Configuring IP Tunnels

This chapter contains the following sections:

Information About IP Tunnels

IP tunnels can encapsulate a same-layer or higher-layer protocol and transport the result over IP through a tunnel created between two devices.

IP tunnels consists of the following three main components:

- Passenger protocol—The protocol that needs to be encapsulated. IPv4 is an example of a passenger protocol.
- Carrier protocol—The protocol that is used to encapsulate the passenger protocol. Cisco NX-OS supports generic routing encapsulation (GRE), and IP-in-IP encapsulation and decapsulation as carrier protocols.
- Transport protocol—The protocol that is used to carry the encapsulated protocol. IPv4 is an example of a transport protocol.

An IP tunnel takes a passenger protocol, such as IPv4, and encapsulates that protocol within a carrier protocol, such as GRE. The device then transmits this carrier protocol over a transport protocol, such as IPv4.

You configure a tunnel interface with matching characteristics on each end of the tunnel.

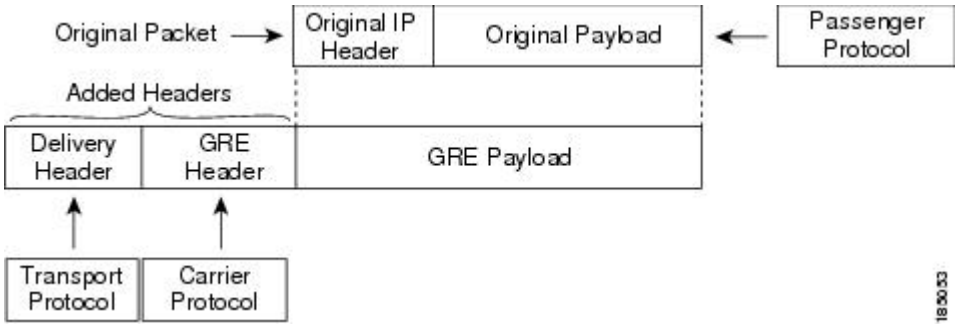
You must enable the tunnel feature before you can configure it.

GRE Tunnels

You can use GRE as the carrier protocol for a variety of passenger protocols. The selection of tunnel interfaces can also be based on the PBR policy.

The figure shows the IP tunnel components for a GRE tunnel. The original passenger protocol packet becomes the GRE payload and the device adds a GRE header to the packet. The device then adds the transport protocol header to the packet and transmits it.

Figure 1: GRE PDU



Point-to-Point IP-in-IP Tunnel Encapsulation and Decapsulation

Point-to-point IP-in-IP encapsulation and decapsulation is a type of tunnel that you can create to send encapsulated packets from a source tunnel interface to a destination tunnel interface. The selection of these tunnel interfaces can also be based on the PBR policy. This type of tunnel will carry both inbound and outbound traffic.

Multi-Point IP-in-IP Tunnel Decapsulation

Multi-point IP-in-IP decapsulate-any is a type of tunnel that you can create to decapsulate packets from any number of IP-in-IP tunnels to one tunnel interface. This tunnel will not carry any outbound traffic. However, any number of remote tunnel endpoints can use a tunnel configured this way as their destination.

Licensing Requirements for IP Tunnels

Product	License Requirement
Cisco NX-OS	IP tunnels require an Enterprise Services license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for IP Tunnels

IP tunnels have the following prerequisites:

- You must be familiar with TCP/IP fundamentals to configure IP tunnels.
- You are logged on to the switch.
- You have installed the Enterprise Services license for Cisco NX-OS.
- You must enable the tunneling feature in a device before you can configure and enable any IP tunnels.

