, the outer VLAN tag, or a combination of both. You can classify on the inner VLAN tag and CoS or the outer VLAN tag and CoS. You can classify on CoS, or CoS inner or any combination of these filters.
- The following MQC actions are permitted:
  - Shaping
  - Bandwidth
  - Two priority queues per policy
  - **Set cos, set cos-inner**, **set cos cos-inner**, **set cos-inner cos**, and **set mpls exp** commands
  - **set mpls exp** is supported in ingress (for Eompls only).
  - WRED aggregate
  - Queue-limit
- For marking, you can copy or rewrite the inner cos, the outer cos, or both inner and outer cos, or your can copy the inner cos to outer cos or vice versa.

EVC Configuration Examples

This example shows ingress QoS on scalable EoMPLS.

```
Router(config)# interface gig 1/0/0
Router(config-if)# service instance 1 ethernet
Router(config-if-srv)# encapsulation dot1q 100
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
Router(config-if)# xconnect 2.2.2.2 100 pw-class vlan-xconnect
Router(config-pmap-c)# service-policy input mark-it-in
Router(config)# policy-map mark-it-in
Router(config-pmap)# class cos0
Router(config-pmap-c)# set mpls exp 5
```

In this example of a single tag VLAN configuration, because the encapsulation dot1q 10 is already classified, only the inner VLAN and CoS values are configured.

```
Router(config)# interface gig 1/0/0
Router(config-if)# service instance 1
Router(config-if-srv)# encapsulation dot1q 10 second-dot1q any
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
Router(config-if-srv)# bridge domain 200
Router(config-pmap-c)# service-policy input mark-it-in
Router(config)# policy-map mark-it-in
Router(config-pmap)# class innervlan20
Router(config-pmap-c)# set cos 0
Router(config-pmap-c)# set cos-inner 0
```

This is an example of a single tag VLAN connect ingress policy.

```
Router(config)# interface interface GigabitEthernet1/0/1
Router(config-if)# service instance 100 ethernet
Router(config-if-srv)# encapsulation dot1q 10 second-dot1q any
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
Router(config-pmap-c)# service-policy in mark-it-in
Router(config)# interface interface GigabitEthernet1/0/2
Router(config-if)# service instance 101 ethernet
Router(config-if-srv)# encapsulation dot1q 11 second-dot1q any
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
Router(config-pmap-c)# service-policy in mark-it-in
Router(config-if-srv)# connect EVC1 GigabitEthernet1/0/1 100 GigabitEthernet1/0/2 101
Router(config)# policy-map mark-it-in
Router(config-pmap)# class vlaninner20cosinner5
Router(config-pmap-c)# set cos 0

This is an example of an egress double tag VLAN connect hierarchical configuration.

Router(config)# interface interface GigabitEthernet1/0/1
Router(config-if)# service instance 100 ethernet
Router(config-if-srv)# encapsulation dot1q 10 second-dot1q 20
Router(config-if-srv)# rewrite ingress tag pop 2  symmetric
Router(config-pmap-c)# service-policy out parent-out-100
Router(config)# interface interface GigabitEthernet1/0/2
Router(config-if)# service instance 101 ethernet
Router(config-if-srv)# encapsulation dot1q 11 second-dot1q 21
Router(config-if-srv)# rewrite ingress tag pop 2 symmetric
Router(config-pmap-c)# service-policy out parent-out-101
Router(config-if-srv)# connect EVC1 GigabitEthernet/0/1 100 GigabitEthernet 1/0/2 101
Router(config)# policy-map child-out-100
Router(config-pmap)# class cos5
Router(config-pmap-c)# bandwidth percent 10
Router(config-pmap-c)# set cos 0
Router(config-pmap-c)# set cos-inner 0
Router(config)# policy-map parent-out-100
Router(config-pmap)# class class-default
Router(config-pmap-c)# shape average 10000000
Router(config-pmap-c)# service-policy child-out-100
Router(config)# policy-map child-out-101
Router(config-pmap)# class cos0
Router(config-pmap-c)# bandwidth percent 10
Router(config-pmap-c)# set cos 5
Router(config-pmap-c)# set cos-inner 5
Router(config)# policy-map parent-out-101
Router(config-pmap)# class class-default
Router(config-pmap-c)# shape average 10000000
Router(config-pmap-c)# service-policy child-out-101

This is an example of an egress double tag VLAN connect flat configuration.

Router(config)# policy-map flat-100
Router(config-pmap)# class cos5
Router(config-pmap-c)# shape average 10000000
Router(config-pmap-c)# set cos 0
Router(config-pmap-c)# set cos-inner 0
Router(config-pmap)# class class-default  <-- required class
Router(config-pmap-c)# shape average 10000000 <-- required queuing action
Router(config-pmap-c)# set cos 6
Router(config)# policy-map flat-101
Router(config-pmap)# class cos0
Router(config-pmap-c)# shape average 10000000
Router(config-pmap-c)# set cos 5
Router(config-pmap-c)# set cos-inner 5
Router(config-pmap)# class class-default  <-- required class
Router(config-pmap-c)# shape average 10000000 <-- required queuing action
Router(config-pmap-c)# set cos 4
```

The next examples show a selective QinQ MPB configuration and the service policy that can be applied to it. With this configuration, a range of VLANs exists in the outer tag of the **encapsulation** command. When VLAN ranges are configured, **rewrite** commands are not allowed so the packets pass through the bridge-domain without change. An inner tag cannot exist in this configuration. The first example is a hierarchical policy and the second example is a flat policy.

```
Router(config)# interface GigabitEthernet 1/0/2
Router(config-if)# service instance 1
Router(config-if-srv)# encapsulation dot1q 10-20
Router(config-if-srv)# bridge-domain 200
Router(config-pmap-c)# service-policy output parent-out
Router(config)# policy-map child-out
Router(config-pmap)# class cos5
Router(config-pmap-c)# bandwidth percent 10
Router(config-pmap-c)# set cos 0
Router(config)# policy-map parent-out
Router(config-pmap)# class outervlan10
Router(config-pmap-c)# shape average 10000000
Router(config-pmap-c)# service-policy child-out

Router(config)# interface GigabitEthernet 1/0/2
Router(config-if)# service instance 1
Router(config-if-srv)# encapsulation dot1q 10
Router(config-if-srv)# bridge-domain 200
Router(config-pmap-c)# service-policy out flat
Router(config)# policy-map flat
Router(config-pmap)# class vlanouter10cos5
Router(config-pmap-c)# shape average 10000000
Router(config-pmap-c)# set cos 0
Router(config-pmap-c)# set cos-inner 0
Router(config-pmap)# class class-default:   <-- required class
Router(config-pmap-c)# shape average 10000000 <-- required queueing action
Router(config-pmap-c)# set cos3
Router(config-pmap-c)# set cos-inner 3
```

The next two examples apply to MPBE configurations except Selective QinQ. The first is a hierarchical policy and the second is a flat policy.

```
!Sample Hier Config 1
Router(config)# interface interface GigabitEthernet 1/0/2
Router(config-if-srv)# service instance 1
Router(config-if-srv)# encapsulation dot1qencap dot1q 10 second-dot1q any
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
Router(config-if-srv)# bridge-domain 200
Router(config)# policy-map child-out
Router(config-pmap)# class vlaninner10
Router(config-pmap-c)# shape average 10000000
Router(config-pmap-c)# set cos 0
Router(config-pmap-c)# set cos-inner 0
Router(config-pmap)# class class-default:  <-- not required
Router(config)# policy-map parent-out
Router(config-pmap)# class class-default
Router(config-pmap-c)# shape average 10000000
Router(config-pmap-c)# service-policy child-out

!Sample Flat Config
Router(config)# interface interface GigabitEthernet 1/0/2
Router(config-if-srv)# service instance 1
Router(config-if-srv)# encapsulation dot1q 10 second-dot1q any
Router(config-if-srv)# rewrite ingress tag pop 1 symmetric
Router(config-if-srv)# bridge-domain 200
Router(config)# policy-map flat
Router(config-pmap)# class vlaninner10
Router(config-pmap-c)# shape average 10000000
Router(config-pmap-c)# set cos 0
Router(config-pmap-c)# set cos-inner 0
Router(config-pmap)# class class-default <-- required class
Router(config-pmap-c)# shape average 10000000 <--  required queueing action
Router(config-pmap-c)# set cos 0
Router(config-pmap-c)# set cos-inner 0
```

Configuring Bandwidth Remaining Ratio (BRR) - Utilizing Unused Bandwidth Support

BRR feature provides the functionality to utilize the unused bandwidth allocated to logical interfaces. BRR enables sharing of unused bandwidth among logical interfaces such as EVCs and L3 subinterfaces through oversubscription.

BRR is implemented on logical interfaces using the following maps:

- Hierarchical Policy-Maps
- Flat Policy- Maps

You can configure Bandwidth Remaining Ratio action in the policy-map of a parent or a child class. BRR can be configured to a minimum ratio of 1 and maximum of 1000 on a logical interface.

For more information on BRR, refer to:
http://www.cisco.com/en/US/docs/routers/7600/ios/12.2SR/configuration/guide/ipsuboe.html#wp1129803

QoS supports BRR CLI in the parent and child classes over a logical interface. You can use the **shape** command to oversubscribe the bandwidth in parent and child classes.

You cannot use the CLI to toggle the priority rate propagation to **ON** in ES20 line cards. For more information, refer to:
http://www.cisco.com/en/US/docs/routers/7600/ios/12.2SR/configuration/guide/ipsuboe.html#wp1131793

In the following BRR sample configuration:

- The bandwidth is configured in the **shape** command oversubscribed in the parent and child classes.
- The priority status of LLQ packets is propagated to the parent queue.
- The class and class-default within a parent policy-map is allocated an excess bandwidth of 250 Mb/s (25% or 1/4 of 1 Gb/s) assuming the maximum link capacity is 1 Gb/s. The excess bandwidth 150 Mb/s (250Mb/s less 100 Mb/s from LLQ class) is allocated to the child policies. The class and childclass2, receives 5/6 or 125 Mb/s as minimum bandwidth, and the class, class-default, receives default BRR of 1 or 1/6 or 25 Mb/s as minimum bandwidth.
- The parent2 policy-map is allocated an excess bandwidth of 750 Mb/s (75% or 3/4 of 1 Gb/s). The excess bandwidth of 650 Mb/s (750Mb/s less 100Mb/s from LLQ class) is allocated to the child policy. The class, childclass2, receives 5/6 or 541,666,666 Mb/s as minimum bandwidth and the class, class-default, receives default BRR of 1 or 1/6 or 108,333,333 Mb/s as minimum bandwidth.
- NonBRR policy on a maininterface: The complete 100% link bandwidth is used to calculate the cir value in all the classes. The 1% reserved bandwidth is used from the class default. Ensure that the **cir** value of a class-default policy receives more than 1% of the port bandwidth. If not, an “exceed guarantee rate” error message is displayed.
- BRR policy on a maininterface: In classes, **cir** values are calculated from 99% link bandwidth for all the classes with a BRR ratio. The 1% reservation value is not relayed from the **class default** and **cir** value is not adjusted for the class-default queue.

Note Reserved link bandwidth is the maximum bandwidth available for distribution that is 1% less than the maximum link capacity.

```
Policy-map child
```

```
   Class voip
      priority
      police 100000000
  class childclass2
    shape ave 1000000000
 bandwidth remaining ratio 5
  class class-default  # unspecified classes default to BRR of 1
    shape ave 1000000000

policy-map parent
  class class-default
    shape ave 1000000000
    bandwidth remaining ratio 1
    service-policy child
policy-map parent2
  class class-default
    shape ave 1000000000
    bandwidth remaining ratio 3
    service-policy child
int gig 4/0/0
  service inst 1 et
    encap dot1q 10-20
    bridge-domain 200
    service-policy output parent
  service inst 1 et
    encap dot1q 30-40
    bridge-domain 200
    service-policy output parent2
```

Hierarchical Policy-Maps

Hierarchical policy-maps implement parent-child relationships between various policy-maps applied on an interface. You can create a child policy-map from another policy-map to inherit its functionality. If BRR is implemented in a hierarchical policy-map for a class, all the other classes in the policy-map are automatically set to BRR with ratio one. Similarly, if BRR is implemented in a hierarchical policy-map over a logical interface, all the other hierarchical policy-maps applied to other interfaces on the same port automatically implement BRR with ratio 1.

