Configuring ASBRs to Exchange IPv4 Routes and MPLS Labels 39
Configuring the Route Reflectors to Exchange VPN-IPv4 Routes 41
Configuring the Route Reflector to Reflect Remote Routes in its AS 43
Providing VPN Connectivity Across Multiple Autonomous Systems with MPLS VPN Inter-AS with ASBRs Exchanging VPN-IPv4 Addresses 46
Configuring the ASBRs to Exchange VPN-IPv4 Addresses for IP Tunnels 46
Configuring a Static Route to an ASBR Peer 49
Configuring EBGP Routing to Exchange VPN Routes Between Subautonomous Systems in a Confederation 50
Configuring MPLS Forwarding for ASBR Confederations 53
Configuring a Static Route to an ASBR Confederation Peer 54
Configuring Carrier Supporting Carrier 55
Identifying the Carrier Supporting Carrier Topology 55
Configuring the Backbone Carrier Core 56
Configuring the CSC-PE and CSC-CE Routers 56
Configuring a Static Route to a Peer 57
Verifying the MPLS Layer 3 VPN Configuration 58
Configuring L3VPN over GRE 62
Creating a GRE Tunnel between Provider Edge Routers 62
Configuring IGP between Provider Edge Routers 63
Configuring LDP/GRE on the Provider Edge Routers 65
Configuring L3VPN 68
Configuration Examples for Implementing MPLS Layer 3 VPNs 73
Configuring an MPLS VPN Using BGP: Example 73
Configuring the Routing Information Protocol on the PE Router: Example 74
Configuring the PE Router Using EIGRP: Example 75
Configuration Examples for MPLS VPN CSC 75
Configuring the Backbone Carrier Core: Examples 75
Configuring the Links Between CSC-PE and CSC-CE Routers: Examples 75
Configuring a Static Route to a Peer: Example 76
Configuring L3VPN over GRE: Example 76

CHAPTER 3  Implementing IPv6 VPN Provider Edge Transport over MPLS 81
Prerequisites for Implementing 6PE/VPE 81
Route Targets  135
Route Re-origination  135
Route Address-Family and Encoded Address-Family  135
Local VPNv4 or VPNv6 Routes Advertisement  136
Data Center VXLAN with Support for MP-BGP  136
Default-Originate Forwarding to BGP EVPN Neighbor  136
Configure Data Center Interconnect Router  136
  Configure VRF and route targets import/export rules  137
Configure Bridge Domain for DCI Gateway  138
Configure VTEP  139
Configure EVPN BGP neighbor and route advertisements  141
Configure L3VPN BGP neighbor relationship and route advertisements  143
Verification of Data Center Gateway Configuration  144
Example: Data Center Interconnection Layer 3 Gateway Configuration  153
OpFlex  155
OpFlex Topology  155
Restrictions  156
Configure OpFlex  156
  Configure BGP  156
    Configure BGP Session on the Fabric Side  157
    Configure BGP Session on the WAN Side  159
  Configure DCI Underlay for Fabric and WAN Interfaces  160
  Configure IGP for ACI and WAN Reachability  161
  Configure MPLS towards WAN  163
  Configure FTI Auto-Configuration Parameters  165
  Configure OpFlex Session  166
EVPN Default VRF Route Leaking  168
  EVPN Default VRF Route Leaking on the DCI for Internet Connectivity  169
  Leaking Routes from Default-VRF to Data Center-VRF  169
  Leaking Routes to Default-VRF from Data Center-VRF  172
    Sample Router Configuration  173
    Sample Router Configuration: with default-originate  174
EVPN Service VRF Route Leaking  175
  EVPN Service VRF Route Leaking on the DCI for Service Connectivity  177
Preface

From Release 6.1.2 onwards, Cisco introduces support for the 64-bit Linux-based IOS XR operating system. Extensive feature parity is maintained between the 32-bit and 64-bit environments. Unless explicitly marked otherwise, the contents of this document are applicable for both the environments. For more details on Cisco IOS XR 64 bit, refer to the Release Notes for Cisco ASR 9000 Series Routers, Release 6.1.2 document.

This guide describes the Cisco ASR 9000 Series Router configurations. The preface for the L2VPN and Ethernet Services Configuration Guide for Cisco ASR 9000 Series Routers contains these sections:

- Changes to This Document, on page xi
- Obtaining Documentation and Submitting a Service Request, on page xi

Changes to This Document

The following table lists the technical changes made to this document since it was first published.

<table>
<thead>
<tr>
<th>Date</th>
<th>Change Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>September 2017</td>
<td>Initial release of this document.</td>
</tr>
<tr>
<td>March 2018</td>
<td>Republished for Release 6.3.2.</td>
</tr>
</tbody>
</table>

Obtaining Documentation and Submitting a Service Request

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


Subscribe to the What's New in Cisco Product Documentation as a Really Simple Syndication (RSS) feed and set content to be delivered directly to your desktop using a reader application. The RSS feeds are a free service and Cisco currently supports RSS version 2.0.
New and Changed L3VPN Features

Table 1: L3VPN Features Added or Modified in IOS XR Release 6.3.x

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Changed in Release</th>
<th>Where Documented</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>No new features introduced</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>
CHAPTER 2

Implementing MPLS Layer 3 VPNs

A Multiprotocol Label Switching (MPLS) Layer 3 Virtual Private Network (VPN) consists of a set of sites that are interconnected by means of an MPLS provider core network. At each customer site, one or more customer edge (CE) routers attach to one or more provider edge (PE) routers.

This module provides the conceptual and configuration information for MPLS Layer 3 VPNs on Cisco IOS XR software.

Note
You must acquire an evaluation or permanent license in order to use MPLS Layer 3 VPN functionality. However, if you are upgrading from a previous version of the software, MPLS Layer 3 VPN functionality will continue to work using an implicit license for 90 days (during which time, you can purchase a permanent license). For more information about licenses, see the Software Entitlement on the Cisco ASR 9000 Series Router module in the System Management Configuration Guide for Cisco ASR 9000 Series Routers.

Note
For a complete description of the commands listed in this module, refer to the VPN and Ethernet Services Command Reference for Cisco ASR 9000 Series Routers. To locate documentation of other commands that appear in this chapter, use the command reference master index, or search online.

Feature History for Implementing MPLS Layer 3 VPNs

- Prerequisites for Implementing MPLS L3VPN, on page 3
- MPLS L3VPN Restrictions, on page 4
- Information About MPLS Layer 3 VPNs, on page 5
- Inter-AS Support for L3VPN, on page 9
- Carrier Supporting Carrier Support for L3VPN, on page 15
- How to Implement MPLS Layer 3 VPNs, on page 18
- Configuration Examples for Implementing MPLS Layer 3 VPNs, on page 73

Prerequisites for Implementing MPLS L3VPN

The following prerequisites are required to configure MPLS Layer 3 VPN:
To perform these configuration tasks, your Cisco IOS XR software system administrator must assign you to a user group associated with a task group that includes the corresponding command task IDs. All command task IDs are listed in individual command references and in the Cisco IOS XR Task ID Reference Guide.

If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

You must be in a user group associated with a task group that includes the proper task IDs for:

- BGP commands
- MPLS commands (generally)
- MPLS Layer 3 VPN commands

To configure MPLS Layer 3 VPNs, routers must support MPLS forwarding and Forwarding Information Base (FIB).

The following prerequisites are required for configuring MPLS VPN Inter-AS with autonomous system boundary routers (ASBRs) exchanging VPN-IPv4 addresses or IPv4 routes and MPLS labels:

- Before configuring external Border Gateway Protocol (eBGP) routing between autonomous systems or subautonomous systems in an MPLS VPN, ensure that all MPLS VPN routing instances and sessions are properly configured (see the How to Implement MPLS Layer 3 VPNs, for procedures)

These following tasks must be performed:

- Define VPN routing instances
- Configure BGP routing sessions in the MPLS core
- Configure PE-to-PE routing sessions in the MPLS core
- Configure BGP PE-to-CE routing sessions
- Configure a VPN-IPv4 eBGP session between directly connected ASBRs

### MPLS L3VPN Restrictions

The following are restrictions for implementing MPLS Layer 3 VPNs:

- Multihop VPN-IPv4 eBGP is not supported for configuring eBGP routing between autonomous systems or subautonomous systems in an MPLS VPN.
- MPLS VPN supports only IPv4 address families.

The following restrictions apply when configuring MPLS VPN Inter-AS with ASBRs exchanging IPv4 routes and MPLS labels:

- For networks configured with eBGP multihop, a label switched path (LSP) must be configured between non adjacent routers.
- Inter-AS supports IPv4 routes only. IPv6 is not supported.
The physical interfaces that connect the BGP speakers must support FIB and MPLS.

The following restrictions apply to routing protocols OSPF and RIP:

- IPv6 is not supported on OSPF and RIP.

Information About MPLS Layer 3 VPNs

To implement MPLS Layer 3 VPNs, you need to understand the following concepts:

MPLS L3VPN Overview

Before defining an MPLS VPN, VPN in general must be defined. A VPN is:

- An IP-based network delivering private network services over a public infrastructure
- A set of sites that are allowed to communicate with each other privately over the Internet or other public or private networks

Conventional VPNs are created by configuring a full mesh of tunnels or permanent virtual circuits (PVCs) to all sites in a VPN. This type of VPN is not easy to maintain or expand, as adding a new site requires changing each edge device in the VPN.

MPLS-based VPNs are created in Layer 3 and are based on the peer model. The peer model enables the service provider and the customer to exchange Layer 3 routing information. The service provider relays the data between the customer sites without customer involvement.

MPLS VPNs are easier to manage and expand than conventional VPNs. When a new site is added to an MPLS VPN, only the edge router of the service provider that provides services to the customer site needs to be updated.

The components of the MPLS VPN are described as follows:

- Provider (P) router—Router in the core of the provider network. PE routers run MPLS switching and do not attach VPN labels to routed packets. VPN labels are used to direct data packets to the correct private network or customer edge router.
- PE router—Router that attaches the VPN label to incoming packets based on the interface or subinterface on which they are received, and also attaches the MPLS core labels. A PE router attaches directly to a CE router.
- Customer (C) router—Router in the Internet service provider (ISP) or enterprise network.
- Customer edge (CE) router—Edge router on the network of the ISP that connects to the PE router on the network. A CE router must interface with a PE router.

This following figures shows a basic MPLS VPN topology.
MPLS L3VPN Benefits

MPLS L3VPN provides the following benefits:

- Service providers can deploy scalable VPNs and deliver value-added services.
- Connectionless service guarantees that no prior action is necessary to establish communication between hosts.
- Centralized Service: Building VPNs in Layer 3 permits delivery of targeted services to a group of users represented by a VPN.
- Scalability: Create scalable VPNs using connection-oriented, point-to-point overlays, Frame Relay, or ATM virtual connections.
- Security: Security is provided at the edge of a provider network (ensuring that packets received from a customer are placed on the correct VPN) and in the backbone.
- Integrated Quality of Service (QoS) support: QoS provides the ability to address predictable performance and policy implementation and support for multiple levels of service in an MPLS VPN.
- Straightforward Migration: Service providers can deploy VPN services using a straightforward migration path.
- Migration for the end customer is simplified. There is no requirement to support MPLS on the CE router and no modifications are required for a customer intranet.

How MPLS L3VPN Works

MPLS VPN functionality is enabled at the edge of an MPLS network. The PE router performs the following tasks:

- Exchanges routing updates with the CE router
- Translates the CE routing information into VPN version 4 (VPNv4) routes.
- Exchanges VPNv4 and VPNv6 routes with other PE routers through the Multiprotocol Border Gateway Protocol (MP-BGP)
Virtual Routing and Forwarding Tables

Each VPN is associated with one or more VPN routing and forwarding (VRF) instances. A VRF defines the VPN membership of a customer site attached to a PE router. A VRF consists of the following components:

- An IP version 4 (IPv4) unicast routing table
- A derived FIB table
- A set of interfaces that use the forwarding table
- A set of rules and routing protocol parameters that control the information that is included in the routing table

These components are collectively called a VRF instance.

A one-to-one relationship does not necessarily exist between customer sites and VPNs. A site can be a member of multiple VPNs. However, a site can associate with only one VRF. A VRF contains all the routes available to the site from the VPNs of which it is a member.

Packet forwarding information is stored in the IP routing table and the FIB table for each VRF. A separate set of routing and FIB tables is maintained for each VRF. These tables prevent information from being forwarded outside a VPN and also prevent packets that are outside a VPN from being forwarded to a router within the VPN.

VPN Routing Information: Distribution

The distribution of VPN routing information is controlled through the use of VPN route target communities, implemented by BGP extended communities. VPN routing information is distributed as follows:

- When a VPN route that is learned from a CE router is injected into a BGP, a list of VPN route target extended community attributes is associated with it. Typically, the list of route target community extended values is set from an export list of route targets associated with the VRF from which the route was learned.

- An import list of route target extended communities is associated with each VRF. The import list defines route target extended community attributes that a route must have for the route to be imported into the VRF. For example, if the import list for a particular VRF includes route target extended communities A, B, and C, then any VPN route that carries any of those route target extended communities—A, B, or C—is imported into the VRF.

BGP Distribution of VPN Routing Information

A PE router can learn an IP prefix from the following sources:

- A CE router by static configuration
- An eBGP session with the CE router
- A Routing Information Protocol (RIP) exchange with the CE router
- Open Shortest Path First (OSPF), Enhanced Interior Gateway Routing Protocol (EIGRP), and RIP as Interior Gateway Protocols (IGPs)

The IP prefix is a member of the IPv4 address family. After the PE router learns the IP prefix, the PE converts it into the VPN-IPv4 prefix by combining it with a 64-bit route distinguisher. The generated prefix is a member of the VPN-IPv4 address family. It uniquely identifies the customer address, even if the customer site is using
globally nonunique (unregistered private) IP addresses. The route distinguisher used to generate the VPN-IPv4 prefix is specified by the `rd` command associated with the VRF on the PE router.

BGP distributes reachability information for VPN-IPv4 prefixes for each VPN. BGP communication takes place at two levels:

- Within the IP domain, known as an autonomous system.
- Between autonomous systems.

PE to PE or PE to route reflector (RR) sessions are iBGP sessions, and PE to CE sessions are eBGP sessions. PE to CE eBGP sessions can be directly or indirectly connected (eBGP multihop).

BGP propagates reachability information for VPN-IPv4 prefixes among PE routers by the BGP protocol extensions (see RFC 2283, Multiprotocol Extensions for BGP-4), which define support for address families other than IPv4. Using the extensions ensures that the routes for a given VPN are learned only by other members of that VPN, enabling members of the VPN to communicate with each other.

**MPLS Forwarding**

Based on routing information stored in the VRF IP routing table and the VRF FIB table, packets are forwarded to their destination using MPLS.

A PE router binds a label to each customer prefix learned from a CE router and includes the label in the network reachability information for the prefix that it advertises to other PE routers. When a PE router forwards a packet received from a CE router across the provider network, it labels the packet with the label learned from the destination PE router. When the destination PE router receives the labeled packet, it pops the label and uses it to direct the packet to the correct CE router. Label forwarding across the provider backbone is based on either dynamic label switching or traffic engineered paths. A customer data packet carries two levels of labels when traversing the backbone:

- The top label directs the packet to the correct PE router.
- The second label indicates how that PE router should forward the packet to the CE router.

More labels can be stacked if other features are enabled. For example, if traffic engineering (TE) tunnels with fast reroute (FRR) are enabled, the total number of labels imposed in the PE is four (Layer 3 VPN, Label Distribution Protocol (LDP), TE, and FRR).

**Automatic Route Distinguisher Assignment**

To take advantage of iBGP load balancing, every network VRF must be assigned a unique route distinguisher. VRF is require a route distinguisher for BGP to distinguish between potentially identical prefixes received from different VPNs.

With thousands of routers in a network each supporting multiple VRFs, configuration and management of route distinguishers across the network can present a problem. Cisco IOS XR software simplifies this process by assigning unique route distinguisher to VRFs using the `rd auto` command.

