Cisco IOS XE Interface and Hardware Component Configuration Guide

Release 2

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

Cisco IOS XE Interface and Hardware Component Configuration Guide

About Cisco IOS XE Software Documentation

Last Updated: December 1, 2009

This document describes the objectives, audience, conventions, and organization used in Cisco IOS XE software documentation. Also included are resources for obtaining technical assistance, additional documentation, and other information from Cisco. This document is organized into the following sections:

Documentation Objectives

Cisco IOS XE documentation describe the tasks and commands available to configure and maintain Cisco networking devices.

Audience

The Cisco IOS XE documentation set is intended for users who configure and maintain Cisco networking devices (such as routers and switches) but who may not be familiar with the configuration and maintenance tasks, the relationship among tasks, or the Cisco IOS commands necessary to perform particular tasks. The Cisco IOS XE documentation set is also intended for those users experienced with Cisco IOS XE software who need to know about new features, new configuration options, and new software characteristics in the current Cisco IOS XE release.

Documentation Conventions

In Cisco IOS XE documentation, the term *router* may be used to refer to various Cisco products; for example, routers, access servers, and switches. These and other networking devices that support Cisco IOS XE software are shown interchangeably in examples and are used only for illustrative purposes. An example that shows one product does not necessarily mean that other products are not supported.

This section contains the following topics:

Typographic Conventions

Cisco IOS XE documentation uses the following typographic conventions:

Convention	Description
^ or Ctrl	Both the ^ symbol and Ctrl represent the Control (Ctrl) key on a keyboard. For example, the key combination **^D** or **Ctrl-D** means that you 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 a Simple Network Management Protocol (SNMP) community string to *public*, do not use quotation marks around the string; otherwise, the string will include the quotation marks.

Command Syntax Conventions

Cisco IOS XE documentation uses the following command syntax conventions:

Convention	Description
bold	Bold text indicates commands and keywords that you enter as shown.
italic	Italic text indicates arguments for which you supply values.
[x]	Square brackets enclose an optional keyword or argument.
...	An ellipsis (three consecutive nonbolded periods without spaces) after a syntax element indicates that the element can be repeated.
|	A vertical line, called a pipe, indicates a choice within a set of keywords or arguments.
[x | y]	Square brackets enclosing keywords or arguments separated by a pipe indicate an optional choice.
{x | y}	Braces enclosing keywords or arguments separated by a pipe indicate a required choice.
[x {y | z}]	Braces and a pipe within square brackets indicate a required choice within an optional element.

Software Conventions

Cisco IOS XE software uses the following conventions:

Convention	Description
`Courier font`	Courier font is used for information that is displayed on a PC or terminal screen.
`Bold Courier font`	Bold Courier font indicates text that the user must enter.
< >	Angle brackets enclose text that is not displayed, such as a password. Angle brackets also are used in contexts in which the italic font style is not supported; for example, ASCII text.
!	An exclamation point at the beginning of a line indicates that the text that follows is a comment, not a line of code. An exclamation point is also displayed by the Cisco IOS XE software for certain processes.
[ ]	Square brackets enclose default responses to system prompts.

Reader Alert Conventions

Cisco IOS XE documentation uses the following conventions for reader alerts:

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 material not covered in the manual.

Timesaver Means *the described action saves time*. You can save time by performing the action described in the paragraph.

Documentation Organization

This section describes the Cisco IOS XE documentation set, how it is organized, and how to access it on Cisco.com. Listed are configuration guides, command references, and supplementary references and resources that comprise the documentation set.

Cisco IOS XE Documentation Set

The Cisco IOS XE documentation set consists of the following:

- Release notes and caveats provide information about platform, technology, and feature support for a release and describe severity 1 (catastrophic), severity 2 (severe), and severity 3 (moderate) defects in released Cisco IOS XE software. Review release notes before other documents to learn whether updates have been made to a feature.
- Sets of configuration guides and command references organized by technology and published for each standard Cisco IOS XE release.
  - Configuration guides—Compilations of documents that provide conceptual and task-oriented descriptions of Cisco IOS XE features.
  - Command references—Alphabetical compilations of command pages that provide detailed information about the commands used in the Cisco IOS XE features and the processes that comprise the related configuration guides. For each technology, there is a single command reference that covers all Cisco IOS XE releases and that is updated at each standard release.
- Command reference book for **debug** commands.
- Lists of all the commands in a specific release and all commands that are new, modified, removed, or replaced in the release.
- Reference book for system messages for all Cisco IOS XE releases.

Cisco IOS XE Documentation on Cisco.com

The following sections describe the documentation organization and how to access various document types.

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

New Features List

The New Features List for each release provides a list of all features in the release with hyperlinks to the feature guides in which they are documented.

Configuration Guides

Configuration guides are provided by technology and release and comprise a set of individual feature guides relevant to the release and technology.

Command References

Command reference books describe Cisco IOS XE commands that are supported in many different software releases and on many different platforms. The books are organized by technology. For information about all Cisco IOS XE commands, use the Command Lookup Tool at http://tools.cisco.com/Support/CLILookup or the *Cisco IOS Master Command List, All Releases*, at http://www.cisco.com/en/US/docs/ios/mcl/allreleasemcl/all_book.html.

Cisco IOS XE Supplementary Documents and Resources

Supplementary documents and resources are listed in Table 2 on page x.

Configuration Guides, Command References, and Supplementary Resources

Table 1 lists, in alphabetical order, Cisco IOS XE software configuration guides and command references, including brief descriptions of the contents of the documents. The command references contain commands for both Cisco IOS software and Cisco IOS XE software, for all releases. The command references support many different software releases and platforms. Your Cisco IOS XE software release or platform may not support all these technologies.

Table 2 lists documents and resources that supplement the Cisco IOS XE software configuration guides and command references. These supplementary resources include release notes and caveats; master command lists; new, modified, removed, and replaced command lists; system messages; and the debug command reference.

For additional information about configuring and operating specific networking devices, and to access Cisco IOS documentation, go to the Product/Technologies Support area of Cisco.com at the following location:

http://www.cisco.com/go/techdocs

Table 1 Cisco IOS XE Configuration Guides and Command References

Configuration Guide and Command Reference Titles	Features/Protocols/Technologies
• *Cisco ASR 1000 Series Aggregation Services Routers SIP and SPA Software Configuration Guide*	Configuration and troubleshooting of SPA interface processors (SIPs) and shared port adapters (SPAs) that are supported on the Cisco ASR 1000 Series Router.
• *Cisco ASR 1000 Series Aggregation Services Routers Software Configuration Guide*	Overview of software functionality that is specific to the Cisco ASR 1000 Series Aggregation Services Routers.
• *Cisco IOS XE Access Node Control Protocol Configuration Guide* • *Cisco IOS Access Node Control Protocol Command Reference*	Communication protocol between digital subscriber line access multiplexers (DSLAMs) and a broadband remote access server (BRAS).
• *Cisco IOS XE Asynchronous Transfer Mode Configuration Guide* • *Cisco IOS Asynchronous Transfer Mode Command Reference*	LAN ATM, multiprotocol over ATM (MPoA), and WAN ATM.
• *Cisco IOS XE Broadband Access Aggregation and DSL Configuration Guide* • *Cisco IOS Broadband Access Aggregation and DSL Command Reference*	PPP over Ethernet (PPPoE).
• *Cisco IOS XE Carrier Ethernet Configuration Guide* • *Cisco IOS Carrier Ethernet Command Reference*	IEEE 802.3ad Link Bundling; Link Aggregation Control Protocol (LACP) support for Ethernet and Gigabit Ethernet links and EtherChannel bundles; LACP support for stateful switchover (SSO), in service software upgrade (ISSU), Cisco nonstop forwarding (NSF), and nonstop routing (NSR) on Gigabit EtherChannel bundles; and IEEE 802.3ad Link Aggregation MIB.
• *Cisco IOS XE Configuration Fundamentals Configuration Guide* • *Cisco IOS Configuration Fundamentals Command Reference*	Autoinstall, Setup, Cisco IOS command-line interface (CLI), Cisco IOS file system (IFS), Cisco IOS web browser user interface (UI), basic file transfer services, and file management.

Table 1 Cisco IOS XE Configuration Guides and Command References (continued)

Configuration Guide and Command Reference Titles	Features/Protocols/Technologies
• *Cisco IOS XE DECnet Configuration Guide* • *Cisco IOS DECnet Command Reference*	DECnet protocol.
• *Cisco IOS XE Dial Technologies Configuration Guide* • *Cisco IOS Dial Technologies Command Reference*	Asynchronous communications, dial backup, dialer technology, Multilink PPP (MLP), PPP, and virtual private dialup network (VPDN).
• *Cisco IOS XE High Availability Configuration Guide* • *Cisco IOS High Availability Command Reference*	A variety of high availability (HA) features and technologies that are available for different network segments (from enterprise access to service provider core) to facilitate creation of end-to-end highly available networks. Cisco IOS HA features and technologies can be categorized in three key areas: system-level resiliency, network-level resiliency, and embedded management for resiliency.
• *Cisco IOS XE Intelligent Services Gateway Configuration Guide* • *Cisco IOS Intelligent Services Gateway Command Reference*	Subscriber identification, service and policy determination, session creation, session policy enforcement, session life-cycle management, accounting for access and service usage, and session state monitoring.
• *Cisco IOS XE Interface and Hardware Component Configuration Guide* • *Cisco IOS Interface and Hardware Component Command Reference*	LAN interfaces, logical interfaces, serial interfaces, virtual interfaces, and interface configuration.
• *Cisco IOS XE IP Addressing Services Configuration Guide* • *Cisco IOS IP Addressing Services Command Reference*	IP addressing, Address Resolution Protocol (ARP), Network Address Translation (NAT), Domain Name System (DNS), Dynamic Host Configuration Protocol (DHCP), and Next Hop Address Resolution Protocol (NHRP).
• *Cisco IOS XE IP Application Services Configuration Guide* • *Cisco IOS IP Application Services Command Reference*	Enhanced Object Tracking (EOT), Gateway Load Balancing Protocol (GLBP), Hot Standby Router Protocol (HSRP), IP Services, TCP, Web Cache Communication Protocol (WCCP), User Datagram Protocol (UDP), and Virtual Router Redundancy Protocol (VRRP).
• *Cisco IOS XE IP Multicast Configuration Guide* • *Cisco IOS IP Multicast Command Reference*	Protocol Independent Multicast (PIM) sparse mode (PIM-SM), bidirectional PIM (bidir-PIM), Source Specific Multicast (SSM), Multicast Source Discovery Protocol (MSDP), Internet Group Management Protocol (IGMP), and Multicast VPN (MVPN).
• *Cisco IOS XE IP Routing: BFD Configuration Guide*	Bidirectional forwarding detection (BFD).
• *Cisco IOS XE IP Routing: BGP Configuration Guide* • *Cisco IOS IP Routing: BGP Command Reference*	Border Gateway Protocol (BGP), multiprotocol BGP, multiprotocol BGP extensions for IP multicast.
• *Cisco IOS XE IP Routing: EIGRP Configuration Guide* • *Cisco IOS IP Routing: EIGRP Command Reference*	Enhanced Interior Gateway Routing Protocol (EIGRP).
• *Cisco IOS XE IP Routing: ISIS Configuration Guide* • *Cisco IOS IP Routing: ISIS Command Reference*	Intermediate System-to-Intermediate System (IS-IS).

Table 1 Cisco IOS XE Configuration Guides and Command References (continued)

Configuration Guide and Command Reference Titles	Features/Protocols/Technologies
• *Cisco IOS XE IP Routing: ODR Configuration Guide* • *Cisco IOS IP Routing: ODR Command Reference*	On-Demand Routing (ODR).
• *Cisco IOS XE IP Routing: OSPF Configuration Guide* • *Cisco IOS IP Routing: OSPF Command Reference*	Open Shortest Path First (OSPF).
• *Cisco IOS XE IP Routing: Protocol-Independent Configuration Guide* • *Cisco IOS IP Routing: Protocol-Independent Command Reference*	IP routing protocol-independent features and commands. Generic policy-based routing (PBR) features and commands are included.
• *Cisco IOS XE IP Routing: RIP Configuration Guide* • *Cisco IOS IP Routing: RIP Command Reference*	Routing Information Protocol (RIP).
• *Cisco IOS XE IP SLAs Configuration Guide* • *Cisco IOS IP SLAs Command Reference*	Cisco IOS IP Service Level Agreements (IP SLAs).
• *Cisco IOS XE IP Switching Configuration Guide* • *Cisco IOS IP Switching Command Reference*	Cisco Express Forwarding.
• *Cisco IOS XE IPv6 Configuration Guide* • *Cisco IOS IPv6 Command Reference*	For a list of IPv6 features, protocols, and technologies, go to the IPv6 “Start Here” document at the following URL: http://www.cisco.com/en/US/docs/ios/ios_xe/ipv6/configuration/guide/ip6-roadmap_xe.html
• *Cisco IOS XE ISO CLNS Configuration Guide* • *Cisco IOS ISO CLNS Command Reference*	ISO Connectionless Network Service (CLNS).
• *Cisco IOS XE LAN Switching Configuration Guide* • *Cisco IOS LAN Switching Command Reference*	VLANs and multilayer switching (MLS).
• *Cisco IOS XE Multiprotocol Label Switching Configuration Guide* • *Cisco IOS Multiprotocol Label Switching Command Reference*	MPLS Label Distribution Protocol (LDP), MPLS Layer 2 VPNs, MPLS Layer 3 VPNs, MPLS Traffic Engineering (TE), and MPLS Embedded Management (EM) and MIBs.
• *Cisco IOS XE NetFlow Configuration Guide* • *Cisco IOS NetFlow Command Reference*	Network traffic data analysis, aggregation caches, and export features.
• *Cisco IOS XE Network Management Configuration Guide* • *Cisco IOS Network Management Command Reference*	Basic system management, system monitoring and logging, Cisco IOS Scripting with Tool Control Language (Tcl), Cisco networking services (CNS), Embedded Event Manager (EEM), Embedded Syslog Manager (ESM), HTTP, Remote Monitoring (RMON), and SNMP.
• *Cisco IOS XE Novell IPX Configuration Guide* • *Cisco IOS Novell IPX Command Reference*	Novell Internetwork Packet Exchange (IPX) protocol.