Guidelines and Limitations for IP Tunnels

IP tunnels have the following configuration guidelines and limitations:

- Cisco NX-OS software supports the GRE header defined in IETF RFC 2784. Cisco NX-OS software does not support tunnel keys and other options from IETF RFC 1701.
- The Cisco Nexus device supports the following maximum number tunnels:
  - GRE and IP-in-IP regular tunnels-8 tunnels
  - Multipoint IP-in-IP tunnels-32 tunnels
- Each tunnel will consume one Equal Cost Multipath (ECMP) adjacency.
- The Cisco Nexus device does not support the following features:
  - Path maximum transmission unit (MTU) discovery
  - Tunnel interface statistics
  - Access control lists (ACLs)
  - Unicast reverse path forwarding (URPF)
  - Multicast traffic and associated multicast protocols such as Internet Group Management Protocol (IGMP) and Protocol Independent Multicast (PIM)
- Cisco NX-OS software does not support the Web Cache Control Protocol (WCCP) on tunnel interfaces.
- Cisco NX-OS software supports only Layer-3 traffic.
- Cisco NX-OS software supports ECMP across tunnels and ECMP for tunnel destination.
- IPv6-in-IPv6 tunnels is not supported.
- Limited control protocols, such as Border Gateway Protocol (BGP), Open Shortest Path First (OSPF), and Enhanced Interior Gateway Routing Protocol (EIGRP), are supported for GRE tunnels.
- Starting with Release 6.0(2)U5(1), Cisco Nexus 3000 Series switches drop all the packets when the tunnel is not configured. The packets are also dropped when the tunnel is configured but the tunnel interface is not configured or the tunnel interface is in shut down state.

Point to Point tunnel (Source and Destination) – Cisco Nexus 3000 Series switches decapsulate all IP-in-IP packets destined to it when the command **feature tunnel** is configured and there is an operational tunnel interface configured with the tunnel source and the destination address that matches the incoming packets' outer source and destination addresses. If there is not a source and destination packet match or if the interface is in shutdown state, the packet is dropped.

Decapsulate Tunnel (Source only) - Cisco Nexus 3000 Series switches decapsulate all IP-in-IP packets destined to it when the command **feature tunnel** is configured and there is an operational tunnel interface configured with the tunnel source address that matches the incoming packets' outer destination addresses. If there is not a source packet match or if the interface is in shutdown state, the packet is dropped.

- Starting with Release 6.0(2)U6(1), Cisco Nexus 3000 Series switches support IPv6 in IPv4 with GRE header only. The new control protocols that are supported on the tunnel are:
  - BGP with v6
  - OSPFv3
  - EIGRP over v6
- The Cisco Nexus 3000 Series switches ASIC supports the GRE encapsulation and decapsulation in the hardware.
- On the encapsulation side, the Cisco Nexus 3000 Series switches performs a single lookup in the hardware.
- Since Cisco Nexus 3000 Series switches perform a single lookup in the hardware, the software has to keep the hardware information up-to-date with any changes related to the second lookup, for example, the tunnel destination adjacency.
- On the decapsulation side, the Cisco Nexus 3000 Series switches have a separate table to perform the outer IP header lookup and it does not need an ACL for the same.

Default Settings for IP Tunneling

The following table lists the default settings for IP tunnel parameters.

Table 1: Default IP Tunnel Parameters

Parameters	Default
Tunnel feature	Disabled

Configuring IP Tunnels

Enabling Tunneling

Before You Begin

You must enable the tunneling feature before you can configure any IP tunnels.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **feature tunnel**	Enables the tunnel feature on the switch.
Step 3	switch(config)# **exit**	Returns to configuration mode.
Step 4	switch(config)# **show feature**	Displays the tunnel feature on the switch.
Step 5	switch# **copy running-config startup-config**	(Optional) Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

This example shows how to enable the tunnel feature:

```
switch# configure terminal
switch(config)# feature tunnel
switch(config)# exit
switch(config)# copy running-config startup-config
```

Creating a Tunnel Interface

You can create a tunnel interface and then configure this logical interface for your IP tunnel. GRE mode is the default tunnel mode.

Before You Begin

Both the tunnel source and the tunnel destination must exist within the same virtual routing and forwarding (VRF) instance.