Note BRR does not propagate to the child level. For implicit propagation of BRR to the child classes, you should configure BRR in one of the child classes.

In the following example, two policy-maps, parentbr and parentbw, are applied to two different logical interfaces on the same port. BRR is implemented only in parentbr policy-map. In the configuration below, the policy-map, parentbw, results in a default BRR of 1. Applying the policy-map, parentbw, results in an error message if the sum of the child bandwidth CIR exceeds the excess ratio allocated for parentbw. If the calculated excess minimum CIR is greater then the guaranteed rates in the child, then classes cos5 and class-default are configured with guaranteed rate plus an evenly distributed fair share of excess bandwidth. The following sample displays the BRR implementation for parentbr and parentbw policy-maps:

```
policy-map testbw
    class VOICE
      police 128000
      priority
    class cos5
      bandwidth 100000
    class class-default
```

```
      bandwidth 100000
policy-map parentbw
    class class-default
      shape ave 300000000
      service-policy testbw

policy-map testbrr
    class VOICE
      police 128000
      priority
    class cos5
      bandwidth remaining ratio 2
    class class-default
      bandwidth remaining ratio 2
policy-map parentbrr
    class class-default
      bandwidth remaining ratio 4
      shape ave 200000000
      service-policy testbrr
```

You can configure BRR on the following policy-maps:

- HQoS Policies with Bandwidth and Shape in Child Classes.
- HQoS Polices with Shape in Child Classes.
- HQoS Policies with Police+Priority (LLQ) and Shape in Child Classes.

Note You can individually or jointly configure BRR on child classes or parent classes.

HQoS Policies with Bandwidth and Shape in Child Classes

In the following sample configuration, the class-default, defined in the parent policy-map is allocated Committed Information Rate (CIR) share of 1/3 of the total available bandwidth. The total bandwidth is 1 Gb/s and available bandwidth is total bandwidth minus the reserved bandwidth (10 Mb/s). Hence, the class, class-default, is allocated 330 Mb/s (1/3 of 1 Gb/s minus the reserved bandwidth of 10 Mb/s per port, 990/3 = 330 Mb/s). As BRR is enabled on the parent, the platform calculates the CIR share for the parents.

In the following example, the minimum bandwidth that the parent requires is 200 Mb/s to accept requests from its child classes. The parent class overrides the shape value configured in the child class and uses the CIR as shape value for the child. Though the child class can receive excess bandwidth of 330 Mb/s, as the parent class is shaped with a CIR value of 300 Mb/s, the excess bandwidth of 30Mb/s is unused. The child class, cos5, is allocated a CIR value of 100 Mb/s and the maximum acceptable CIR value is 300 Mb/s (parent shaper). The class, class-default, is allocated a CIR value of 100 Mb/s.

The class, class-default, defined in parent2 policy-map is allocated 2/3 share of 990 Mb/s that is 660 Mb/s. Class cos5 is allocated 100 Mb/s and class-default is allocated the remaining share of 550 Mb/s (660-100 Mb/s).

EIR values of the user-defined classes are proportional to the CIR values. If the EIR value is high, so is the CIR value.

```
Policy-map child
Class cos5
Bandwidth 100000 kbps
class class-default
shape average 100000000
```

```
policy-map parent
class class-default
Shape average 300000000
service-policy child
Bandwidth remaining ratio 1

Policy-map parent2
class class-default
Shape average 800000000
Service-policy child
Bandwidth remaining ratio 2

int gig 4/0/0
  service inst 1 et
    encap dot1q 10-20
    bridge-domain 200
    service-policy output parent
  service inst 2 et
    encap dot1q 30-40
    bridge-domain 200
    service-policy output parent2
```

The following policy configurations exhibit similar behavior to the HQoS Policies with bandwidth and shape in the child classes:

- HQoS Polices with Shape in Child Classes, page 3-60
- HQoS Policies with Police+Priority (LLQ) and Shape in Child Classes, page 3-61.

Restrictions and Usage Guidelines

When configuring HQoS Policies with Bandwidth and Shape in the child classes, follow these restrictions and usage guidelines:

- CIR allocated to parent class should be greater than sum of child CIR. CIR is calculated using the following formula:

  ((Link rate – reserved bandwidth) * BRR) / (total BRR parts of all parent classes within the link)
- To avoid configuration failure when CIR share for a parent is reduced due to increase in parent classes, you should also configure BRR in child classes.
- You can configure BRR without shape within a class.

HQoS Polices with Shape in Child Classes

In the following sample configuration, the class, class-default, defined in the parent policy-map is allocated CIR share of 1/3 of the total available bandwidth. The total bandwidth is 1 Gb/s and available bandwidth is total bandwidth– reserved bandwidth (10 Mb/s). Hence, the class, class-default, is allocated 330 Mb/s (1/3 of 1 Gb/s– reserved bandwidth of 10 Mb/s per port = 990/3 = 330 Mb/s). With BRR enabled on the parent, the platform calculates the CIR share.

In the following example, the excess bandwidth is divided between the child class, cos5 and class-default, in bandwidth ratio of 2:1. The class, class-default, is configured to BRR of one. Hence, the class, cos5, is allocated 220 Mb/s and the class, class-default, is allocated 110 Mb/s of bandwidth.

The class, class-default, in parent2 policy-map is allocated excess bandwidth of 660 Mb/s, which is further divided between the child classes in the parent2 policy-map. The class, cos5, is allocated 440Mb/s (2/3) and the class, class-default, is allocated 220Mb/s (1/3). The following example configuration shows the HQoS polices with only shape in child classes:

```
policy-map child
class cos5
shape average 100000000
Bandwidth Remaining Ratio 2
class class-default
shape avergae 100000000

policy-map parent
class class-default
shape average 300000000
service-policy child
bandwidth remaining ratio 1

policy-map parent2
class class-default
shape average 300000000
service-policy child
bandwidth remaining ratio 2

int gig 4/0/0
  service inst 1 et
    encap dot1q 10-20
    bridge-domain 200
    service-policy output parent
  service inst 2 et
    encap dot1q 30-40
    bridge-domain 200
    service-policy output parent2
```

Restrictions and Usage Guideline

When configuring HQoS polices with only shape in child classes, follow this restriction and usage guideline:

- You can configure BRR in both the child and parent classes.

HQoS Policies with Police+Priority (LLQ) and Shape in Child Classes

In the following sample configuration, the class, class-default, in the parent policy-map is allocated CIR share of 1/3 of the total available bandwidth. The total bandwidth is 1 Gb/s and available bandwidth is total bandwidth– reserved bandwidth (10 Mb/s). Hence, the class, class-default, is allocated 330 Mb/s (1/3 of 1 Gb/s– reserved bandwidth of 10 Mb/s per port = 990/3 = 330 Mb/s). With BRR enabled on the parent, the platform calculates the CIR share.

The excess bandwidth is allocated to the child classes. The priority class, cos3, is allocated CIR equal to 100Mb/s. The remaining excess bandwidth of 230 Mb/s (330 Mb/s-100 Mb/s) is allocated to the classes, cos5 and class-default. The class cos5 is allocated 153 Mb/s and class-default is allocated 76 Mb/s.

Similarly, Parent2 policy-map is configured to CIR of 660 Mb/s (2/3 of total available bandwidth of 1 Gb/s). The LLQ class is allocated 100 Mb/s. The remaining bandwidth of 550 Mb/s is divided between the two classes, cos5 and class-default, in the ratio of 2:1. The class, cos5, is allocated 373 Mb/s and the class, class-default, is allocated 186 Mb/s. Because the class, class-default, is configured to shape limit of 100 Mb/s, the excess bandwidth is not utilized. The sample configuration shows HQoS Policies with Police+Priority(LLQ) and Shape in the child classes:

```
Policy-map child
Class cos3
Police cir 100000000
Priority
Class cos5
Bandwidth remaining ratio 2
class class-default
Bandwidth remaining ratio 1
shape average 100000000

policy-map parent
Shape average 300000000
service-policy child
Bandwidth remaining ratio 1

Policy-map parent2
Shape average 300000000
Service-policy child
Bandwidth remaining ratio 2

int gig 4/0/0
service inst 1 et
encap dot1q 10-20
bridge-domain 200
service-policy output parent
service inst 2 et
encap dot1q 30-40
bridge-domain 200
service-policy output parent2
```

Restrictions and Usage Guideline

When configuring HQoS hierarchical policies with Police+Priority(LLQ) and Shape in the child classes, follow this restriction and usage guideline:

- You can configure BRR in both the child and parent classes.

Flat Policy- Maps

If you implement BRR in one Flat policy-map over a logical interface, all the other Flat policy-maps applied on the other logical interface in the same port are not affected, and are allocated guaranteed bandwidth. The excess bandwidth is not available to other Flat policy-maps on the same interface.

In the following example, two policy-maps, testbrr and testbw, are applied on two different logical interfaces on the same port. BRR is implemented only in testbrr policy-map.

```
policy-map testbw
    class cos5
      shape ave 100000000
    class class-default
      shape ave 100000000
policy-map testbrr
    class cos5
      bandwidth remaining ratio 2
    class class-default
      bandwidth remaining ratio
```

Note You cannot simultaneously configure bandwidth and BRR in the child classes.

BRR Propagation

When you apply BRR on a:

- Hierarchical Policy—All other policies on the same logical interface are configured to BRR value of 1.
- Flat policy—
  - All other policies on the same logical interface are not configured to default BRR settings (refers to an implicit BRR value of 1).
  - When BRR is configured in one or more classes within the flat policy-map, classes within the same flat policy-map has a default BRR value of 1.
- Flat policy coexisting with HQoS policies —BRR on flat policy-maps does not propagate the default BRR settings to HQoS policy-maps and vice versa.

Restrictions and Usage Guidelines

When you configure BRR on a logical interface, follow these restrictions and usage guidelines:

- You can implement BRR on a logical and maininterface.
- Bandwidth CIR is blocked in the parent policies that are applied to logical interfaces.
- You cannot combine BRR and bandwidth CIR in a policy-map, class-map, or LLQ action.
- The line card automatically configures the queue limit when BRR is configured. No explicit queue-limit statements are supported.
- You can propagate BRR only if the same policy-map is applied across multiple interfaces within a single port.
- The sum of the parent shape rates cannot exceed the link bandwidth if BRR is not configured. The shape sum can exceed the link bandwidth on an interface only if the BRR is available in the parent policy.
- PRP state is set to 'OFF'.
- Hybrid policies are not supported with BRR. For more information on hybrid policies, refer section "Example of a Hybrid Policy" section on page 3-63.

Note If the calculated excess cir is greater then the shape rate, the shape rate is used instead of the calculated BRR minimum rate. If the shape rate is modified, the guaranteed rate and shape rate for the specific queue is updated.

Example of a Hybrid Policy

The following example displays a sample configuration of a BRR on a hybrid policy:

```
Policy Map hybrid
    Class vlan-range
      Average Rate Traffic Shaping
      cir 100000000 (bps)
      service-policy child
    Class class-default
      Average Rate Traffic Shaping
      cir 200000000 (bps)

  Policy Map child
    Class cos5
      Average Rate Traffic Shaping
      cir 1000000 (bps)
    Class cos6
      Average Rate Traffic Shaping
      cir 10000000 (bps)
          Class class-default
      Average Rate Traffic Shaping
cir 1000000 (bps)
```

Hardware Restrictions

When you configure BRR on a logical interface, follow these restrictions and usage guidelines:

- If the sum of LLQ and non LLQ bandwidth per class is:
  - More than the CIR share calculated through BRR, the line card rejects the policy-map.
  - Is less than the allocated BRR, excess bandwidth is not split among the remaining subscribers based on the configured ratio.
- Excess unused bandwidth is distributed to active oversubscribed queues but not in the configured BRR ratio.