Table 1 Cisco IOS XE Configuration Guides and Command References (continued)

Configuration Guide and Command Reference Titles	Features/Protocols/Technologies
• *Cisco IOS XE Quality of Service Solutions Configuration Guide* • *Cisco IOS Quality of Service Solutions Command Reference*	Class-based weighted fair queueing (CBWFQ), low latency queueing (LLQ), Modular Quality of Service (QoS) Command-Line Interface (CLI) (MQC), Network-Based Application Recognition (NBAR), priority queueing, Multilink PPP (MLP) for QoS, header compression, Resource Reservation Protocol (RSVP), weighted fair queueing (WFQ), and weighted random early detection (WRED).
• *Cisco IOS Security Command Reference*	Access control lists (ACLs); authentication, authorization, and accounting (AAA); firewalls; IP security and encryption; neighbor router authentication; network access security; public key infrastructure (PKI); RADIUS; and TACACS+.
• *Cisco IOS XE Security Configuration Guide: Secure Connectivity*	Internet Key Exchange (IKE) for IPsec VPNs; security for VPNs with IPsec; VPN availability features (reverse route injection, IPsec preferred peer, and real-time resolution for the IPsec tunnel peer); IPsec data plane features; IPsec management plane features; Public Key Infrastructure (PKI); Dynamic Multipoint VPN (DMVPN); Easy VPN; and Cisco Group Encrypted Transport VPN (GET VPN).
• *Cisco IOS XE Security Configuration Guide: Securing the Data Plane*	Access Control Lists (ACLs); Firewalls: Context-Based Access Control (CBAC) and Zone-Based Firewall; Cisco IOS Intrusion Prevention System (IPS); Flexible Packet Matching; Unicast Reverse Path Forwarding (uRPF); Threat Information Distribution Protocol (TIDP) and TMS.
• *Cisco IOS XE Security Configuration Guide: Securing User Services*	AAA (includes Network Admission Control [NAC]); Security Server Protocols (RADIUS and TACACS+); Secure Shell (SSH); Secure Access for Networking Devices (includes Autosecure and Role-Based CLI access); Lawful Intercept.
• *Cisco IOS XE Service Advertisement Framework Configuration Guide* • *Cisco IOS Service Advertisement Framework Command Reference*	Cisco Service Advertisement Framework.
• *Cisco IOS XE VPDN Configuration Guide* • *Cisco IOS VPDN Command Reference*	Multihop by Dialed Number Identification Service (DNIS), timer and retry enhancements for L2TP and Layer 2 Forwarding (L2F), RADIUS Attribute 82 (tunnel assignment ID), shell-based authentication of VPDN users, and tunnel authentication via RADIUS on tunnel terminator.
• *Cisco IOS XE Wide-Area Networking Configuration Guide* • *Cisco IOS Wide-Area Networking Command Reference*	Frame Relay; L2VPN Pseudowire Redundancy; and Media-Independent PPP and Multilink PPP.

Table 1 Cisco IOS XE Configuration Guides and Command References (continued)

Configuration Guide and Command Reference Titles	Features/Protocols/Technologies
• *Cisco Unified Border Element (Enterprise) Configuration Guide* • *Cisco IOS Voice Command Reference*	The Cisco Unified Border Element (Enterprise) on the Cisco ASR 1000 brings a scalable option for enterprise customers. Running as a process on the Cisco ASR 1000 and utilizing the high-speed RTP packet processing path, the Cisco Unified Border Element (Enterprise) is used as an IP-to-IP gateway by enterprises and commercial customers to interconnect SIP and H.323 voice and video networks. The Cisco UBE (Enterprise) provides a network-to-network demarcation interface for signaling interworking, media interworking, address and port translations, billing, security, quality of service (QoS), and bandwidth management.
• *Cisco Unified Border Element (SP Edition) Configuration Guide: Distributed Model* • *Cisco Unified Border Element (SP Edition) Command Reference: Distributed Model*	The Cisco Unified Border Element (SP Edition) is a session border controller (SBC) that is VoIP-enabled and deployed at the edge of networks. For Cisco IOS XE Release 2.3 and earlier releases, Cisco Unified Border Element (SP Edition) is supported only in the distributed mode. Operating in the distributed mode, the SBC is a toolkit of functions that can be used to deploy and manage VoIP services, such as signaling interworking, network hiding, security, and quality of service.
• *Cisco Unified Border Element (SP Edition) Configuration Guide: Unified Model* • *Cisco Unified Border Element (SP Edition) Command Reference: Unified Model*	The Cisco Unified Border Element (SP Edition) is a highly scalable, carrier-grade session border controller (SBC) that is designed for service providers and that is generally deployed at the border of the enterprise or SP networks to enable the easy deployment and management of VoIP services. Cisco Unified Border Element (SP Edition) is integrated into Cisco routing platforms and can use a large number of router functions to provide a very feature-rich and intelligent SBC application. Formerly known as Integrated Session Border Controller, Cisco Unified Border Element (SP Edition) provides a network-to-network demarcation interface for signaling interworking, media interworking, address and port translations, billing, security, quality of service, call admission control, and bandwidth management. For Cisco IOS XE Release 2.4 and later releases, Cisco Unified Border Element (SP Edition) can operate in two modes or deployment models: unified and distributed. The configuration guide documents the features in the unified mode.

Table 2 lists documents and resources that supplement the Cisco IOS XE software configuration guides and command references.

Table 2 Cisco IOS XE Software Supplementary Documents and Resources

Document Title or Resource	Description
Cisco IOS Master Command List, All Releases	Alphabetical list of all the commands documented in all Cisco IOS XE software releases.
Cisco IOS Debug Command Reference	Alphabetical list of **debug** commands including brief descriptions of use, command syntax, and usage guidelines.
Cisco IOS XE system messages	List of Cisco IOS XE system messages and descriptions. System messages may indicate problems with your system, may be informational only, or may help diagnose problems with communications lines, internal hardware, or the system software.
Release notes and caveats	Information about new and changed features, system requirements, and other useful information about specific software releases; information about defects in specific Cisco IOS XE software releases.
MIBs	Files used for network monitoring. To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator at the following URL: http://www.cisco.com/go/mibs
RFCs	Standards documents maintained by the Internet Engineering Task Force (IETF) that Cisco IOS XE documentation references where applicable. The full text of referenced RFCs may be obtained at the following URL: http://www.rfc-editor.org/

Additional Resources and Documentation Feedback

What's New in Cisco Product Documentation is updated monthly and describes all new and revised Cisco technical documentation. The *What's New in Cisco Product Documentation* publication also provides information about obtaining the following resources:

- Technical documentation
- Cisco product security overview
- Product alerts and field notices
- Technical assistance

Cisco IOS XE software technical documentation includes embedded feedback forms where you can rate documents and provide suggestions for improvement. Your feedback helps us improve our documentation.

Using the Command-Line Interface in Cisco IOS XE Software

Last Updated: December 1, 2009

This document provides basic information about the command-line interface (CLI) in Cisco IOS XE software and how you can use some of the CLI features. This document contains the following sections:

For more information about using the CLI, see "Part 1: Using the Cisco IOS Command-Line Interface (CLI)" of the *Cisco IOS XE Configuration Fundamentals Configuration Guide*.

For information about the software documentation set, see the "About Cisco IOS XE Software Documentation" document.

Initially Configuring a Device

Initially configuring a device varies by platform. For information about performing an initial configuration, see the hardware installation documentation that is provided with the original packaging of the product or go to the Product Support area of Cisco.com at http://www.cisco.com/go/techdocs.

After you have performed the initial configuration and connected the device to your network, you can configure the device by using the console port or a remote access method, such as Telnet or Secure Shell (SSH), to access the CLI or by using the configuration method provided on the device, such as Security Device Manager.

Changing the Default Settings for a Console or AUX Port

There are only two settings that you can change on a console port or an AUX port:

- Change the port speed with the **config-register 0x** command. Changing the port speed is not recommended. The well-known default speed is 9600.
- Change the behavior of the port; for example, by adding a password or changing the timeout value.

Note The AUX port on the Route Processor (RP) installed in a Cisco ASR 1000 series router does not serve any useful customer purpose and should be accessed only under the advisement of a customer support representative.

Using the CLI

This section describes the following topics:

Understanding Command Modes

The CLI command mode structure is hierarchical, and each mode supports a set of specific commands. This section describes the most common of the many modes that exist.

Table 1 lists common command modes with associated CLI prompts, access and exit methods, and a brief description of how each mode is used.

Table 1 CLI Command Modes

Command Mode	Access Method	Prompt	Exit Method	Mode Usage
User EXEC	Log in.	`Router>`	Issue the **logout** or **exit** command.	• Change terminal settings. • Perform basic tests. • Display device status.
Privileged EXEC	From user EXEC mode, issue the **enable** command.	`Router#`	Issue the **disable** command or the **exit** command to return to user EXEC mode.	• Issue **show** and **debug** commands. • Copy images to the device. • Reload the device. • Manage device configuration files. • Manage device file systems.
Global configuration	From privileged EXEC mode, issue the **configure terminal** command.	`Router(config)#`	Issue the **exit** command or the **end** command to return to privileged EXEC mode.	Configure the device.
Interface configuration	From global configuration mode, issue the **interface** command.	`Router(config-if)#`	Issue the **exit** command to return to global configuration mode or the **end** command to return to privileged EXEC mode.	Configure individual interfaces.
Line configuration	From global configuration mode, issue the **line vty** or **line console** command.	`Router(config-line)#`	Issue the **exit** command to return to global configuration mode or the **end** command to return to privileged EXEC mode.	Configure individual terminal lines.

Table 1 CLI Command Modes (continued)

Command Mode	Access Method	Prompt	Exit Method	Mode Usage
ROM monitor	From privileged EXEC mode, issue the **reload** command. Press the **Break** key during the first 60 seconds while the system is booting.	`rommon # >` The # symbol represents the line number and increments at each prompt.	Issue the **continue** command.	• Run as the default operating mode when a valid image cannot be loaded. • Access the fall-back procedure for loading an image when the device lacks a valid image and cannot be booted. • Perform password recovery when a CTRL-Break sequence is issued within 60 seconds of a power-on or reload event.
Diagnostic	The router boots or enters diagnostic mode in the following scenarios. When a Cisco IOS XE process or processes fail, in most scenarios the router will reload. • A user-configured access policy was configured using the **transport-map** command, which directed the user into diagnostic mode. • The router was accessed using an RP auxiliary port. • A break signal (**Ctrl-C**, **Ctrl-Shift-6**, or the **send break** command) was entered, and the router was configured to enter diagnostic mode when the break signal was received.	`Router(diag)#`	If a Cisco IOS XE process failure is the reason for entering diagnostic mode, the failure must be resolved and the router must be rebooted to exit diagnostic mode. If the router is in diagnostic mode because of a transport-map configuration, access the router through another port or use a method that is configured to connect to the Cisco IOS XE CLI. If the RP auxiliary port was used to access the router, use another port for access. Accessing the router through the auxiliary port is not useful for customer purposes.	• Inspect various states on the router, including the Cisco IOS XE state. • Replace or roll back the configuration. • Provide methods of restarting the Cisco IOS XE software or other processes. • Reboot hardware, such as the entire router, an RP, an ESP, a SIP, a SPA, or other hardware components. • Transfer files into or off of the router using remote access methods such as FTP, TFTP, and SCP.

EXEC commands are not saved when the software reboots. Commands that you issue in a configuration mode can be saved to the startup configuration. If you save the running configuration to the startup configuration, these commands will execute when the software is rebooted. Global configuration mode is the highest level of configuration mode. From global configuration mode, you can enter a variety of other configuration modes, including protocol-specific modes.

ROM monitor mode is a separate mode that is used when the software cannot load properly. If a valid software image is not found when the software boots or if the configuration file is corrupted at startup, the software might enter ROM monitor mode. Use the question symbol (?) to view the commands that you can use while the device is in ROM monitor mode.

```
rommon 1 > ?
alias               set and display aliases command
boot                boot up an external process
confreg             configuration register utility
cont                continue executing a downloaded image
context             display the context of a loaded image
cookie              display contents of cookie PROM in hex
.
.
.
rommon 2 >
```

The following example shows how the command prompt changes to indicate a different command mode:

```
Router> enable
Router# configure terminal
Router(config)# interface ethernet 1/1
Router(config-if)# ethernet
Router(config-line)# exit
Router(config)# end
Router#
```

Note A keyboard alternative to the **end** command is Ctrl-Z.

Using the Interactive Help Feature

The CLI includes an interactive Help feature. Table 2 describes how to use the Help feature.

Table 2 CLI Interactive Help Commands

Command	Purpose
help	Provides a brief description of the Help feature in any command mode.
?	Lists all commands available for a particular command mode.
*partial command***?**	Provides a list of commands that begin with the character string (no space between the command and the question mark).
partial command<**Tab**>	Completes a partial command name (no space between the command and <Tab>).
command **?**	Lists the keywords, arguments, or both associated with the command (space between the command and the question mark).
command keyword **?**	Lists the arguments that are associated with the keyword (space between the keyword and the question mark).

The following examples show how to use the help commands:

help

```
Router> help

Help may be requested at any point in a command by entering a question mark '?'. If nothing
matches, the help list will be empty and you must backup until entering a '?' shows the
available options.

Two styles of help are provided:

1. Full help is available when you are ready to enter a command argument (e.g. 'show ?')
and describes each possible argument.

2. Partial help is provided when an abbreviated argument is entered and you want to know
what arguments match the input (e.g. 'show pr?'.)
```

?

```
Router# ?
Exec commands:
  access-enable        Create a temporary access-List entry
  access-profile       Apply user-profile to interface
  access-template      Create a temporary access-List entry
  alps                 ALPS exec commands
  archive              manage archive files
<snip>
```

***partial command*?**

```
Router(config)# zo?
zone  zone-pair
```

***partial command*<Tab>**

```
Router(config)# we<Tab> webvpn
```

***command* ?**

```
Router(config-if)# pppoe ?
  enable        Enable pppoe
  max-sessions  Maximum PPPOE sessions
```

***command keyword* ?**

```
Router(config-if)# pppoe enable ?
  group  attach a BBA group
  <cr>
```

Understanding Command Syntax

Command syntax is the format in which a command should be entered in the CLI. Commands include the name of the command, keywords, and arguments. Keywords are alphanumeric strings that are used literally. Arguments are placeholders for values that a user must supply. Keywords and arguments may be required or optional.

Specific conventions convey information about syntax and command elements. Table 3 describes these conventions.