To assign a unique route distinguisher for each router, you must ensure that each router has a unique BGP router-id. If so, the `rd auto` command assigns a Type 1 route distinguisher to the VRF using the following format: `ip-address:number`. The IP address is specified by the BGP router-id statement and the number (which is derived as an unused index in the 0 to 65535 range) is unique across the VRFs.
Finally, route distinguisher values are checkpointed so that route distinguisher assignment to VRF is persistent across failover or process restart. If an route distinguisher is explicitly configured for a VRF, this value is not overridden by the autoroute distinguisher.

### MPLS L3VPN Major Components

An MPLS-based VPN network has three major components:

- **VPN route target communities**—A VPN route target community is a list of all members of a VPN community. VPN route targets need to be configured for each VPN community member.

- **Multiprotocol BGP (MP-BGP) peering of the VPN community PE routers**—MP-BGP propagates VRF reachability information to all members of a VPN community. MP-BGP peering needs to be configured in all PE routers within a VPN community.

- **MPLS forwarding**—MPLS transports all traffic between all VPN community members across a VPN service-provider network.

A one-to-one relationship does not necessarily exist between customer sites and VPNs. A given site can be a member of multiple VPNs. However, a site can associate with only one VRF. A customer-site VRF contains all the routes available to the site from the VPNs of which it is a member.

### Inter-AS Support for L3VPN

This section contains the following topics:

### Inter-AS Support: Overview

An autonomous system (AS) is a single network or group of networks that is controlled by a common system administration group and uses a single, clearly defined routing protocol.

As VPNs grow, their requirements expand. In some cases, VPNs need to reside on different autonomous systems in different geographic areas. In addition, some VPNs need to extend across multiple service providers (overlapping VPNs). Regardless of the complexity and location of the VPNs, the connection between autonomous systems must be seamless.

An MPLS VPN Inter-AS provides the following benefits:

- **Allows a VPN to cross more than one service provider backbone.**

  Service providers, running separate autonomous systems, can jointly offer MPLS VPN services to the same end customer. A VPN can begin at one customer site and traverse different VPN service provider backbones before arriving at another site of the same customer. Previously, MPLS VPN could traverse only a single BGP autonomous system service provider backbone. This feature lets multiple autonomous systems form a continuous, seamless network between customer sites of a service provider.

- **Allows a VPN to exist in different areas.**

  A service provider can create a VPN in different geographic areas. Having all VPN traffic flow through one point (between the areas) allows for better rate control of network traffic between the areas.

- **Allows confederations to optimize iBGP meshing.**
Internal Border Gateway Protocol (iBGP) meshing in an autonomous system is more organized and manageable. You can divide an autonomous system into multiple, separate subautonomous systems and then classify them into a single confederation. This capability lets a service provider offer MPLS VPNs across the confederation, as it supports the exchange of labeled VPN-IPv4 Network Layer Reachability Information (NLRI) between the subautonomous systems that form the confederation.

**Inter-AS and ASBRs**

Separate autonomous systems from different service providers can communicate by exchanging IPv4 NLRI and IPv6 in the form of VPN-IPv4 addresses. The ASBRs use eBGP to exchange that information. Then an Interior Gateway Protocol (IGP) distributes the network layer information for VPN-IPV4 prefixes throughout each VPN and each autonomous system. The following protocols are used for sharing routing information:

- Within an autonomous system, routing information is shared using an IGP.
- Between autonomous systems, routing information is shared using an eBGP. An eBGP lets service providers set up an interdomain routing system that guarantees the loop-free exchange of routing information between separate autonomous systems.

The primary function of an eBGP is to exchange network reachability information between autonomous systems, including information about the list of autonomous system routes. The autonomous systems use EBGP border edge routers to distribute the routes, which include label switching information. Each border edge router rewrites the next-hop and MPLS labels.

Inter-AS configurations supported in an MPLS VPN can include:

- Interprovider VPN—MPLS VPNs that include two or more autonomous systems, connected by separate border edge routers. The autonomous systems exchange routes using eBGP. No IGP or routing information is exchanged between the autonomous systems.
- BGP Confederations—MPLS VPNs that divide a single autonomous system into multiple subautonomous systems and classify them as a single, designated confederation. The network recognizes the confederation as a single autonomous system. The peers in the different autonomous systems communicate over eBGP sessions; however, they can exchange route information as if they were iBGP peers.

**Confederations**

A confederation is multiple subautonomous systems grouped together. A confederation reduces the total number of peer devices in an autonomous system. A confederation divides an autonomous system into subautonomous systems and assigns a confederation identifier to the autonomous systems. A VPN can span service providers running in separate autonomous systems or multiple subautonomous systems that form a confederation.

In a confederation, each subautonomous system is fully meshed with other subautonomous systems. The subautonomous systems communicate using an IGP, such as Open Shortest Path First (OSPF) or Intermediate System-to-Intermediate System (IS-IS). Each subautonomous system also has an eBGP connection to the other subautonomous systems. The confederation eBGP (CEBGP) border edge routers forward next-hop-self addresses between the specified subautonomous systems. The next-hop-self address forces the BGP to use a specified address as the next hop rather than letting the protocol choose the next hop.

You can configure a confederation with separate subautonomous systems two ways:
• Configure a router to forward next-hop-self addresses between only the CEBGP border edge routers (both directions). The subautonomous systems (iBGP peers) at the subautonomous system border do not forward the next-hop-self address. Each subautonomous system runs as a single IGP domain. However, the CEBGP border edge router addresses are known in the IGP domains.

• Configure a router to forward next-hop-self addresses between the CEBGP border edge routers (both directions) and within the iBGP peers at the subautonomous system border. Each subautonomous system runs as a single IGP domain but also forwards next-hop-self addresses between the PE routers in the domain. The CEBGP border edge router addresses are known in the IGP domains.

---

**Note**

eBGP Connection Between Two Subautonomous Systems in a Confederation figure illustrates how two autonomous systems exchange routes and forward packets. Subautonomous systems in a confederation use a similar method of exchanging routes and forwarding packets.

The figure below illustrates a typical MPLS VPN confederation configuration. In this configuration:

- The two CEBGP border edge routers exchange VPN-IPv4 addresses with labels between the two autonomous systems.

- The distributing router changes the next-hop addresses and labels and uses a next-hop-self address.

- IGP-1 and IGP-2 know the addresses of CEBGP-1 and CEBGP-2.

**Figure 2: eBGP Connection Between Two Subautonomous Systems in a Confederation**

In this confederation configuration:

- CEBGP border edge routers function as neighboring peers between the subautonomous systems. The subautonomous systems use eBGP to exchange route information.

- Each CEBGP border edge router (CEBG-1 and CEBGP-2) assigns a label for the router before distributing the route to the next subautonomous system. The CEBGP border edge router distributes the route as a VPN-IPv4 address by using the multiprotocol extensions of BGP. The label and the VPN identifier are encoded as part of the NLRI.
• Each PE and CEBGP border edge router assigns its own label to each VPN-IPv4 address prefix before redistributing the routes. The CEBGP border edge routers exchange IPV-IPv4 addresses with the labels. The next-hop-self address is included in the label (as the value of the eBGP next-hop attribute). Within the subautonomous systems, the CEBGP border edge router address is distributed throughout the iBGP neighbors, and the two CEBGP border edge routers are known to both confederations.

• For more information about how to configure confederations, see the .

MPLS VPN Inter-AS BGP Label Distribution

Note

This section is not applicable to Inter-AS over IP tunnels.

You can set up the MPLS VPN Inter-AS network so that the ASBRs exchange IPv4 routes with MPLS labels of the provider edge (PE) routers. Route reflectors (RRs) exchange VPN-IPv4 routes by using multihop, multiprotocol external Border Gateway Protocol (eBGP). This method of configuring the Inter-AS system is often called MPLS VPN Inter-AS BGP Label Distribution.

Configuring the Inter-AS system so that the ASBRs exchange the IPv4 routes and MPLS labels has the following benefits:

• Saves the ASBRs from having to store all the VPN-IPv4 routes. Using the route reflectors to store the VPN-IPv4 routes and forward them to the PE routers results in improved scalability compared with configurations in which the ASBR holds all the VPN-IPv4 routes and forwards the routes based on VPN-IPv4 labels.

• Having the route reflectors hold the VPN-IPv4 routes also simplifies the configuration at the border of the network.

• Enables a non-VPN core network to act as a transit network for VPN traffic. You can transport IPv4 routes with MPLS labels over a non-MPLS VPN service provider.

• Eliminates the need for any other label distribution protocol between adjacent label switch routers (LSRs). If two adjacent LSRs are also BGP peers, BGP can handle the distribution of the MPLS labels. No other label distribution protocol is needed between the two LSRs.

Exchanging IPv4 Routes with MPLS labels

Note

This section is not applicable to Inter-AS over IP tunnels.

You can set up a VPN service provider network to exchange IPv4 routes with MPLS labels. You can configure the VPN service provider network as follows:

• Route reflectors exchange VPN-IPv4 routes by using multihop, multiprotocol eBGP. This configuration also preserves the next-hop information and the VPN labels across the autonomous systems.

• A local PE router (for example, PE1 in the figure below) needs to know the routes and label information for the remote PE router (PE2).
This information can be exchanged between the PE routers and ASBRs in one of two ways:

- Internal Gateway Protocol (IGP) and Label Distribution Protocol (LDP): The ASBR can redistribute the IPv4 routes and MPLS labels it learned from eBGP into IGP and LDP and from IGP and LDP into eBGP.

- Internal Border Gateway Protocol (iBGP) IPv4 label distribution: The ASBR and PE router can use direct iBGP sessions to exchange VPN-IPv4 and IPv4 routes and MPLS labels.

Alternatively, the route reflector can reflect the IPv4 routes and MPLS labels learned from the ASBR to the PE routers in the VPN. This reflecting of learned IPv4 routes and MPLS labels is accomplished by enabling the ASBR to exchange IPv4 routes and MPLS labels with the route reflector. The route reflector also reflects the VPN-IPv4 routes to the PE routers in the VPN. For example, in VPN1, RR1 reflects to PE1 the VPN-IPv4 routes it learned and IPv4 routes and MPLS labels learned from ASBR1. Using the route reflectors to store the VPN-IPv4 routes and forward them through the PE routers and ASBRs allows for a scalable configuration.

Figure 3: VPNs Using eBGP and iBGP to Distribute Routes and MPLS Labels

**BGP Routing Information**

BGP routing information includes the following items:

- Network number (prefix), which is the IP address of the destination.

- Autonomous system (AS) path, which is a list of the other ASs through which a route passes on the way to the local router. The first AS in the list is closest to the local router; the last AS in the list is farthest from the local router and usually the AS where the route began.

- Path attributes, which provide other information about the AS path, for example, the next hop.

**BGP Messages and MPLS Labels**

MPLS labels are included in the update messages that a router sends. Routers exchange the following types of BGP messages:

- Open messages—After a router establishes a TCP connection with a neighboring router, the routers exchange open messages. This message contains the number of the autonomous system to which the router belongs and the IP address of the router that sent the message.

- Update messages—When a router has a new, changed, or broken route, it sends an update message to the neighboring router. This message contains the NLRI, which lists the IP addresses of the usable routes. The update message includes any routes that are no longer usable. The update message also includes
path attributes and the lengths of both the usable and unusable paths. Labels for VPN-IPv4 routes are encoded in the update message, as specified in RFC 2858. The labels for the IPv4 routes are encoded in the update message, as specified in RFC 3107.

- Keepalive messages—Routers exchange keepalive messages to determine if a neighboring router is still available to exchange routing information. The router sends these messages at regular intervals. (Sixty seconds is the default for Cisco routers.) The keepalive message does not contain routing data; it contains only a message header.

- Notification messages—When a router detects an error, it sends a notification message.

Sending MPLS Labels with Routes

When BGP (eBGP and iBGP) distributes a route, it can also distribute an MPLS label that is mapped to that route. The MPLS label mapping information for the route is carried in the BGP update message that contains the information about the route. If the next hop is not changed, the label is preserved.

When you issue the `show bgp neighbors ip-address` command on both BGP routers, the routers advertise to each other that they can then send MPLS labels with the routes. If the routers successfully negotiate their ability to send MPLS labels, the routers add MPLS labels to all outgoing BGP updates.

Generic Routing Encapsulation Support for L3VPN

Generic Routing Encapsulation (GRE) is a tunneling protocol that can encapsulate many types of packets to enable data transmission using a tunnel. The GRE tunneling protocol enables:

- High assurance Internet Protocol encryptor (HAIPE) devices for encryption over the public Internet and nonsecure connections.
- Service providers (that do not run MPLS in their core network) to provide VPN services along with the security services.

GRE is used with IP to create a virtual point-to-point link to routers at remote points in a network. For detailed information about configuring GRE tunnel interfaces, see the `<module-name>` module of the Cisco IOS XR Interfaces and Hardware Components Configuration Guide.

Note

GRE is used with IP to create a virtual point-to-point link to routers at remote points in a network. For detailed information about configuring GRE tunnel interfaces, refer to the Cisco IOS XR Interfaces and Hardware Components Configuration Guide. For a PE to PE (core) link, enable LDP (with implicit null) on the GRE interfaces for L3VPN.

GRE Restriction for L3VPN

The following restrictions are applicable to L3VPN forwarding over GRE:

- Carrier Supporting Carrier (CsC) or Inter-AS is not supported.
- GRE-based L3VPN does not interwork with MPLS or IP VPNs.
- GRE tunnel is supported only as a core link (PE-PE, PE-P, P-P, P-PE). A PE-CE (edge) link is not supported.
VPNv6 forwarding using GRE tunnels is not supported.

VPNv4 Forwarding Using GRE Tunnels

This section describes the working of VPNv4 forwarding over GRE tunnels. The following description assumes that GRE is used only as a core link between the encapsulation and decapsulation provider edge (PE) routers that are connected to one or more customer edge (CE) routers.

Ingress of Encapsulation Router

On receiving prefixes from the CE routers, Border Gateway Protocol (BGP) assigns the VPN label to the prefixes that need to be exported. These VPN prefixes are then forwarded to the Forwarding Information Base (FIB) using the Route Information Base (RIB) or the label switched database (LSD). The FIB then populates the prefix in the appropriate VRF table. The FIB also populates the label in the global label table. Using BGP, the prefixes are then relayed to the remote PE router (decapsulation router).

Egress of Encapsulation Router

The forwarding behavior on egress of the encapsulation PE router is similar to the MPLS VPN label imposition. Regardless of whether the VPN label imposition is performed on the ingress or egress side, the GRE tunnel forwards a packet that has an associated label. This labeled packet is then encapsulated with a GRE header and forwarded based on the IP header.

Ingress of Decapsulation Router

The decapsulation PE router learns the VPN prefixes and label information from the remote encapsulation PE router using BGP. The next-hop information for the VPN prefix is the address of the GRE tunnel interface connecting the two PE routers. BGP downloads these prefixes to the RIB. The RIB downloads the routes to the FIB and the FIB installs the routes in the hardware.

Egress of Decapsulation Router

The egress forwarding behavior on the decapsulation PE router is similar to VPN disposition and forwarding, based on the protocol type of the inner payload.

Carrier Supporting Carrier Support for L3VPN

This section provides conceptual information about MPLS VPN Carrier Supporting Carrier (CSC) functionality and includes the following topics:

- CSC Prerequisites
- CSC Benefits
- Configuration Options for the Backbone and Customer Carriers

Throughout this document, the following terminology is used in the context of CSC:

- **backbone carrier**—Service provider that provides the segment of the backbone network to the other provider. A backbone carrier offers BGP and MPLS VPN services.
- **customer carrier**—Service provider that uses the segment of the backbone network. The customer carrier may be an Internet service provider (ISP) or a BGP/MPLS VPN service provider.
CE router—A customer edge router is part of a customer network and interfaces to a provider edge (PE) router. In this document, the CE router sits on the edge of the customer carrier network.

PE router—A provider edge router is part of a service provider's network connected to a customer edge (CE) router. In this document, the PE router sits on the edge of the backbone carrier network.

ASBR—An autonomous system boundary router connects one autonomous system to another.