Troubleshooting QoS Features in a ES20 Line Card

Table 3-13 lists some of the QoS troubleshooting scenarios in a ES20 line card.

Table 3-13 QoS troubleshooting scenarios

Problem	Solution
Ingress Policer issues	One rate two color (1R2C) Policer issues: Ensure that: • You configure mls QoS for the EARL policing to work. • You did not configure 1R2C on user defined classes. • You did not configure policer in a child class on the EVC. • You did not configure the police actions. • If the issue persists, contact TAC. Two rate threee color (2R3C) Policer issues: Ensure that: • You created a single 1r0c/2r3c policer per EVC/SG per bay irrespective of the number of member links on that bay and at least one member link is part of EVC PC. • If the issue persists, contact TAC.
Queue limit and WRED issues	Ensure that: • You configure only four WRED statements in a class-map including the default WRED statement, i.e. *random-detect aggregate*. • You did not exceed the acceptable limit of 120 different queue-limits. • You did not exceed the acceptable limit of 120 WRED classes when each class uses different WRED statements. If two classes have the same WRED configurations, then they would share the same WRED group. • If the issue persists, contact TAC.

Problem	Solution
Traffic drops resulting in buffer space issues while using the BRR feature.	• Ensure that you correctly configured the **queue-limit** <*x*> value for each child class queue. • To check buffer exhaustion issues, execute the **show policy-map int** command and confirm the value of the *queue depth* parameter in the output. If the queue does not receive the guaranteed bandwidth, and the queue depth is close to zero or much less than its queue limit, traffic drops and buffer space issues occur. • If the issue persists, contact TAC.
EXCEEDGUARANTEE rate error message (BRR feature) is displayed	• Check if the class-default of the policy map has received less than one percent of of port bandwidth as its CIR value. • If the issue persists, contact TAC.
Rate counters are not accurate	• Rate counters are updated in fixed intervals. Use the **load-interval** *seconds* command to increase the load interval for accurate rate counter values.

Problem	Solution
Traffic classified incorrectly	• Use the **show run class-map** command to check the class-map definition. • Use the **show policy-map interface** *interface* command to check the classification statistics. • Use the **show tcam interface interface qos type1/type2 ip** *detail* (type1 is for input policy, type2 for output policy) command to verify that the classification hardware parameters are configured correctly and packets are relayed to the right class as shown in the example:

```
Router#sh tcam interface gig10/1 qos type1
ip detail
* Global Defaults shared
DPort - Destination Port   SPort - Source
Port        TCP-F - U -URG            Pro
- Protocol
I      - Inverted LOU       TOS   - TOS
Value                - A -ACK
rtr   - Router
MRFM  - M -MPLS Packet     TN    - T -Tcp
Control           - P -PSH            COD
- C -Bank Care Flag
      - R -Recirc. Flag          - N
-Non-cachable        - R -RST
- I -OrdIndep. Flag
      - F -Fragment Flag   CAP   - Capture
Flag             - S -SYN
- D -Dynamic Flag
      - M -More Fragments  F-P   -
FlowMask-Prior.         - F -FIN
T      - V(Value)/M(Mask)/R(Result)
X      - XTAG               (*)   - Bank
Priority
Interface: 1018   label: 513   lookup_type:
1
protocol: IP   packet-type: 0
|T|Index|  Dest Ip Addr | Source Ip Addr|
DPort     |     SPort     | TCP-F
|Pro|MRFM|X|TOS|TN|COD|F-P|
V 36828        0.0.0.0        0.0.0.0
P=0             P=0        ------   0 ----
0   0 -- --- 0-0  <-
 M 36836       0.0.0.0       0.0.0.0
0              0       ------   0 X--- 0
0                <-
 R rslt: 142811A8           <-

 V 36829        0.0.0.0        0.0.0.0
P=0             P=0        ------   0 M---
0   0 -- --- 0-0
 M 36836       0.0.0.0       0.0.0.0
0              0       ------   0 X--- 0
0
```

Problem	Solution
	`R rslt: 142811A8` **Note** **<-** indicates the class where the packets are classified.

Excessive drops in packets	• Use the **show policy map interface** command to check if the offered rate exceeds the configured policer shape rate.
	• Queuing on ES20/SIP-600 is performed on the L1 frame with an overhead of 24 bytes. The ES20 supports user-defined overhead accounting for shape/wfq classes.
	• Drops also occur due to low queue-limit configured. Increase the queue-limits value if unaccounted drops are seen.
	• In case of excessive drops, check if WRED can be used as, illustrated in this example.
	`Router#s`**`h run policy-map queuing`**
	`Building configuration...`
	
	`Current configuration : 276 bytes`
	`!`
	`policy-map queuing`
	` class prec1`
	`  bandwidth 200000`
	` class prec2`
	`  shape average 100000000`
	`  random-detect`
	`  random-detect precedence 1 100 1000`
	`  random-detect precedence 2 150 1500`
	`  random-detect precedence 3 200 800`
	` class class-default`
	`  shape average 100000000`
	`!`
	`end`

<table>
<tr><th>Problem</th><th>Solution</th></tr>
<tr><td></td><td>

```
Router#
Router#sh pol
Router#sh policy-map int gig10/6
 GigabitEthernet10/6
  Service-policy output: queuing
Counters last updated 00:00:27 ago

    Class-map: prec1 (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop
rate 0000 bps
      Match: ip precedence 1
      Queueing
      queue limit 65536 packets
      (queue depth/total drops/no-buffer
drops) 0/0/0    <<< drops due to bandwidth
over subscription
      (pkts output/bytes output) 0/0
      bandwidth 200000 kbps        <<<<
bandwidth configured in the policy-map

    Class-map: prec2 (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop
rate 0000 bps
      Match: ip precedence 2
      Match:  precedence 2
      Queueing
      queue limit 32768 packets
      (queue depth/total drops/no-buffer
drops) 0/0/0        <<< drops due to shaper
      (pkts output/bytes output) 0/0
      shape (average) cir 100000000, bc
400000, be 400000
      target shape rate 100000000
<<<  shaper configured in the policy-map by
the user
```

</td></tr>
</table>

Problem	Solution

```
Exp-weight-constant: 9 (1/512)
        Mean queue depth: 0 packets
        class      Random drop      Tail
drop          Minimum        Maximum
Mark
            pkts/bytes    pkts/bytes
thresh         thresh     prob

       0          3457/5687000
0/0             8192        16384  1/10
<<< wred packet/bytes counts and threshold
values
      1        0/0          0/0
100          1000  1/10
      2        0/0          0/0
150          1500  1/10
       3          1408/4508780
0/0               200           800  1/10
      4        0/0          0/0
12288         16384  1/10
      5        0/0          0/0
13312         16384  1/10
      6        0/0          0/0
14336         16384  1/10
      7        0/0          0/0
15360         16384  1/10

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop
rate 0000 bps
      Match: any
      Queueing
      queue limit 32768 packets
      (queue depth/total drops/no-buffer
drops) 0/0/0
      (pkts output/bytes output) 0/0
      shape (average) cir 100000000, bc
400000, be 400000
      target shape rate 100000000 <<<
shaper configured in the policy-map  by the
user
```

Problem	Solution
Bandwidth not met	Check if the queue limits are configured right. Based on this example, check if the bandwidth and priority class has been configured correctly.

```
Router#sh run policy-map queuing
Building configuration...

Current configuration : 276 bytes
!
policy-map queuing
 class prec1
  bandwidth 200000
 class prec2
  shape average 100000000
  random-detect
  random-detect precedence 1 100 1000
  random-detect precedence 2 150 1500
  random-detect precedence 3 200 800
 class class-default
  shape average 100000000
!
end

Router#

Router#sh pol
Router#sh policy-map int gig10/6
 GigabitEthernet10/6

  Service-policy output: queuing

  Counters last updated 00:00:27 ago

    Class-map: prec1 (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop
rate 0000 bps
      Match: ip precedence 1
      Queueing
      queue limit 65536 packets
      (queue depth/total drops/no-buffer
drops) 0/0/0    <<< drops due to bandwidth
over subscription
      (pkts output/bytes output) 0/0
      bandwidth 200000 kbps        <<<<
bandwidth configured in the policy-map
```

Problem	Solution
Debug QoS policing traffic issues in EARL	• Check the policer configured on the hardware. For aggregate policer and microflow policer, check the *aggregate Id* value using the **sh mls qos [ip|mpls|ipv6|arp]** command. If the value is 0 or n/a, it indicates a failure. • Use the **show tcam interface** command to check whether or not the TCAM is programmed correctly for the interface. The result from the output can be used to find different fields in the QoS hardware on that interface. **Warning** **Use the show tcam interface command with the module option to view the tcam programming on the DFCs.** • Check the rate and burst configured on the policer. • Use embedded logic analyzer (ELAM) tool to gather (captures the packets routed internally, checks dbus and rbus. dbus contains the packet details which is going from the line card to router and rbus has the packet details from router to line card) packet informations. Share the output information with TAC for further troubleshooting.
Policer not receiving the packets	• Use the **sh mls qos [ip|mpls|ipv6|arp]** and **sh policy-map interface** commands to confirm if the policer receives the packets. If not, share the output information with TAC to troubleshoot line card issues.
Incorrect QoS ACLs in TCAM	• Use the **sh qm int xxx** and **sh tcam int xx qos [type1|type2] [ip|mpls|ipv6|other|arp] det** commands to verify if the correct QoS ACLs are displayed in the TCAM. If not, Share the output information with TAC for further troubleshooting.

<table>
<tr><th>Problem</th><th>Solution</th></tr>
<tr><td>Microflow policing issues</td><td>

- Use the **sh fm int xxx** and **sh mls netflow ip detail** commands to view the output. Share the information with TAC for troubleshooting.
- Validation of microflow policing: Use the **show mls netflow ip qos module** *module* command to validate the microflow policing. In this output, *Pkts/Bytes* indicates total forwarded packets/bytes, and the police count column indicates the drop count.

<pre>
Router#sh mls netflow ip qos module 10
Displaying Netflow entries in module 10
DstIP   SrcIP
Prot:SrcPort:DstPort Src i/f
:AdjPtr

Pkts   Bytes   LastSeen   QoS
PoliceCount Threshold   Leak

Drop Bucket

20.1.1.2   10.1.1.2   255 :0
:0   --   0x0
12857116   591427336   17:35:40   0x80
5117484352   0   0
NO   3145792
</pre>
</td></tr>
<tr><td>Expected CIR/PIR rate not reached</td><td>

1. TCP traffic displays rates below the CIR due to the slow-start algorithms and retransmissions.
2. To increase the CIR/PIR rates, use a traffic generator or UDP traffic .
3. Use large burst values to police TCP traffic.

</td></tr>
<tr><td>Egress packet drop issues</td><td>

- Check the traffic type.
- Raise the bandwidth. Eg: old 10M users – gig link in a 10 gig backbone network.
- Modify the queue limit or introduce WRED.

</td></tr>
</table>

CHAPTER 4

Troubleshooting the Cisco 7600 Series Ethernet Services 20G Line Card

This chapter describes techniques that you can use to troubleshoot the operation of your Cisco 7600 Series Ethernet Services 20G (ES20) line card.

It includes the following sections:

- General Troubleshooting Information, page 4-1
- Using the Cisco IOS Event Tracer to Troubleshoot Problems, page 4-2
- Troubleshooting SFP Issues, page 4-3
- Preparing for Online Insertion and Removal of Cisco 7600 Series Ethernet Services 20G Line Card, page 4-3
- Line Card Online Diagnostics, page 4-7
- Onboard Failure Logging, page 4-7

The first section provides information about basic interface troubleshooting. If you are having a problem with your SFP and XFP modules, use the steps in the "Using the Cisco IOS Event Tracer to Troubleshoot Problems" section to begin your investigation of a possible interface configuration problem.