Table 3 CLI Syntax Conventions

Symbol/Text	Function	Notes
< > (angle brackets)	Indicate that the option is an argument.	Sometimes arguments are displayed without angle brackets.
A.B.C.D.	Indicates that you must enter a dotted decimal IP address.	Angle brackets (< >) are not always used to indicate that an IP address is an argument.
WORD (all capital letters)	Indicates that you must enter one word.	Angle brackets (< >) are not always used to indicate that a WORD is an argument.
LINE (all capital letters)	Indicates that you must enter more than one word.	Angle brackets (< >) are not always used to indicate that a LINE is an argument.
<cr> (carriage return)	Indicates the end of the list of available keywords and arguments, and also indicates when keywords and arguments are optional. When <cr> is the only option, you have reached the end of the branch or the end of the command if the command has only one branch.	—

The following examples show syntax conventions:

```
Router(config)# ethernet cfm domain ?
  WORD  domain name

Router(config)# ethernet cfm domain dname ?
  level

Router(config)# ethernet cfm domain dname level ?
  <0-7>  maintenance level number

Router(config)# ethernet cfm domain dname level 7 ?
  <cr>

Router(config)# snmp-server file-transfer access-group 10 ?
  protocol  protocol options
  <cr>

Router(config)# logging host ?
  Hostname or A.B.C.D  IP address of the syslog server
  ipv6                 Configure IPv6 syslog server
```

Understanding Enable and Enable Secret Passwords

Some privileged EXEC commands are used for actions that impact the system, and it is recommended that you set a password for these commands to prevent unauthorized use. Two types of passwords, enable (not encrypted) and enable secret (encrypted), can be set. The following commands set these passwords and are issued in global configuration mode:

- **enable** *password*
- **enable secret** *password*

Using an enable secret password is recommended because it is encrypted and more secure than the enable password. When you use an enable secret password, text is encrypted (unreadable) before it is written to the config.text file. When you use an enable password, the text is written as entered (readable) to the config.text file.

Each type of password is case sensitive, can contain from 1 to 25 uppercase and lowercase alphanumeric characters, and can start with a number. Spaces are also valid password characters; for example, "two words" is a valid password. Leading spaces are ignored, but trailing spaces are recognized.

Note Both password commands have numeric keywords that are single integer values. If you choose a number for the first character of your password followed by a space, the system will read the number as if it were the numeric keyword and not as part of your password.

When both passwords are set, the enable secret password takes precedence over the enable password.

To remove a password, use the **no** form of the commands: **no enable** *password* or **no enable secret** *password*.

For more information about password recovery procedures for Cisco products, see http://www.cisco.com/en/US/products/sw/iosswrel/ps1831/products_tech_note09186a00801746e6.shtml.

Using the Command History Feature

The command history feature saves the commands that you enter during a session in a command history buffer. The default number of commands saved is 10, but the number is configurable within the range of 0 to 256. This command history feature is particularly useful for recalling long or complex commands.

To change the number of commands saved in the history buffer for a terminal session, issue the **terminal history size** command:

```
Router# terminal history size num
```

A command history buffer is also available in line configuration mode with the same default and configuration options. To set the command history buffer size for a terminal session in line configuration mode, issue the **history** command:

```
Router(config-line)# history [size num]
```

To recall commands from the history buffer, use the following methods:

- Press Ctrl-P or the Up Arrow key—Recalls commands beginning with the most recent command. Repeat the key sequence to recall successively older commands.
- Press Ctrl-N or the Down Arrow key—Recalls the most recent commands in the history buffer after they have been recalled using Ctrl-P or the Up Arrow key. Repeat the key sequence to recall successively more recent commands.

Note The arrow keys function only on ANSI-compatible terminals such as the VT100.

- Issue the **show history** command in user EXEC or privileged EXEC mode—Lists the most recent commands that you entered. The number of commands that are displayed is determined by the setting of the **terminal history size** and **history** commands.

  The command history feature is enabled by default. To disable this feature for a terminal session, issue the **terminal no history** command in user EXEC or privileged EXEC mode or the **no history** command in line configuration mode.

Abbreviating Commands

Typing a complete command name is not always required for the command to execute. The CLI recognizes an abbreviated command when the abbreviation contains enough characters to uniquely identify the command. For example, the **show version** command can be abbreviated as **sh ver**. It cannot be abbreviated as **s ver** because **s** could mean **show**, **set**, or **systat**. The **sh v** abbreviation also is not valid because the **show** command has **vrrp** as a keyword in addition to **version**.

Using Aliases for CLI Commands

To save time and the repetition of entering the same command multiple times, you can use a command alias. An alias can be configured to do anything that can be done at the command line, but an alias cannot move between modes, type in passwords, or perform any interactive functions.

Table 4 shows the default command aliases.

Table 4 Default Command Aliases

Command Alias	Original Command
h	help
lo	logout
p	ping
s	show
u or **un**	undebug
w	where

To create a command alias, issue the **alias** command in global configuration mode. The syntax of the command is **alias** *mode command-alias original-command*. Following are some examples:

- Router(config)# **alias exec prt partition**—privileged EXEC mode
- Router(config)# **alias configure sb source-bridge**—global configuration mode
- Router(config)# **alias interface rl rate-limit**—interface configuration mode

To view both default and user-created aliases, issue the **show alias** command.

For more information about the **alias** command, see http://www.cisco.com/en/US/docs/ios/fundamentals/command/reference/cf_book.html.

Using the no and default Forms of Commands

Most configuration commands have a **no** form that is used to reset a command to its default value or disable a feature or function. For example, the **ip routing** command is enabled by default. To disable this command, you would issue the **no ip routing** command. To re-enable IP routing, you would issue the **ip routing** command.

Configuration commands may also have a **default** form, which returns the command settings to their default values. For commands that are disabled by default, using the **default** form has the same effect as using the **no** form of the command. For commands that are enabled by default and have default settings, the **default** form enables the command and returns the settings to their default values.

The **no** form is documented in the command pages of command references. The **default** form is generally documented in the command pages only when the **default** form performs a different function than the plain and **no** forms of the command. To see what **default** commands are available on your system, enter **default ?** in the appropriate command mode.

Using the debug Command

A **debug** command produces extensive output that helps you troubleshoot problems in your network. These commands are available for many features and functions within Cisco IOS XE software. Some **debug** commands are **debug all**, **debug aaa accounting**, and **debug mpls packets**. To use **debug** commands during a Telnet session with a device, you must first enter the **terminal monitor** command. To turn off debugging completely, you must enter the **undebug all** command.

For more information about **debug** commands, see the *Cisco IOS Debug Command Reference* at http://www.cisco.com/en/US/docs/ios/debug/command/reference/db_book.html.

Caution Debugging is a high priority and high CPU utilization process that can render your device unusable. Use **debug** commands only to troubleshoot specific problems. The best times to run debugging are during periods of low network traffic and when few users are interacting with the network. Debugging during these periods decreases the likelihood that the **debug** command processing overhead will affect network performance or user access or response times.

Filtering Output Using Output Modifiers

Many commands produce lengthy output that may use several screens to display. You can use output modifiers to filter this output to show only the information that you want to see.

The following three output modifiers are available:

- **begin** *regular-expression*—Displays the first line in which a match of the regular expression is found and all lines that follow.
- **include** *regular-expression*—Displays all lines in which a match of the regular expression is found.
- **exclude** *regular-expression*—Displays all lines except those in which a match of the regular expression is found.

To use one of these output modifiers, type the command followed by the pipe symbol (|), the modifier, and the regular expression that you want to search for or filter. A regular expression is a case-sensitive alphanumeric pattern. It can be a single character or number, a phrase, or a more complex string.

The following example illustrates how to filter output of the **show interface** command to display only lines that include the expression “protocol.”

```
Router# show interface | include protocol

FastEthernet0/0 is up, line protocol is up
Serial4/0 is up, line protocol is up
Serial4/1 is up, line protocol is up
Serial4/2 is administratively down, line protocol is down
Serial4/3 is administratively down, line protocol is down
```

Understanding CLI Error Messages

You may encounter some error messages while using the CLI. Table 5 shows the common CLI error messages.

Table 5 Common CLI Error Messages

Error Message	Meaning	How to Get Help
% Ambiguous command: “show con”	You did not enter enough characters for the command to be recognized.	Reenter the command followed by a space and a question mark (?). The keywords that you are allowed to enter for the command appear.
% Incomplete command.	You did not enter all the keywords or values required by the command.	Reenter the command followed by a space and a question mark (?). The keywords that you are allowed to enter for the command appear.
% Invalid input detected at “^” marker.	You entered the command incorrectly. The caret (^) marks the point of the error.	Enter a question mark (?) to display all the commands that are available in this command mode. The keywords that you are allowed to enter for the command appear.

For more system error messages, see the *System Messages for Cisco IOS XE* document.

Saving Changes to a Configuration

To save changes that you made to the configuration of a device, you must issue the **copy running-config startup-config** command or the **copy system:running-config nvram:startup-config** command. When you issue these commands, the configuration changes that you made are saved to the startup configuration and saved when the software reloads or power to the device is turned off or interrupted. The following example shows the syntax of the **copy running-config startup-config** command:

```
Router# copy running-config startup-config
Destination filename [startup-config]?
```

You press Enter to accept the startup-config filename (the default), or type a new filename and then press Enter to accept that name. The following output is displayed indicating that the configuration was saved:

```
Building configuration...
[OK]
Router#
```

On most platforms, the configuration is saved to NVRAM. On platforms with a Class A flash file system, the configuration is saved to the location specified by the CONFIG_FILE environment variable. The CONFIG_FILE variable defaults to NVRAM.

Additional Information

- "Part 1: Using the Cisco IOS Command-Line Interface (CLI)" of the *Cisco IOS XE Configuration Fundamentals Configuration Guide*

  http://www.cisco.com/en/US/docs/ios/ios_xe/fundamentals/configuration/guide/2_xe/cf_xe_book.html

  or

  "Using Cisco IOS XE Software" chapter of the *Cisco ASR 1000 Series Aggregation Services Routers Software Configuration Guide*

  http://www.cisco.com/en/US/docs/routers/asr1000/configuration/guide/chassis/Using_CLI.html
- Cisco Product Support Resources

  http://www.cisco.com/go/techdocs
- Support area on Cisco.com (also search for documentation by task or product)

  http://www.cisco.com/en/US/support/index.html
- Software Download Center (downloads; tools; licensing, registration, advisory, and general information) (requires Cisco.com user ID and password)

  http://www.cisco.com/kobayashi/sw-center/
- Error Message Decoder, a tool to help you research and resolve error messages for Cisco IOS XE software

  http://www.cisco.com/pcgi-bin/Support/Errordecoder/index.cgi

- Command Lookup Tool, a tool to help you find detailed descriptions of Cisco IOS XE commands (requires Cisco.com user ID and password)

  http://tools.cisco.com/Support/CLILookup

- Output Interpreter, a troubleshooting tool that analyzes command output of supported **show** commands

  https://www.cisco.com/pcgi-bin/Support/OutputInterpreter/home.pl\

Configuring Physical Interfaces

First Published: May 12, 2009
Last Updated: May 12, 2009

The Cisco ASR 1000 Series Aggregation Services Routers support many different types of physical (hardware) interfaces such as Gigabit Ethernet, Packet over SONET (POS), and serial shared port adapter (SPA) interfaces. For hardware technical descriptions and information about installing interfaces, refer to the hardware installation and configuration publication for your product.

Configuration Information

- For information about using the Gigabit Ethernet Management Ethernet interface, see the "Using the Management Ethernet Interface" chapter of the *Cisco ASR 1000 Series Aggregation Services Routers Software Configuration Guide* at:

  http://www.cisco.com/en/US/docs/routers/asr1000/configuration/guide/chassis/asrswcfg.html

- For information about configuring and troubleshooting SPA interface processors (SIPs) and SPAs that are supported on a Cisco ASR 1000 Series Aggregation Services Router, see the *Cisco ASR 1000 Series Aggregation Services Routers SIP and SPA Software Configuration Guide* at:

  http://cisco.com/en/US/docs/interfaces_modules/shared_port_adapters/configuration/ASR1000/ASRspasw.html

Command Reference Information

- Complete descriptions of the commands used to configure interfaces are included in the *Cisco IOS Interface and Hardware Component Command Reference* at:

  http://www.cisco.com/en/US/docs/ios/interface/command/reference/ir_book.html

- For information about other Cisco IOS XE commands, use the Command Lookup Tool at http://tools.cisco.com/Support/CLILookup or the *Cisco IOS Master Command List, All Releases*, at http://www.cisco.com/en/US/docs/ios/mcl/allreleasemcl/all_book.html.

Americas Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

Configuring Virtual Interfaces

First Published: May 12, 2009
Last Updated: May 12, 2009

Virtual interfaces are software-based interfaces that you create in the memory of the networking device using Cisco IOS XE commands. Virtual interfaces do not have a hardware component such as the RJ-45 female port on a 100BASE-T Fast Ethernet network interface card. This module describes the four common types of virtual, or logical, interfaces that can be configured using Cisco IOS XE software:

- Loopback interfaces
- Null interfaces
- Subinterfaces
- Tunnel interfaces

Contents

Prerequisites for Configuring Virtual Interfaces

Before virtual interfaces can be used in your network, you must have some physical (hardware) interfaces configured and you must be able to communicate between the networking devices on which you wish to use virtual interfaces.

Americas Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

Information About Configuring Virtual Interfaces

To configure virtual interfaces, you should understand the following concepts:

Virtual Interfaces

Virtual interfaces are network interfaces that are not associated with a physical interface. Physical interfaces have some form of physical element—for example, an RJ-45 male connector on an Ethernet cable. Virtual interfaces exist only in software; there are no physical elements. You identify an individual virtual interface using a numerical ID after the virtual interface name. For example: loopback 0, tunnel 1, and fastethernet 0/0/0.1. The ID is unique per virtual interface type to make the entire name string unique; for example both a loopback 0 interface and a null 0 interface can exist, but two loopback 0 interfaces cannot exist in a single networking device.

Cisco IOS XE software supports four types of virtual interfaces:

- Loopback
- Null
- Subinterface
- Tunnel

Benefits of Virtual Interfaces

- A loopback interface can provide a stable interface on which you can assign a Layer 3 address such as an IP or IPX address. This address can be configured as the source address when the networking device needs to send data for protocols such as NetFlow or Cisco Discovery Protocol (CDP) to another device in your network and you want the receiving device to always see the same source IP address from the networking device. This is an issue in networks with multiple equal-cost paths because under normal circumstances the packets that are generated by a networking device use the IP address from the outbound interface as the source address for the packets and because in a network with two or more equal-cost paths from the networking device to the receiving host each packet might use a different outbound interface.
- A null interface provides an alternative method of filtering without the overhead involved with using access lists. For example, instead of creating an outbound access list that prevents traffic to a destination network from being transmitted out an interface, you can configure a static route for the destination network that points to the null interface.

- Subinterfaces were invented as a method of virtually subdividing a physical interface into two or more interfaces so that the IP routing protocols would see the network connection to each remote networking device as a separate physical interface even though the subinterfaces share a common physical interface. One of the first uses of subinterfaces was to resolve the problem with split horizon on Frame Relay WANs.
- The following are several situations in which tunneling (encapsulating traffic in another protocol) is useful:
  - To enable multiprotocol local networks over a single-protocol backbone.
  - To provide workarounds for networks that use protocols that have limited hop counts; for example, RIP version 1, AppleTalk.
  - To connect discontiguous subnetworks.
  - To allow virtual private networks across WANs.