Ensure that you have enabled the tunneling feature.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# [**no**] **interface tunnel** *number*	Creates a new tunnel interface.
Step 3	switch(config)# **tunnel mode** {**gre ip** | **ipip** {**ip** | **decapsulate-any**}}	Sets this tunnel mode to GRE, ipip, or ipip decapsulate-only. The **gre** and **ip** keywords specify that GRE encapsulation over IP will be used. The **ipip** keyword specifies that IP-in-IP encapsulation will be used. The optional **decapsulate-any** keyword terminates IP-in-IP tunnels at one tunnel interface. This

	Command or Action	Purpose
		keyword creates a tunnel that will not carry any outbound traffic. However, remote tunnel endpoints can use a tunnel configured as their destination.
Step 4	switch(config)# **tunnel source** {*ip address* | *interface-name*}	Configures the source address for this IP tunnel.
Step 5	switch(config)# **tunnel destination** {*ip address* | *host-name*}	Configures the destination address for this IP tunnel.
Step 6	switch(config)# **tunnel use-vrf** *vrf-name*	(Optional) Uses the configured VRF instance to look up the tunnel IP destination address.
Step 7	switch(config)# **show interface tunnel** *number*	(Optional) Displays the tunnel interface statistics.
Step 8	switch# **copy running-config startup-config**	(Optional) Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

This example shows how to create a tunnel interface:

```
switch# configure terminal
switch(config)# interface tunnel 1
switch(config)# tunnel source ethernet 1/2
switch(config)# tunnel destination 192.0.2.1
switch(config)# copy running-config startup-config
```

Configuring a Tunnel Interface Based on Policy Based Routing

You can create a tunnel interface and then configure this logical interface for your IP tunnel based on the PBR policy.

Before You Begin

Ensure that you have enabled the tunneling feature.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# [**no**] **interface tunnel** *number*	Creates a new tunnel interface.
Step 3	switch(config)# **ip address** *ip address*	Configures an IP address for this interface.

	Command or Action	Purpose
Step 4	switch(config)# **route-map** *map-name*	Assigns a route map for IPv4 policy-based routing to the interface
Step 5	switch(config-route-map)# **match ip address access-list-name** *name*	Matches an IPv4 address against one or more IP access control lists (ACLs). This command is used for policy-based routing and is ignored by route filtering or redistribution.
Step 6	switch(config-route-map)# **set ip next-hop** *address*	Sets the IPv4 next-hop address for policy-based routing. Tunnel IP addresses can be specified as next-hop addresses to select tunnel interfaces. This command uses the first valid next-hop address if multiple addresses are configured. Use the **load-share** option to select ECMP across next-hop entries.

This example shows how to configure a tunnel interface that is based on PBR:

```
switch# configure terminal
switch(config)# interface tunnel 1
switch(config)# ip address 1.1.1.1/24
switch(config)# route-map pbr1
switch(config-route-map)# match ip address access-list-name pbr1
switch(config-route-map)# set ip next-hop 1.1.1.1
```

Configuring a GRE Tunnel

You can set a tunnel interface to GRE tunnel mode, ipip mode, or ipip decapsulate-only mode. GRE mode is the default tunnel mode. Starting with Release 6.0(2)U6(1), Cisco Nexus 3000 Series switches support IPv6 in IPv4 with GRE header only.

Before You Begin

Ensure that you have enabled the tunneling feature.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface tunnel** *number*	Enters a tunnel interface configuration mode.
Step 3	switch(config-if)# **tunnel mode** {**gre ip** | **ipip** {**ip** | **decapsulate-any**}}	Sets this tunnel mode to GRE, ipip, or ipip decapsulate-only. The **gre** and **ip** keywords specify that GRE encapsulation over IP will be used. The ipip keyword specifies that IP-in-IP encapsulation will be used. The optional **decapsulate-any** keyword

	Command or Action	Purpose
		terminates IP-in-IP tunnels at one tunnel interface. This keyword creates a tunnel that will not carry any outbound traffic. However, remote tunnel endpoints can use a tunnel configured as their destination.
Step 4	switch(config-if)# **tunnel use-vrf** *vrf-name*	Configures tunnel VRF name.
Step 5	switch(config-if)# **ipv6 address** *IPv6 address*	Configures the IPv6 address. **Note** The tunnel source and the destination addresses are still the same (IPv4 address.)
Step 6	switch(config-if)# **show interface tunnel** *number*	(Optional) Displays the tunnel interface statistics.
Step 7	switch(config-if)# **mtu** *value*	Sets the maximum transmission unit (MTU) of IP packets sent on an interface.
Step 8	switch(config-if)# **copy running-config startup-config**	(Optional) Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