General Troubleshooting Information

This section describes general information for troubleshooting the ES20 line card. It includes the following sections:

- Interpreting Console Error Messages, page 4-1
- Using debug Commands, page 4-2
- Using show Commands, page 4-2

Interpreting Console Error Messages

To view the explanations and recommended actions for Cisco 7600 series router error messages, including messages related to Cisco 7600 series router ES20 line card, refer to the *Cisco 7600 Series Cisco IOS System Message Guide, 12.2SR* at
http://www.cisco.com/en/US/docs/ios/system/messages/guide/consol_smg.html

System error messages are organized in the documentation according to the particular system facility that produces the messages. The ES20 line card error messages use the following facility names:

- 7600-ES20-2X10G—ESM20
- 7600-ES20-20XG—ESM20

Using debug Commands

Along with the other **debug** commands supported on the Cisco 7600 series router, you can obtain specific debug information for the ES20 line card on the Cisco 7600 series router using the **debug hw-module** privileged EXEC command.

The **debug hw-module** command is intended for use by Cisco technical support personnel.

Caution Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use **debug** commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco technical support staff. Moreover, it is best to use **debug** commands during periods of lower network traffic and fewer users. Debugging during these periods decreases the likelihood that increased **debug** command processing overhead will affect system use.

For information about other **debug** commands supported on the Cisco 7600 series routers, refer to the *Cisco IOS Debug Command Reference, Release 12.2 SR* at http://www.cisco.com/en/US/docs/ios/12_2/debug/command/reference/dbfser.html

Using show Commands

There are several **show** commands that you can use to monitor and troubleshoot the ES20 line card on the Cisco 7600 series routers. See Chapter A, "Command Summary for the Cisco 7600 Series Ethernet Services 20G Line Card".

Using the Cisco IOS Event Tracer to Troubleshoot Problems

Note The Event Tracer feature is intended for use as a software diagnostic tool and should be configured only under the direction of a Cisco Technical Assistance Center (TAC) representative.

The Event Tracer feature provides a binary trace facility for troubleshooting Cisco IOS software. This feature gives Cisco service representatives additional insight into the operation of the Cisco IOS software and can be useful in helping to diagnose problems in the unlikely event of an operating system malfunction or, in the case of redundant systems, Route Processor switchover.

Event tracing works by reading informational messages from specific Cisco IOS software subsystem components that have been preprogrammed to work with event tracing, and by logging messages from those components into system memory. Trace messages stored in memory can be displayed on the screen or saved to a file for later analysis.

For more information about using the Event Tracer feature, refer to the following URL:

www.cisco.com/en/US/docs/ios/netmgmt/configuration/guide/nm_event_tracer_ps6441_TSD_Products_Configuration_Guide_Chapter.html

Troubleshooting SFP Issues

Use the following steps when troubleshooting an small form-factor pluggable (SFP) issue:

Note The ES20 line card uses a slot/bay/port numbering scheme. The *slot* refers to whichever slot the line card occupies in the router. The *bay* number is always 0. The *port* number is either 0 or 1 in the 2-port card or 0 through 19 in the 20-port card.

Step 1 Connect to the card with the problematic SFP module using the **execute-on** command:

```
Router# execute-on 1 show tech
```

Step 2 Use the **test sfp** command to display information about the problematic SFP module on line card slot 1, port 0.

```
# test interfaces gigabitEthernet 1/0/0
```

Step 3 Use the debug commands to provide additional information. Use the following command to debug all interfaces:

```
# debug ethernet-interface
```

Use the following command to debug a specific interface in slot 1, port 0:

```
# debug interface gigabitEthernet 1/0/0
```

Step 4 Use the show controller command to view additional information:

```
Router# show controller gigabitethernet 2 1 sfp
SFP disabled or link problems (0x32)
```

Preparing for Online Insertion and Removal of Cisco 7600 Series Ethernet Services 20G Line Card

The Cisco 7600 series router supports online insertion and removal (OIR) of the ES20 line card, in addition to each of the small form-factor pluggable (SFP or XFP) optical transceivers.

Therefore, you can remove an ES20 line card with its optical transceivers still intact, or you can remove an optical transceiver independently from the ES20 line card, leaving the ES20 line card installed in the router.

This section includes the following topics on OIR support:

- Preparing for Online Removal of a Cisco 7600 Series Ethernet Services 20G Line Card, page 4-4
- Verifying Deactivation and Activation of a Cisco 7600 Series Ethernet Services 20G Line Card, page 4-5
- Deactivation and Activation Configuration Examples, page 4-6

Preparing for Online Removal of a Cisco 7600 Series Ethernet Services 20G Line Card

The Cisco 7600 series router supports OIR of the ES20 line card. To do this, you can power down an ES20 line card (which automatically deactivates any installed optical transceivers) and remove the ES20 line card still intact.

Although graceful deactivation of an ES20 line card is preferred using the **no power enable module** command, the Cisco 7600 series router does support removal of the ES20 line card without deactivating it first. If you plan to remove an ES20 line card, you can deactivate the ES20 line card first, using the **no power enable module** global configuration command. When you deactivate an ES20 line card using this command, it automatically deactivates each of the optical transceivers that are installed in that ES20 line card. Therefore, it is not necessary to deactivate each of the optical transceivers prior to deactivating the ES20 line card.

Either a blank filler plate or a functional optical transceiver should reside in every subslot of an ES20 line card during normal operation.

For more information about the recommended procedures for physical removal of the ES20 line card, refer to the *Cisco 7600 Series Ethernet Services 20G Line Card Hardware Installation Guide*.

Deactivating a Cisco 7600 Series Ethernet Services 20G Line Card

To deactivate an ES20 line card and its installed optical transceivers prior to removal of the line card, use the following command in global configuration mode:

Command	Purpose
Router(config)# **no power enable module** *slot*	Shuts down any installed interfaces, and deactivates the ES20 line card in the specified slot, where: • *slot*—Specifies the chassis slot number where the line card is installed.

For information on how to specify the physical locations of an ES20 line card on the Cisco 7600 series routers, see Identifying Slots and Subslots for the Cisco 7600 Series Ethernet Services 20G Line Card, page 2-3.

Reactivating a Cisco 7600 Series Ethernet Services 20G Line Card

Once you deactivate an ES20 line card, whether or not you have performed an OIR, you must use the **power enable module** global configuration command to reactivate the ES20 line card.

If you did not issue a command to deactivate the optical transceivers installed in an ES20 line card, but you did deactivate the ES20 line card using the **no power enable module** command, then you do not need to reactivate the optical transceivers after an OIR of the ES20 line card. The installed optical transceivers automatically reactivate upon reactivation of the ES20 line card in the router.

For example, consider the case in which you remove an ES20 line card from the router to replace it with another ES20 line card. You reinstall the same optical transceivers into the new ES20 line card. When you enter the **power enable module** command on the router, the optical transceivers will automatically reactivate with the new ES20 line card.

To activate an ES20 line card and its installed optical transceivers after the ES20 line card has been deactivated, use the following command in global configuration mode:

Command	Purpose
Router(config)# **power enable module** *slot*	Activates the ES20 line card in the specified slot and its installed optical transceivers, where: • *slot*—Specifies the chassis slot number where the ES20 line card is installed.

For information on how to specify the physical locations of an ES20 line card on the Cisco 7600 series routers, see Identifying Slots and Subslots for the Cisco 7600 Series Ethernet Services 20G Line Card, page 2-3.

Verifying Deactivation and Activation of a Cisco 7600 Series Ethernet Services 20G Line Card

To verify the deactivation of an ES20 line card, enter the **show module** command in privileged EXEC configuration mode. Observe the Status field associated with the ES20 line card that you want to verify.

The following example shows that the ES20 line card located in slot 2 is deactivated. This is indicated by its “PwrDown” status.

```
Router# show module 2
Mod Ports Card Type                              Model              Serial No.
--- ----- -------------------------------------- ------------------ -----------
  2    0  ESM20G                                 7600-ES20-BASE     JAB1030007C

Mod MAC addresses                       Hw    Fw           Sw           Status
--- ---------------------------------- ------ ------------ ------------ -------
  2  00e0.aabb.cc00 to 00e0.aabb.cc00   1.0   12.2(2006032 12.2(2006110 PwrDown

Mod  Sub-Module                  Model              Serial       Hw     Status
---- --------------------------- ------------------ ----------- ------- -------
  2  ESM20G/PFC3C Distributed Fo 7600-ES20-D3C      JAB1030008H  1.0     PwrDown

Mod  Online Diag Status
---- -------------------
  2  Not Applicable
Router#
```

To verify activation and proper operation of an ES20 line card, enter the **show module** command and observe “Ok” in the Status field as shown in the following example:

```
Router# show module 4
Mod Ports Card Type                              Model              Serial No.
--- ----- -------------------------------------- ------------------ -----------
  4    2  ESM20G                                 7600-ES20-BASE     JAB10230687

Mod MAC addresses                       Hw    Fw           Sw           Status
--- ---------------------------------- ------ ------------ ------------ -------
  4  00e0.aabb.cc00 to 00e0.aabb.cc00   1.0   12.2(2006032 12.2(nightly Ok

Mod  Sub-Module                  Model              Serial       Hw     Status
---- --------------------------- ------------------ ----------- ------- -------
  4  ESM20G Distributed Forwardi 7600-ES20-D3CXL    JAB10230672  1.0     Ok
 4/0 1x10GE XFP Port             7600-ES20-2X10G    JAB1023069L  1.0     Ok
 4/1 1x10GE XFP Port            7600-ES20-2X10G    JAB1023069L  1.0    Ok

Mod  Online Diag Status
---- -------------------
  4  Bypass
 4/0 Bypass
 4/1 Bypass
Router#
```

For information on how to specify the physical locations of an ES20 line card on the Cisco 7600 series routers, see Identifying Slots and Subslots for the Cisco 7600 Series Ethernet Services 20G Line Card, page 2-3.

Deactivation and Activation Configuration Examples

This section provides the following examples of deactivating and activating an ES20 line card and optical transceivers:

- Deactivation of a Cisco 7600 Series Ethernet Services 20G Line Card Configuration Example, page 4-6
- Activation of a Cisco 7600 Series Ethernet Services 20G Line Card Configuration Example, page 4-6

Deactivation of a Cisco 7600 Series Ethernet Services 20G Line Card Configuration Example

Deactivate an ES20 line card when you want to perform OIR of the ES20 line card. The following example deactivates the ES20 line card that is installed in slot 5 of the router, its optical transceivers, and all of the interfaces. The corresponding console messages are shown:

```
Router# configure terminal
Router(config)# no power enable module 5
1w4d: %OIR-6-REMCARD: Card removed from slot 5, interfaces disabled
1w4d: %C6KPWR-SP-4-DISABLED: power to module in slot 5 set off (admin request)
```

Activation of a Cisco 7600 Series Ethernet Services 20G Line Card Configuration Example

Activate an ES20 line card if you have previously deactivated it. If you did not deactivate the optical transceivers, the optical transceivers automatically reactivate with reactivation of the ES20 line card.

```
The following example activates the ES20 line card that is installed in slot 5 of the
router, its optical transceivers, and all of the interfaces (as long as the hw-module
subslot shutdown command was not issued to also deactivate the optical transceivers):

Router# configure terminal
Router(config)# power enable module 5

Notice that there are no corresponding console messages shown with activation. If you
re-enter the power enable module command, a message is displayed indicating that the
module is already enabled:

Router(config)# power enable module 5
% module is already enabled
```

Line Card Online Diagnostics

Note Output from this procedure will vary slightly depending on which line card you are using, but the basic information will be the same.