Loopback Interfaces

You can specify a software-only interface called a loopback interface to emulate a physical interface. Loopback interfaces are supported on all platforms. A loopback interface is a virtual interface on a Cisco router that remains up (active) after you issue the **no shutdown** command until you disable it with the **shutdown** command. Unlike subinterfaces, loopback interfaces are independent of the state of any physical interface.

The loopback interface can be considered stable because once you enable it, it will remain up until you shut it down. This makes loopback interfaces ideal for assigning Layer 3 addresses such as IP addresses when you want a single address as a reference that is independent of the status of any physical interfaces in the networking device. A good example of this is using the IP address of a loopback interface as the IP address for the DNS host address for the networking device. Before loopback interfaces were available, network administrators had to configure a DNS host entry for every interface on a router that had an IP address assigned to it because they could never be certain which interface IP address might be available at any given time for managing the router. In the sample interface configuration and DNS entries for Router A shown below, you can see that there is a DNS entry for each interface.

Router A Interface Configuration Before Loopback

```
GigabitEthernet0 10.10.10.1 255.255.255.0
GigabitEthernet1 10.10.11.1 255.255.255.0
GigabitEthernet2 10.10.12.1 255.255.255.0
GigabitEthernet3 10.10.13.1 255.255.255.0
GigabitEthernet4 10.10.14.1 255.255.255.0
GigabitEthernet5 10.10.15.1 255.255.255.0
```

Router A DNS Entries Before Loopback

```
RouterA   IN  A  10.10.10.1
          IN  A  10.10.11.1
          IN  A  10.10.12.1
          IN  A  10.10.13.1
          IN  A  10.10.14.1
          IN  A  10.10.15.1
```

Interfaces on networking devices can fail, and they can also be taken out of service for maintenance. If any of the interfaces in Router A fails or is taken out of service, another networking device will not be able to access that interface. When you configure a networking device with a loopback interface and assign it an IP address that is advertised throughout the network, the networking device will be reachable by using this IP address as long as the networking device has at least one network interface capable of

sending and receiving IP traffic. In the sample interface configuration and DNS entries for Router A after a loopback interface is configured, you can see that there is now only one DNS entry that can be used to reach the router over any of its physical interfaces.

Router A Interface Configuration After Loopback

```
Loopback 172.16.78.1 255.255.255.0
GigabitEthernet0 10.10.10.1 255.255.255.0
GigabitEthernet1 10.10.11.1 255.255.255.0
GigabitEthernet2 10.10.12.1 255.255.255.0
GigabitEthernet3 10.10.13.1 255.255.255.0
GigabitEthernet4 10.10.14.1 255.255.255.0
GigabitEthernet5 10.10.15.1 255.255.255.0
```

Router A DNS Entries After Loopback

```
RouterA   IN  A  172.16.78.1
```

The configured IP address of the loopback interface—172.16.78.1—can be used as the source address for packets generated by the router and forwarded to networking management applications and routing protocols. Unless this loopback interface is explicitly shut down, it is always reachable.

You can use the loopback interface as the termination address for OSPF or BGP sessions. A loopback interface can also be used to establish a Telnet session from the console port of the device to its auxiliary port when all other interfaces are down. In applications where other routers or access servers attempt to reach this loopback interface, you should configure a routing protocol to distribute the subnet assigned to the loopback address.

IP Packets routed to the loopback interface are rerouted back to the router or access server and processed locally. IP packets routed out the loopback interface but not destined to the loopback interface are dropped. Under these two conditions, the loopback interface can behave like a null interface.

Loopback Interfaces Versus Loopback Mode

Loopback interfaces provide a stable source interface to ensure that the IP address assigned to the interface is always reachable as long as the IP routing protocols continue to advertise the subnet assigned to the loopback interface. Loopback mode, however, is used to test and diagnose issues with WAN (serial) links such as bit loss or data corruption. The idea is to configure a loop to return the data packets that were received by the interface back out the same interface to the device that originated the traffic. Loopback mode is used to troubleshoot problems by checking that the data packets are returned in the same condition in which they were sent. Errors in the data packets indicate a problem with the WAN infrastructure. Many types of serial interfaces have their own form of loopback command syntax that is entered under interface or controller configuration mode.

Null Interfaces

The null interface is a virtual network interface that is similar to the loopback interface. Whereas traffic to the loopback interface is directed to the router itself, traffic sent to the null interface is discarded. This interface is always up and can never forward or receive traffic; encapsulation always fails. The null interface functions similarly to the null devices available on most operating systems.

Null interfaces are used as a low-overhead method of discarding unnecessary network traffic. For example, if you do not want your network users to be able to reach certain IP subnets, you can create static IP routes for the subnets that point to the null interface of a networking device. Using the static IP

routes takes less CPU time for the networking device than using IP access lists. The static-route configuration is also easier to configure than IP access lists because it is done in global configuration mode instead of in interface configuration mode.

The null interface may not be configured with an address. Traffic can be sent to this interface only by configuring a static route where the next hop is the null interface—represented by Null 0. One example of configuring the next hop to be the null interface is to create a route to an aggregate network that can then be announced through the BGP, or to ensure that traffic to a particular range of addresses is not propagated through the router, perhaps for security purposes.

The router always has a single null interface. By default, a packet sent to the null interface causes the router to respond by sending an ICMP unreachable message to the source IP address of the packet. You can configure the router either to send these responses or to drop the packets silently.

Subinterfaces

Subinterfaces are associated with physical interfaces. Subinterfaces are enabled when the physical interface with which they are associated is enabled, and subinterfaces are disabled when the physical interface is shut down.

Note Subinterfaces can be enabled and shut down independently of the physical port with which they are associated. However, you cannot enable a subinterface of a physical interface that has been shut down.

Subinterfaces are created by subdividing the physical interface into two or more virtual interfaces on which you can assign unique Layer 3 network addresses such as IP subnets. One of the first uses of subinterfaces was to resolve the problem with split horizon on Frame Relay WANs. Split horizon is a behavior associated with IP routing protocols such as RIP in which IP subnets are not advertised back out the same physical interface that they were learned over. Split horizon was implemented to prevent routing loops in IP networks. A routing loop can be created when the networking devices at both ends of a network connection advertise the same IP routes to each other. Split horizon was an issue for Frame Relay multipoint network interfaces—interfaces that connect to two or more remote networking devices over a single physical interface—because the default behavior of many networking devices was to implement split horizon, which means that the networking device did not advertise the IP routes that were learned over an interface back out the interface to other devices that were also reachable via the same physical interface. Subinterfaces were invented as a method of virtually subdividing a physical interface into two or more interfaces so that the IP routing protocols would see the network connection to each remote networking device as a separate physical interface even though the subinterfaces share a common physical interface. Although TCP/IP now disables split horizon limitations by default, protocols such as AppleTalk and IPX are still constrained by split horizon.

Subinterfaces are identified by a prefix that consists of the hardware interface descriptor (IDB) followed by a period and then by a number that is unique for that prefix. The full subinterface number must be unique to the networking device. For example, the first subinterface for GigabitEthernet interface 0/0/0 might be named GigabitEthernet 0/0/0.1 where .1 indicates the subinterface.

Tunnel Interfaces

Tunneling provides a way to encapsulate arbitrary packets inside a transport protocol. Tunnels are implemented as a virtual interface to provide a simple interface for configuration. The tunnel interface is not tied to specific “passenger” or “transport” protocols, but, rather, it is an architecture that is designed to provide the services necessary to implement any standard point-to-point encapsulation scheme.

There are several ways to implement tunnel interfaces depending on the connectivity that you need to provide. One common use for tunnels is to carry data traffic for a network protocol such as IPX over devices in your network that do not support IPX. For instance, if your network uses IPX in sites at the edge of your network but not in the core of your network, you can connect the IPX sites at the network edges by tunneling IPX in IP over the core of the network.

For more details about the various types of tunneling techniques available using Cisco IOS XE software, see the “Implementing Tunnels” module of the *Cisco IOS XE Interface and Hardware Component Configuration Guide, Release 2*.

How to Configure Virtual Interfaces

This section contains the following tasks:

Configuring a Loopback Interface

This task explains how to configure a loopback interface. A loopback interface can be considered stable because once you enable it, it will remain up until you shut it down. This makes loopback interfaces ideal for assigning Layer 3 addresses such as IP addresses when you want to have a single address to use as a reference that is independent of the status of any of the physical interfaces in the networking device.

Prerequisites

The IP address for the loopback interface must be unique and not in use by another interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface loopback** *number*
4. **ip address** *ip-address mask* [**secondary**]
5. **end**
6. **show interfaces loopback** *number*
7. **exit**

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 loopback** *number* **Example:** `Router(config)# interface loopback 0`	Specifies a loopback interface and enters interface configuration mode. • Use the *number* argument to specify the number of the loopback interface that you want to create or configure. **Note** There is no limit on the number of loopback interfaces that you can create.
Step 4	**ip address** *ip-address mask* [**secondary**] **Example:** `Router(config-if)# ip address 10.20.1.2 255.255.255.0`	Specifies an IP address for the loopback interface and enables IP processing on the interface. • Use the *ip-address* and *mask* arguments to specify the subnet for the loopback address.
Step 5	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 6	**show interfaces loopback** *number* **Example:** `Router# show interfaces loopback 0`	(Optional) Displays information about loopback interfaces. • Use the *number* argument to display information about one particular loopback interface. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS Interface and Hardware Component Command Reference*.
Step 7	**exit** **Example:** `Router# exit`	Exits privileged EXEC mode.

Examples

The following is sample output for the **show interfaces loopback** command.

```
Router# show interfaces loopback

Loopback0 is up, line protocol is up
  Hardware is Loopback
  Internet address is 10.20.1.2/24
  MTU 1514 bytes, BW 8000000 Kbit, DLY 5000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation LOOPBACK, loopback not set
  Last input never, output never, output hang never
  Last clearing of "show interface" counters never
  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
     0 packets input, 0 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     0 packets output, 0 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 output buffer failures, 0 output buffers swapped out
```

Configuring a Null Interface

This task explains how to configure a null interface. Null interfaces provide an alternative method to access control lists for filtering traffic. All unwanted traffic can be directed to the null interface; the null interface cannot receive or forward traffic, or allow its traffic to be encapsulated.

The only interface configuration command that you can specify for the null interface is the **no ip unreachables** command.

ICMP Unreachable Messages from Null Interfaces

By default, a packet sent to the null interface causes the router to respond by sending an Internet Control Message Protocol (ICMP) unreachable message to the source IP address of the packet. You can configure the router either to send these responses or to drop the packets silently.

To disable the sending of ICMP unreachable messages in response to packets sent to the null interface, use the **no ip unreachables** command in interface configuration mode. To reenable the sending of ICMP unreachable messages in response to packets sent to the null interface, use the **ip unreachables** command in interface configuration mode.

Restrictions

Only one null interface can be configured on each networking device.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface null** *number*

4. **no ip unreachables**
5. **end**
6. **show interfaces null** [*number*] [**accounting**]

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 null** *number* **Example:** `Router(config)# interface null 0`	Specifies a null interface and number, and enters interface configuration mode. • The number argument is always 0.
Step 4	**no ip unreachables** **Example:** `Router(config-if)# no ip unreachables`	Prevents the generation of ICMP unreachable messages on an interface. • This command affects all types of ICMP unreachable messages.
Step 5	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 6	**show interfaces null** [*number*] [**accounting**] **Example:** `Router# show interfaces null 0`	(Optional) Displays information about null interfaces. • For null interfaces, the *number* argument is always 0. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS Interface and Hardware Component Command Reference*.

Examples

The following is sample output for the **show interfaces null** command.

```
Router# show interfaces null

Null0 is up, line protocol is up
  Hardware is Unknown
  MTU 1500 bytes, BW 10000000 Kbit, DLY 0 usec,
     reliability 0/255, txload 0/255, rxload 0/255
  Encapsulation ARPA, loopback not set
  Last input never, output never, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  5 minute input rate 0 bits/sec, 0 packets/sec
```

```
5 minute output rate 0 bits/sec, 0 packets/sec
   0 packets input, 0 bytes, 0 no buffer
   Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
   0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
   0 packets output, 0 bytes, 0 underruns
   0 output errors, 0 collisions, 0 interface resets

   0 output buffer failures, 0 output buffers swapped out
```

Configuration Examples for Virtual Interfaces

This section contains the following examples:

- Configuring a Loopback Interface: Example, page 10
- Configuring a Null Interface: Example, page 10

Configuring a Loopback Interface: Example

The following example shows how to configure a loopback interface, loopback 0.

```
interface loopback 0
 ip address 10.20.1.2 255.255.255.0
 end
```

Configuring a Null Interface: Example

The following example shows how to configure a null interface and to drop the ICMP unreachable messages. All packets sent to the null interface are dropped and in this example, the ICMP messages usually sent in response to packets being sent to the null interface are dropped.

```
interface null 0
 no ip unreachables
 end
```

Where to Go Next

- If you want to implement tunnels in your network, see the "Implementing Tunnels" module of the *Cisco IOS XE Interface and Hardware Component Configuration Guide, Release 2*.
- If you want to implement physical (hardware) interfaces (such as Gigabit Ethernet or serial interfaces) in your network, see the "Configuring Physical Interfaces" module of the *Cisco IOS XE Interface and Hardware Component Configuration Guide, Release 2*.

Additional References

The following sections provide references related to virtual interfaces.

Related Documents

Related Topic	Document Title
Interface commands: complete command syntax, command mode, defaults, command history, usage guidelines, and examples	*Cisco IOS Interface and Hardware Component Command Reference*
All Cisco IOS XE commands	• *Cisco IOS Master Command List, All Releases*. • Command Lookup Tool
Cisco IOS XE Interface and Hardware Component configuration modules	*Cisco IOS XE Interface and Hardware Component Configuration Guide, Release 2*
Configuration example showing how to use loopback interfaces with BGP	*Sample Configuration for iBGP and eBGP With or Without a Loopback Address*

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	—

MIBs

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

RFCs

RFC	Title
No new or modified RFCs are supported, and support for existing RFCs has not been modified.	—

Technical Assistance

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

Implementing Tunnels

First Published: May 02, 2005
Last Updated: June 30, 2009

This module describes the various types of tunneling techniques that are available using Cisco IOS XE software. Configuration details and examples are provided for the tunnel types that use physical or virtual interfaces. Many tunneling techniques are implemented using technology-specific commands, and links are provided to the appropriate technology modules.

Tunneling provides a way to encapsulate arbitrary packets inside a transport protocol. Tunnels are implemented as a virtual interface to provide a simple interface for configuration. The tunnel interface is not tied to specific "passenger" or "transport" protocols, but, rather, it is an architecture that is designed to provide the services necessary to implement any standard point-to-point encapsulation scheme.