CSC Prerequisites

The following prerequisites are required to configure CSC:

• You must be able to configure MPLS VPNs with end-to-end (CE-to-CE router) pings working.
• You must be able to configure Interior Gateway Protocols (IGPs), MPLS Label Distribution Protocol (LDP), and Multiprotocol Border Gateway Protocol (MP-BGP).
• You must ensure that CSC-PE and CSC-CE routers support BGP label distribution.

Note

BGP is the only supported label distribution protocol on the link between CE and PE.

CSC Benefits

This section describes the benefits of CSC to the backbone carrier and customer carriers.

Benefits to the Backbone Carrier

• The backbone carrier can accommodate many customer carriers and give them access to its backbone.
• The MPLS VPN carrier supporting carrier feature is scalable.
• The MPLS VPN carrier supporting carrier feature is a flexible solution.

Benefits to the Customer Carriers

• The MPLS VPN carrier supporting carrier feature removes from the customer carrier the burden of configuring, operating, and maintaining its own backbone.
• Customer carriers who use the VPN services provided by the backbone carrier receive the same level of security that Frame Relay or ATM-based VPNs provide.
• Customer carriers can use any link layer technology to connect the CE routers to the PE routers.
• The customer carrier can use any addressing scheme and still be supported by a backbone carrier.

Benefits of Implementing MPLS VPN CSC Using BGP

The benefits of using BGP to distribute IPv4 routes and MPLS label routes are:

• BGP takes the place of an IGP and LDP in a VPN forwarding and routing instance (VRF) table.
• BGP is the preferred routing protocol for connecting two ISPs.
Configuration Options for the Backbone and Customer Carriers

To enable CSC, the backbone and customer carriers must be configured accordingly:

- The backbone carrier must offer BGP and MPLS VPN services.

- The customer carrier can take several networking forms. The customer carrier can be:
  - An ISP with an IP core (see the “Customer Carrier: ISP with IP Core”).
  - An MPLS service provider with or without VPN services (see “Customer Carrier: MPLS Service Provider”).

An IGP in the customer carrier network is used to distribute next hops and loopbacks to the CSC-CE. IBGP with label sessions are used in the customer carrier network to distribute next hops and loopbacks to the CSC-CE.

Customer Carrier: ISP with IP Core

The following figure shows a network configuration where the customer carrier is an ISP. The customer carrier has two sites, each of which is a point of presence (POP). The customer carrier connects these sites using a VPN service provided by the backbone carrier. The backbone carrier uses MPLS or IP tunnels to provide VPN services. The ISP sites use IP.

*Figure 4: Network: Customer Carrier Is an ISP*

The links between the CE and PE routers use eBGP to distribute IPv4 routes and MPLS labels. Between the links, the PE routers use multiprotocol iBGP to distribute VPNv4 routes.

Customer Carrier: MPLS Service Provider

The following figure shows a network configuration where the backbone carrier and the customer carrier are BGP/MPLS VPN service providers. The customer carrier has two sites. The customer carrier uses MPLS in its network while the backbone carrier may use MPLS or IP tunnels in its network.
In Network: Customer Carrier Is an MPLS VPN Service Provider configuration, the customer carrier can configure its network in one of these ways:

- The customer carrier can run an IGP and LDP in its core network. In this case, the CSC-CE1 router in the customer carrier redistributes the eBGP routes it learns from the CSC-PE1 router of the backbone carrier to an IGP.
- The CSC-CE1 router of the customer carrier system can run an IPv4 and labels iBGP session with the PE1 router.

How to Implement MPLS Layer 3 VPNs

This section contains instructions for the following tasks:

Configuring the Core Network

Configuring the core network includes the following tasks:

Assessing the Needs of MPLS VPN Customers

Before configuring an MPLS VPN, the core network topology must be identified so that it can best serve MPLS VPN customers. Perform this task to identify the core network topology.

SUMMARY STEPS

1. Identify the size of the network.
2. Identify the routing protocols in the core.
3. Determine if MPLS High Availability support is required.
4. Determine if BGP load sharing and redundant paths are required.

DETAILED STEPS

Step 1 Identify the size of the network.

Identify the following to determine the number of routers and ports required:
• How many customers will be supported?
• How many VPNs are required for each customer?
• How many virtual routing and forwarding (VRF) instances are there for each VPN?

Step 2 Identify the routing protocols in the core.
Determine which routing protocols are required in the core network.

Step 3 Determine if MPLS High Availability support is required.
MPLS VPN nonstop forwarding and graceful restart are supported on select routers and Cisco IOS XR software releases.

Step 4 Determine if BGP load sharing and redundant paths are required.
Determine if BGP load sharing and redundant paths in the MPLS VPN core are required.

Configuring Routing Protocols in the Core
To configure a routing protocol, see the Routing Configuration Guide for Cisco ASR 9000 Series Routers.

Configuring MPLS in the Core
To enable MPLS on all routers in the core, you must configure a Label Distribution Protocol (LDP). You can use either of the following as an LDP:

• MPLS LDP—See the Implementing MPLS Label Distribution Protocol chapter in the MPLS Configuration Guide for Cisco ASR 9000 Series Routers for configuration information.


Determining if FIB Is Enabled in the Core
Forwarding Information Base (FIB) must be enabled on all routers in the core, including the provider edge (PE) routers. For information on how to determine if FIB is enabled, see the Implementing Cisco Express Forwarding module in the IP Addresses and Services Configuration Guide for Cisco ASR 9000 Series Routers.

Configuring Multiprotocol BGP on the PE Routers and Route Reflectors
Perform this task to configure multiprotocol BGP (MP-BGP) connectivity on the PE routers and route reflectors.

SUMMARY STEPS
1. configure
2. router bgp autonomous-system-number
3. address-family vpv4 unicast or address-family vpv6 unicast
4. neighbor ip-address remote-as autonomous-system-number
5. address-family vpv4 unicast or address-family vpv6 unicast
6. Use the commit or end command.
DETAILED STEPS

Step 1 configure
Example:

RP/0/RSP0/CPU0:router# configure
Enters the Global Configuration mode.

Step 2 router bgp autonomous-system-number
Example:

RP/0/RSP0/CPU0:router(config)# router bgp 120
Enters BGP configuration mode allowing you to configure the BGP routing process.

Step 3 address-family vpnv4 unicast or address-family vpnv6 unicast
Example:

RP/0/RSP0/CPU0:router(config-bgp)# address-family vpnv4 unicast
Enters VPNv4 or VPNv6 address family configuration mode for the VPNv4 or VPNv6 address family.

Step 4 neighbor ip-address remote-as autonomous-system-number
Example:

RP/0/RSP0/CPU0:router(config-bgp)# neighbor 172.168.40.24 remote-as 2002
Creates a neighbor and assigns it a remote autonomous system number.

Step 5 address-family vpnv4 unicast or address-family vpnv6 unicast
Example:

RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family vpnv4 unicast
Enters VPNv4 or VPNv6 address family configuration mode for the VPNv4 or VPNv6 address family.

Step 6 Use the commit or end command.
commit - Saves the configuration changes and remains within the configuration session.
end - Prompts user to take one of these actions:
  • Yes - Saves configuration changes and exits the configuration session.
  • No - Exits the configuration session without committing the configuration changes.
  • Cancel - Remains in the configuration mode, without committing the configuration changes.
Connecting MPLS VPN Customers

To connect MPLS VPN customers to the VPN, perform the following tasks:

Defining VRFs on the PE Routers to Enable Customer Connectivity

Perform this task to define VPN routing and forwarding (VRF) instances.

SUMMARY STEPS

1. configure
2. vrf vrf-name
3. address-family ipv4 unicast
4. import route-policy policy-name
5. import route-target [ as-number:nn | ip-address:nn ]
6. export route-policy policy-name
7. export route-target [ as-number:nn | ip-address:nn ]
8. exit
9. exit
10. router bgp autonomous-system-number
11. vrf vrf-name
12. rd [ as-number | ip-address | auto ]
13. Use the commit or end command.

DETAILED STEPS

Step 1 configure

Example:

```
RP/0/RSP0/CPU0:router# configure
```
Enters Global Configuration mode.

Step 2 vrf vrf-name

Example:

```
RP/0/RSP0/CPU0:router(config)# vrf vrf_1
```
Configures a VRF instance and enters VRF configuration mode.

Step 3 address-family ipv4 unicast

Example:

```
RP/0/RSP0/CPU0:router(config-vrf)# address-family ipv4 unicast
```
Enters VRF address family configuration mode for the IPv4 address family.

Step 4 import route-policy policy-name
Step 5

import route-target [ as-number:nn | ip-address:nn ]

Example:

RP/0/RSP0/CPU0:router(config-vrf) # import route-target 120:1

Allows exported VPN routes to be imported into the VPN if one of the route targets of the exported route matches one of the local VPN import route targets.

Step 6

export route-policy policy-name

Example:

RP/0/RSP0/CPU0:router(config-vrf) # export route-policy policy_B

Step 7

export route-target [ as-number:nn | ip-address:nn ]

Example:

RP/0/RSP0/CPU0:router(config-vrf) # export route-target 120:2

Step 8

exit

Example:

RP/0/RSP0/CPU0:router(config-vrf) # exit

Step 9

exit

Example:

RP/0/RSP0/CPU0:router(config-vrf) # exit

Step 10

router bgp autonomous-system-number

Example:

RP/0/RSP0/CPU0:router(config) # router bgp 120
Enters BGP configuration mode allowing you to configure the BGP routing process.

Step 11  
```
vrf vrf-name
```

Example:
```
RP/0/RSP0/CPU0:router(config-bgp)# vrf vrf_1
```

Configures a VRF instance and enters VRF configuration mode for BGP routing.

Step 12  
```
rd { as-number | ip-address | auto }
```

Example:
```
RP/0/RSP0/CPU0:router(config-bgp-vrf)# rd auto
```

Automatically assigns a unique route distinguisher (RD) to vrf_1.

Step 13  
Use the `commit` or `end` command.

`commit` - Saves the configuration changes and remains within the configuration session.

`end` - Prompts user to take one of these actions:
- `Yes` - Saves configuration changes and exits the configuration session.
- `No` - Exits the configuration session without committing the configuration changes.
- `Cancel` - Remains in the configuration mode, without committing the configuration changes.

---

**Configuring VRF Interfaces on PE Routers for Each VPN Customer**

Perform this task to associate a VPN routing and forwarding (VRF) instance with an interface or a subinterface on the PE routers.

**Note**  
You must remove IPv4/IPv6 addresses from an interface prior to assigning, removing, or changing an interface's VRF. If this is not done in advance, any attempt to change the VRF on an IP interface is rejected.

**SUMMARY STEPS**

1. `configure`
2. `interface type interface-path-id`
3. `vrf vrf-name`
4. `ipv4 address ipv4-address mask`
5. Use the `commit` or `end` command.

**DETAILED STEPS**

Step 1  
`configure`
Configuring BGP as the Routing Protocol Between the PE and CE Routers

Perform this task to configure PE-to-CE routing sessions using BGP.

**SUMMARY STEPS**

1. configure
2. router bgp autonomous-system-number
3. bgp router-id {ip-address}
4. vrf vrf-name
5. label-allocation-mode per-ce
6. address-family ipv4 unicast

**Example:**

RP/0/RSP0/CPU0:router# configure

Enters Global Configuration mode.

**Step 2**

`interface type interface-path-id`

**Example:**

RP/0/RSP0/CPU0:router(config)# interface TenGigE 0/3/0/0

Enters interface configuration mode.

**Step 3**

`vrf vrf-name`

**Example:**

RP/0/RSP0/CPU0:router(config-if)# vrf vrf_A

Configures a VRF instance and enters VRF configuration mode.

**Step 4**

`ipv4 address ipv4-address mask`

**Example:**

RP/0/RSP0/CPU0:router(config-if)# ipv4 address 192.168.1.27 255.255.255.0

Configures a primary IPv4 address for the specified interface.

**Step 5**

Use the `commit` or `end` command.

- `commit` - Saves the configuration changes and remains within the configuration session.
- `end` - Prompts user to take one of these actions:
  - `Yes` - Saves configuration changes and exits the configuration session.
  - `No` - Exits the configuration session without committing the configuration changes.
  - `Cancel` -Remains in the configuration mode, without committing the configuration changes.
Implementing MPLS Layer 3 VPNs

Configuring BGP as the Routing Protocol Between the PE and CE Routers

7. Do one of the following:
   - redistribute connected [ metric metric-value ] [ route-policy route-policy-name ]
   - redistribute isis process-id [ level { 1 | 1-inter-area | 2 } ] [ metric metric-value ] [ route-policy route-policy-name ]
   - redistribute ospf process-id [ match { external { 1 | 2 } | internal | nssa-external { 1 | 2 } } ] [ metric metric-value ] [ route-policy route-policy-name ]
   - redistribute static [ metric metric-value ] [ route-policy route-policy-name ]

8. aggregate-address address/mask-length [as-set] [as-confed-set] [summary-only] [route-policy route-policy-name]
9. network {ip-address/prefix-length | ip-address mask} [ route-policy route-policy-name ]
10. exit
11. neighbor ip-address
12. remote-as autonomous-system-number
13. password { clear | encrypted } password
14. ebgp-multihop [ ttl-value ]
15. address-family ipv4 unicast
16. allowas-in [as-occurrence-number]
17. route-policy route-policy-name in
18. route-policy route-policy-name out
19. Use the commit or end command.

DETAILED STEPS

---

Step 1 configure
Example:

RP/0/RSP0/CPU0:router# configure

Enters Global Configuration mode.

Step 2 router bgp autonomous-system-number
Example:

RP/0/RSP0/CPU0:router(config)# router bgp 120

Enters Border Gateway Protocol (BGP) configuration mode allowing you to configure the BGP routing process.

Step 3 bgp router-id {ip-address}
Example:

RP/0/RSP0/CPU0:router(config-bgp)# bgp router-id 192.168.70.24

Configures the local router with a router ID of 192.168.70.24.

Step 4 vrf vrf-name
Example:

RP/0/RSP0/CPU0:router(config-bgp)# vrf vrf_1

Configures a VPN routing and forwarding (VRF) instance and enters VRF configuration mode for BGP routing.

Step 5 label-allocation-mode per-ce

Example:

RP/0/RSP0/CPU0:router(config-bgp-vrf)# label-allocation-mode per-ce

Sets the MPLS VPN label allocation mode for each customer edge (CE) label mode allowing the provider edge (PE) router to allocate one label for every immediate next-hop.

Step 6 address-family ipv4 unicast

Example:

RP/0/RSP0/CPU0:router(config-bgp-vrf)# address-family ipv4 unicast

Enters VRF address family configuration mode for the IPv4 address family.

Step 7 Do one of the following:

- **redistribute connected** [metric metric-value] [route-policy route-policy-name]
- **redistribute isis process-id** [level {1 | 1-inter-area | 2}] [metric metric-value] [route-policy route-policy-name]
- **redistribute ospf process-id** [match {external [1 | 2] | internal | nssa-external [1 | 2]}] [metric metric-value] [route-policy route-policy-name]
- **redistribute static** [metric metric-value] [route-policy route-policy-name]

Example:

RP/0/RSP0/CPU0:router(config-bgp-vrf-af)# redistribute connected

Causes routes to be redistributed into BGP. The routes that can be redistributed into BGP are:

- Connected
- Intermediate System-to-Intermediate System (IS-IS)
- Open Shortest Path First (OSPF)
- Static

Step 8 aggregate-address address/mask-length [as-set] [as-confed-set] [summary-only] [route-policy route-policy-name]

Example:

RP/0/RSP0/CPU0:router(config-bgp-vrf-af)# aggregate-address 10.0.0.0/8 as-set

Creates an aggregate address. The path advertised for this route is an autonomous system set consisting of all elements contained in all paths that are being summarized.

- The as-set keyword generates autonomous system set path information and community information from contributing paths.
• The **as-confed-set** keyword generates autonomous system confederation set path information from contributing paths.
• The **summary-only** keyword filters all more specific routes from updates.
• The **route-policy route-policy-name** keyword and argument specify the route policy used to set the attributes of the aggregate route.