Line card field diagnostic software is bundled with the main Cisco IOS software to enable you to test whether a suspect line card is faulty. For information on running online diagnostics, see Configuring Online Diagnostics in the *Cisco 7600 Series Cisco IOS Software Configuration Guide, 12.2SR* at http://www.cisco.com/en/US/docs/routers/7600/ios/12.2SR/configuration/guide/swcg.html

Onboard Failure Logging

The onboard failure logging (OBFL) feature gathers boot, environmental, and critical hardware failure data for field-replaceable units (FRUs), and stores the information in the nonvolatile memory of the FRU. This information is used for troubleshooting, testing, and diagnosis if a failure or other error occurs.

Because OBFL is on by default, data is collected and stored as soon as the card is installed. If a problem occurs, the data can provide information about historical environmental conditions, uptime, downtime, errors, and other operating conditions.

Caution OBFL is activated by default in all cards and should not be deactivated. OBFL is used to diagnose problems in FRUs and to display a history of FRU data.

For information on configuring OBFL, see *Onboard Failure Logging* at http://www.cisco.com/en/US/docs/ios/12_0s/feature/guide/12sobfl.html

CHAPTER 5

Upgrading Field-Programmable Devices

In general terms, field-programmable devices (FPDs) are hardware devices implemented on router cards that support separate upgrades. The term “FPD” has been introduced to collectively and generically describe any type of programmable hardware device on the Cisco 7600 Series ES20 line card.

This chapter describes the information that you need to know to verify image versions and to perform Cisco 7600 Series ES20 line card FPD upgrades.

For more information about the commands used in this chapter, see the *Cisco IOS Release 12.2 SR Command References at* http://www.cisco.com/en/US/products/ps6922/prod_command_reference_list.html

This chapter includes the following sections:

- FPD Quick Upgrade, page 5-1
- Overview of FPD Images and Packages, page 5-2
- Upgrading FPD Images, page 5-3
- Optional FPD Procedures, page 5-6
- FPD Image Upgrade Examples, page 5-11

FPD Quick Upgrade

This section provides information if you simply want to upgrade FPDs for Cisco 7600 Series ES20 line cards as quickly as possible. These instructions are not always feasible for operating network environments and are not the only methods available for upgrading FPDs. If these methods of upgrade are not suitable for your situation, see the various other sections of this document for other methods of upgrading FPDs.

This section addresses the following topics:

- FPD Quick Upgrade Before Upgrading your Cisco IOS Release (Recommended), page 5-2
- FPD Quick Upgrade After Upgrading your Cisco IOS Release, page 5-2

FPD Quick Upgrade Before Upgrading your Cisco IOS Release (Recommended)

Step 1 When getting your Cisco IOS image, download the FPD image package for the Cisco IOS release that you are upgrading to any Flash disk on your router before booting the new version of Cisco IOS. The FPD image package can be retrieved from the same site where you went to get your Cisco IOS image. Do not change the name of the FPD image package.

Step 2 Boot using the new version of Cisco IOS. When the new Cisco IOS boots, it by default searches for the FPD image package in the router flash file systems and the FPD images will be updated automatically as part of the IOS boot process.

FPD Quick Upgrade After Upgrading your Cisco IOS Release

Step 1 An FPD upgrade is not always necessary after Cisco IOS is reloaded. If you have already reloaded your Cisco IOS, enter the **show hw-module all fpd** command to see if all system FPDs are compatible. If the FPDs are compatible, no further action is necessary. If at least one FPD needs an upgrade, proceed to Step 2.

Step 2 Go to the cisco.com site where you downloaded your specific Cisco IOS software and locate the FPD image package, if you haven't already.

Step 3 Download this FPD image package to a Flash disk on your router. Do not change the name of the FPD image package.

Do not change any FPD-related settings on your system (if **upgrade fpd auto** or **upgrade fpd path** has been changed, change the settings back to the default settings using the **no** form of the command). Reboot your Cisco IOS release software. When the new Cisco IOS boots, it by default searches for the FPD image package in the Flash file systems and the FPD images will be updated automatically as part of the IOS boot process.

Overview of FPD Images and Packages

An FPD image package is used to upgrade FPD images. Whenever a Cisco IOS image is released that supports the Cisco 7600 Series ES20 line cards, a companion FPD image package is also released for that Cisco IOS software release. The FPD image package is available from Cisco.com and is accessible from the Cisco Software Center page where you also go to download your Cisco IOS software image.

If you are running Cisco 7600 Series ES20 line cards on your router and are upgrading your Cisco IOS image, you should download the FPD image package file before booting the router using the new Cisco IOS release. If the Cisco 7600 Series ES20 line card requires an FPD upgrade and the Cisco IOS image is unable to locate an FPD image package, the system messages will indicate that the FPD image is incompatible and you will need to go to the Cisco Software Center on Cisco.com to download the FPD image package for your Cisco IOS software release. An FPD incompatibility on a Cisco 7600 Series ES20 line card disables all interfaces on that Cisco 7600 Series ES20 line card until the incompatibility is addressed.

Note The FPD automatic upgrade feature only searches for the FPD image package file that is the same version number as the Cisco IOS release being used by the system. For example, if the Cisco IOS release being used is Cisco IOS Release 12.2(33)SRD, then the system will search for the FPD image package file that supports the specific Cisco IOS release (c7600-fpd-pkg.122-33.SRD.pkg). Therefore, ensure the FPD image package file on your system is compatible with your Cisco IOS release and do not change the name of the FPD image package file.

Upgrading FPD Images

This section documents some of the common scenarios where FPD image updates are necessary. It discusses the following scenarios:

- Migrating to a Newer Cisco IOS Release, page 5-3
- Upgrading FPD Images in a Production System, page 5-4

Migrating to a Newer Cisco IOS Release

This section discusses the following topics:

- Upgrading FPD Images Before Upgrading Cisco IOS Release (Recommended), page 5-3
- Upgrade FPD Images after Upgrading the New Cisco IOS Release, page 5-4
- Upgrading FPD Images Using Fast Software Upgrade, page 5-6

Upgrading FPD Images Before Upgrading Cisco IOS Release (Recommended)

If you are still running your old Cisco IOS Release but are preparing to load a newer version of Cisco IOS, you can upgrade FPD for the new Cisco IOS Release using the following method:

- Placing FPD Image Package on Flash Disk Before Upgrading IOS (Recommended), page 5-3

Placing FPD Image Package on Flash Disk Before Upgrading IOS (Recommended)

Placing the FPD image package for the IOS release that you are upgrading to before upgrading IOS is the recommended method for upgrading FPD because it is simple in addition to being fast. To perform this type of FPD upgrade, follow these steps:

Step 1 While still running the Cisco IOS release that will be upgraded, place the FPD image package for the new version of Cisco IOS onto one of your router’s Flash file systems. For instance, if you are running Cisco IOS Release 12.2(33)SRD and are upgrading to a newer release, place the FPD image package for the newer release onto a Flash file system while still running Cisco IOS Release 12.2(33)SRD. You can locate the FPD image package for a specific IOS release on cisco.com from the same area where you download that Cisco IOS software image. Your router and Cisco 7600 Series ES20 line cards should continue to operate normally since this action will have no impact on the current FPDs.

Caution Do not change the filename of the FPD image package file. The Cisco IOS searches for the FPD image package file by filename, so the FPD image package file cannot be found if it has been renamed.

Step 2 Reboot your router using the new upgraded Cisco IOS image. As part of the bootup process, the router will search for the FPD image package. Since the default settings for the FPD image package search are to check for the FPD image package for the specific Cisco IOS Release in a Flash file system, the FPD image package will be located during the bootup procedure and all FPDs that required upgrades will be upgraded.

Step 3 When the router has booted, verify the upgrade was successful by entering the **show hw-module all fpd** command.

Upgrade FPD Images after Upgrading the New Cisco IOS Release

The following steps explain how to upgrade FPD images if you have already upgraded your Cisco IOS release but still need to upgrade your FPD images.

To perform an FPD upgrade after the new Cisco release has been booted, follow these steps:

Step 1 If you are unsure if your FPD images for your Cisco 7600 Series ES20 line cards are compatible, enter the **show hw-module all fpd** command to verify compatibility of all Cisco 7600 Series ES20 line cards. If all of your Cisco 7600 Series ES20 line cards are compatible, there is no reason to perform this upgrade.

Step 2 If an FPD upgrade is necessary, place the FPD image package for the new version of Cisco IOS onto the router's Flash Disk or on an accessible FTP or TFTP server. You can locate the FPD image package on cisco.com from the same area where you downloaded your Cisco IOS software image.

Step 3 Enter the **upgrade hw-module [slot** *slot-number*] *file-url* **[force]** command. The *file-url* command should direct users to the location of the FPD image package. For instance, if you had placed the FPD image package for Release 12.2(33)SRD on the TFTP server abrick/muck/myfolder, you would enter **upgrade hw-module [slot** *slot-number*] **tftp://abrick/muck/myfolder/c7600-fpd-pkg.122-33.SRD.pkg** to complete this step.

If multiple Cisco 7600 Series ES20 line cards require upgrades, the different pieces of hardware will have to be updated individually.

Note the **force** option is used in this command. This option will force an FPD upgrade even if no FPD mismatch is detected. In instances where the **upgrade hw-module** command is entered, this option is almost never necessary and should only be entered if requested by a technical support representative.

Step 4 Verify the upgrade was successful by entering the **show hw-module all fpd** command.

Upgrading FPD Images in a Production System

Adding a Cisco 7600 Series ES20 line card to a production system presents the possibility that the Cisco 7600 Series ES20 line card may contain versions of FPD images that are incompatible with the Cisco IOS release currently running the router. In addition, the FPD upgrade operation can be a very CPU-intensive operation and therefore the upgrade operation may take more time when it is performed on a production system. The performance impact will vary depending on various factors, including network traffic load, the type of processing engine used, type of Cisco 7600 Series ES20 line card, and the type of service configured.

For these reasons, we recommend that one of the following alternatives be used to perform the FPD upgrade on a production system if possible:

- Using a NonProduction System to Upgrade the Cisco 7600 Series ES20 Line Card FPD Image, page 5-5
- Upgrading FPD Images Using Fast Software Upgrade, page 5-6

Using a NonProduction System to Upgrade the Cisco 7600 Series ES20 Line Card FPD Image

Before beginning the upgrade, ensure:

- The spare system is running the same version of the Cisco IOS software release that the target production system is running.
- The automatic upgrade feature is enabled on the spare system (the automatic upgrade feature is enabled by default. It can also be enabled using the **upgrade fpd auto** command).

Use the following procedure to perform an upgrade on a spare system:

Step 1 Download the FPD image package file to the router's flash file system or TFTP or FTP server accessible by the spare system. In most cases, it is preferable to place the file in a Flash file system since the router, by default, searches for the FPD image package in the Flash file systems. If the Flash file systems are full, use the **upgrade fpd path** command to direct the router to search for the FPD image package in the proper location.

Step 2 Insert the ES20 line card into the spare system.

If an upgrade is required, the system will perform the necessary FPD image updates so that when this ES20 line card is inserted to the target production system it will not trigger an FPD upgrade operation there.

Step 3 Verify the upgrade was successful by entering the **show hw-module all fpd** command.

Step 4 Remove the ES20 line card from the spare system after the upgrade.

Step 5 Insert the ES20 line card into the target production system.

Verifying System Compatibility First

If a spare system is not available to perform an upgrade, you can check for system compatibility by disabling the automatic upgrade feature before inserting the ES20 line card (the automatic upgrade feature is enabled by default. It can be disabled using the **no upgrade fpd auto** command).

- If the FPD images on the ES20 line card are compatible with the system, you will only need to reenable the automatic upgrade feature (the automatic upgrade feature can be reenabled using the **upgrade fpd auto** command).
- If the FPD images on the ES20 line card are not compatible with the system, the ES20 line card is disabled but will not impact system performance by attempting to perform an automatic upgrade.