Finding Feature Information

For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "Feature Information for Implementing Tunnels" section on page 34.

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

Contents

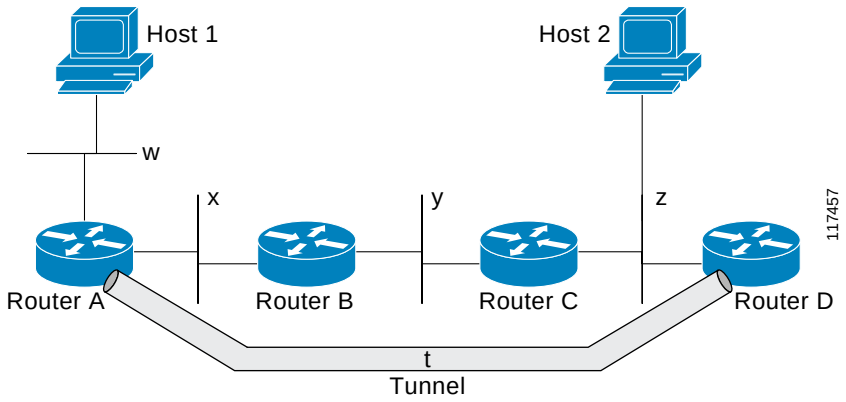
Americas Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

Restrictions for Implementing Tunnels

- It is important to allow the tunnel protocol through a firewall and to allow it to pass access control list (ACL) checking.
- Multiple point-to-point tunnels can saturate the physical link with routing information if the bandwidth is not configured correctly on the tunnel interface.
- A tunnel looks like one hop, and routing protocols may prefer a tunnel over a multihop physical path. This can be deceptive because the tunnel, although it may look like a single hop, may traverse a slower path than a multihop link. A tunnel is as robust and fast, or as unreliable and slow, as the links that it actually traverses. Routing protocols that make their decisions on the sole basis of hop count will often prefer a tunnel over a set of physical links. A tunnel might appear to be a one-hop, point-to-point link and have the lowest-cost path, but may actually cost more in terms of latency than an alternative physical topology.

  For example, in the topology shown in Figure 1, packets from Host 1 will appear to travel across networks w, t, and z to get to Host 2 instead of taking the path w, x, y, and z because the tunnel hop count appears shorter. In fact, the packets going through the tunnel will still be traveling across Router A, B, and C, but they must also travel to Router D before coming back to Router C.

Figure 1 Tunnel Precautions: Hop Counts



- If routing is not carefully configured, the tunnel may have a recursive routing problem. When the best path to the “tunnel destination” is via the tunnel itself, recursive routing causes the tunnel interface to flap. To avoid recursive routing problems, keep the control-plane routing separate from the tunnel routing using the following methods:
  - Use a different autonomous system number or tag.
  - Use a different routing protocol.
  - Use static routes to override the first hop (but watch for routing loops).

  When you have recursive routing to the tunnel destination, the following error appears:

```
%TUN-RECURDOWN Interface Tunnel 0
temporarily disabled due to recursive routing
```

Information About Implementing Tunnels

To configure tunnels, you should understand the following concepts:

Tunneling Versus Encapsulation

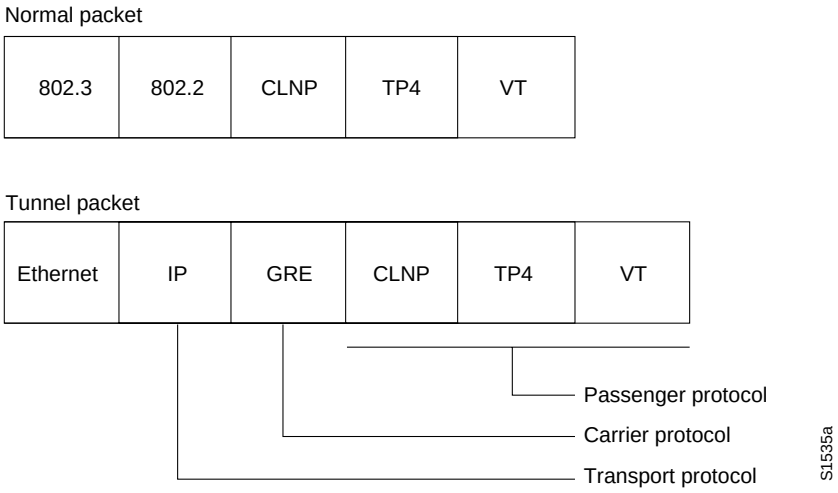
To understand how tunnels work, it is important to distinguish between the concepts of encapsulation and tunneling. Encapsulation is the process of adding headers to data at each layer of a particular protocol stack. The Open Systems Interconnection (OSI) reference model describes the functions of a network as seven layers stacked on top of each other. When data has to be sent from one host (a PC for example) on a network to another host, the process of encapsulation is used to add a header in front of the data at each layer of the protocol stack in descending order. The header must contain a data field that indicates the type of data encapsulated at the layer immediately above the current layer. As the packet ascends the protocol stack on the receiving side of the network, each encapsulation header is removed in the reverse order.

Tunneling encapsulates data packets from one protocol inside a different protocol and transports the data packets unchanged across a foreign network. Unlike encapsulation, tunneling allows a lower-layer protocol, or same-layer protocol, to be carried through the tunnel. A tunnel interface is a virtual (or logical) interface. Although many different types of tunnels have been created to solve different network problems, tunneling consists of three main components:

- Passenger protocol—The protocol that you are encapsulating. Examples of passenger protocols are IPv4 and IPv6.
- Carrier protocol—The protocol that does the encapsulating. Examples of carrier protocols are GRE and MPLS.
- Transport protocol—The protocol used to carry the encapsulated protocol. The main transport protocol is IP.

Figure 2 illustrates IP tunneling terminology and concepts.

Figure 2 IP Tunneling Terminology and Concepts



Tunnel ToS

Tunnel type of service (ToS) allows you to tunnel your network traffic and group all your packets in the same specific ToS byte value. The ToS byte values and Time-to-Live (TTL) hop-count value can be set in the encapsulating IP header of tunnel packets for an IP tunnel interface on a router. The Tunnel ToS feature is supported for Cisco Express Forwarding (CEF), fast switching, and process switching.

The ToS and TTL byte values are defined in RFC 791. RFC 2474 and RFC 2780 obsolete the use of the ToS byte as defined in RFC 791. RFC 791 specifies that bits 6 and 7 of the ToS byte (the first two least significant bits) are reserved for future use and should be set to 0. Currently, the Tunnel ToS feature does not conform to this standard and allows you to set the whole ToS byte value, including bits 6 and 7, and decide to which RFC standard the ToS byte of your packets should conform.

Generic Routing Encapsulation

Generic routing encapsulation (GRE) is defined in RFC 2784. GRE is a carrier protocol that can be used with a variety of underlying transport protocols and that can carry a variety of passenger protocols. RFC 2784 also covers the use of GRE with IPv4 as the transport protocol and the passenger protocol. Cisco IOS XE software supports GRE as the carrier protocol with many combinations of passenger and transport protocols such as:

- GRE over an IPv4 network (GRE/IPv4)—GRE is the carrier protocol, and IPv4 is the transport protocol. This is the most common type of GRE tunnel. For configuration details, see the "Configuring a GRE Tunnel" section on page 11. Cisco IOS XE software supports many passenger protocols for GRE/IPv4 such as AppleTalk, IPX, IPv4, and IPv6. For more details about IPv6 as a passenger protocol with GRE/IPv4, see the "GRE/IPv4 Tunnel Support for IPv6 Traffic" section on page 5.
- GRE over an IPv6 network (GRE/IPv6)—GRE is the carrier protocol, and IPv6 is the transport protocol. Cisco IOS XE software supports IPv4 and IPv6 as passenger protocols with GRE/IPv6. For configuration details about IPv4 and IPv6 as passenger protocols with GRE/IPv6, see the "Configuring GRE/IPv6 Tunnels" section on page 15.

The following descriptions of GRE tunnels are included in this section:

GRE Tunnel IP Source and Destination VRF Membership

GRE Tunnel IP Source and Destination VRF Membership allows you to configure the source and destination of a tunnel to belong to any virtual private network (VPN) routing/forwarding (VRFs) tables. A VRF table stores routing data for each VPN. The VRF table defines the VPN membership of a customer site attached to the network access server (NAS). Each VRF table comprises an IP routing table, a derived Cisco Express Forwarding (CEF) table, and guidelines and routing protocol parameters that control the information that is included in the routing table.

Previously, Generic Routing Encapsulation (GRE) IP tunnels required the IP tunnel destination to be in the global routing table. The implementation of this feature allows you to configure a tunnel source and destination to belong to any VRF. As with existing GRE tunnels, the tunnel becomes disabled if no route to the tunnel destination is defined.

GRE/IPv4 Tunnel Support for IPv6 Traffic

IPv6 traffic can be carried over IPv4 generic routing encapsulation (GRE) tunnels using the standard GRE tunneling technique that is designed to provide the services necessary to implement any standard point-to-point encapsulation scheme. As in IPv6 manually configured tunnels, GRE tunnels are links between two points, with a separate tunnel for each link. The tunnels are not tied to a specific passenger or transport protocol, but in this case IPv6 is the passenger protocol, GRE is the carrier protocol, and IPv4 is the transport protocol.

The primary use of GRE tunnels is for stable connections that require regular secure communication between two edge routers or between an edge router and an end system. The edge routers and the end systems must be dual-stack implementations.

GRE has a protocol field that identifies the passenger protocol. GRE tunnels allow IS-IS or IPv6 to be specified as a passenger protocol, allowing both IS-IS and IPv6 traffic to run over the same tunnel. If GRE did not have a protocol field, it would be impossible to distinguish whether the tunnel was carrying IS-IS or IPv6 packets. The GRE protocol field is why it is desirable that you tunnel IS-IS and IPv6 inside GRE.

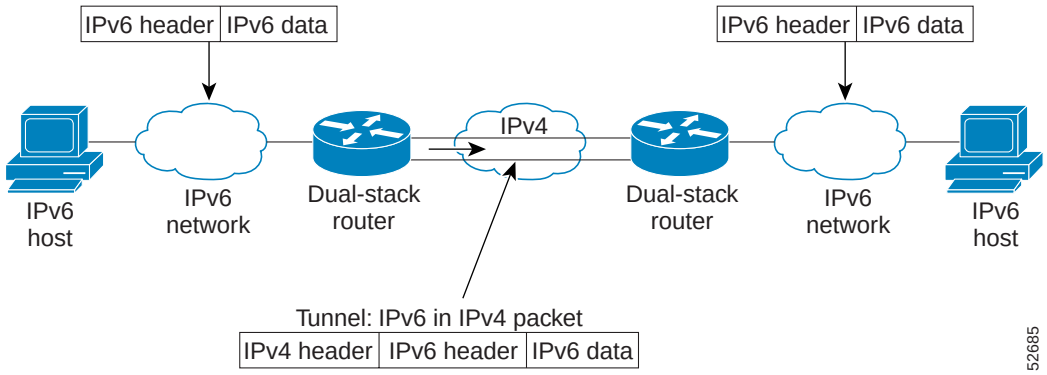
Overlay Tunnels for IPv6

Overlay tunneling encapsulates IPv6 packets in IPv4 packets for delivery across an IPv4 infrastructure (a core network or the Internet). (See Figure 3.) By using overlay tunnels, you can communicate with isolated IPv6 networks without upgrading the IPv4 infrastructure between them. Overlay tunnels can be configured between border routers or between a border router and a host; however, both tunnel endpoints must support both the IPv4 and IPv6 protocol stacks. Cisco IOS XE IPv6 currently supports the following types of overlay tunneling mechanisms:

- Manual
- Generic routing encapsulation (GRE)
- IPv4-compatible
- 6to4

- Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)

Figure 3 Overlay Tunnels



Note Overlay tunnels reduce the maximum transmission unit (MTU) of an interface by 20 octets (assuming that the basic IPv4 packet header does not contain optional fields). A network that uses overlay tunnels is difficult to troubleshoot. Therefore, overlay tunnels that connect isolated IPv6 networks should not be considered as a final IPv6 network architecture. The use of overlay tunnels should be considered as a transition technique toward a network that supports both the IPv4 and IPv6 protocol stacks or just the IPv6 protocol stack.

Use Table 1 to help you determine which type of tunnel you want to configure to carry IPv6 packets over an IPv4 network.

Table 1 Suggested Usage of Tunnel Types to Carry IPv6 Packets over an IPv4 Network

Tunneling Type	Suggested Usage	Usage Notes
Manual	Simple point-to-point tunnels that can be used within a site or between sites.	Can carry IPv6 packets only.
GRE/IPv4	Simple point-to-point tunnels that can be used within a site or between sites.	Can carry IPv6, CLNS, and many other types of packets.
6to4	Point-to-multipoint tunnels that can be used to connect isolated IPv6 sites.	Sites use addresses from the 2002::/16 prefix.
ISATAP	Point-to-multipoint tunnels that can be used to connect systems within a site.	Sites can use any IPv6 unicast addresses.

Individual tunnel types are discussed in more detail in the following concepts, and we recommend that you review and understand the information on the specific tunnel type that you want to implement. When you are familiar with the type of tunnel you need, Table 2 provides a quick summary of the tunnel configuration parameters that you may find useful.

Table 2 Overlay Tunnel Configuration Parameters by Tunneling Type

Overlay Tunneling Type	Overlay Tunnel Configuration Parameter			
	Tunnel Mode	Tunnel Source	Tunnel Destination	Interface Prefix/Address
Manual	ipv6ip	An IPv4 address or a reference to an interface on which IPv4 is configured.	An IPv4 address.	An IPv6 address.
GRE/IPv4	gre ip		An IPv4 address.	An IPv6 address.
6to4	ipv6ip 6to4		Not required. These are all point-to-multipoint tunneling types. The IPv4 destination address is calculated, on a per-packet basis, from the IPv6 destination.	An IPv6 address. The prefix must embed the tunnel source IPv4 address.
ISATAP	ipv6ip isatap			An IPv6 prefix in modified eui-64 format. The IPv6 address is generated from the prefix and the tunnel source IPv4 address.

IPv6 Manually Configured Tunnels

A manually configured tunnel is equivalent to a permanent link between two IPv6 domains over an IPv4 backbone. The primary use is for stable connections that require regular secure communication between two edge routers or between an end system and an edge router, or for connection to remote IPv6 networks.

An IPv6 address is manually configured on a tunnel interface, and manually configured IPv4 addresses are assigned to the tunnel source and the tunnel destination. The host or router at each end of a configured tunnel must support both the IPv4 and IPv6 protocol stacks. Manually configured tunnels can be configured between border routers or between a border router and a host. Cisco Express Forwarding (CEF) switching can be used for IPv6 manually configured tunnels, or CEF switching can be disabled if process switching is needed.