### Step 9  
**network {ip-address/prefix-length | ip-address mask} [route-policy route-policy-name]**

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-vrf-af)# network 172.20.0.0/16
```

Configures the local router to originate and advertise the specified network.

### Step 10  
**exit**

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-vrf-af)# exit
```

Exits VRF address family configuration mode and returns the router to VRF configuration mode for BGP routing.

### Step 11  
**neighbor ip-address**

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-vrf)# neighbor 172.168.40.24
```

Places the router in VRF neighbor configuration mode for BGP routing and configures the neighbor IP address 172.168.40.24 as a BGP peer.

### Step 12  
**remote-as autonomous-system-number**

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-vrf-nbr)# remote-as 2002
```

Creates a neighbor and assigns it a remote autonomous system number.

### Step 13  
**password {clear | encrypted} password**

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-vrf-nbr)# password clear pswd123
```

Configures neighbor 172.168.40.24 to use MD5 authentication with the password pswd123.

### Step 14  
**ebgp-multihop [ttl-value]**

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-vrf-nbr)# ebgp-multihop
```

Allows a BGP connection to neighbor 172.168.40.24.
Configuring RIPv2 as the Routing Protocol Between the PE and CE Routers

Perform this task to configure provider edge (PE)-to-customer edge (CE) routing sessions using Routing Information Protocol version 2 (RIPv2).

SUMMARY STEPS

1. configure
2. router rip
3. vrf vrf-name

Configuring RIPv2 as the Routing Protocol Between the PE and CE Routers

Step 15 address-family ipv4 unicast

Example:

RP/0/RSP0/CPU0:router(config-bgp-vrf-nbr)# address-family ipv4 unicast

Enters VRF neighbor address family configuration mode for BGP routing.

Step 16 allowas-in [as-occurrence-number]

Example:

RP/0/RSP0/CPU0:router(config-bgp-vrf-nbr-af)# allowas-in 3

Replaces the neighbor autonomous system number (ASN) with the PE ASN in the AS path three times.

Step 17 route-policy route-policy-name in

Example:

RP/0/RSP0/CPU0:router(config-bgp-vrf-nbr-af)# route-policy In-Ipv4 in

Applies the In-Ipv4 policy to inbound IPv4 unicast routes.

Step 18 route-policy route-policy-name out

Example:

RP/0/RSP0/CPU0:router(config-bgp-vrf-nbr-af)# route-policy In-Ipv4 in

Applies the In-Ipv4 policy to outbound IPv4 unicast routes.

Step 19 Use the commit or end command.

commit - Saves the configuration changes and remains within the configuration session.
end - Prompts user to take one of these actions:
• Yes - Saves configuration changes and exits the configuration session.
• No - Exits the configuration session without committing the configuration changes.
• Cancel - Remains in the configuration mode, without committing the configuration changes.
4. **interface** type instance
5. **site-of-origin** \{ as-number : number | ip-address : number \}
6. **exit**
7. Do one of the following:
   - **redistribute bgp** as-number \[ [ external | internal | local ] \ [ route-policy name \]
   - **redistribute connected** \[ route-policy name \]
   - **redistribute isis** process-id \[ level-1 | level-1-2 | level-2 \] \[ route-policy name \]
   - **redistribute eigrp** as-number \[ route-policy name \]
   - **redistribute ospf** process-id \[ match \{ external [ 1 | 2 ] | internal | nssa-external [ 1 | 2 ] \} \[ route-policy name \]
   - **redistribute static** \[ route-policy name \]
8. Use the commit or end command.

**DETAILED STEPS**

**Step 1**
configure

**Example:**

```
RP/0/RSP0/CPU0:router# configure
```

Enters Global Configuration mode.

**Step 2**
router rip

**Example:**

```
RP/0/RSP0/CPU0:router(config)# router rip
```

Enters the Routing Information Protocol (RIP) configuration mode allowing you to configure the RIP routing process.

**Step 3**
**vrf** vrf-name

**Example:**

```
RP/0/RSP0/CPU0:router(config-rip)# vrf vrf_1
```

Configures a VPN routing and forwarding (VRF) instance and enters VRF configuration mode for RIP routing.

**Step 4**
**interface** type instance

**Example:**

```
RP/0/RSP0/CPU0:router(config-rip-vrf)# interface TenGigE 0/3/0/0
```

Enters VRF interface configuration mode.

**Step 5**
**site-of-origin** \{ as-number : number | ip-address : number \}

**Example:**
Identifies routes that have originated from a site so that the re-advertisement of that prefix back to the source site can be prevented. Uniquely identifies the site from which a PE router has learned a route.

**Step 6**

`exit`

**Example:**

```
RP/0/RSP0/CPU0:router(config-rip-vrf-if)# exit
```

Exits VRF interface configuration mode, and returns the router to VRF configuration mode for RIP routing.

**Step 7**

Do one of the following:

- `redistribute bgp as-number [ [ external | internal | local ] [ route-policy name ] ]`
- `redistribute connected [ route-policy name ]`
- `redistribute isis process-id [ level-1 | level-1-2 | level-2 ] [ route-policy name ]`
- `redistribute eigrp as-number [ route-policy name ]`
- `redistribute ospf process-id [ match { external [ 1 | 2 ] | internal | nssa-external [ 1 | 2 ] } ] [ route-policy name ]`
- `redistribute static [ route-policy name ]`

**Example:**

```
RP/0/RSP0/CPU0:router(config-rip-vrf-if)# redistribute connected
```

Causes routes to be redistributed into RIP. The routes that can be redistributed into RIP are:

- Border Gateway Protocol (BGP)
- Connected
- Enhanced Interior Gateway Routing Protocol (EIGRP)
- Intermediate System-to-Intermediate System (IS-IS)
- Open Shortest Path First (OSPF)
- Static

**Step 8**

Use the `commit` or `end` command.

- `commit` - Saves the configuration changes and remains within the configuration session.
- `end` - Prompts user to take one of these actions:
  - `Yes` - Saves configuration changes and exits the configuration session.
  - `No` - Exits the configuration session without committing the configuration changes.
  - `Cancel` - Remains in the configuration mode, without committing the configuration changes.

---

**Configuring Static Routes Between the PE and CE Routers**

Perform this task to configure provider edge (PE)-to-customer edge (CE) routing sessions that use static routes.
You must remove IPv4/IPv6 addresses from an interface prior to assigning, removing, or changing an interface's VRF. If this is not done in advance, any attempt to change the VRF on an IP interface is rejected.

SUMMARY STEPS

1. configure
2. router static
3. vrf vrf-name
4. address-family ipv4 unicast
5. prefix/mask [ vrf vrf-name ] { ip-address | type interface-path-id }
6. prefix/mask [vrf vrf-name] bfd fast-detect
7. Use the commit or end command.

DETAILED STEPS

Step 1 configure
Example:

```
RP/0/RSP0/CPU0:router# configure
```

Enters Global Configuration mode.

Step 2 router static
Example:

```
RP/0/RSP0/CPU0:router(config)# router static
```

Enters static routing configuration mode allowing you to configure the static routing process.

Step 3 vrf vrf-name
Example:

```
RP/0/RSP0/CPU0:router(config-static)# vrf vrf_1
```

Configures a VPN routing and forwarding (VRF) instance and enters VRF configuration mode for static routing.

Step 4 address-family ipv4 unicast
Example:

```
RP/0/RSP0/CPU0:router(config-static-vrf)# address-family ipv4 unicast
```

Enters VRF address family configuration mode for the IPv4 address family.

Step 5 prefix/mask [ vrf vrf-name ] { ip-address | type interface-path-id }
Configuring OSPF as the Routing Protocol Between the PE and CE Routers

Example:

RP/0/RSP0/CPU0:router(config-static-vrf-afi)# 172.168.40.24/24 vrf vrf_1 10.1.1.1

Assigns the static route to vrf_1.

Step 6  
**prefix/mask [vrf vrf-name] bfd fast-detect**

Example:

RP/0/RSP0/CPU0:router(config-static-vrf-afi)# 172.168.40.24/24 vrf vrf_1 bfd fast-detect

Enables bidirectional forwarding detection (BFD) to detect failures in the path between adjacent forwarding engines. This option is available is when the forwarding router address is specified in Step 5.

Step 7  
Use the **commit** or **end** command.

**commit** - Saves the configuration changes and remains within the configuration session.

**end** - Prompts user to take one of these actions:

- **Yes** - Saves configuration changes and exits the configuration session.
- **No** - Exits the configuration session without committing the configuration changes.
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.

Configuring OSPF as the Routing Protocol Between the PE and CE Routers

Perform this task to configure provider edge (PE)-to-customer edge (CE) routing sessions that use Open Shortest Path First (OSPF).

**SUMMARY STEPS**

1. **configure**
2. **router ospf process-name**
3. **vrf vrf-name**
4. **router-id {router-id | type interface-path-id}**
5. Do one of the following:

   - **redistribute bgp process-id [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]**
   - **redistribute connected [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]**
   - **redistribute ospf process-id [match {external [1 | 2] | internal | nssa-external [1 | 2]}] [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]**
   - **redistribute static [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]**
   - **redistribute eigrp process-id [match {external [1 | 2] | internal | nssa-external [1 | 2]}) [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]**
   - **redistribute rip [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]**
6. **area** area-id
7. **interface** type interface-path-id
8. Use the **commit** or **end** command.

### DETAILED STEPS

**Step 1**

```plaintext
configure
```

Example:

```
RP/0/RSP0/CPU0:router# configure
```

Enters Global Configuration mode.

**Step 2**

```plaintext
router ospf process-name
```

Example:

```
RP/0/RSP0/CPU0:router(config)# router ospf 109
```

Enters OSPF configuration mode allowing you to configure the OSPF routing process.

**Step 3**

```plaintext
vrf vrf-name
```

Example:

```
RP/0/RSP0/CPU0:router(config-ospf)# vrf vrf_1
```

Configures a VPN routing and forwarding (VRF) instance and enters VRF configuration mode for OSPF routing.

**Step 4**

```plaintext
router-id {router-id | type interface-path-id}
```

Example:

```
RP/0/RSP0/CPU0:router(config-ospf-vrf)# router-id 172.20.10.10
```

Configures the router ID for the OSPF routing process.

**Step 5**

Do one of the following:

- ```plaintext
  redistribute bgp process-id [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]
  ```
- ```plaintext
  redistribute connected [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]
  ```
- ```plaintext
  redistribute ospf process-id [match {external {1 | 2} | internal | nssa-external {1 | 2}}] [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]
  ```
- ```plaintext
  redistribute static [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]
  ```
- ```plaintext
  redistribute eigrp process-id [match {external {1 | 2} | internal | nssa-external {1 | 2}}] [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]
  ```
- ```plaintext
  redistribute rip [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]
  ```

Example:
Configuring EIGRP as the Routing Protocol Between the PE and CE Routers

Perform this task to configure provider edge (PE)-to-customer edge (CE) routing sessions that use Enhanced Interior Gateway Routing Protocol (EIGRP).

Using EIGRP between the PE and CE routers allows you to transparently connect EIGRP customer networks through an MPLS-enable Border Gateway Protocol (BGP) core network so that EIGRP routes are redistributed through the VPN across the BGP network as internal BGP (iBGP) routes.

Before you begin

BGP is configured in the network. See the Implementing BGP module in the Routing Configuration Guide for Cisco ASR 9000 Series Routers.
You must remove IPv4/IPv6 addresses from an interface prior to assigning, removing, or changing an interface's VRF. If this is not done in advance, any attempt to change the VRF on an IP interface is rejected.

**SUMMARY STEPS**

1. configure
2. router eigrp as-number
3. vrf vrf-name
4. address-family ipv4
5. router-id router-id
6. autonomous-system as-number
7. default-metric bandwidth delay reliability loading mtu
8. redistribute { { bgp | connected | isis | ospf | rip | static } [ as-number | instance-name ] } [ route-policy name ]
9. interface type interface-path-id
10. site-of-origin { as-number:number | ip-address : number } 
11. Use the commit or end command.

**DETAILED STEPS**

**Step 1** configure

*Example:*

```
RP/0/RSP0/CPU0:router# configure
```

Enters Global Configuration mode.

**Step 2** router eigrp as-number

*Example:*

```
RP/0/RSP0/CPU0:router(config)# router eigrp 24
```

Enters EIGRP configuration mode allowing you to configure the EIGRP routing process.

**Step 3** vrf vrf-name

*Example:*

```
RP/0/RSP0/CPU0:router(config-eigrp)# vrf vrf_1
```

Configures a VPN routing and forwarding (VRF) instance and enters VRF configuration mode for EIGRP routing.

**Step 4** address-family ipv4

*Example:*

```
RP/0/RSP0/CPU0:router(config-eigrp-vrf)# address family ipv4

Enters VRF address family configuration mode for the IPv4 address family.

**Step 5**

`router-id router-id`  
**Example:**

RP/0/RSP0/CPU0:router(config-eigrp-vrf-af)# router-id 172.20.0.0

Configures the router ID for the Enhanced Interior Gateway Routing Protocol (EIGRP) routing process.

**Step 6**

`autonomous-system as-number`  
**Example:**

RP/0/RSP0/CPU0:router(config-eigrp-vrf-af)# autonomous-system 6

Configures the EIGRP routing process to run within a VRF.

**Step 7**

`default-metric bandwidth delay reliability loading mtu`  
**Example:**

RP/0/RSP0/CPU0:router(config-eigrp-vrf-af)# default-metric 100000 4000 200 45 4470

Sets the metrics for an EIGRP.

**Step 8**

`redistribute { { bgp | connected | isis | ospf | rip | static } [ as-number | instance-name ] } [ route-policy name ]`  
**Example:**

RP/0/RSP0/CPU0:router(config-eigrp-vrf-af)# redistribute connected

Causes connected routes to be redistributed into EIGRP.

**Step 9**

`interface type interface-path-id`  
**Example:**

RP/0/RSP0/CPU0:router(config-eigrp-vrf-af)# interface TenGigE 0/3/0/0

Associates interface TenGigE 0/3/0/0 with the EIGRP routing process.

**Step 10**

`site-of-origin { as-number:number | ip-address : number }`  
**Example:**

RP/0/RSP0/CPU0:router(config-eigrp-vrf-af-if)# site-of-origin 201:1

Configures site of origin (SoO) on interface TenGigE 0/3/0/0.

**Step 11**

Use the `commit` or `end` command.
commit - Saves the configuration changes and remains within the configuration session.
end - Prompts user to take one of these actions:
• Yes - Saves configuration changes and exits the configuration session.
• No - Exits the configuration session without committing the configuration changes.
• Cancel - Remains in the configuration mode, without committing the configuration changes.

Configuring EIGRP Redistribution in the MPLS VPN

Perform this task for every provider edge (PE) router that provides VPN services to enable Enhanced Interior Gateway Routing Protocol (EIGRP) redistribution in the MPLS VPN.

Before you begin

The metric can be configured in the route-policy configuring using the redistribute command (or configured with the default-metric command). If an external route is received from another EIGRP autonomous system or a non-EIGRP network without a configured metric, the route is not installed in the EIGRP database. If an external route is received from another EIGRP autonomous system or a non-EIGRP network without a configured metric, the route is not advertised to the CE router. See the Implementing EIGRP module in the Routing Configuration Guide for Cisco ASR 9000 Series Routers.

Restriction
Redistribution between native EIGRP VPN routing and forwarding (VRF) instances is not supported. This behavior is designed.

SUMMARY STEPS

1. configure
2. router eigrp as-number
3. vrf vrf-name
4. address-family ipv4
5. redistribute bgp [as-number] [route-policy policy-name]
6. Use the commit or end command.

DETAILED STEPS

Step 1 configure
Example:

RP/0/RSP0/CPU0:router# configure

Enters Global Configuration mode.

Step 2 router eigrp as-number
Example:
RP/0/RSP0/CPU0:router(config)# router eigrp 24

Enters EIGRP configuration mode allowing you to configure the EIGRP routing process.

Step 3  vrf vrf-name

Example:

RP/0/RSP0/CPU0:router(config-eigrp)# vrf vrf_1

Configures a VRF instance and enters VRF configuration mode for EIGRP routing.

Step 4  address-family ipv4

Example:

RP/0/RSP0/CPU0:router(config-eigrp-vrf)# address family ipv4

Enters VRF address family configuration mode for the IPv4 address family.

Step 5  redistribute bgp [as-number] [route-policy policy-name]

Example:

RP/0/RSP0/CPU0:router(config-eigrp-vrf-af)# redistribute bgp 24 route-policy policy_A

Causes Border Gateway Protocol (BGP) routes to be redistributed into EIGRP.

Step 6  Use the commit or end command.

commit - Saves the configuration changes and remains within the configuration session.
end - Prompts user to take one of these actions:

• Yes - Saves configuration changes and exits the configuration session.
• No - Exits the configuration session without committing the configuration changes.
• Cancel - Remains in the configuration mode, without committing the configuration changes.

Providing VPN Connectivity Across Multiple Autonomous Systems with MPLS VPN Inter-AS with ASBRs Exchanging IPv4 Routes and MPLS Labels

This section is not applicable to Inter-AS over IP tunnels.

This section contains instructions for the following tasks:
Configuring ASBRs to Exchange IPv4 Routes and MPLS Labels

Perform this task to configure the autonomous system boundary routers (ASBRs) to exchange IPv4 routes and MPLS labels.

**SUMMARY STEPS**

1. configure
2. router bgp autonomous-system-number
3. address-family ipv4 unicast
4. allocate-label all
5. neighbor ip-address
6. remote-as autonomous-system-number
7. address-family ipv4 labeled-unicast
8. route-policy route-policy-name in
9. route-policy route-policy-name out
10. Use the commit or end command.

**DETAILED STEPS**

**Step 1**

`configure`

Example:

```
RP/0/RSP0/CPU0:router# configure
```

Enters Global Configuration mode.

**Step 2**

`router bgp autonomous-system-number`

Example:

```
RP/0/RSP0/CPU0:router(config)# router bgp 120
RP/0/RSP0/CPU0:router(config-bgp)#
```

Enters Border Gateway Protocol (BGP) configuration mode allowing you to configure the BGP routing process.

**Step 3**

`address-family ipv4 unicast`

Example:

```
RP/0/RSP0/CPU0:router(config-bgp)# address-family ipv4 unicast
RP/0/RSP0/CPU0:router(config-bgp-af)#
```

Enters global address family configuration mode for the IPv4 unicast address family.

**Step 4**

`allocate-label all`

Example:

```
RP/0/CPU0:router(config-bgp-af)# allocate-label all
```
Allocates the MPLS labels for a specific IPv4 unicast or VPN routing and forwarding (VRF) IPv4 unicast routes so that the BGP router can send labels with BGP routes to a neighboring router that is configured for a labeled-unicast session.

**Step 5**

**neighbor ip-address**

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-af)# neighbor 172.168.40.24
RP/0/RSP0/CPU0:router(config-bgp-nbr)#
```

Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address 172.168.40.24 as a BGP peer.

**Step 6**

**remote-as autonomous-system-number**

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 2002
```

Creates a neighbor and assigns it a remote autonomous system number.

**Step 7**

**address-family ipv4 labeled-unicast**

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family ipv4 labeled-unicast
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)
```

Enters neighbor address family configuration mode for the IPv4 labeled-unicast address family.

**Step 8**

**route-policy route-policy-name in**

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# route-policy pass-all in
```

Applies a routing policy to updates that are received from a BGP neighbor.

- Use the `route-policy-name` argument to define the name of the of route policy. The example shows that the route policy name is defined as pass-all.
- Use the `in` keyword to define the policy for inbound routes.

**Step 9**

**route-policy route-policy-name out**

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# route-policy pass-all out
```

Applies a routing policy to updates that are sent to a BGP neighbor.

- Use the `route-policy-name` argument to define the name of the of route policy. The example shows that the route policy name is defined as pass-all.
- Use the `out` keyword to define the policy for outbound routes.
Step 10  Use the **commit** or **end** command.

**commit** - Saves the configuration changes and remains within the configuration session.

**end** - Prompts user to take one of these actions:
- **Yes** - Saves configuration changes and exits the configuration session.
- **No** - Exits the configuration session without committing the configuration changes.
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.

---

**Configuring the Route Reflectors to Exchange VPN-IPv4 Routes**

Perform this task to enable the route reflectors to exchange VPN-IPv4 routes by using multihop. This task specifies that the next-hop information and the VPN label are to be preserved across the autonomous system.

**SUMMARY STEPS**

1. **configure**
2. **router bgp** *autonomous-system-number*
3. **neighbor** *ip-address*
4. **remote-as** *autonomous-system-number*
5. **ebgp-multihop** [*ttl-value*]
6. **update-source** *type* *interface-path-id*
7. **address-family** *vpnv4* *unicast*
8. **route-policy** *route-policy-name* *in*
9. **route-policy** *route-policy-name* *out*
10. **next-hop-unchanged**
11. Use the **commit** or **end** command.

**DETAILED STEPS**

**Step 1**

**configure**

**Example:**

```
RP/0/RSP0/CPU0:router# configure
```

Enters Global Configuration mode.

**Step 2**

**router bgp** *autonomous-system-number*

**Example:**

```
RP/0/RSP0/CPU0:router(config)# router bgp 120
RP/0/RSP0/CPU0:router(config-bgp)#
```

Enters Border Gateway Protocol (BGP) configuration mode allowing you to configure the BGP routing process.

**Step 3**

**neighbor** *ip-address*

**Example:**
RP/0/RSP0/CPU0:router(config-bgp)# neighbor 172.168.40.24
RP/0/RSP0/CPU0:router(config-bgp-nbr)#

Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address 172.168.40.24 as a BGP peer.

**Step 4**

`remote-as autonomous-system-number`

**Example:**

RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 2002

Creates a neighbor and assigns it a remote autonomous system number.

**Step 5**

`ebgp-multihop [ttl-value]`

**Example:**

RP/0/RSP0/CPU0:router(config-bgp-nbr)# ebgp-multihop

Enables multihop peerings with external BGP neighbors.

**Step 6**

`update-source type interface-path-id`

**Example:**

RP/0/RSP0/CPU0:router(config-bgp-nbr)# update-source loopback0

Allows BGP sessions to use the primary IP address from a particular interface as the local address.

**Step 7**

`address-family vpnv4 unicast`

**Example:**

RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family vpnv4 unicast
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)#

Configures VPNv4 address family.

**Step 8**

`route-policy route-policy-name in`

**Example:**

RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# route-policy pass-all in

Applies a routing policy to updates that are received from a BGP neighbor.

- Use the `route-policy-name` argument to define the name of the route policy. The example shows that the route policy name is defined as pass-all.
- Use the `in` keyword to define the policy for inbound routes.

**Step 9**

`route-policy route-policy-name out`

**Example:**
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# route-policy pass-all out

Applies a routing policy to updates that are sent to a BGP neighbor.

• Use the route-policy-name argument to define the name of the route policy. The example shows that the route policy name is defined as pass-all.
• Use the out keyword to define the policy for outbound routes.

Step 10  next-hop-unchanged

Example:

RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# next-hop-unchanged

Disables overwriting of the next hop before advertising to external Border Gateway Protocol (eBGP) peers.

Step 11  Use the commit or end command.

commit - Saves the configuration changes and remains within the configuration session.
end - Prompts user to take one of these actions:

• Yes - Saves configuration changes and exits the configuration session.
• No - Exits the configuration session without committing the configuration changes.
• Cancel - Remains in the configuration mode, without committing the configuration changes.

Configuring the Route Reflector to Reflect Remote Routes in its AS

Perform this task to enable the route reflector (RR) to reflect the IPv4 routes and labels learned by the autonomous system boundary router (ASBR) to the provider edge (PE) routers in the autonomous system. This task is accomplished by making the ASBR and PE route reflector clients of the RR.

SUMMARY STEPS

1. configure
2. router bgp autonomous-system-number
3. address-family ipv4 unicast
4. allocate-label all
5. neighbor ip-address
6. remote-as autonomous-system-number
7. update-source type interface-path-id
8. address-family ipv4 labeled-unicast
9. route-reflector-client
10. neighbor ip-address
11. remote-as autonomous-system-number
12. update-source type interface-path-id
13. address-family ipv4 labeled-unicast
14. route-reflector-client
15. Use the commit or end command.
## DETAILED STEPS

**Step 1** configure

**Example:**

```
RP/0/RSP0/CPU0:router# configure
```

Enters Global Configuration mode.

**Step 2** router bgp autonomous-system-number

**Example:**

```
RP/0/RSP0/CPU0:router(config)# router bgp 120
```

Enters Border Gateway Protocol (BGP) configuration mode allowing you to configure the BGP routing process.

**Step 3** address-family ipv4 unicast

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp)# address-family ipv4 unicast
RP/0/RSP0/CPU0:router(config-bgp-af)#
```

Enters global address family configuration mode for the IPv4 unicast address family.

**Step 4** allocate-label all

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-af)# allocate-label all
```

Allocates the MPLS labels for a specific IPv4 unicast or VPN routing and forwarding (VRF) IPv4 unicast routes so that the BGP router can send labels with BGP routes to a neighboring router that is configured for a labeled-unicast session.

**Step 5** neighbor ip-address

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-af)# neighbor 172.168.40.24
RP/0/RSP0/CPU0:router(config-bgp-nbr)#
```

Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address 172.168.40.24 as an ASBR eBGP peer.

**Step 6** remote-as autonomous-system-number

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 2002
```
Step 7  **update-source type interface-path-id**

Example:

```
RP/0/RSP0/CPU0:router(config-bgp-nbr)# update-source loopback0
```

Allows BGP sessions to use the primary IP address from a particular interface as the local address.

Step 8  **address-family ipv4 labeled-unicast**

Example:

```
RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family ipv4 labeled-unicast
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)#
```

Enters neighbor address family configuration mode for the IPv4 labeled-unicast address family.

Step 9  **route-reflector-client**

Example:

```
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# route-reflector-client
```

Configures the router as a BGP route reflector and neighbor 172.168.40.24 as its client.

Step 10  **neighbor ip-address**

Example:

```
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# neighbor 10.40.25.2
RP/0/RSP0/CPU0:router(config-bgp-nbr)#
```

Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address .40.25.2 as an VPNv4 iBGP peer.

Step 11  **remote-as autonomous-system-number**

Example:

```
RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 2002
```

Creates a neighbor and assigns it a remote autonomous system number.

Step 12  **update-source type interface-path-id**

Example:

```
RP/0/RSP0/CPU0:router(config-bgp-nbr)# update-source loopback0
```

Allows BGP sessions to use the primary IP address from a particular interface as the local address.

Step 13  **address-family ipv4 labeled-unicast**
Example:

Example:

Enters neighbor address family configuration mode for the IPv4 labeled-unicast address family.

Step 14  route-reflector-client

Example:

Configures the neighbor as a route reflector client.

Step 15  Use the commit or end command.

commit - Saves the configuration changes and remains within the configuration session.
end - Prompts user to take one of these actions:

• Yes - Saves configuration changes and exits the configuration session.
• No - Exits the configuration session without committing the configuration changes.
• Cancel - Remains in the configuration mode, without committing the configuration changes.

Providing VPN Connectivity Across Multiple Autonomous Systems with MPLS VPN Inter-AS with ASBRs Exchanging VPN-IPv4 Addresses

This section contains instructions for the following tasks:

Configuring the ASBRs to Exchange VPN-IPv4 Addresses for IP Tunnels

Perform this task to configure an external Border Gateway Protocol (eBGP) autonomous system boundary router (ASBR) to exchange VPN-IPv4 routes with another autonomous system.

SUMMARY STEPS

1. configure
2. router bgp autonomous-system-number
3. address-family { ipv4 tunnel }
4. address-family { vpnv4 unicast }
5. neighbor ip-address
6. remote-as autonomous-system-number
7. address-family { vpnv4 unicast }
8. route-policy route-policy-name { in }
9. route-policy route-policy-name { out }
10. neighbor ip-address
11. remote-as autonomous-system-number
12. update-source type interface-path-id
13. address-family { ipv4 tunnel }
14. address-family { vpnv4 unicast }
15. Use the commit or end command.

DETAILED STEPS

Step 1 configure
Example:
RP/0/RSP0/CPU0:router# configure

Enters the Global Configuration mode.

Step 2 router bgp autonomous-system-number
Example:
RP/0/RSP0/CPU0:router(config)# router bgp 120
RP/0/RSP0/CPU0:router(config-bgp)#

Enters Border Gateway Protocol (BGP) configuration mode allowing you to configure the BGP routing process.

Step 3 address-family { ipv4 tunnel }
Example:
RP/0/RSP0/CPU0:router(config-bgp)# address-family ipv4 tunnel
RP/0/RSP0/CPU0:router(config-bgp-af)#

Configures IPv4 tunnel address family.

Step 4 address-family { vpnv4 unicast }
Example:
RP/0/RSP0/CPU0:router(config-bgp-af)# address-family vpnv4 unicast

Configures VPNv4 address family.

Step 5 neighbor ip-address
Example:
RP/0/RSP0/CPU0:router(config-bgp-af)# neighbor 172.168.40.24
RP/0/RSP0/CPU0:router(config-bgp-nbr)#

Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address 172.168.40.24 as an ASBR eBGP peer.

Step 6 remote-as autonomous-system-number
Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 2002

Creates a neighbor and assigns it a remote autonomous system number.

Step 7 address-family { vpnv4 unicast }

Configuring the ASBRs to Exchange VPN-IPv4 Addresses for IP Tunnels
Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family vpnv4 unicast
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)#

Configures VPNv4 address family.

Step 8
route-policy route-policy-name { in }

Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# route-policy pass-all in

Applies a routing policy to updates that are received from a BGP neighbor.

- Use the route-policy-name argument to define the name of the route policy. The example shows that the route policy name is defined as pass-all.
- Use the in keyword to define the policy for inbound routes.

Step 9
route-policy route-policy-name { out }

Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# route-policy pass-all out

Applies a routing policy to updates that are sent from a BGP neighbor.

- Use the route-policy-name argument to define the name of the route policy. The example shows that the route policy name is defined as pass-all.
- Use the out keyword to define the policy for outbound routes.

Step 10
neighbor ip-address

Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# neighbor 175.40.25.2
RP/0/RSP0/CPU0:router(config-bgp-nbr)#

Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address 175.40.25.2 as a VPNv4 iBGP peer.

Step 11
remote-as autonomous-system-number

Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# remote-as 2002
RP/0/RSP0/CPU0:router(config-bgp-nbr)#

Creates a neighbor and assigns it a remote autonomous system number.

Step 12
update-source type interface-path-id

Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# update-source loopback0
RP/0/RSP0/CPU0:router(config-bgp-nbr)#

Allows BGP sessions to use the primary IP address from a particular interface as the local address.