Use the following procedure to check the FPD images on the ES20 line card for system compatibility:

Step 1 Disable the automatic upgrade feature using the **no upgrade fpd auto** global configuration command.

Step 2 Insert the ES20 line card into the system.

If the FPD images are compatible, the ES20 line card will operate successfully after bootup.

If the FPD images are not compatible, the ES20 line card is disabled. At this point we recommend that you wait for a scheduled maintenance when the system is offline to manually perform the FPD upgrade using one of the procedures outlined in the "Upgrading FPD Images" section on page 5-3.

Step 3 Reenable the automatic upgrade feature using the **upgrade fpd auto** global configuration command.

Upgrading FPD Images Using Fast Software Upgrade

The fast software upgrade (FSU) procedure supported by Route Processor Redundancy (RPR) allows you to upgrade the Cisco IOS image on supervisor engines without reloading the system.

When using FSU to upgrade the Cisco IOS image, remember that Cisco IOS software is configured, by default, to automatically load the new FPD images from a flash file system on the router. Therefore, if the FPD image package for the new Cisco IOS has not been downloaded to the router flash file system, the FPD image that needs to be upgraded will not get upgraded if the new supervisor engine with the upgraded Cisco IOS becomes the primary supervisor engine. To ensure FPD is upgraded at the time of the FSU, place the FPD image package for the new version of Cisco IOS onto the flash file system before upgrading the Cisco IOS and follow the instructions in the "Upgrading FPD Images Before Upgrading Cisco IOS Release (Recommended)" section on page 5-3.

If an ES20 line card is disabled after FSU is used to upgrade Cisco IOS and the supervisor engine with the upgraded Cisco IOS has become the primary supervisor engine, follow the instructions in the "Upgrade FPD Images after Upgrading the New Cisco IOS Release" section on page 5-4 to verify and, if necessary, upgrade FPD.

Optional FPD Procedures

This section provides information for optional FPD-related functions. None of the topics discussed in this section are necessary for completing FPD upgrades, but may be useful in some FPD-related scenarios. It covers the following topics:

- Manually Upgrading ES20 Line Card FPD Images, page 5-6
- Upgrading FPD from an FTP or TFTP Server, page 5-7
- Modifying the Default Path for the FPD Image Package File Location, page 5-8
- Displaying Current and Minimum Required FPD Image Versions, page 5-8
- Displaying Information About the Default FPD Image Package, page 5-10

Manually Upgrading ES20 Line Card FPD Images

To manually upgrade the current FPD version on an ES20 line card, use the following command:

```
Router# upgrade hw-module [slot slot-number] file file-url [force]
```

In this example, *slot-number* is the slot where the ES20 line card is installed, *file-url* is the location and name of the FPD image package file, and **force** is an option that forces the SPA to perform an FPD upgrade even if FPD is compatible (the **force** option is almost never necessary and should only be entered if requested by a technical support representative).

Caution An image upgrade can require a long period of time to complete depending on the ES20 line card.

Upgrading FPD from an FTP or TFTP Server

The generally recommended method to perform an FPD image upgrade is to download the FPD image package to a Flash file system and use the FPD automatic upgrade. By default, the system searches the Flash file system for the FPD image package file when an FPD incompatibility is detected.

This default behavior of loading an FPD image from Flash can be changed using the **upgrade fpd path** global configuration command, which sets the path to search for the FPD image package file to a location other than the router's Flash file systems.

For large deployments where all the systems are being upgraded to a specific Cisco IOS software release, we recommend that the FPD image package file be placed on an FTP or TFTP server that is accessible to all the affected systems, and then use the **upgrade fpd path** global configuration command to configure the routers to look for the FPD image package file from the FTP or TFTP server prior to the reloading of the system with the new Cisco IOS release.

Note This approach can also be used if there is not enough disk space on the system Flash card to hold the FPD image package file.

To download an FPD image package file to an FTP or TFTP server, use the following procedure:

Step 1 Copy the FPD image package file to the FTP or TFTP server.

Step 2 From global configuration mode, use the **upgrade fpd path** command to instruct the router to locate the FPD image package file from the FTP or TFTP server location.

For example, enter one of the following global configuration commands from the target system's console:

```
Router(config)# upgrade fpd path tftp://my_tftpserver/fpd_pkg_dir/
```

or

```
Router(config)# upgrade fpd path ftp://login:password@my_ftpserver/fpd_pkg_dir/
```

Note The final "/" at the end of each of the above examples is required. If the path is specified without the trailing "/" character, the command will not work properly.

In these examples, *my_tftpserver* or *my_ftpserver* is the path to server name, *fpd_pkg_dir* is the directory on the TFTP server where the FPD image package is located, and *login:password* is your FTP login name and password.

Step 3 Make sure that the FPD automatic upgrade feature is enabled by examining the output of the **show running-config** command. (Look for the *upgrade fpd auto* configuration line in the output. If there are no upgrade commands in the output, then **upgrade fpd auto** is enabled because it is the default setting.) If automatic upgrades are disabled, use the **upgrade fpd auto** global configuration command to enable automatic FPD upgrades.

Step 4 Enter the **show upgrade fpd file** command to ensure your router is connecting properly to the default FPD image package. If you are able to generate output related to the FPD image package using this command, the upgrade should work properly.

Step 5 Save the configuration and reload the system with the new Cisco IOS release.

During the system startup after the reload, the necessary FPD image version check for all the ES20 line cards will be performed and any upgrade operation will occur automatically if an upgrade is required. In each upgrade operation, the system extracts the necessary FPD images to the ES20 line card from the FPD image package file located on the FTP or TFTP server.

Modifying the Default Path for the FPD Image Package File Location

By default, the Cisco IOS software looks for the FPD image package file on a Flash file system when performing an automatic FPD image upgrade.

Note Be sure there is enough space on one of your Flash file systems to accommodate the FPD image package file.

Alternatively, you can store an FPD image package file elsewhere. However, because the system looks on the Flash file systems by default, you need to change the FPD image package file location so that the system is directed to search an alternate location (such an FTP or TFTP server) that is accessible by the Cisco IOS software. Enter the **upgrade fpd path** *fpd-pkg-dir-url* global configuration command, where *fpd-pkg-dir-url* is the alternate location, to instruct the router to search for the FPD image package elsewhere.

When specifying the *fpd-pkg-dir-url*, be aware of the following:

- The *fpd-pkg-dir-url* is the path to the FPD image package, but the FPD image package should not be specified as part of the *fpd-pkg-dir-url*. For instance, if the c7600-fpd-pkg.122-33.SRD.pkg file can be found on the TFTP server using the path mytftpserver/myname/myfpdpkg/c7600-fpd-pkg.122-33.SRD.pkg and you wanted the router to utilize this FPD image package for FPD upgrades, the **upgrade fpd path tftp://mytftpserver/myname/myfpdpkg/** command should be entered so the router knows where to find the file. The actual filename should not be specified.
- The final "/" character in the *fpd-pkg-dir-url* is required. In the preceding example, note that the *fpd-pkg-dir-url* is **tftp://mytftpserver/myname/myfpdpkg/.** Entering **tftp://mytftpserver/myname/myfpdpkg** (note: the final "/" character is missing) as the *fpd-pkg-dir-url* in that scenario would not work.

If the **upgrade fpd path** global configuration command has not been entered to direct the router to locate an FPD image package file in an alternate location, the system searches the Flash file systems on the Cisco 7600 series router for the FPD image package file.

Failure to locate an FPD image package file when an upgrade is required will disable the ES20 line card. Because ES20 line cards will not come online until FPD is compatible, the ES20 line card will also be disabled if it requires an FPD upgrade and the automatic upgrade feature is disabled.

Displaying Current and Minimum Required FPD Image Versions

To display the current version of FPD images on the ES20 line cards installed on your router, use the **show hw-module** [*slot-number* | **all**] **fpd** command, where *slot-number* is the slot number where the ES20 line card is installed. Entering the **all** keyword shows information for hardware in all router slots.

The following examples show the output when using this **show** command.

The output display in this example shows that FPD versions on the ES20 line cards in the system meet the minimum requirements:

```
Router# show hw-module all fpd

==== ====================== ====== ==============================================
                             H/W   Field Programmable   Current   Min. Required
Slot Card Type              Ver.  Device: "ID-Name"    Version      Version
==== ====================== ====== ================== ========== ==============
   1 7600-ES20-GE3CXL        1.0  1-ROMMON              1.4         1.4
                                  2-I/O FPGA            0.21        0.21
                                  3-PKT ENG FPGA        0.5         0.5
                                  5-20x1GE LINK FPGA    0.7         0.7
---- ---------------------- ------ ------------------ ----------- --------------
   4 7600-SIP-400            2.4  1-ROMMON              1.3         1.3
                                  2-I/O FPGA            0.82        0.82
                                  3-SWITCH FPGA         0.39        0.39
---- ---------------------- ------ ------------------ ----------- --------------
 4/0 SPA-2X1GE               2.2  1-GE I/O FPGA         1.10        1.10
---- ---------------------- ------ ------------------ ----------- --------------
 4/1 SPA-2X1GE               2.2  1-GE I/O FPGA         1.10        1.10
---- ---------------------- ------ ------------------ ----------- --------------
 4/2 SPA-2X1GE               2.2  1-GE I/O FPGA         1.10        1.10
---- ---------------------- ------ ------------------ ----------- --------------
   7 7600-ES20-GE3CXL        1.0  1-ROMMON              1.4         1.4
                                  2-I/O FPGA            0.21        0.21
                                  3-PKT ENG FPGA        0.5         0.5
                                  5-20x1GE LINK FPGA    0.7         0.7
---- ---------------------- ------ ------------------ ----------- --------------
   8 7600-ES20-10G3CXL       1.1  1-ROMMON              1.4         1.4
                                  2-I/O FPGA            0.21        0.21
                                  3-PKT ENG FPGA        0.5         0.5
                                  4-2x10GE LINK FPGA    0.9         0.9
---- ---------------------- ------ ------------------ ----------- --------------
   9 7600-ES2040G3CXL         0.303 1-ROMMON             1.1         1.1
                                  2-I/O FPGA            0.17        0.17
                                  3-SELENE              0.15        0.15
     ---------------------- ------ ------------------ ----------- --------------
     7600-ES203CXL            0.400 4-PKT EN FPGA XL     0.8         0.8
                                  11-Kp FPGA XL          1.1         1.1
     ---------------------- ------ ------------------ ----------- --------------
     7600-ES2040G             0.401 6-40x1G LinkFPGA     0.15        0.15
                                  10-40x1G LedFPGA       0.2         0.2
==== ====================== ====== ==============================================
```

This example shows the output when verifying the FPD for the ES20 card in a specific slot:

```
Router# show hw-module slot 9 fpd

==== ====================== ====== ==============================================
                             H/W   Field Programmable   Current   Min. Required
Slot Card Type              Ver.  Device: "ID-Name"    Version      Version
==== ====================== ====== ================== ========== ==============
   9 7600-ES2040G3CXL         0.303 1-ROMMON             1.1         1.1
                                  2-I/O FPGA            0.17        0.17
                                  3-SELENE              0.15        0.15
     ---------------------- ------ ------------------ ----------- --------------
     7600-ES203CXL            0.400 4-PKT EN FPGA XL     0.8         0.8
                                  11-Kp FPGA XL          1.1         1.1
     ---------------------- ------ ------------------ ----------- --------------
     7600-ES2040G             0.401 6-40x1G LinkFPGA     0.15        0.15
                                  10-40x1G LedFPGA       0.2         0.2
==== ====================== ====== ==============================================


Router#
```