Automatic 6to4 Tunnels

An automatic 6to4 tunnel allows isolated IPv6 domains to be connected over an IPv4 network to remote IPv6 networks. The key difference between automatic 6to4 tunnels and manually configured tunnels is that the tunnel is not point-to-point; it is point-to-multipoint. In automatic 6to4 tunnels, routers are not configured in pairs because they treat the IPv4 infrastructure as a virtual nonbroadcast multiaccess (NBMA) link. The IPv4 address embedded in the IPv6 address is used to find the other end of the automatic tunnel.

An automatic 6to4 tunnel may be configured on a border router in an isolated IPv6 network, which creates a tunnel on a per-packet basis to a border router in another IPv6 network over an IPv4 infrastructure. The tunnel destination is determined by the IPv4 address of the border router extracted from the IPv6 address that starts with the prefix 2002::/16, where the format is 2002:*border-router-IPv4-address*::/48. Following the embedded IPv4 address are 16 bits that can be used to number networks within the site. The border router at each end of a 6to4 tunnel must support both the IPv4 and IPv6 protocol stacks. 6to4 tunnels are configured between border routers or between a border router and a host.

The simplest deployment scenario for 6to4 tunnels is to interconnect multiple IPv6 sites, each of which has at least one connection to a shared IPv4 network. This IPv4 network could be the global Internet or a corporate backbone. The key requirement is that each site have a globally unique IPv4 address; the Cisco IOS XE software uses this address to construct a globally unique 6to4/48 IPv6 prefix. As with other tunnel mechanisms, appropriate entries in a Domain Name System (DNS) that map between hostnames and IP addresses for both IPv4 and IPv6 allow the applications to choose the required address.

ISATAP Tunnels

The Intra-Site Automatic Tunnel Addressing Protocol (ISATAP) is an automatic overlay tunneling mechanism that uses the underlying IPv4 network as a nonbroadcast multiaccess (NBMA) link layer for IPv6. ISATAP is designed for transporting IPv6 packets *within* a site where a native IPv6 infrastructure is not yet available; for example, when sparse IPv6 hosts are deployed for testing. ISATAP tunnels allow individual IPv4/IPv6 dual-stack hosts within a site to communicate with other such hosts on the same virtual link, basically creating an IPv6 network using the IPv4 infrastructure.

The ISATAP router provides standard router advertisement network configuration support for the ISATAP site. This feature allows clients to automatically configure themselves as they would do if they were connected to an Ethernet. It can also be configured to provide connectivity out of the site. ISATAP uses a well-defined IPv6 address format composed of any unicast IPv6 prefix (/64), which can be link-local or global (including 6to4 prefixes), enabling IPv6 routing locally or on the Internet. The IPv4 address is encoded in the last 32 bits of the IPv6 address, enabling automatic IPv6-in-IPv4 tunneling.

While the ISATAP tunneling mechanism is similar to other automatic tunneling mechanisms, such as IPv6 6to4 tunneling, ISATAP is designed for transporting IPv6 packets *within* a site, not *between* sites.

ISATAP uses unicast addresses that include a 64-bit IPv6 prefix and a 64-bit interface identifier. The interface identifier is created in modified EUI-64 format in which the first 32 bits contain the value 000:5EFE to indicate that the address is an IPv6 ISATAP address. Table 3 shows the layout of an ISATAP address.

Table 3 ISATAP Address Example

64 Bits	32 Bits	32 Bits
Link local or global IPv6 unicast prefix	0000:5EFE	IPv4 address of the ISATAP link

As shown in Table 3, an ISATAP address consists of an IPv6 prefix and the ISATAP interface identifier. This interface identifier includes the IPv4 address of the underlying IPv4 link. The following example shows what an actual ISATAP address would look like if the prefix is 2001:0DB8:1234:5678::/64 and the embedded IPv4 address is 10.173.129.8. In the ISATAP address, the IPv4 address is expressed in hexadecimal as 0AAD:8108.

Example

2001:0DB8:1234:5678:0000:5EFE:0AAD:8108

Path MTU Discovery

Path MTU Discovery (PMTUD) can be enabled on a GRE or IP-in-IP tunnel interface. When PMTUD (RFC 1191) is enabled on a tunnel interface, the router performs PMTUD processing for the GRE (or IP-in-IP) tunnel IP packets. The router always performs PMTUD processing on the original data IP

packets that enter the tunnel. When PMTUD is enabled, packet fragmentation is not permitted for packets that traverse the tunnel because the Don't Fragment (DF) bit is set on all the packets. If a packet that enters the tunnel encounters a link with a smaller MTU, the packet is dropped and an ICMP message is sent back to the sender of the packet. This message indicates that fragmentation was required (but not permitted) and provides the MTU of the link that caused the packet to be dropped.

Note PMTUD on a tunnel interface requires that the tunnel endpoint be able to receive ICMP messages generated by routers in the path of the tunnel. Check that ICMP messages can be received before using PMTUD over firewall connections.

Use the **tunnel path-mtu-discovery** command to enable PMTUD for the tunnel packets, and use the **show interfaces tunnel** command to verify the tunnel PMTUD parameters. PMTUD currently works only on GRE and IP-in-IP tunnel interfaces.

QoS Options for Tunnels

A tunnel interface supports many of the same quality of service (QoS) features as a physical interface. QoS provides a way to ensure that mission-critical traffic has an acceptable level of performance. QoS options for tunnels include support for applying generic traffic shaping (GTS) directly on the tunnel interface and support for class-based shaping using the modular QoS command-line interface (MQC). Tunnel interfaces also support class-based policing, but they do not support committed access rate (CAR).

GRE tunnels allow the router to copy the IP precedence bit values of the type of service (ToS) byte to the tunnel or the GRE IP header that encapsulates the inner packet. Intermediate routers between the tunnel endpoints can use the IP precedence values to classify the packets for QoS features such as policy routing, weighted fair queueing (WFQ), and weighted random early detection (WRED).

When packets are encapsulated by tunnel or encryption headers, QoS features are unable to examine the original packet headers and correctly classify the packets. Packets that travel across the same tunnel have the same tunnel headers, so the packets are treated identically if the physical interface is congested. Tunnel packets can, however, be classified before tunneling and encryption can occur by using the QoS preclassify feature on the tunnel interface or on the crypto map.

Note Class-based WFQ (CBWFQ) inside class-based shaping is not supported on a multipoint interface.

For examples of how to implement some QoS features on a tunnel interface, see the "Configuring QoS Options on Tunnel Interfaces: Examples" section on page 30.

How to Implement Tunnels

This section contains the following tasks:

- Configuring 6to4 Tunnels, page 21 (optional)
- Configuring ISATAP Tunnels, page 23 (optional)
- Verifying Tunnel Configuration and Operation, page 25 (optional)

Determining the Tunnel Type

Before configuring a tunnel, you must determine what type of tunnel you need to create.

SUMMARY STEPS

1. Determine the passenger protocol.
2. Determine the **tunnel mode** command keyword, if appropriate.

DETAILED STEPS

Step 1 Determine the passenger protocol.

The passenger protocol is the protocol that you are encapsulating.

Step 2 Determine the **tunnel mode** command keyword, if appropriate.

Table 4 shows how to determine the appropriate keyword to use with the **tunnel mode** command. In the tasks that follow in this module, only the relevant keywords for the **tunnel mode** command are displayed.

Table 4 Determining the tunnel mode Command Keyword

Keyword	Purpose
dvmrp	Use the **dvmrp** keyword to specify that the Distance Vector Multicast Routing Protocol encapsulation will be used.
gre ip	Use the **gre ip** keywords to specify that GRE encapsulation over IP will be used.
gre ipv6	Use the **gre ipv6** keywords to specify that GRE encapsulation over IPv6 will be used.
ipip [**decapsulate-any**]	Use the **ipip** keyword to specify that IP-in-IP encapsulation will be used. The optional **decapsulate-any** keyword terminates any number of IP-in-IP tunnels at one tunnel interface. Note that this tunnel will not carry any outbound traffic; however, any number of remote tunnel endpoints can use a tunnel configured this way as their destination.
ipv6	Use the **ipv6** keyword to specify that generic packet tunneling in IPv6 will be used.
ipv6ip	Use the **ipv6ip** keyword to specify that IPv6 will be used as the passenger protocol and IPv4 as both the carrier (encapsulation) and transport protocol. When additional keywords are not used, manual IPv6 tunnels are configured. Additional keywords can be used to specify IPv4-compatible, 6to4, or ISATAP tunnels.
mpls	Use the **mpls** keyword to specify that MPLS will be used for configuring Traffic Engineering (TE) tunnels.

What to Do Next

- To configure a tunnel to carry IP data packets, proceed to the "Configuring a GRE Tunnel" section on page 11.
- To configure a tunnel to carry IPv6 data packets, review the "Overlay Tunnels for IPv6" section on page 5 and proceed to one of the following tasks:
  - "Configuring GRE/IPv6 Tunnels" section on page 15
  - "Configuring Manual IPv6 Tunnels" section on page 18
  - "Configuring 6to4 Tunnels" section on page 21
  - "Configuring ISATAP Tunnels" section on page 23

Configuring a GRE Tunnel

Perform this task to configure a GRE tunnel. A tunnel interface is used to pass protocol traffic across a network that does not normally support the protocol. To build a tunnel, a tunnel interface must be defined on each of two routers and the tunnel interfaces must reference each other. At each router, the tunnel

interface must be configured with a Layer 3 address. The tunnel endpoints, tunnel source, and tunnel destination must be defined, and the type of tunnel must be selected. Optional steps can be performed to customize the tunnel.

Remember to configure the router at each end of the tunnel. If only one side of a tunnel is configured, the tunnel interface may still come up and stay up (unless keepalive is configured), but packets going into the tunnel will be dropped.

GRE Tunnel Keepalive

Keepalive packets can be configured to be sent over IP-encapsulated GRE tunnels. You can specify the rate at which keepalives will be sent and the number of times that a device will continue to send keepalive packets without a response before the interface becomes inactive. GRE keepalive packets may be sent from both sides of a tunnel or from just one side.

Prerequisites

Ensure that the physical interface to be used as the tunnel source in this task is up and configured with the appropriate IP address. For hardware technical descriptions and information about installing interfaces, see the hardware installation and configuration publication for your product.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **bandwidth** *kbps*
5. **keepalive** [*period* [*retries*]]
6. **tunnel source** {*ip-address* | *interface-type interface-number*}
7. **tunnel destination** {*hostname* | *ip-address*}
8. **tunnel key** *key-number*
9. **tunnel mode** {**gre ip** | **gre multipoint**}
10. **ip mtu** *bytes*
11. **ip tcp mss** *mss-value*
12. **tunnel path-mtu-discovery** [**age-timer** {*aging-mins* | **infinite**}]
13. **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** *type number* **Example:** `Router(config)# interface tunnel 0`	Specifies the interface type and number and enters interface configuration mode. • To configure a tunnel, use **tunnel** for the *type* argument.
Step 4	**bandwidth** *kbps* **Example:** `Router(config-if)# bandwidth 1000`	Sets the current bandwidth value for an interface and communicates it to higher-level protocols. Specifies the tunnel bandwidth to be used to transmit packets. • Use the *kbps* argument to set the bandwidth, in kilobits per second (kbps). **Note** This is a routing parameter only; it does not affect the physical interface. The default bandwidth setting on a tunnel interface is 9.6 kbps. You should set the bandwidth on a tunnel to an appropriate value.
Step 5	**keepalive** [*period* [*retries*]] **Example:** `Router(config-if)# keepalive 3 7`	(Optional) Specifies the number of times that the device will continue to send keepalive packets without response before bringing the tunnel interface protocol down. • GRE keepalive packets may be configured either on only one side of the tunnel or on both. • If GRE keepalive is configured on both sides of the tunnel, the *period* and *retries* arguments can be different at each side of the link. **Note** This command is supported only on GRE point-to-point tunnels. **Note** The GRE tunnel keepalive feature should not be configured on a VRF tunnel. This combination of features is not supported.
Step 6	**tunnel source** {*ip-address* | *interface-type interface-number*} **Example:** `Router(config-if)# tunnel source GigabitEthernet 0/0/0`	Configures the tunnel source. • Use the *ip-address* argument to specify the source IP address. • Use the *interface-type* and *interface-number* arguments to specify the interface to use. **Note** The tunnel source and destination IP addresses must be defined on two separate devices.

	Command or Action	Purpose
Step 7	**tunnel destination** {*hostname* | *ip-address*} **Example:** `Router(config-if)# tunnel destination 172.17.2.1`	Configures the tunnel destination. • Use the *hostname* argument to specify the name of the host destination. • Use the *ip-address* argument to specify the IP address of the host destination. **Note** The tunnel source and destination IP addresses must be defined on two separate devices.
Step 8	**tunnel key** *key-number* **Example:** `Router(config-if)# tunnel key 1000`	(Optional) Enables an ID key for a tunnel interface. • Use the *key-number* argument to identify a tunnel key that is carried in each packet. • Tunnel ID keys can be used as a form of weak security to prevent improper configuration or injection of packets from a foreign source. **Note** This command is supported only on GRE tunnel interfaces. We do not recommend relying on this key for security purposes.
Step 9	**tunnel mode** {**gre ip** | **gre multipoint**} **Example:** `Router(config-if)# tunnel mode gre ip`	Specifies the encapsulation protocol to be used in the tunnel. • Use the **gre ip** keywords to specify that GRE over IP encapsulation will be used. • Use the **gre multipoint** keywords to specify that multipoint GRE (mGRE) will be used.
Step 10	**ip mtu** *bytes* **Example:** `Router(config-if)# ip mtu 1400`	(Optional) Set the maximum transmission unit (MTU) size of IP packets sent on an interface. • If an IP packet exceeds the MTU set for the interface, the Cisco IOS XE software will fragment it unless the DF bit is set. • All devices on a physical medium must have the same protocol MTU in order to operate. • For IPv6 packets, use the **ipv6 mtu** command. **Note** If the **tunnel path-mtu-discovery** command is enabled in Step 12, do not configure this command.
Step 11	**ip tcp mss** *mss-value* **Example:** `Router(config-if)# ip tcp mss 250`	(Optional) Specifies the maximum segment size (MSS) for TCP connections that originate or terminate on a router. • Use the *mss-value* argument to specify the maximum segment size for TCP connections, in bytes.