Step 13
address-family { ipv4 tunnel }

Example:
Step 14

```
address-family { vpnv4 unicast }
```

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# address-family vpnv4 unicast
```

Configures VPNv4 address family.

Step 15

Use the `commit` or `end` command.

- **commit** - Saves the configuration changes and remains within the configuration session.
- **end** - Prompts user to take one of these actions:
  - **Yes** - Saves configuration changes and exits the configuration session.
  - **No** - Exits the configuration session without committing the configuration changes.
  - **Cancel** - Remains in the configuration mode, without committing the configuration changes.

### Configuring a Static Route to an ASBR Peer

Perform this task to configure a static route to an ASBR peer.

**SUMMARY STEPS**

1. `configure`
2. `router static`
3. `address-family ipv4 unicast`
4. `A.B.C.D/length next-hop`
5. Use the `commit` or `end` command.

**DETAILED STEPS**

Step 1

```
configure
```

**Example:**

```
RP/0/RSP0/CPU0:router# configure
```

Enters the Global Configuration mode.

Step 2

```
router static
```

**Example:**

```
RP/0/RSP0/CPU0:router(config)# router static
RP/0/RSP0/CPU0:router(config-static)#
```

Enters router static configuration mode.
**Step 3**  
address-family ipv4 unicast  
Example:  
```
RP/0/RSP0/CPU0:router(config-static)# address-family ipv4 unicast
RP/0/RSP0/CPU0:router(config-static-afi)#
```  
Enables an IPv4 address family.

**Step 4**  
A.B.C.D/length next-hop  
Example:  
```
RP/0/RSP0/CPU0:router(config-static-afi)# 10.10.10.10/32 10.9.9.9
```  
Enters the address of the destination router (including IPv4 subnet mask).

**Step 5**  
Use the **commit** or **end** command.  
- **commit** - Saves the configuration changes and remains within the configuration session.  
- **end** - Prompts user to take one of these actions:  
  
  • **Yes** - Saves configuration changes and exits the configuration session.  
  • **No** - Exits the configuration session without committing the configuration changes.  
  • **Cancel** - Remains in the configuration mode, without committing the configuration changes.

---

### Configuring EBGP Routing to Exchange VPN Routes Between Subautonomous Systems in a Confederation

Perform this task to configure external Border Gateway Protocol (eBGP) routing to exchange VPN routes between subautonomous systems in a confederation.

**Note**  
To ensure that host routes for VPN-IPv4 eBGP neighbors are propagated (by means of the Interior Gateway Protocol [IGP]) to other routers and PE routers, specify the **redistribute connected** command in the IGP configuration portion of the confederation eBGP (CEBGP) router. If you are using Open Shortest Path First (OSPF), make sure that the OSPF process is not enabled on the CEBGP interface in which the “redistribute connected” subnet exists.

### SUMMARY STEPS

1. configure  
2. router bgp autonomous-system-number  
3. bgp confederation peers peer autonomous-system-number  
4. bgp confederation identifier autonomous-system-number  
5. address-family vpnv4 unicast  
6. neighbor ip-address  
7. remote-as autonomous-system-number
8. address-family vpnv4 unicast
9. route-policy route-policy-name in
10. route-policy route-policy-name out
11. next-hop-self
12. Use the commit or end command.

DETAILED STEPS

Step 1 configure
Example:
RP/0/RSP0/CPU0:router# configure
Enters Global Configuration mode.

Step 2 router bgp autonomous-system-number
Example:
RP/0/RSP0/CPU0:router(config)# router bgp 120
RP/0/RSP0/CPU0:router(config-bgp)#
Enters BGP configuration mode allowing you to configure the BGP routing process.

Step 3 bgp confederation peers peer autonomous-system-number
Example:
RP/0/RSP0/CPU0:router(config-bgp)# bgp confederation peers 8
Configures the peer autonomous system number that belongs to the confederation.

Step 4 bgp confederation identifier autonomous-system-number
Example:
RP/0/RSP0/CPU0:router(config-bgp)# bgp confederation identifier 5
Specifies the autonomous system number for the confederation ID.

Step 5 address-family vpnv4 unicast
Example:
RP/0/RSP0/CPU0:router(config-bgp)# address-family vpnv4 unicast
RP/0/RSP0/CPU0:router(config-bgp-af)#
Configures VPNv4 address family.

Step 6 neighbor ip-address
Example:
Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address 10.168.40.24 as a BGP peer.

**Step 7**

remote-as autonomous-system-number

**Example:**

RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 2002

Creates a neighbor and assigns it a remote autonomous system number.

**Step 8**

address-family vpnv4 unicast

**Example:**

RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family vpnv4 unicast

Configures VPNv4 address family.

**Step 9**

route-policy route-policy-name in

**Example:**

RP/0/RSP0/CPU0:router(config-bgp-nbr)# route-policy In-Ipv4 in

Applies a routing policy to updates received from a BGP neighbor.

**Step 10**

route-policy route-policy-name out

**Example:**

RP/0/RSP0/CPU0:router(config-bgp-nbr)# route-policy Out-Ipv4 out

Applies a routing policy to updates advertised to a BGP neighbor.

**Step 11**

next-hop-self

**Example:**

RP/0/RSP0/CPU0:router(config-bgp-nbr)# next-hop-self

Disables next-hop calculation and let you insert your own address in the next-hop field of BGP updates.

**Step 12**

Use the **commit** or **end** command.

- **commit** - Saves the configuration changes and remains within the configuration session.
- **end** - Prompts user to take one of these actions:
  - **Yes** - Saves configuration changes and exits the configuration session.
- **No** - Exits the configuration session without committing the configuration changes.
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.

---

**Configuring MPLS Forwarding for ASBR Confederations**

Perform this task to configure MPLS forwarding for autonomous system boundary router (ASBR) confederations (in BGP) on a specified interface.

---

**Note**

This configuration adds the implicit NULL rewrite corresponding to the peer associated with the interface, which is required to prevent BGP from automatically installing rewrites by LDP (in multihop instances).

---

**SUMMARY STEPS**

1. `configure`
2. `router bgp as-number`
3. `mpls activate`
4. `interface type interface-path-id`
5. Use the **commit** or **end** command.

---

**DETAILED STEPS**

**Step 1**

`configure`

**Example:**

```
RP/0/RSP0/CPU0:router# configure
```

Enters Global Configuration mode.

**Step 2**

`router bgp as-number`

**Example:**

```
RP/0/RSP0/CPU0:router(config)# router bgp 120
RP/0/RSP0/CPU0:router(config-bgp)
```

Enters BGP configuration mode allowing you to configure the BGP routing process.

**Step 3**

`mpls activate`

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp)# mpls activate
RP/0/RSP0/CPU0:router(config-bgp-mpls)#
```

Enters BGP MPLS activate configuration mode.
Step 4  interface type interface-path-id  
Example: 

```plaintext
RP/0/RSP0/CPU0:router(config-bgp-mpls)# interface GigabitEthernet 0/3/0/0
```

Enables MPLS on the interface.

Step 5  Use the commit or end command.  

- **commit** - Saves the configuration changes and remains within the configuration session.  
- **end** - Prompts user to take one of these actions:  
  - **Yes** - Saves configuration changes and exits the configuration session.  
  - **No** - Exits the configuration session without committing the configuration changes.  
  - **Cancel** - Remains in the configuration mode, without committing the configuration changes.

---

**Configuring a Static Route to an ASBR Confederation Peer**

Perform this task to configure a static route to an Inter-AS confederation peer. For more detailed information, see “Configuring a Static Route to a Peer” section.

**SUMMARY STEPS**

1.  configure  
2.  router static  
3.  address-family ipv4 unicast  
4.  A.B.C.D/length next-hop  
5.  Use the commit or end command.

**DETAILED STEPS**

Step 1  configure  
Example: 

```plaintext
RP/0/RSP0/CPU0:router# configure
```

Enters Global Configuration mode.

Step 2  router static  
Example: 

```plaintext
RP/0/RSP0/CPU0:router(config)# router static
RP/0/RSP0/CPU0:router(config-static)#
```

Enters router static configuration mode.

Step 3  address-family ipv4 unicast
Example:

```
RP/0/RSP0/CPU0:router(config-static)# address-family ipv4 unicast
RP/0/RSP0/CPU0:router(config-static-afi)#
```

Enables an IPv4 address family.

**Step 4 A.B.C.D/length next-hop**

Example:

```
RP/0/RSP0/CPU0:router(config-static-afi)# 10.10.10.10/32 10.9.9.9
```

Enters the address of the destination router (including IPv4 subnet mask).

**Step 5** Use the **commit** or **end** command.

- **commit** - Saves the configuration changes and remains within the configuration session.
- **end** - Prompts user to take one of these actions:
  - **Yes** - Saves configuration changes and exits the configuration session.
  - **No** - Exits the configuration session without committing the configuration changes.
  - **Cancel** - Remains in the configuration mode, without committing the configuration changes.

---

**Configuring Carrier Supporting Carrier**

Perform the tasks in this section to configure Carrier Supporting Carrier (CSC):

**Identifying the Carrier Supporting Carrier Topology**

Before you configure the MPLS VPN CSC with BGP, you must identify both the backbone and customer carrier topology.

**Note**

You can connect multiple CSC-CE routers to the same PE, or you can connect a single CSC-CE router to multiple CSC-PEs using more than one CSC-CE interface to provide redundancy and multiple path support in a CSC topology.

**SUMMARY STEPS**

1. Identify the type of customer carrier, ISP, or MPLS VPN service provider.
2. Identify the CE routers.
3. Identify the customer carrier core router configuration.
4. Identify the customer carrier edge (CSC-CE) routers.
5. Identify the backbone carrier router configuration.
DETAILED STEPS

Step 1  Identify the type of customer carrier, ISP, or MPLS VPN service provider.
Sets up requirements for configuration of carrier supporting carrier network.

Step 2  Identify the CE routers.
Sets up requirements for configuration of CE to PE connections.

Step 3  Identify the customer carrier core router configuration.
Sets up requirements for configuration between core (P) routers and between P routers and edge routers (PE and CSC-CE routers).

Step 4  Identify the customer carrier edge (CSC-CE) routers.
Sets up requirements for configuration of CSC-CE to CSC-PE connections.

Step 5  Identify the backbone carrier router configuration.
Sets up requirements for configuration between CSC core routers and between CSC core routers and edge routers (CSC-CE and CSC-PE routers).

Configuring the Backbone Carrier Core

Configuring the backbone carrier core requires setting up connectivity and routing functions for the CSC core and the CSC-PE routers. To do so, you must complete the following high-level tasks:

- Verify IP connectivity in the CSC core.
- Verify LDP configuration in the CSC core.

This task is not applicable to CSC over IP tunnels.

- Configure VRFs for CSC-PE routers.
- Configure multiprotocol BGP for VPN connectivity in the backbone carrier.

Configuring the CSC-PE and CSC-CE Routers

Perform the following tasks to configure links between a CSC-PE router and the carrier CSC-CE router for an MPLS VPN CSC network that uses BGP to distribute routes and MPLS labels:

The following figure shows the configuration for the peering with directly connected interfaces between CSC-PE and CSC-CE routers. This configuration is used as the example in the tasks that follow.

Figure 6: Configuration for Peering with Directly Connected Interfaces Between CSC-PE and CSC-CE Routers

![Figure 6](image-url)
Configuring a Static Route to a Peer

Perform this task to configure a static route to an Inter-AS or CSC-CE peer.

When you configure an Inter-AS or CSC peer, BGP allocates a label for a /32 route to that peer and performs a NULL label rewrite. When forwarding a labeled packet to the peer, the router removes the top label from the label stack; however, in such an instance, BGP expects a /32 route to the peer. This task ensures that there is, in fact, a /32 route to the peer.

Please be aware of the following facts before performing this task:

- A /32 route is not required to establish BGP peering. A route using a shorter prefix length will also work.
- A shorter prefix length route is not associated with the allocated label; even though the BGP session comes up between the peers, without the static route, forwarding will not work.

**Note**

To configure a static route on a CSC-PE, you must configure the router under the VRF (as noted in the detailed steps).

**SUMMARY STEPS**

1. `configure`
2. `router static`
3. `address-family ipv4 unicast`
4. `A.B.C.D/length next-hop`
5. Use the `commit` or `end` command.

**DETAILED STEPS**

**Step 1** `configure`

*Example:*

```
RP/0/RSP0/CPU0:router# configure
```

Enters the Global Configuration mode.

**Step 2** `router static`

*Example:*

```
RP/0/RSP0/CPU0:router(config)# router static
```

Enters router static configuration mode.

**Step 3** `address-family ipv4 unicast`

*Example:*

```
RP/0/RSP0/CPU0:router(config-static)# address-family ipv4 unicast
```
Enables an IPv4 address family.

Note To configure a static route on a CSC-PE, you must first configure the VRF using the `vrf` command before `address-family`.

Step 4 A.B.C.D/length  `next-hop`

Example:

```
RP/0/RSP0/CPU0:router(config-static-afi)# 10.10.10.10/32 10.9.9.9
```

Enters the address of the destination router (including IPv4 subnet mask).

Step 5 Use the `commit` or `end` command.

`commit` - Saves the configuration changes and remains within the configuration session.

`end` - Prompts user to take one of these actions:

- **Yes** - Saves configuration changes and exits the configuration session.
- **No** - Exits the configuration session without committing the configuration changes.
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.

---

**Verifying the MPLS Layer 3 VPN Configuration**

Perform this task to verify the MPLS Layer 3 VPN configuration.

**SUMMARY STEPS**

1. `show running-config router bgp as-number vrf vrf-name`
2. `show running-config routes`
3. `show ospf vrf vrf-name database`
4. `show running-config router bgp as-number vrf vrf-name neighbor ip-address`
5. `show bgp vrf vrf-name summary`
6. `show bgp vrf vrf-name neighbors ip-address`
7. `show bgp vrf vrf-name`
8. `show route vrf vrf-name ip-address`
9. `show bgp vpn unicast summary`
10. `show running-config router isis`
11. `show running-config mpls`
12. `show isis adjacency`
13. `show mpls ldp forwarding`
14. `show bgp vpnv4 unicast` or `show bgp vrf vrf-name`
15. `show bgp vrf vrf-name imported-routes`
16. `show route vrf vrf-name ip-address`
17. `show cef vrf vrf-name ip-address`
18. `show cef vrf vrf-name ip-address location node-id`
19. `show bgp vrf vrf-name ip-address`
20. `show ospf vrf vrf-name database`

**DETAILED STEPS**

**Step 1**

`show running-config router bgp as-number vrf vrf-name`

**Example:**

```
RP/0/RSP0/CPU0:router# show running-config router bgp 3 vrf vrf_A
```

Displays the specified VPN routing and forwarding (VRF) content of the currently running configuration.

**Step 2**

`show running-config routes`

**Example:**

```
RP/0/RSP0/CPU0:router# show running-config routes
```

Displays the Open Shortest Path First (OSPF) routes table in the currently running configuration.

**Step 3**

`show ospf vrf vrf-name database`

**Example:**

```
RP/0/RSP0/CPU0:router# show ospf vrf vrf_A database
```

Displays lists of information related to the OSPF database for a specified VRF.

**Step 4**

`show running-config router bgp as-number vrf vrf-name neighbor ip-address`

**Example:**

```
RP/0/RSP0/CPU0:router# show running-config router bgp 3 vrf vrf_A neighbor 172.168.40.24
```

Displays the Border Gateway Protocol (BGP) VRF neighbor content of the currently running configuration.

**Step 5**

`show bgp vrf vrf-name summary`

**Example:**

```
RP/0/RSP0/CPU0:router# show bgp vrf vrf_A summary
```

Displays the status of the specified BGP VRF connections.

**Step 6**

`show bgp vrf vrf-name neighbors ip-address`

**Example:**

```
RP/0/RSP0/CPU0:router# show bgp vrf vrf_A neighbors 172.168.40.24
```

Displays information about BGP VRF connections to the specified neighbors.

**Step 7**

`show bgp vrf vrf-name`
Example:

RP/0/RSP0/CPU0:router# show bgp vrf vrf_A

Displays information about a specified BGP VRF.

Step 8  show route vrf vrf-name ip-address

Example:

RP/0/RSP0/CPU0:router# show route vrf vrf_A 10.0.0.0

Displays the current routes in the Routing Information Base (RIB) for a specified VRF.

Step 9  show bgp vpn unicast summary

Example:

RP/0/RSP0/CPU0:router# show bgp vpn unicast summary

Displays the status of all BGP VPN unicast connections.

Step 10  show running-config router isis

Example:

RP/0/RSP0/CPU0:router# show running-config router isis

Displays the Intermediate System-to-Intermediate System (IS-IS) content of the currently running configuration.

Step 11  show running-config mpls

Example:

RP/0/RSP0/CPU0:router# show running-config mpls

Displays the MPLS content of the currently running-configuration.

Step 12  show isis adjacency

Example:

RP/0/RSP0/CPU0:router# show isis adjacency

Displays IS-IS adjacency information.