Displaying Information About the Default FPD Image Package

You can use the **show upgrade fpd package default** command to find out which ES20 line cards are supported with your current Cisco IOS release and which FPD image package you need for an upgrade.

```
Router# show upgrade fpd package default

****************************************************************************
This Cisco IOS software image requires the following default FPD Image
Package for the automatic upgrade of FPD images (the package is available
from Cisco.com and is accessible from the Cisco Software Center page where
this IOS software image can be downloaded):
****************************************************************************

Version: 12.2(nightly.SR080616)

Package Filename: c7600-fpd-pkg.122-nightly.SR.pkg

        List of card type supported in this package:

                                                              Minimal
                 No. Card Type                                HW Ver.
                ---- ---------------------------------------- -------
                  1) 2 port adapter Enhanced FlexRouterN            1.0
                  2) 2 port adapter Enhanced FlexRouterN            2.0
                  3) 24xT1E1 CE/ATM SPA                         1.0
                  4) 1xOC3STM1 CE/ATM SPA                       1.0
                  5) 1xOC3STM1 CE/ATM SPA                       2.0
                  6) 2xT3E3 CE/ATM SPA                          1.0
                  7) 1xCHSTM1 SPA                               0.0
                  8) 2xCT3 SPA                                  0.100
                  9) 2xCT3 SPA                                  0.200
                 10) 4xCT3 SPA                                  0.100
                 11) 4xCT3 SPA                                  0.200
                 12) 10xGE SPA                                  0.0
                 13) 8xGE SPA                                   0.0
                 14) 8xFE TX SPA                                0.0
                 15) 4xFE TX SPA                                0.0
                 16) 5xGE SPA                                   0.0
                 17) 2xGE SPA                                   0.0
                 18) 1x10GE XFP SPA                             0.0
                 19) 10xGE SPA                                  0.0
                 20) 8xGE SPA                                   0.0
                 21) 8xFE TX SPA                                0.0
                 22) 4xFE TX SPA                                0.0
                 23) 5xGE SPA                                   0.0
                 24) 1x10GE XFP SPA                             0.0
                 25) 1x10GE DWDM SPA                            0.0
                 26) 2xGE V2 SPA                                0.0
                 27) 8xCHT1/E1 SPA                              0.140
                 28) 8xCHT1/E1 SPA                              0.0
                 29) 4xT SERIAL SPA                             0.0
                 30) 4xT SERIAL SPA                             2.0
                 31) 2xOC3 POS SPA                              0.0
                 32) 2xOC3 POS SPA                              0.200
                 33) 4xOC3 POS SPA                              0.0
                 34) 4xOC3 POS SPA                              0.200
                 35) 1xOC12 POS SPA                             0.0
                 36) 1xOC12 POS SPA                             0.200
                 37) 1xOC192 POS/RPR XFP SPA                    0.0
                 38) 1xOC192 POS/RPR SPA                        0.0
                 39) 1xOC48 POS/RPR SPA                         0.0
```

```
 40) 2xOC48 POS/RPR SPA                         0.0
 41) 4xOC48 POS/RPR SPA                         0.0
 42) 4-subslot SPA Interface Processor-200      0.100
 43) 4-subslot SPA Interface Processor-200      0.450
 44) 4-subslot SPA Interface Processor-200      0.500
 45) 4-subslot SPA Interface Processor-200      0.550
 46) 4-subslot SPA Interface Processor-200      0.600
 47) 4-subslot SPA Interface Processor-200      2.0
 48) 4-subslot SPA Interface Processor-400      0.1
 49) 1-subslot SPA Interface Processor-600      0.1
 50) ESM20G                                     0.1
 51) 2-subslot Services SPA Carrier-400         0.3
 52) 2-subslot Services SPA Carrier-400         0.4
 53) 2-subslot Services SPA Carrier-400         0.5
 54) 2-subslot Services SPA Carrier-600         0.1
 55) 7600 ES20                                   0.100
 56) 7600 ES20                                   0.300
 57) 7600 ES20                                   0.400
 58) 7600 ES20 DFC XL                            0.100
 59) 7600 ES20 DFC XL                            0.300
 60) 7600 ES20 DFC LITE                          0.100
 61) 7600 ES20 DFC LITE                          0.300
 62) 7600 ES20 40xGE SFP                         0.100
 63) 7600 ES20 40xGE SFP                         0.200
 64) 7600 ES20 40xGE SFP                         0.400
 65) 7600 ES20 20xGE SFP                         0.100
 66) 7600 ES20 20xGE SFP                         0.200
 67) 7600 ES20 20xGE SFP                         0.400
 68) 7600 ES20 4x10GE XFP                        0.100
 69) 7600 ES20 4x10GE XFP                        0.200
 70) 7600 ES20 4x10GE XFP                        0.150
 71) 7600 ES20 2x10GE XFP                        0.100
 72) 7600 ES20 2x10GE XFP                        0.200
 73) 7600 ES20 2x10GE XFP                        0.150
 74) 2xT3E3 SPA                                 0.0
 75) 4xT3E3 SPA                                 0.0
 76) 2 Gbps IPSec SPA                           0.1
 77) 2 Gbps C12000 IPSec SPA                    0.1
 78) 2xOC3 ATM SPA                              0.0
 79) 4xOC3 ATM SPA                              0.0
 80) 1xOC12 ATM SPA                             0.0
 81) 1xOC48 ATM SPA                             0.0
---- ---------------------------------------- -------
```

FPD Image Upgrade Examples

This section provides examples of automatic and manual FPD image upgrades. It includes the following examples:

- Automatic FPD Image Upgrade Example, page 5-11
- Manual FPD Image Upgrade Example, page 5-12

Automatic FPD Image Upgrade Example

The following example uses the **upgrade fpd auto to do an automatic upgrade**.

```
Router# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
```

```
Router(config)# upgrade fpd ?
  auto  Auto upgrade all FPD images
  path  Set path to locate the FPD image package file for auto upgrade

Router(config)#
Router(config)# upgrade fpd auto ?
  <cr>

Router(config)# upgrade fpd auto
Router(config)#
Router(config)#^Z
Router# sgh
*Jun 18 10:27:00.078 sum08: %SYS-5-CONFIG_I: Configured from console by consoh ver
Cisco IOS Software, rsp72043_rp Software (rsp72043_rp-ADVENTERPRISEK9_DBG-M), Version
12.2(nightly.SR080616) NIGHTLY BUILD, synced to rainier
RAINIER_BASE_FOR_V122_33_SRA_THROTTLE
Copyright (c) 1986-2008 by Cisco Systems, Inc.
Compiled Tue 17-Jun-08 00:10 by cuotran

ROM: System Bootstrap, Version 12.2(33r)SRB3, RELEASE SOFTRouterRE (fc1)

 Router uptime is 22 hours, 29 minutes
Uptime for this control processor is 22 hours, 29 minutes
System returned to ROM by reload (SP by reload)
System image file is "disk0:rsp72043-adventerprisek9_dbg-mz.autobahn76_061608"
Last reload type: Normal Reload
```

Manual FPD Image Upgrade Example

In the following example, FPD for the ES20 line card in slot 8 is upgraded manually:

```
Router#
Router# upgrade hw-module slot 8 ?
  fpd  Field programmable device upgrade option
Router# upgrade hw-module slot 8 fpd ?
  file  Upgrade with field programmable device package/bundle file

Router# upgrade hw-module slot 8 fpd fi
Router# upgrade hw-module slot 8 fpd file c
Router# upgrade hw-module slot 8 fpd file d
*Jun 17 13:24:12.531 sum08: %FPD_MGMT-3-INCOMP_IMG_VER: Incompatible I/O FPGA (FPD ID=2)
image version detected for 7600-ES2040G3CXL card in slot 8. Detected version = 0.16,
minimum required version = 0.17. Current HW version = 0.118.
*Jun 17 13:24:12.531 sum08: %FPD_MGMT-3-INCOMP_IMG_VER: Incompatible 40x1G LinkFPGA (FPD
ID=6) image version detected for 7600-ES2040G card in slot-dc 8-2. Detected version =
0.14, minimum required version = 0.15. Current HW version = 0.106.
*Jun 17 13:24:12.531 sum08: %FPD_MGMT-5-UPGRADE_ATTEMPT: Attempting to automatically
upgrade the FPD image(s) for 7600-ES2040G3CXL card in slot 8. Use 'show upgrade fpd
progress' command to view the upgrade progress ...
*Jun 17 13:24:12.547 sum08: %FPD_MGMT-6-BUNDLE_DOWNLOAD: Downloading FPD image bundle for
7600-ES2040G3CXL card in slot 8 ...i
Router#upgrade hw-module slot 8 fpd file disk
*Jun 17 16:24:12.551: %FABRIC_INTF_ASIC-DFC8-5-FABRICSYNC_DONE: Fabric ASIC 0 Channel 1:
Fabric sync done.
*Jun 17 16:24:12.575: %FABRIC_INTF_ASIC-DFC8-5-FABRICSYNC_DONE: Fabric ASIC 1 Channel 1:
Fabric sync done.
```

APPENDIX A

Command Summary for the Cisco 7600 Series Ethernet Services 20G Line Card

Table A-1 provides an alphabetical list of some of the related commands to monitor and maintain the Cisco 7600 Series Ethernet Services 20G (ES20) line card. Table A-2 provides an alphabetical list of the EVC commands used with the ES20 line card.

Note **keepalive** command in the interface configuration mode is not supported in a ES20 line card.

For more information about the commands used in this chapter, see the *Cisco IOS Release 12.2SR Command References at* http://www.cisco.com/en/US/products/ps6922/prod_command_reference_list.html.

If a command is not located in those publications, refer to the Cisco IOS command reference and master index publications.

Table A-1 Monitoring Command Summary

Command	Purpose
Router# **hw-module module** *slot* **reset**	Turns power off and on to the ES20 line card in the specified chassis slot.
Router(config)# **no power enable module** *slot*	Powers down an ES20 line card in the specified chassis slot.
Router(config)# **power enable module** *slot*	Powers on an ES20 line card in the specified chassis slot.
Router# **show diagbus** *slot*	Displays information about the ES20 line card installed in the specified chassis slot, and status of any small form-factor pluggable (SFP or XFP) optical transceivers. hardware installed in that ES20 line card.
Router# **show hw-module slot** *slot* **align** [**cpu** {**0** | **1**}]	Displays alignment data for an ES20 line card in the specified chassis slot.
Router# **show hw-module slot** *slot* **logging** [**cpu** {**0** | **1**}]	Displays logging information for an ES20 line card in the specified chassis slot.
Router# **show hw-module slot** *slot* **proc cpu** [**cpu** {**0** | **1**}]	Displays CPU utilization for each process on an ES20 line card in the specified chassis slot.
Router# **show hw-module slot** *slot* **tech-support** [**cpu** {**0** | **1**}]	Displays system information to troubleshoot a problem for an ES20 line card in the specified chassis slot.

Table A-1 Monitoring Command Summary (continued)

Command	Purpose
Router# **show idprom module** *slot* [**clei**]	Displays IDPROM information for an ES20 line card in the specified chassis slot, and optionally any Common Language Equipment Identification (CLEI) information for the ES20 line card.
Router# **show module** [*slot* | **all** | **version**]	Displays status and information for an ES20 line card in the specified chassis slot.