This example shows how to create the tunnel interface to GRE:

```
switch# configure terminal
switch(config)# interface tunnel 1
switch(config-if)# tunnel mode gre ip
switch(config-if)# tunnel use-vrf red
switch(config-if)# ipv6 address 2:2::2/64
switch(config-if)# copy running-config startup-config
```

This example shows how to view the tunnel interface to GRE:

```
switch(config)# show int tunnel 2
Tunnel2 is up
    Internet address(es):
    2.2.2.2/24
    2:2::2/64
    MTU 1476 bytes, BW 9 Kbit
    Transport protocol is in VRF "default"
    Tunnel protocol/transport GRE/IP
    Tunnel source 2.2.3.2, destination 2.2.3.1
    Last clearing of "show interface" counters never
    Tx
    0 packets output, 0 bytes
```

This example shows how to create an ipip tunnel:

```
switch# configure terminal
switch(config)# interface tunnel 1
switch(config-if)# tunnel mode ipip
switch(config-if)# mtu 1400
switch(config-if)# copy running-config startup-config
switch(config-if)# no shut
```

Assigning VRF Membership to a Tunnel Interface

You can add a tunnel interface to a VRF.

Before You Begin

Ensure that you have enabled the tunneling feature.

Assign the IP address for a tunnel interface after you have configured the interface for a VRF.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface tunnel** *number*	Enters interface configuration mode.
Step 3	switch(config)# **vrf member** *vrf-name*	Adds this interface to a VRF.
Step 4	switch(config)# **ip address** *ip-prefix/length*	Configures an IP address for this interface. You must do this step after you assign this interface to a VRF.
Step 5	switch(config)# **show vrf** [*vrf-name*] **interface** *interface-type number*	(Optional) Displays VRF information.
Step 6	switch(config-if)# **copy running-config startup-config**	(Optional) Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

This example shows how to add a tunnel interface to the VRF:

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

Verifying the IP Tunnel Configuration

Use the following commands to verify the configuration:

Command	Purpose
show interface tunnel *number*	Displays the configuration for the tunnel interface (MTU, protocol, transport, and VRF). Displays input and output packets, bytes, and packet rates.
show interface tunnel *number* **brief**	Displays the operational status, IP address, encapsulation type, and MTU of the tunnel interface.

Command	Purpose
show interface tunnel *number* **description**	Displays the configured description of the tunnel interface.
show interface tunnel *number* **status**	Displays the operational status of the tunnel interface.
show interface tunnel *number* **status err-disabled**	Displays the error disabled status of the tunnel interface.

Configuration Examples for IP Tunneling

This example shows a simple GRE tunnel. Ethernet 1/2 is the tunnel source for router A and the tunnel destination for router B. Ethernet interface 1/3 is the tunnel source for router B and the tunnel destination for router A.

```
router A:
feature tunnel
interface tunnel 0
  ip address 209.165.20.2/8
  tunnel source ethernet 1/2
  tunnel destination 192.0.2.2
  tunnel mode gre ip
interface ethernet1/2
  ip address 192.0.2.55/8

router B:
feature tunnel
interface tunnel 0
  ip address 209.165.20.1/8
  tunnel source ethernet 1/3
  tunnel destination 192.0.2.55
  tunnel mode gre ip
interface ethernet 1/3
ip address 192.0.2.2/8
```

Related Documents for IP Tunnels

Related Topics	Document Title
IP tunnel commands	*Cisco Nexus 3000 Series Interfaces Command Reference*

Standards for IP Tunnels

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

Feature History for Configuring IP Tunnels

Table 2: Feature History for Configuring IP Tunnels

Feature Name	Release	Feature Information
Multi-point and Point-to-Point IP-in-IP encapsulation and decapsulation	6.0(2)U2(1)	Support for these tunnel modes was added.
IP tunnels	5.0(3)U4(1)	This feature was introduced.