	Command or Action	Purpose
Step 12	**tunnel path-mtu-discovery** [**age-timer** {*aging-mins* | **infinite**}] **Example:** `Router(config-if)# tunnel path-mtu-discovery`	(Optional) Enables Path MTU Discovery (PMTUD) on a GRE or IP-in-IP tunnel interface. • When PMTUD is enabled on a tunnel interface, PMTUD will operate for GRE IP tunnel packets to minimize fragmentation in the path between the tunnel endpoints.
Step 13	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

What to Do Next

Proceed to the "Verifying Tunnel Configuration and Operation" section on page 25.

Configuring GRE/IPv6 Tunnels

This task explains how to configure a GRE tunnel on an IPv6 network. GRE tunnels can be configured to run over an IPv6 network layer and to transport IPv6 packets in IPv6 tunnels and IPv4 packets in IPv6 tunnels.

Prerequisites

When GRE/IPv6 tunnels are configured, IPv6 addresses are assigned to the tunnel source and the tunnel destination. The tunnel interface can have either IPv4 or IPv6 addresses assigned (this is not shown in the task below). The host or router at each end of a configured tunnel must support both the IPv4 and IPv6 protocol stacks.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *tunnel-number*
4. **tunnel source** {*ipv6-address* | *interface-type interface-number*}
5. **tunnel destination** *ipv6-address*
6. **tunnel mode gre ipv6**
7. **ipv6 mtu** *bytes*
8. **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 tunnel** *tunnel-number* **Example:** `Router(config)# interface tunnel 0`	Specifies a tunnel interface and number, and enters interface configuration mode.
Step 4	**tunnel source** {*ipv6-address* | *interface-type interface-number*} **Example:** `Router(config-if)# tunnel source GigabitEthernet 0/0/0`	Specifies the source IPv6 address or the source interface type and number for the tunnel interface. • If an interface type and number are specified, that interface must be configured with an IPv6 address. **Note** Only the syntax used in this context is displayed. For more details, see the *Cisco IOS IPv6 Command Reference*.
Step 5	**tunnel destination** *ipv6-address* **Example:** `Router(config-if)# tunnel destination 2001:0DB8:0C18:2::300`	Specifies the destination IPv6 address for the tunnel interface. **Note** Only the syntax used in this context is displayed. For more details, see the *Cisco IOS IPv6 Command Reference*.
Step 6	**tunnel mode gre ipv6** **Example:** `Router(config-if)# tunnel mode gre ipv6`	Specifies a GRE IPv6 tunnel. **Note** The **tunnel mode gre ipv6** command specifies GRE as the encapsulation protocol for the tunnel.
Step 7	**ipv6 mtu** *bytes* **Example:** `Router(config-if)# ipv6 mtu 1400`	(Optional) Set the maximum transmission unit (MTU) size of IPv6 packets sent on an interface.
Step 8	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

What to Do Next

Proceed to the "Verifying Tunnel Configuration and Operation" section on page 25.

Configuring GRE Tunnel IP Source and Destination VRF Membership

This task explains how to configure the source and destination of a tunnel to belong to any virtual private network (VPN) routing/forwarding (VRFs) tables.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *slot*
4. **ip vrf forwarding** *vrf-name*
5. **ip address** *ip-address subnet-mask*
6. **tunnel source** {*ip-address* | *type number*}
7. **tunnel destination** *ip-address* {*hostname* | *ip-address*}
8. **tunnel vrf** *vrf-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables higher privilege levels, such as privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel** *slot* **Example:** `Router(config)# interface tunnel 0`	Enters interface configuration mode for the specified interface.
Step 4	**ip vrf forwarding** *vrf-name* **Example:** `Router(config-if)# ip vrf forwarding green`	Defines the VRF associated with the tunnel interface.
Step 5	**ip address** *ip-address subnet-mask* **Example:** `Router(config-if)# ip address 10.7.7.7 255.255.255.255`	Specifies the ip address and subnet mask.
Step 6	**tunnel source** {*ip-address* | *type number*} **Example:** `Router(config-if)# tunnel source loop 0`	Specifies the tunnel source.
Step 7	**tunnel destination** {*hostname* | *ip-address*} **Example:** `Router(config-if)# tunnel destination 10.5.5.5`	Defines the tunnel destination.
Step 8	**tunnel vrf** *vrf-name* **Example:** `Router(config-if)# tunnel vrf finance1`	Defines the VRF associated with the physical interface from which tunnel packets are sent.

What to Do Next

Proceed to the "Verifying Tunnel Configuration and Operation" section on page 25.

Configuring Manual IPv6 Tunnels

This task explains how to configure a manual IPv6 overlay tunnel.

Prerequisites

With manually configured IPv6 tunnels, an IPv6 address is configured on a tunnel interface and manually configured IPv4 addresses are assigned to the tunnel source and the tunnel destination. The host or router at each end of a configured tunnel must support both the IPv4 and IPv6 protocol stacks.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *tunnel-number*
4. **ipv6 address** *ipv6-prefix/prefix-length* [**eui-64**]
5. **tunnel source** {*ip-address* | *interface-type interface-number*}
6. **tunnel destination** *ip-address*
7. **tunnel mode ipv6ip**
8. **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 tunnel** *tunnel-number* **Example:** `Router(config)# interface tunnel 0`	Specifies a tunnel interface and number, and enters interface configuration mode.
Step 4	**ipv6 address** *ipv6-prefix/prefix-length* [**eui-64**] **Example:** `Router(config-if)# ipv6 address 2001:0DB8:1234:5678::3/126`	Specifies the IPv6 network assigned to the interface and enables IPv6 processing on the interface. **Note** See the "Configuring Basic Connectivity for IPv6" module for more information on configuring IPv6 addresses.
Step 5	**tunnel source** {*ip-address* | *interface-type interface-number*} **Example:** `Router(config-if)# tunnel source GigabitEthernet 0/0/0`	Specifies the source IPv4 address or the source interface type and number for the tunnel interface. • If an interface is specified, the interface must be configured with an IPv4 address.
Step 6	**tunnel destination** *ip-address* **Example:** `Router(config-if)# tunnel destination 192.168.30.1`	Specifies the destination IPv4 address for the tunnel interface.
Step 7	**tunnel mode ipv6ip** **Example:** `Router(config-if)# tunnel mode ipv6ip`	Specifies a manual IPv6 tunnel. **Note** The **tunnel mode ipv6ip** command specifies IPv6 as the passenger protocol and IPv4 as both the carrier (encapsulation) and transport protocol for the manual IPv6 tunnel.
Step 8	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

What to Do Next

Proceed to the "Verifying Tunnel Configuration and Operation" section on page 25.

Configuring 6to4 Tunnels

This task explains how to configure a 6to4 overlay tunnel.

Prerequisites

With 6to4 tunnels, the tunnel destination is determined by the border-router IPv4 address, which is concatenated to the prefix 2002::/16 in the format 2002:*border-router-IPv4-address*::/48. The border router at each end of a 6to4 tunnel must support both the IPv4 and IPv6 protocol stacks.

Restrictions

The configuration of only one IPv4-compatible tunnel and one 6to4 IPv6 tunnel is supported on a router. If you choose to configure both of these tunnel types on the same router, we strongly recommend that they not share the same tunnel source.

The reason that a 6to4 tunnel and an IPv4-compatible tunnel cannot share the same interface is that both of them are NBMA "point-to-multipoint" access links and only the tunnel source can be used to reorder the packets from a multiplexed packet stream into a single packet stream for an incoming interface. So when a packet with an IPv4 protocol type of 41 arrives on an interface, that packet is mapped to an IPv6 tunnel interface on the basis of the IPv4 address. However, if both the 6to4 tunnel and the IPv4-compatible tunnel share the same source interface, the router cannot determine the IPv6 tunnel interface to which it should assign the incoming packet.

IPv6 manually configured tunnels can share the same source interface because a manual tunnel is a "point-to-point" link, and both the IPv4 source and IPv4 destination of the tunnel are defined.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *tunnel-number*
4. **ipv6 address** *ipv6-prefix*/*prefix-length* [**eui-64**]
5. **tunnel source** {*ip-address* | *interface-type interface-number*}
6. **tunnel mode ipv6ip 6to4**
7. **exit**
8. **ipv6 route** *ipv6-prefix*/*prefix-length* **tunnel** *tunnel-number*

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 tunnel** *tunnel-number* **Example:** `Router(config)# interface tunnel 0`	Specifies a tunnel interface and number, and enters interface configuration mode.
Step 4	**ipv6 address** *ipv6-prefix/prefix-length* [**eui-64**] **Example:** `Router(config-if)# ipv6 address 2002:c0a8:6301:1::1/64`	Specifies the IPv6 address assigned to the interface and enables IPv6 processing on the interface. • The 32 bits following the initial 2002::/16 prefix correspond to an IPv4 address assigned to the tunnel source. **Note** See the "Configuring Basic Connectivity for IPv6" module for more information on configuring IPv6 addresses.
Step 5	**tunnel source** {*ip-address* | *interface-type interface-number*} **Example:** `Router(config-if)# tunnel source GigabitEthernet 0/0/0`	Specifies the source IPv4 address or the source interface type and number for the tunnel interface. **Note** The interface type and number specified in the **tunnel source** command must be configured with an IPv4 address.
Step 6	**tunnel mode ipv6ip 6to4** **Example:** `Router(config-if)# tunnel mode ipv6ip 6to4`	Specifies an IPv6 overlay tunnel using a 6to4 address.
Step 7	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 8	**ipv6 route** *ipv6-prefix/prefix-length* **tunnel** *tunnel-number* **Example:** `Router(config)# ipv6 route 2002::/16 tunnel 0`	Configures a static route for the IPv6 6to4 prefix 2002::/16 to the specified tunnel interface. **Note** When configuring a 6to4 overlay tunnel, you must configure a static route for the IPv6 6to4 prefix 2002::/16 to the 6to4 tunnel interface. • The tunnel number specified in the **ipv6 route** command must be the same tunnel number specified in the **interface tunnel** command.

What to Do Next

Proceed to the "Verifying Tunnel Configuration and Operation" section on page 25.

Configuring ISATAP Tunnels

This task describes how to configure an ISATAP overlay tunnel.

Prerequisites

The **tunnel source** command used in the configuration of an ISATAP tunnel must point to an interface that is configured with an IPv4 address. The ISATAP IPv6 address and prefix (or prefixes) advertised are configured for a native IPv6 interface. The IPv6 tunnel interface must be configured with a modified EUI-64 address because the last 32 bits in the interface identifier are constructed using the IPv4 tunnel source address.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *tunnel-number*
4. **ipv6 address** *ipv6-prefix/prefix-length* [**eui-64**]
5. **no ipv6 nd suppress-ra**
6. **tunnel source** {*ip-address* | *interface-type interface-number*}
7. **tunnel mode ipv6ip isatap**
8. **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 tunnel** *tunnel-number* **Example:** `Router(config)# interface tunnel 1`	Specifies a tunnel interface and number, and enters interface configuration mode.
Step 4	**ipv6 address** *ipv6-prefix/prefix-length* [**eui-64**] **Example:** `Router(config-if)# ipv6 address 2001:0DB8:6301::/64 eui-64`	Specifies the IPv6 address assigned to the interface and enables IPv6 processing on the interface. **Note** See the "Configuring Basic Connectivity for IPv6" module for more information on configuring IPv6 addresses.
Step 5	**no ipv6 nd suppress-ra** **Example:** `Router(config-if)# no ipv6 nd suppress-ra`	Enables the sending of IPv6 router advertisements to allow client autoconfiguration. • Sending of IPv6 router advertisements is disabled by default on tunnel interfaces.
Step 6	**tunnel source** {*ip-address* | *interface-type interface-number*} **Example:** `Router(config-if)# tunnel source GigabitEthernet 0/0/1`	Specifies the source IPv4 address or the source interface type and number for the tunnel interface. **Note** The interface type and number specified in the **tunnel source** command must be configured with an IPv4 address.
Step 7	**tunnel mode ipv6ip isatap** **Example:** `Router(config-if)# tunnel mode ipv6ip isatap`	Specifies an IPv6 overlay tunnel using an ISATAP address.
Step 8	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

What to Do Next

Proceed to the "Verifying Tunnel Configuration and Operation" section on page 25.

Verifying Tunnel Configuration and Operation

This optional task explains how to verify tunnel configuration and operation. The commands contained in the task steps can be used in any sequence and may need to be repeated. The following commands can be used for GRE tunnels, IPv6 manually configured tunnels, and IPv6 over IPv4 GRE tunnels.

SUMMARY STEPS

1. **enable**
2. **show interfaces tunnel** *number* [**accounting**]
3. **ping** [*protocol*] *destination*
4. **show ip route** [*address* [*mask*]]
5. **ping** [*protocol*] *destination*

DETAILED STEPS

Step 1 enable

Enables privileged EXEC mode. Enter your password if prompted.

```
Router> enable
```

Step 2 show interfaces tunnel *number* [accounting]

Assuming a generic example suitable for both IPv6 manually configured tunnels and IPv6 over IPv4 GRE tunnels, two routers are configured to be endpoints of a tunnel. Router A has GigabitEthernet interface 0/0/0 configured as the source for tunnel interface 0 with an IPv4 address of 10.0.0.1 and an IPv6 prefix of 2001:0DB8:1111:2222::1/64. Router B has GigabitEthernet interface 0/0/0 configured as the source for tunnel interface 1 with an IPv4 address of 10.0.0.2 and an IPv6 prefix of 2001:0DB8:1111:2222::2/64.

To verify that the tunnel source and destination addresses are configured, use the **show interfaces tunnel** command on Router A.

```
RouterA# show interfaces tunnel 0

Tunnel0 is up, line protocol is up
  Hardware is Tunnel
  MTU 1514 bytes, BW 9 Kbit, DLY 500000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive not set
  Tunnel source 10.0.0.1 (GigabitEthernet0/0/0), destination 10.0.0.2, fastswitch TTL 255
  Tunnel protocol/transport GRE/IP, key disabled, sequencing disabled
  Tunnel TTL 255
  Checksumming of packets disabled,  fast tunneling enabled
  Last input 00:00:14, output 00:00:04, output hang never
  Last clearing of "show interface" counters never
  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
     4 packets input, 352 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     8 packets output, 704 bytes, 0 underruns
```

```
     0 output errors, 0 collisions, 0 interface resets
     0 output buffer failures, 0 output buffers swapped out
```

Step 3 **ping** [*protocol*] *destination*

To check that the local endpoint is configured and working, use the **ping** command on Router A.

```
RouterA# ping 2001:0DB8:1111:2222::2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:0DB8:1111:2222::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 20/20/20 ms
```

Step 4 **show ip route** [*address* [*mask*]]

To check that a route exists to the remote endpoint address, use the **show ip route** command.

```
RouterA# show ip route 10.0.0.2

Routing entry for 10.0.0.0/24
  Known via "connected", distance 0, metric 0 (connected, via interface)
  Routing Descriptor Blocks:
  * directly connected, via GigabitEthernet0/0/0
      Route metric is 0, traffic share count is 1
```

Step 5 **ping** [*protocol*] *destination*

To check that the remote endpoint address is reachable, use the **ping** command on Router A.