Table A-2 EVC Command Summary

Command	Purpose
Router(config-if-srv)# **[bridge-domain bridge-id**	Binds the service instance to a bridge-domain instance.
Router(config-if-srv)# **connect evc-name interface1 efp-id1 interface2 efp-id2**	Creates a direct and transparent connection between two service instances.
Router#**debug ethernet service evc** [**id** *evc-id*]	Enables Ethernet virtual connection (EVC) debugging. If no EVC ID is specified, debugging is enabled for all EVCs on the system.
Router#**debug ethernet service instance** [**id** *instance-id* **interface** *interface-id* | **interface** *interface-id*]	Enables service instance debugging. If no options are specified, debugging for all service instances is enabled. If a service instance ID and interface are specified, only debug messages for the associated service instance are output. If only an interface is specified, debug messages for all service instances on that interface are displayed.
Router#**debug ethernet service interface** [*interface-id*]	Enables Port Data Block (PDB) debugging.
Router#**debug ethernet service api**	Enables debugging between Ethernet Services Infrastructure and its clients.
Router#**debug ethernet service oam-mgr**	Enables OAM Manager debugging, to debug OAM inter-working.
Router#**debug ethernet service error**	Enables ethernet service error debugging.
Router#**debug ethernet service all**	Enables all EI debugging messages for all PDBs, EVCs and service instances.
Router#**ethernet evc** ***service-name***	Creates an EVC (or a Customer Service Instance) on global configuration mode. It sets the device into the config-srv submode.
Router(config-if-srv)#**encapsulation untagged**	Defines the matching criteria to be used in order to map untagged Ethernet frames ingress on an interface to the appropriate service instance.
Router(config-if-srv)#**encapsulation default**	Configures the default service instance on a port. Anything that does not meet the criteria of other service instances on the same physical interface falls into this service instance

Table A-2 EVC Command Summary (continued)

Command	Purpose
Router(config-if-srv)#**encapsulation dot1q** *vlan-id[,vlan-id[-vlain-id]]* **[native]**	Defines the matching criteria to be used in order to map dot1q frames ingress on an interface to the appropriate service instance. The criteria for this command are: single VLAN, range of VLANs and lists of the previous two.
Router(config-if-srv)#**encapsulation dot1ad** *vlan-id*[,*vlan-id*[-*vlain-id*]] [**native**]	Defines the matching criteria to be used in order to map single-tagged 802.1ad frames ingress on an interface to the appropriate service instance. The criteria for this command are: single VLAN, range of VLANs and lists of the previous two.
Router(config-if-srv)#**encapsulation dot1q** <*vlan-id*> **second-dot1q** {**any** | *vlan-id*[,*vlan-id*[-*vlain-id*]]}	Defines the matching criteria to be used in order to map QinQ frames ingress on an interface to the appropriate service instance. The criteria for this command are: outer tag must be unique and inner tag may be single VLAN, range of VLANs or lists of the previous two.
Router(config-if-srv)#**encapsulation dot1ad** <*vlan-id*> **dot1q** {**any** | *vlan-id*[,*vlan-id*[-*vlain-id*]]}	Defines the matching criteria to be used in order to map double-tagged 802.1ad frames ingress on an interface to the appropriate service instance. The criteria for this command are: outer tag must be unique and inner tag may be single VLAN, range of VLANs or lists of the previous two.
Router(config-if-srv)#**rewrite ingress tag** {**push {dot1q** *vlan-id* | **dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **pop** {**1** | **2**} | **translate** {**1-to-1** {**dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **2-to-1 dot1q** *vlan-id* | **dot1ad** *vlan-id*}| **1-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*} | **2-to-2** {**dot1q** *vlan-id* **second-dot1q** *vlan-id* | **dot1ad** *vlan-id* **dot1q** *vlan-id*}} [**symmetric**]	Specifies the encapsulation adjustment that is to be performed on the frame ingress to the service instance.
Router(config-if-srv)#**service instance** *id* {**Ethernet** [service-name]}	Creates an service instance (instantiation of an EVC) on an interface. It sets the device into the config-if-srv sub-mode.
Router#**show ethernet service evc [id evc-id | interface interface-id] [detail]**	Displays information pertaining to a specific EVC if an EVC ID is specified, or pertaining to all EVCs on an interface if an interface is specified. The **detailed** option provides additional information on the EVC.
Router#**show ethernet service instance [id <instance-id> interface interface-id | interface interface-id] [detail]**	Displays information about one or more service instances: If a service instance ID and interface are specified, only data pertaining to that particular service instance will be displayed. If only an interface ID is specified, this command will display data for all service instances on the given interface.
Router#**show ethernet service interface** [*interface-id*] [**detail**]	Displays information in the PDB.
Router(config-if-srv)#**xconnect** *peer-id vc-id* **encapsulation mpls**	Configures scalable EoMPLS on a service instance. On the ingress side, after proper encapsulation manipulations, the packet is tunneled in a EoMPLS VC and transmitted on the core.

INDEX

A

activation
  verifying for ES20 line card 5
automatic FPD image upgrade
  (example) 11
  disabling 5
  re-enabling 6

B

backup interface for flexible UNI feature 150
bridge-domain command 72, 130, 171, 176, 180, 183, 186, 191, 195, 207, 224, 226, 231, 2
BRR on Hybrid Policies 63

C

cautions, usage in text xx
CLEI (Common Language Equipment Identification)
  displaying 2
command syntax
  conventions xix
configuration example
  VPLS, associating the attachment circuit with the VSI at the PE 148
configure terminal command 6
Configuring
  ATM/FR to Ethernet 304
  Dual Rate Three Color Policing 11
Configuring 2R3C 11
  Conform, exceed and violate commands 12
  Detailed steps 9, 13
  Examples 10, 15
  Restrictions 11
  Summary Steps 12
Configuring Asymmetric Carrier-Delay 165
Configuring Custom Ethertype for EVC Interfaces 74
Configuring EVC Etherchannel 69
Configuring Flexible Service Mapping Based on CoS and Ethertype 119
Configuring Gigabit Ethernet Link Aggregation with Advanced Load Balancing 134
Configuring H-Virtual Private LAN Service (VPLS) with Port-Channel as a Core Interface 141
Configuring L2ACL on EVC 156
Configuring Private Host SVI
  Port Classification 254
Configuring Private Host Switch Virtual Interface (VLAN and VPLS) 254
Configuring UniDirectional Link Detection (UDLD)
  Detailed Steps 10
  Disable UDLD on ports with configured EVC 11
  Enable UDLD on Ports with configured EVC 10
  Reset the disabled UDLD on ports with configured EVC. 11
  Summary 9, 39, 40
  UDLD Aggressive Mode 9
connect command 2
Create L2 Access List 158
Custom Ethertype
  Examples 108
  Rewrite Rules 105
  Supported Rewrites 105
  Supported Rewrites on C-Tag 106
Custom Ethetype on EVC interfaces
  Restrictions and Usage Guidelines 106

D

deactivation

- verifying for ES20 line card 5

debug commands 2

debug ethernet service all command 2

debug ethernet service api command 2

debug ethernet service error command 2

debug ethernet service evc command 2

debug ethernet service instance command 2

debug ethernet service interface command 2

debug ethernet service oam-mgr command 2

debug hw-module command 2

DHCP snooping option-82 feature 174

Documentation and Service Request xx

drop command 23

E

encapsulation default command 2

encapsulation dot1ad command 3

encapsulation dot1q command 121, 130, 160, 290, 3

encapsulation frame-relay ietf command 71, 79, 99, 114, 176, 184, 186, 189, 191

encapsulation untagged command 2

error messages 1

- system 2

ES20 line card

- activation (example) 6
- deactivating 4
- deactivation (example) 6
- features 3–??
- features supported 3
- hardware type, displaying 9
- reactivating 4
- release history 1
- resetting 307

ethernet evc command 2

Ethertypes

- 0x8100 – 802.1q 104
- 0x88a8 – 802.1ad 105
- 0x9100 – Q-in-Q 105
- 0x9200 – Q-in-Q 105

event tracer feature 2

F

Flat Policy-Maps 62

FPD image packages

- caution 3, 8
- displaying default information 10
- downloading 7–8
- modifying the default path 8
- overview 2
- version number requirements 3

FPD images

- displaying minimum and current versions 8
- manually upgrading 6
- upgrade scenarios 3
- upgrading in production 4–6

FPDs (field-programmable devices)

- description 1

FTP server, downloading FPD images to 7–8

H

Hierarchical Policy-Maps 58

hierarchical QoS 48

hierarchical QoS (HQoS) 53

hierarchical virtual private LAN services (H-VPLS) 303

HQoS Polices with only Shape in Child Classes 60

HQoS Policies with Bandwidth and Shape in the Child Classes 59

HQoS Policies with Police+Priority(LLQ) and Shape in the Child Classes 61

HQoS with Ethernet MPB 53

http

//www.cisco.com/en/US/docs/interfaces_modules/shared_port_adapters/configuration/7600series/76cfgeth.html 304

hw-module reset command 307, 308, 1

hw-module subslot shutdown command 6

I

IGMP/PIM snooping for VPLS pseudowire 258

interface gigabitethernet command 121, 130, 159, 270, 290

ip igmp snooping command 259

ipv6 mld snooping command 259

L

LACP over EVC Port Channel 69

configuration commands, configuration steps 5, 22, 71, 170, 179, 194, 205, 221, 223, 225, 229

Verification 28, 74

LACP over EVC port channel 69

configuration commands, configuration steps 181, 182, 184, 186, 188, 190, 191, 196, 198, 209, 214, 232, 234, 236, 238, 240, 248

Examples 5, 72, 80, 101

load balancing 69

restrictions 69

LLQ 46

M

MAC address security for EVC bridge-domain 178

marking 29

match access-group command 17

match access-group name command 17

match any command 17

match atm clp command 17

match class-map command 17

match cos command 16, 17

match cos-inner command 16

match destination-address mac command 18

match fr-de command 18

match fr-dlci command 18

match inner-cos command 17

match input vlan command 18

match ip dscp command 18

match ip precedence command 18

match ip rtp command 18

match mac address command 18

match mpls experimental command 18

match not command 19

match packet length command 18

match protocol command 19

match qos-group command 19

match source-address mac command 18

match vlan command 19

match vlan inner command 19

match vlan-inner command 16

MST on EVC bridge-domain 167

Multichassis Support for Link Aggregation Control Protocol 74

multiPoint bridging over Ethernet 128

N

no power enable module command 4, 6, 1

no upgrade fpd auto command 5

notes, usage in text xx

O

OIR 3

OIR (online insertion and removal)

for ES20s 4

onboard failure logging (OBFL) 7

online diagnostics 7

online insertion and removal 3

P

PFC QoS **47**

police flow command **22**

police flow mask command **22**

policing **21**

power enable module command **5**, **6**, **1**

Q

QoS

- Ingress Trust **2**
- marking **29**
- Policing **21**
- queueing **42**
- shaping **34**

queue scheduling **42**

R

rewrite ingress tag command **130**, **3**

RFC 2698 **11**

S

scalable EoMPLS **289**

service instance command **121**, **130**, **159**, **270**, **290**, **3**

set atm-clp command **32**

set cos command **33**

set cos cos-inner command **33**

set cos-inner command **33**

set cos-inner cos command **33**

set discard-class command **32**

set fr-de command **32**

set ip dscp command **32**

set ip precedence command **33**

set mpls experimental imposition command **33**

set mpls experimental topmost command **33**

set qos-group command **33**

set-dscp-transmit command **23**

set-mpls-experimental-imposition-transmit command **23**

set-mpls-experimental-topmost-transmit command **23**

set-prec-transmit command **23**

SFPs

- troubleshooting **3**

shape adaptive command **35**

shape fecn-adapt command **35**

shape peak command **35**

shape peak percent command **35**

shaping **34**

show diagbus command **9**, **1**

show ethernet service evc command **3**

show ethernet service instance command **3**

show ethernet service interface command **3**

show hw-module slot align command **1**

show hw-module slot logging command **1**

show hw-module slot proc cpu command **1**

show hw-module slot tech-support command **1**

show hw-module subslot command **8**

show idprom command **9**

show idprom module command **2**

show module command **9**, **5**, **2**

show policy-map class command **46**

show policy-map command **46**

show policy-map interface command **46**

show running-config command **7**

show upgrade package default command **10**

storm control on switchports and ports having EVC **140**

Supported Rewrites

- Range on C-Tag with a NNI **106**

system error messages **2**

T

TE-FRR Support on VPLS LAG NNI **142**

TFTP server, downloading FPD images to **7**–**8**

tips, usage in text **xx**

transmit command **23**

troubleshooting 1

SFPs 3

U

UDE on ES-20 Line cards

restrictions 3, 260

upgrade fpd auto command 6, 7, 11

upgrade fpd path command 7, 8

upgrade hw-module subslot command 6

V

VPLS (virtural private LAN service) 302

X

xconnect command 3

xconnect vfi command 259