Step 13  show mpls ldp forwarding

Example:

RP/0/RSP0/CPU0:router# show mpls ldp forwarding

Displays the Label Distribution Protocol (LDP) forwarding state installed in MPLS forwarding.
Step 14  show bgp vpnv4 unicast or show bgp vrf vrf-name

Example:

RP/0/RSP0/CPU0:router# show bgp vpnv4 unicast

Displays entries in the BGP routing table for VPNv4 or VPNv6 unicast addresses.

Step 15  show bgp vrf vrf-name imported-routes

Example:

RP/0/RSP0/CPU0:router# show bgp vrf vrf_A imported-routes

Displays BGP information for routes imported into specified VRF instances.

Step 16  show route vrf vrf-name ip-address

Example:

RP/0/RSP0/CPU0:router# show route vrf vrf_A 10.0.0.0

Displays the current specified VRF routes in the RIB.

Step 17  show cef vrf vrf-name ip-address

Example:

RP/0/RSP0/CPU0:router# show cef vrf vrf_A 10.0.0.1

Displays the IPv4 Cisco Express Forwarding (CEF) table for a specified VRF.

Step 18  show cef vrf vrf-name ip-address location node-id

Example:

RP/0/RSP0/CPU0:router# show cef vrf vrf_A 10.0.0.1 location 0/1/cpu0

Displays the IPv4 CEF table for a specified VRF and location.

Step 19  show bgp vrf vrf-name ip-address

Example:

RP/0/RSP0/CPU0:router# show bgp vrf vrf_A 10.0.0.0

Displays entries in the BGP routing table for VRF vrf_A.

Step 20  show ospf vrf vrf-name database

Example:

RP/0/RSP0/CPU0:router# show ospf vrf vrf_A database
Displays lists of information related to the OSPF database for a specified VRF.

Configuring L3VPN over GRE

Perform the following tasks to configure L3VPN over GRE:

Creating a GRE Tunnel between Provider Edge Routers

Perform this task to configure a GRE tunnel between provider edge routers.

SUMMARY STEPS

1. configure
2. interface tunnel-ip number
3. ipv4 address ipv4-address subnet-mask
4. ipv6 address ipv6-prefix/prefix-length
5. tunnel mode gre ipv4
6. tunnel source type path-id
7. tunnel destination ip-address
8. Use the commit or end command.

DETAILED STEPS

Step 1 configure
Example:

RP/0/RSP0/CPU0:router# configure

Enters the Global Configuration mode.

Step 2 interface tunnel-ip number
Example:

RP/0/RSP0/CPU0:router(config)# interface tunnel-ip 4000

Enters tunnel interface configuration mode.

• number is the number associated with the tunnel interface.

Step 3 ipv4 address ipv4-address subnet-mask
Example:

RP/0/RSP0/CPU0:router(config-if)# ipv4 address 10.1.1.1 255.255.255.0

Specifies the IPv4 address and subnet mask for the interface.
• ipv4-address specifies the IP address of the interface.
• subnet-mask specifies the subnet mask of the interface.

Step 4  ipv6 address ipv6-prefix/prefix-length
Example:
RP/0/RSP0/CPU0:router(config-if)# ipv6 address 100:1:1:1::1/64

Specifies an IPv6 network assigned to the interface.

Step 5  tunnel mode gre ipv4
Example:
RP/0/RSP0/CPU0:router(config-if)# tunnel mode gre ipv4

Sets the encapsulation mode of the tunnel interface to GRE.

Step 6  tunnel source type path-id
Example:
RP/0/RSP0/CPU0:router(config-if)# tunnel source TenGigE0/2/0/1

Specifies the source of the tunnel interface.

Step 7  tunnel destination ip-address
Example:
RP/0/RSP0/CPU0:router(config-if)# tunnel destination 145.12.5.2

Defines the tunnel destination.

Step 8  Use the commit or end command.
commit - Saves the configuration changes and remains within the configuration session.
end - Prompts user to take one of these actions:
  • Yes - Saves configuration changes and exits the configuration session.
  • No - Exits the configuration session without committing the configuration changes.
  • Cancel - Remains in the configuration mode, without committing the configuration changes.

---

**Configuring IGP between Provider Edge Routers**

Perform this task to configure IGP between provider edge routers.

**SUMMARY STEPS**

1. configure
2. `router ospf process-name`
3. `nsr`
4. `router-id { router-id }
5. `mpls ldp sync`
6. `dead-interval seconds`
7. `hello-interval seconds`
8. `area area-id`
9. `interface tunnel-ip number`
10. Use the `commit` or `end` command.

**DETAILED STEPS**

**Step 1**
```
configure
```

**Example:**
```
RP/0/RSP0/CPU0:router# configure
```

Enters the Global Configuration mode.

**Step 2**
```
router ospf process-name
```

**Example:**
```
RP/0/RSP0/CPU0:router(config)# router ospf 1
```

Enables OSPF routing for the specified routing process and places the router in router configuration mode.

**Step 3**
```
nsr
```

**Example:**
```
RP/0/RSP0/CPU0:router(config-ospf)# nsr
```

Activates BGP NSR.

**Step 4**
```
router-id { router-id }
```

**Example:**
```
RP/0/RSP0/CPU0:router(config-ospf)# router-id 1.1.1.1
```

Configures a router ID for the OSPF process.

*Note* We recommend using a stable IP address as the router ID.

**Step 5**
```
mpls ldp sync
```

**Example:**
```
RP/0/RSP0/CPU0:router(config-ospf)# mpls ldp sync
```

Enables MPLS LDP synchronization.

**Step 6**
```
dead-interval seconds
```

Implementing MPLS Layer 3 VPNs

Configuring IGP between Provider Edge Routers
Example:
RP/0/RSP0/CPU0:router(config-ospf)# dead-interval 60

Sets the time to wait for a hello packet from a neighbor before declaring the neighbor down.

**Step 7**
hello-interval *seconds*

Example:
RP/0/RSP0/CPU0:router(config-ospf)# hello-interval 15

Specifies the interval between hello packets that OSPF sends on the interface.

**Step 8**
area *area-id*

Example:
RP/0/RSP0/CPU0:router(config-ospf)# area 0

Enters area configuration mode and configures an area for the OSPF process.

**Step 9**
interface tunnel-ip *number*

Example:
RP/0/RSP0/CPU0:router(config-ospf)# interface tunnel-ip 4

Enters tunnel interface configuration mode.
  * number is the number associated with the tunnel interface.

**Step 10**
Use the **commit** or **end** command.

**commit** - Saves the configuration changes and remains within the configuration session.

**end** - Prompts user to take one of these actions:
  * **Yes** - Saves configuration changes and exits the configuration session.
  * **No** - Exits the configuration session without committing the configuration changes.
  * **Cancel** - Remains in the configuration mode, without committing the configuration changes.

---

**Configuring LDP/GRE on the Provider Edge Routers**

Perform this task to configure LDP/GRE on the provider edge routers.

**SUMMARY STEPS**

1. configure
2. mpls ldp
3. router-id { router-id }
4. discovery hello holdtime *seconds*
5. discovery hello interval *seconds*
6. nsr
7. graceful-restart
8. graceful-restart reconnect-timeout seconds
9. graceful-restart forwarding-state-holdtime seconds
10. holdtime seconds
11. neighbor ip-address
12. interface tunnel-ip number
13. Use the commit or end command.

DETAILED STEPS

Step 1 configure
Example:
RP/0/RSP0/CPU0:router# configure

Enters the Global Configuration mode.

Step 2 mpls ldp
Example:
RP/0/RSP0/CPU0:router(config)# mpls ldp

Enables MPLS LDP configuration mode.

Step 3 router-id { router-id }
Example:
RP/0/RSP0/CPU0:router(config-ldp)# router-id 1.1.1.1

Configures a router ID for the OSPF process.
Note We recommend using a stable IP address as the router ID.

Step 4 discovery hello holdtime seconds
Example:
RP/0/RSP0/CPU0:router(config-ldp)# discovery hello holdtime 40

Defines the period of time a discovered LDP neighbor is remembered without receipt of an LDP Hello message from the neighbor.
Note We recommend using a stable IP address as the router ID.

Step 5 discovery hello interval seconds
Example:
RP/0/RSP0/CPU0:router(config-ldp)# discovery hello holdtime 20

Defines the period of time between the sending of consecutive Hello messages.

Step 6 nsr
Example:
Activates BGP NSR.

**Step 7**

**graceful-restart**

**Example:**

```plaintext
RP/0/RSP0/CPU0:router(config-ldp)# graceful-restart
```

Enables graceful restart on the router.

**Step 8**

**graceful-restart reconnect-timeout** `seconds`

**Example:**

```plaintext
RP/0/RSP0/CPU0:router(config-ldp)# graceful-restart reconnect-timeout 180
```

Defines the time for which the neighbor should wait for a reconnection if the LDP session is lost.

**Step 9**

**graceful-restart forwarding-state-holdtime** `seconds`

**Example:**

```plaintext
RP/0/RSP0/CPU0:router(config-ldp)# graceful-restart forwarding-state-holdtime 300
```

Defines the time that the neighbor should retain the MPLS forwarding state during a recovery.

**Step 10**

**holdtime** `seconds`

**Example:**

```plaintext
RP/0/RSP0/CPU0:router(config-ldp)# holdtime 90
```

Configures the hold time for an interface.

**Step 11**

**neighbor** `ip-address`

**Example:**

```plaintext
RP/0/RSP0/CPU0:router(config-ldp)# neighbor 10.1.1.0
```

Defines a neighboring router.

**Step 12**

**interface tunnel-ip** `number`

**Example:**

```plaintext
RP/0/RSP0/CPU0:router(config-ldp)# interface tunnel-ip 4
```

Enters tunnel interface configuration mode.
- number is the number associated with the tunnel interface.

**Step 13**

Use the `commit` or `end` command.

- **commit** - Saves the configuration changes and remains within the configuration session.
- **end** - Prompts user to take one of these actions:
  - **Yes** - Saves configuration changes and exits the configuration session.
• No - Exits the configuration session without committing the configuration changes.
• Cancel - Remains in the configuration mode, without committing the configuration changes.

Configuring L3VPN

Perform this task to configure L3VPN.

SUMMARY STEPS

1. configure
2. vrf vrf-name
3. address-family { ipv4 | ipv6 } unicast
4. import route-target [ as-number:nn | ip-address:nn ]
5. export route-target [ as-number:nn | ip-address:nn ]
6. interface type interface-path-id
7. vrf vrf-name
8. ipv4 address ipv4-address subnet-mask
9. dot1q native vlan vlan-id
10. router bgp as-number
11. nsr
12. bgp router-id ip-address
13. address-family { vpnv4 | vpnv6 } unicast
14. neighbor ip-address
15. remote-as as-number
16. update-source type interface-path-id
17. address-family { vpnv4 | vpnv6 } unicast
18. route-policy route-policy-name in
19. route-policy route-policy-name out
20. vrf vrf-name
21. rd { as-number:nn | ip-address:nn | auto }
22. address-family { ipv4 | ipv6 } unicast
23. redistribute connected [ metric metric-value ] [ route-policy route-policy-name ]
24. redistribute static [ metric metric-value ] [ route-policy route-policy-name ]
25. neighbor ip-address
26. remote-as as-number
27. ebg-multihop ttl-value
28. address-family { ipv4 | ipv6 } unicast
29. route-policy route-policy-name in
30. route-policy route-policy-name out
31. Use the commit or end command.

DETAILED STEPS

Step 1 configure
Example:
RP/0/RSP0/CPU0:router# configure

Enters the Global Configuration mode.

Step 2  
**vrf vrf-name**

Example:
RP/0/RSP0/CPU0:router(config)# vrf vpn1

Configures a VRF instance.

Step 3  
**address-family { ipv4 | ipv6 } unicast**

Example:
RP/0/RSP0/CPU0:router(config-vrf)# address-family { ipv4 | ipv6 } unicast

Specifies either the IPv4 or IPv6 address family and enters address family configuration submode.

Step 4  
**import route-target [ as-number:nn | ip-address:nn ]**

Example:
RP/0/RSP0/CPU0:router(config-vrf)# import route-target 2:1

Specifies a list of route target (RT) extended communities. Only prefixes that are associated with the specified import route target extended communities are imported into the VRF.

Step 5  
**export route-target [ as-number:nn | ip-address:nn ]**

Example:
RP/0/RSP0/CPU0:router(config-vrf)# export route-target 1:1

Specifies a list of route target extended communities. Export route target communities are associated with prefixes when they are advertised to remote PEs. The remote PEs import them into VRFs which have import RTs that match these exported route target communities.

Step 6  
**interface type interface-path-id**

Example:
RP/0/RSP0/CPU0:router(config)# interface TenGigE0/2/0/0.1

Enters interface configuration mode and configures an interface.

Step 7  
**vrf vrf-name**

Example:
RP/0/RSP0/CPU0:router(config-if)# vrf vpn1

Configures a VRF instance.

Step 8  
**ipv4 address ipv4-address subnet-mask**

Example:
Step 9  
\texttt{dot1q native vlan \textit{vlan-id}}

\textbf{Example:}
\begin{verbatim}
RP/0/RSP0/CPU0:router(config-if)# dot1q native vlan 1
\end{verbatim}

Assigns the native VLAN ID of a physical interface trunking 802.1Q VLAN traffic.

Step 10  
\texttt{router bgp as-number}

\textbf{Example:}
\begin{verbatim}
RP/0/RSP0/CPU0:router(config)# router bgp 1
\end{verbatim}

Specifies the autonomous system number and enters the BGP configuration mode, allowing you to configure the BGP routing process.

Step 11  
\texttt{nsr}

\textbf{Example:}
\begin{verbatim}
RP/0/RSP0/CPU0:router(config-bgp)# nsr
\end{verbatim}

Activates BGP NSR.

Step 12  
\texttt{bgp router-id \textit{ip-address}}

\textbf{Example:}
\begin{verbatim}
RP/0/RSP0/CPU0:router(config-bgp)# bgp router-id 1.1.1.1
\end{verbatim}

Configures the local router with a specified router ID.

Step 13  
\texttt{address-family \{vpng4 | vpng6\} unicast}

\textbf{Example:}
\begin{verbatim}
RP/0/RSP0/CPU0:router(config-bgp)# address-family vpn4 unicast
\end{verbatim}

Enters address family configuration submode for the specified address family.

Step 14  
\texttt{neighbor \textit{ip-address}}

\textbf{Example:}
\begin{verbatim}
RP/0/RSP0/CPU0:router(config-bgp)# neighbor 1.1.1.1
\end{verbatim}

Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address as a BGP peer.
Step 15  
**remote-as as-number**  
*Example:*  
RP/0/RSP0/CPU0:router(config-bgp-nbr)#remote-as 1

Creates a neighbor and assigns a remote autonomous system number to it.

Step 16  
**update-source** *type interface-path-id*  
*Example:*  
RP/0/RSP0/CPU0:router(config-bgp-nbr)#update-source Loopback0

Allows sessions to use the primary IP address from a specific interface as the local address when forming a session with a neighbor.

Step 17  
**address-family { vpnv4 | vpnv6 } unicast**  
*Example:*  
RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family vpnv4 unicast

Enters address family configuration submode for the specified address family.

Step 18  
**route-policy route-policy-name in**  
*Example:*  
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)#route-policy pass-all in

Defines a route policy and enters route policy configuration mode.

Step 19  
**route-policy route-policy-name out**  
*Example:*  
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)#route-policy pass-all out

Defines a route policy and enters route policy configuration mode.

Step 20  
**vrf vrf-name**  
*Example:*  
RP/0/RSP0/CPU0:router(config)# vrf vpn1

Configures a VRF instance.

Step 21  
**rd { as-number:nn | ip-address:nn | auto }**  
*Example:*  
RP/0/RSP0/CPU0:router(config-vrf)#rd 1:1

Configures the route distinguisher.

Step 22  
**address-family { ipv4 | ipv6 } unicast**  
*Example:*  


Step 23
redistribute connected [ metric metric-value ] [ route-policy route-policy-name ]

Example:
RP/0/RSP0/CPU0:router(config-vrf-af)# redistribute connected

Configures the local router with a specified router ID.