Note The remote endpoint address may not be reachable using the **ping** command because of filtering, but the tunnel traffic may still reach its destination.

```
RouterA# ping 10.0.0.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.0.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 20/21/28 ms
```

To check that the remote IPv6 tunnel endpoint is reachable, use the **ping** command again on Router A. The same note on filtering also applies to this example.

```
RouterA# ping 1::2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 1::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 20/20/20 ms
```

These steps may be repeated at the other endpoint of the tunnel.

Configuration Examples for Implementing Tunnels

This section contains the following examples:

Configuring GRE/IPv4 Tunnels: Examples

The following example shows a simple configuration of GRE tunneling. Note that GigabitEthernet interface 0/0/1 is the tunnel source for Router A and the tunnel destination for Router B. Fast Ethernet interface 0/0/1 is the tunnel source for Router B and the tunnel destination for Router A.

Router A

```
interface Tunnel0
 ip address 10.1.1.2 255.255.255.0
 tunnel source GigabitEthernet0/0/1
 tunnel destination 192.168.3.2
 tunnel mode gre ip
!
interface GigabitEthernet0/0/1
 ip address 192.168.4.2 255.255.255.0
```

Router B

```
interface Tunnel0
 ip address 10.1.1.1 255.255.255.0
 tunnel source FastEthernet0/0/1
 tunnel destination 192.168.4.2
 tunnel mode gre ip
!
interface FastEthernet0/0/1
 ip address 192.168.3.2 255.255.255.0
```

The following example configures a GRE tunnel running both IS-IS and IPv6 traffic between Router A and Router B.

Router A

```
ipv6 unicast-routing
clns routing
!
interface Tunnel0
 no ip address
 ipv6 address 2001:0DB8:1111:2222::1/64
 ipv6 router isis
 tunnel source GigabitEthernet0/0/0
 tunnel destination 10.0.0.2
 tunnel mode gre ip
!
interface GigabitEthernet0/0/0
 ip address 10.0.0.1 255.255.255.0
!
router isis
 network 49.0000.0000.000a.00
```

Router B

```
ipv6 unicast-routing
clns routing
!
interface Tunnel0
 no ip address
 ipv6 address 2001:0DB8:1111:2222::2/64
 ipv6 router isis
 tunnel source GigabitEthernet0/0/0
 tunnel destination 10.0.0.1
 tunnel mode gre ip
!
interface GigabitEthernet0/0/0
 ip address 10.0.0.2 255.255.255.0
!
router isis
 network 49.0000.0000.000b.00
 address-family ipv6
 redistribute static
 exit-address-family
```

Configuring GRE/IPv6 Tunnels: Example

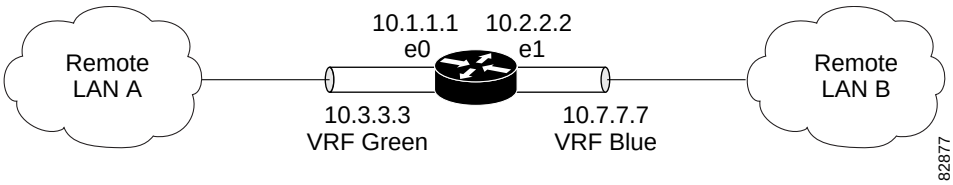
The following example shows how to configure a GRE tunnel over an IPv6 transport. GigabitEthernet0/0/0 has an IPv6 address configured, and this is the source address used by the tunnel interface. The destination IPv6 address of the tunnel is specified directly. In this example, the tunnel carries both IPv4 and IS-IS traffic:

```
interface Tunnel0
 ip address 10.1.1.1 255.255.255.0
 ip router isis
 tunnel source GigabitEthernet0/0/0
 tunnel destination 2001:DB8:1111:2222::1
 tunnel mode gre ipv6
!
interface Ethernet0/0
 no ip address
 ipv6 address 2001:DB8:1111:1111::1/64
!
router isis
 net 49.0001.0000.0000.000a.00
```

Configuring GRE Tunnel IP Source and Destination VRF Membership: Example

In this example, packets received on interface e0 using VRF green, will be forwarded out of the tunnel through interface e1 using VRF blue. Figure 4 shows a simple tunnel scenario:

Figure 4 GRE Tunnel Diagram



The following example shows the configuration for the tunnel in Figure 4:

```
ip vrf blue
 rd 1:1

ip vrf green
 rd 1:2

interface loopback0
 ip vrf forwarding blue
 ip address 10.7.7.7 255.255.255.255

interface tunnel0
 ip vrf forwarding green
 ip address 10.3.3.3 255.255.255.0
 tunnel source loopback 0
 tunnel destination 10.5.5.5
 tunnel vrf blue

interface GigabitEthernet0/0/0
 ip vrf forwarding green
 ip address 10.1.1.1 255.255.255.0

interface GigabitEthernet0/0/1
 ip vrf forwarding blue
 ip address 10.2.2.2 255.255.255.0

 ip route vrf blue 10.5.5.5 255.255.255.0 GigabitEthernet 0/0/1
```

Configuring Manual IPv6 Tunnels: Example

The following example configures a manual IPv6 tunnel between Router A and Router B. In the example, tunnel interface 0 for both Router A and Router B is manually configured with a global IPv6 address. The tunnel source and destination addresses are also manually configured.

Router A

```
interface GigabitEthernet 0/0/0
 ip address 192.168.99.1 255.255.255.0

interface tunnel 0
 ipv6 address 2001:0db8:c18:1::3/126
 tunnel source GigabitEthernet 0/0/0
 tunnel destination 192.168.30.1
 tunnel mode ipv6ip
```

Router B

```
interface GigabitEthernet 0/0/0
 ip address 192.168.30.1 255.255.255.0

interface tunnel 0
 ipv6 address 2001:0db8:c18:1::2/126
 tunnel source GigabitEthernet 0/0/0
 tunnel destination 192.168.99.1
 tunnel mode ipv6ip
```

Configuring 6to4 Tunnels: Example

The following example configures a 6to4 tunnel on a border router in an isolated IPv6 network. The IPv4 address is 192.168.99.1, which translates to the IPv6 prefix of 2002:c0a8:6301::/48. The IPv6 prefix is subnetted into 2002:c0a8:6301::/64 for the tunnel interface: 2002:c0a8:6301:1::/64 for the first IPv6 network and 2002:c0a8:6301:2::/64 for the second IPv6 network. The static route ensures that any other traffic for the IPv6 prefix 2002::/16 is directed to tunnel interface 0 for automatic tunneling.

```
interface GigabitEthernet0/0/0
 description IPv4 uplink
 ip address 192.168.99.1 255.255.255.0
!
interface GigabitEthernet0/0/1
 description IPv6 local network 1
 ipv6 address 2002:c0a8:6301:1::1/64
!
interface GigabitEthernet0/0/2
 description IPv6 local network 2
 ipv6 address 2002:c0a8:6301:2::1/64
!
interface Tunnel0
 description IPv6 uplink
 no ip address
 ipv6 address 2002:c0a8:6301::1/64
 tunnel source GigabitEthernet0/0/0
 tunnel mode ipv6ip 6to4
!
ipv6 route 2002::/16 Tunnel0
```

Configuring ISATAP Tunnels: Example

The following example shows the tunnel source defined on GigabitEthernet 0/0/0 and the **tunnel mode** command used to configure the ISATAP tunnel. Router advertisements are enabled to allow client autoconfiguration.

```
interface Tunnel1
 tunnel source GigabitEthernet 0/0/0
 tunnel mode ipv6ip isatap
 ipv6 address 2001:0DB8::/64 eui-64
 no ipv6 nd suppress-ra
```

Configuring QoS Options on Tunnel Interfaces: Examples

The following sample configuration applies generic traffic shaping (GTS) directly on the tunnel interface. In this example the configuration shapes the tunnel interface to an overall output rate of 500 kbps.

```
interface Tunnel0
 ip address 10.1.2.1 255.255.255.0
 traffic-shape rate 500000 125000 125000 1000
 tunnel source 10.1.1.1
 tunnel destination 10.2.2.2
```

The following sample configuration shows how to apply the same shaping policy to the tunnel interface with the Modular QoS CLI (MQC) commands.

```
policy-map tunnel
 class class-default
```

```
 shape average 500000 125000 125000
!
interface Tunnel0
 ip address 10.1.2.1 255.255.255.0
 service-policy output tunnel
 tunnel source 10.1.35.1
 tunnel destination 10.1.35.2
```

Policing Example

When an interface becomes congested and packets start to queue, you can apply a queueing method to packets that are waiting to be transmitted. Cisco IOS XE logical interfaces—tunnel interfaces in this example—do not inherently support a state of congestion and do not support the direct application of a service policy that applies a queueing method. Instead, you need to apply a hierarchical policy. Create a "child" or lower-level policy that configures a queueing mechanism, such as low latency queueing with the **priority** command and class-based weighted fair queueing (CBWFQ) with the **bandwidth** command.

```
policy-map child
 class voice
 priority 512
```

Create a "parent" or top-level policy that applies class-based shaping. Apply the child policy as a command under the parent policy because admission control for the child class is done according to the shaping rate for the parent class.

```
policy-map tunnel
 class class-default
 shape average 2000000
 service-policy child
```

Apply the parent policy to the tunnel interface.

```
interface tunnel0
 service-policy tunnel
```

In the following example, a tunnel interface is configured with a service policy that applies queueing without shaping. A log message is displayed noting that this configuration is not supported.

```
Router(config)# interface tunnel1
Router(config-if)# service-policy output child

Class Based Weighted Fair Queueing not supported on this interface
```

Additional References

The following sections provide references related to implementing tunnels.

Related Documents

Related Topic	Document Title
Tunnel commands: complete command syntax, command mode, defaults, command history, usage guidelines, and examples	*Cisco IOS Interface and Hardware Component Command Reference*
IPv6 commands: complete command syntax, command mode, defaults, command history, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
All Cisco IOS XE commands	• *Cisco IOS Master Command List, All Releases*. • Command Lookup Tool
Cisco IOS XE Interface and Hardware Component configuration modules	*Cisco IOS XE Interface and Hardware Component Configuration Guide, Release 2*
Cisco IOS XE IPv6 configuration modules	*Cisco IOS XE IPv6 Configuration Guide, Release 2*
Cisco IOS XE Quality of Service Solutions configuration modules	*Cisco IOS XE Quality of Service Solutions Configuration Guide, Release 2*
Cisco IOS XE Multiprotocol Label Switching configuration modules	*Cisco IOS XE Multiprotocol Label Switching Configuration Guide, Release 2*
Configuration example for a VRF aware Dynamic Multipoint VPN (DMVPN)	"Dynamic Multipoint VPN (DMVPN)" configuration module in the *Cisco IOS XE Security Configuration Guide: Secure Connectivity, Release 2*

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	—

MIBs

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

RFCs

RFC	Title
RFC 791	*Internet Protocol*
RFC 1191	*Path MTU Discovery*
RFC 1323	*TCP Extensions for High Performance*

RFC	Title
RFC 1483	*Multiprotocol Encapsulation over ATM Adaptation Layer 5*
RFC 2003	*IP Encapsulation Within IP*
RFC 2018	*TCP Selective Acknowledgment Options*
RFC 2460	*Internet Protocol, Version 6 (IPv6)*
RFC 2473	*Generic Packet Tunneling in IPv6 Specification*
RFC 2474	*Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*
RFC 2516	*A Method for Transmitting PPP over Ethernet (PPPoE)*
RFC 2547	*BGP/MPLS VPNs*
RFC 2780	*IANA Allocation Guidelines for Values in the Internet Protocol and Related Headers*
RFC 2784	*Generic Routing Encapsulation (GRE)*
RFC 2890	*Key and Sequence Number Extensions to GRE*
RFC 2893	*Transition Mechanisms for IPv6 Hosts and Routers*
RFC 3056	*Connection of IPv6 Domains via IPv4 Clouds*
RFC 3147	*Generic Routing Encapsulation over CLNS Networks*

Technical Assistance

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

Feature Information for Implementing Tunnels

Table 5 lists the features in this module and provides links to specific configuration information.

Use Cisco Feature Navigator to find information about platform support and software image support. Cisco Feature Navigator enables you to determine which Cisco IOS XE software images support a specific software release, feature set, or platform. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Note Table 5 lists only the Cisco IOS XE software release that introduced support for a given feature in a given Cisco IOS XE software release train. Unless noted otherwise, subsequent releases of that Cisco IOS XE software release train also support that feature.

Table 5 Feature Information for Implementing Tunnels

Feature Name	Releases	Feature Configuration Information
GRE Tunnel IP Source and Destination VRF Membership	Cisco IOS XE Release 2.2	This feature allows you to configure the source and destination of a tunnel to belong to any VPN VRF table. The following sections provide information about this feature: • GRE Tunnel IP Source and Destination VRF Membership, page 5 • Configuring GRE Tunnel IP Source and Destination VRF Membership, page 17 • Configuring GRE Tunnel IP Source and Destination VRF Membership: Example, page 28 The following command was introduced to support this feature: **tunnel vrf**.
GRE Tunnel Keepalive	Cisco IOS XE Release 2.1	The GRE Tunnel Keepalive feature provides the capability of configuring keepalive packets to be sent over IP-encapsulated generic routing encapsulation (GRE) tunnels. You can specify the rate at which keepalives will be sent and the number of times that a device will continue to send keepalive packets without a response before the interface becomes inactive. GRE keepalive packets may be sent from both sides of a tunnel or from just one side. The following section provides information about this feature: • Configuring a GRE Tunnel, page 11 The following command was introduced by this feature: **keepalive** (tunnel interfaces).

Table 5 Feature Information for Implementing Tunnels (continued)

Feature Name	Releases	Feature Configuration Information
IP over IPv6 Tunnels	Cisco IOS XE Release 2.4	The following sections provide information about this feature: • Configuring GRE/IPv6 Tunnels, page 15 • Configuring GRE/IPv6 Tunnels: Example, page 28 The following commands were modified by this feature: **tunnel destination**, **tunnel mode**, and **tunnel source**.
IP Precedence for GRE Tunnels	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Routers.
Tunnel ToS	Cisco IOS XE Release 2.1	The Tunnel ToS feature allows you to configure the ToS and Time-to-Live (TTL) byte values in the encapsulating IP header of tunnel packets for an IP tunnel interface on a router. The Tunnel ToS feature is supported in Cisco Express Forwarding (CEF), fast switching, and process switching forwarding modes. The following section provides information about this feature: • Tunnel ToS, page 4 The following commands were introduced or modified by this feature: **show interfaces tunnel**, **tunnel tos**, **tunnel ttl**.