Step 24
redistribute static [ metric metric-value ] [ route-policy route-policy-name ]

Example:
RP/0/RSP0/CPU0:router(config-vrf-af)# redistribute static

Causes routes from the specified instance to be redistributed into BGP.

Step 25
neighbor ip-address

Example:
RP/0/RSP0/CPU0:router(config-bgp)# neighbor 150.1.1.2

Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address as a BGP peer.

Step 26
remote-as as-number

Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr)#remote-as 7501

Creates a neighbor and assigns a remote autonomous system number to it.

Step 27
ebg-multihop ttl-value

Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr)#ebgp-multihop 10

Configures the CE neighbor to accept and attempt BGP connections to external peers residing on networks that are not directly connected.

Step 28
address-family { ipv4 | ipv6 } unicast

Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family ipv4 unicast

Specifies either the IPv4 or IPv6 address family and enters address family configuration submode.

Step 29
route-policy route-policy-name in

Example:
Step 30
route-policy route-policy-name out

Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)#route-policy BGP_pass_all out

Step 31
Use the commit or end command.

commit - Saves the configuration changes and remains within the configuration session.
end - Prompts user to take one of these actions:
  • Yes - Saves configuration changes and exits the configuration session.
  • No - Exits the configuration session without committing the configuration changes.
  • Cancel - Remains in the configuration mode, without committing the configuration changes.

Configuration Examples for Implementing MPLS Layer 3 VPNs
The following section provides sample configurations for MPLS L3VPN features:

Configuring an MPLS VPN Using BGP: Example
The following example shows the configuration for an MPLS VPN using BGP on “vrf vpn1”:

```
address-family ipv4 unicast
  import route-target
  100:1
  !
  export route-target
  100:1
  !
  !
route-policy pass-all
  pass
end-policy
!
interface Loopback0
  ipv4 address 10.0.0.1 255.255.255.255
!
interface TenGigE 0/1/0/0
  vrf vpn1
  ipv4 address 10.0.0.2 255.0.0.0
!
interface TenGigE 0/1/0/1
  ipv4 address 10.0.0.1 255.0.0.0
!
router ospf 100
```
area 100
  interface loopback0
  interface TenGigE 0/1/0/1
!
router bgp 100
  address-family vpnv4 unicast
  retain route-target route-policy policy1
  neighbor 10.0.0.3
    remote-as 100
    update-source Loopback0
    address-family vpnv4 unicast
  !
vrf vpn1
  rd 100:1
  address-family ipv4 unicast
    redistribute connected
  !
  neighbor 10.0.0.1
    remote-as 200
    address-family ipv4 unicast
    as-override
    route-policy pass-all in
    route-policy pass-all out
    !
    advertisement-interval 5
  !
  !
mpls ldp
  route-id loopback0
  interface TenGigE 0/1/0/1
!

Configuring the Routing Information Protocol on the PE Router: Example

The following example shows the configuration for the RIP on the PE router:

vrf vpn1
  address-family ipv4 unicast
    import route-target
    100:1
  !
    export route-target
    100:1
  !
  !
  route-policy pass-all
    pass
  end-policy
  !
  interface TenGigE 0/1/0/0
    vrf vpn1
    ipv4 address 10.0.0.2 255.0.0.0
  !
  router rip
  vrf vpn1
  interface TenGigE 0/1/0/0
    !
    timers basic 30 90 90 120
Configuring the PE Router Using EIGRP: Example

The following example shows the configuration for the Enhanced Interior Gateway Routing Protocol (EIGRP) on the PE router:

```conf
Router eigrp 10
  vrf VRF1
    address-family ipv4
      router-id 10.1.1.2
      default-metric 100000 2000 255 1 1500
      as 62
      redistribute bgp 2000
        interface Loopback0
      !
      interface TenGigE 0/6/0/0
```

Configuration Examples for MPLS VPN CSC

Configuration examples for the MPLS VPN CSC include:

Configuring the Backbone Carrier Core: Examples

Configuration examples for the backbone carrier core included in this section are as follows:

Configuring VRFs for CSC-PE Routers: Example

The following example shows how to configure a VPN routing and forwarding instance (VRF) for a CSC-PE router:

```conf
config
  vrf vpn1
    address-family ipv4 unicast
    import route-target 100:1
    export route-target 100:1
end
```

Configuring the Links Between CSC-PE and CSC-CE Routers: Examples

This section contains the following examples:

Configuring a CSC-PE: Example

In this example, a CSC-PE router peers with a PE router, 10.1.0.2, in its own AS. It also has a labeled unicast peering with a CSC-CE router, 10.0.0.1.

```conf
config
  router bgp 2
    address-family vpnv4 unicast
    neighbor 10.1.0.2
      remote-as 2
      update-source loopback0
    address-family vpnv4 unicast
```
vrf customer-carrier
rd 1:100
address-family ipv4 unicast
   allocate-label all
   redistribute static
neighbor 10.0.0.1
remote-as 1
address-family ipv4 labeled-unicast
   route-policy pass-all in
   route-policy pass-all out
   as-override
end

**Configuring a CSC-CE: Example**

The following example shows how to configure a CSC-CE router. In this example, the CSC-CE router peers CSC-PE router 10.0.0.2 in AS 2.

```plaintext
config
router bgp 1
   address-family ipv4 unicast
      redistribute ospf 200
      allocate-label all
      neighbor 10.0.0.2
      remote-as 2
      address-family ipv4 labeled-unicast
      route-policy pass-all in
      route-policy pass-all out
end
```

**Configuring a Static Route to a Peer: Example**

The following example shows how to configure a static route to an Inter-AS or CSC-CE peer:

```plaintext
config
router static
   address-family ipv4 unicast
      10.0.0.2/32 40.1.1.1
end
```

**Configuring a Static Route to a Peer: Example**

This example shows how to configure a static route to an Inter-AS or CSC-CE peer:

```plaintext
config
router static
   address-family ipv4 unicast
      10.0.0.2/32 40.1.1.1
end
```

**Configuring L3VPN over GRE: Example**

The following example shows how to configure L3VPN over GRE:

Sample configuration to create a GRE tunnel between PE1 and PE2:
Configure IGP between PE1 and PE2:

Sample configuration for PE1 is given below. PE2 will also have a similar configuration.

```
RP/0/RSP0/CPU0:PE1#sh run router ospf 1
router ospf 1
  nsr
  router-id 1.1.1.1 <== Loopback0
  mpls ldp sync
  mtu-ignore enable
  dead-interval 60
  hello-interval 15
  area 0
  interface TenGigE0/2/0/1

RP/0/RSP0/CPU0:PE1#sh run router ospf 0
router ospf 0
  nsr
  router-id 1.1.1.1
  mpls ldp sync
  dead-interval 60
  hello-interval 15
  area 0
  interface Loopback0
  interface tunnel-ip1

* Check for OSPF neighbors
RP/0/RSP0/CPU0:PE1#sh ospf neighbor
Neighbors for OSPF 0

<table>
<thead>
<tr>
<th>Neighbor ID</th>
<th>Pri</th>
<th>State</th>
<th>Dead Time</th>
<th>Address</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.4.4.4</td>
<td>1</td>
<td>FULL/</td>
<td>00:00:47</td>
<td>100.1.1.2</td>
<td>tunnel-ip1</td>
</tr>
<tr>
<td>Neighbor PE2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| Neighbor is up for 00:13:40

Neighbors for OSPF 1

<table>
<thead>
<tr>
<th>Neighbor ID</th>
<th>Pri</th>
<th>State</th>
<th>Dead Time</th>
<th>Address</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2.2.2</td>
<td>1</td>
<td>FULL/DR</td>
<td>00:00:50</td>
<td>145.12.1.2</td>
<td>TenGigE0/2/0/1</td>
</tr>
<tr>
<td>Neighbor P1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| Neighbor is up for 00:13:43
```
Configure LDP/GRE on PE1 and PE2:

```
RP/0/RSP0/CPU0:PE1#sh run mpls ldp
mpls ldp
  router-id 1.1.1.1 <=== Loopback0
discovery hello holdtime 45
discovery hello interval 15
nsr
graceful-restart
graceful-restart reconnect-timeout 180
graceful-restart forwarding-state-holdtime 300
holdtime 90
log
neighbor
! interface tunnel-ip1
!
```

*Check for mpls forwarding

```
RP/0/RSP0/CPU0:PE1#sh mpls forwarding prefix 4.4.4.4/32
Local Outgoing Prefix Outgoing Next Hop Bytes
Label Label or ID Interface Switched
---- ------- -------------- -------- ----------- ----------
16003 Pop 4.4.4.4/32 ti1 100.4.1.2 0
```

Configure L3VPN

```
RP/0/RSP0/CPU0:PE1#sh run vrf vpn1
vrf vpn1
  address-family ipv4 unicast
  import route-target
    2:1
  !
  export route-target
    1:1
  !
RP/0/RSP0/CPU0:PE1#sh run int tenGigE 0/2/0/0.1
interface TenGigE0/2/0/0.1
  vrf vpn1
  ipv4 address 150.1.1.1 255.255.255.0
  encapsulation dot1q 1
  !
RP/0/RSP0/CPU0:PE1#sh run router bgp
router bgp 1
  nsr
  bgp router-id 1.1.1.1 <===Loopback0
  address-family vpnv4 unicast
  !
  neighbor 4.4.4.4 <===iBGP session with PE2
    remote-as 1
    update-source Loopback0
    address-family vpnv4 unicast
    route-policy pass-all in
    route-policy pass-all out
  !
```
vrf vpn1
rd 1:1
address-family ipv4 unicast
redistribute connected
redistribute static

neighbor 150.1.1.2
remote-as 7501
ebgp-multihop 10
address-family ipv4 unicast
route-policy BGP_pass_all in
route-policy BGP_pass_all out

* Check vrf ping to the 150.1.1.2.

RP/0/RSP0/CPU0:PE1#ping vrf vpn1 150.1.1.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 150.1.1.2, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/3 ms

* Send traffic to vrf routesadvertised and verify that mpls counters increase in tunnel interface accounting

RP/0/RSP0/CPU0:PE1#sh int tunnel-ip1 accounting
tunnel-ip1
<table>
<thead>
<tr>
<th>Protocol</th>
<th>Pkts In</th>
<th>Chars In</th>
<th>Pkts Out</th>
<th>Chars Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPV4_MULTICAST</td>
<td>3</td>
<td>276</td>
<td>3</td>
<td>276</td>
</tr>
<tr>
<td>MPLS</td>
<td>697747</td>
<td>48842290</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
CHAPTER 3

Implementing IPv6 VPN Provider Edge Transport over MPLS

IPv6 Provider Edge or IPv6 VPN Provider Edge (6PE/VPE) uses the existing MPLS IPv4 core infrastructure for IPv6 transport. 6PE/VPE enables IPv6 sites to communicate with each other over an MPLS IPv4 core network using MPLS label switched paths (LSPs).

This feature relies heavily on multiprotocol Border Gateway Protocol (BGP) extensions in the IPv4 network configuration on the provider edge (PE) router to exchange IPv6 reachability information (in addition to an MPLS label) for each IPv6 address prefix. Edge routers are configured as dual-stack, running both IPv4 and IPv6, and use the IPv4 mapped IPv6 address for IPv6 prefix reachability exchange.

For detailed information about the commands used to configure 6PE/VPE, see the VPN and Ethernet Services Command Reference for Cisco ASR 9000 Series Routers.

Feature History for Implementing 6PE/VPE Transport over MPLS

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release 3.9.1</td>
<td>This feature was introduced.</td>
</tr>
<tr>
<td>Release 4.0.0</td>
<td>Support was added for the 6PE and 6VPE features for IPv6 L3VPN on A9K-SIP-700.</td>
</tr>
<tr>
<td></td>
<td>Support was added for the BGP per VRF/CE label allocation for 6PE feature.</td>
</tr>
<tr>
<td>Release 4.1.0</td>
<td>Support for the Open Shortest Path First version 3 (OSPFv3) IPv6 VPN Provider Edge (6VPE) feature was added.</td>
</tr>
</tbody>
</table>

- Prerequisites for Implementing 6PE/VPE, on page 81
- Information About 6PE/VPE, on page 82
- How to Implement 6PE/VPE, on page 85
- Configuration Examples for 6PE/VPE, on page 93

Prerequisites for Implementing 6PE/VPE

The following prerequisites are required to implement 6PE/VPE:
• You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command.

If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

• Familiarity with MPLS and BGP4 configuration and troubleshooting.

Information About 6PE/VPE

To configure the 6PE/VPE feature, you should understand the concepts that are described in these sections:

Overview of 6PE/VPE

Multiple techniques are available to integrate IPv6 services over service provider core backbones:

• Dedicated IPv6 network running over various data link layers

• Dual-stack IPv4-IPv6 backbone

• Existing MPLS backbone leverage

These solutions are deployed on service providers’ backbones when the amount of IPv6 traffic and the revenue generated are in line with the necessary investments and the agreed-upon risks. Conditions are favorable for the introduction of native IPv6 services, from the edge, in a scalable way, without any IPv6 addressing restrictions and without putting a well-controlled IPv4 backbone in jeopardy. Backbone stability is essential for service providers that have recently stabilized their IPv4 infrastructure.

Service providers running an MPLS/IPv4 infrastructure follow similar trends because several integration scenarios that offer IPv6 services on an MPLS network are possible. Cisco Systems has specially developed Cisco 6PE or IPv6 Provider Edge Router over MPLS, to meet all those requirements.

Inter-AS support for 6PE requires support of Border Gateway Protocol (BGP) to enable the address families and to allocate and distribute PE and ASBR labels.

Note

Cisco IOS XR displays actual IPv4 next-hop addresses for IPv6 labeled-unicast and VPNv6 prefixes. IPv4-mapped-to-Ipv6 format is not supported.

Benefits of 6PE/VPE

Service providers who currently deploy MPLS experience these benefits of Cisco 6PE/VPE:

• Minimal operational cost and risk—No impact on existing IPv4 and MPLS services.

• Provider edge routers upgrade only—A 6PE/VPE router can be an existing PE router or a new one dedicated to IPv6 traffic.

• No impact on IPv6 customer edge routers—The ISP can connect to any customer CE running Static, IGP or EGP.

• Production services ready—An ISP can delegate IPv6 prefixes.
IPv6 on the Provider Edge and Customer Edge Routers

Service Provider Edge Routers

6PE is particularly applicable to service providers who currently run an MPLS network. One of its advantages is that there is no need to upgrade the hardware, software, or configuration of the core network, and it eliminates the impact on the operations and the revenues generated by the existing IPv4 traffic. MPLS is used by many service providers to deliver services to customers. MPLS as a multiservice infrastructure technology is able to provide layer 3 VPN, QoS, traffic engineering, fast re-routing and integration of ATM and IP switching.

Customer Edge Routers

Using tunnels on the CE routers is the simplest way to deploy IPv6 over MPLS networks. It has no impact on the operation or infrastructure of MPLS and requires no changes to the P routers in the core or to the PE routers. However, tunnel meshing is required as the number of CEs to connect increases, and it is difficult to delegate a global IPv6 prefix for an ISP.

The following figure illustrates the network architecture using tunnels on the CE routers.

Figure 7: IPv6 Using Tunnels on the CE Routers

IPv6 Provider Edge Multipath

Internal and external BGP multipath for IPv6 allows the IPv6 router to load balance between several paths (for example, same neighboring autonomous system (AS) or sub-AS, or the same metric) to reach its destination. The 6PE multipath feature uses multiprotocol internal BGP (MP-IBGP) to distribute IPv6 routes over the MPLS IPv4 core network and to attach an MPLS label to each route.
When MP-IBGP multipath is enabled on the 6PE router, all labeled paths are installed in the forwarding table with MPLS information (label stack) when MPLS information is available. This functionality enables 6PE to perform load balancing.

**OSPFv3 6VPE**

The Open Shortest Path First version 3 (OSPFv3) IPv6 VPN Provider Edge (6VPE) feature adds VPN routing and forwarding (VRF) and provider edge-to-customer edge (PE-CE) routing support to Cisco IOS XR OSPFv3 implementation. This feature allows:

- Multiple VRF support per OSPFv3 routing process
- OSPFv3 PE-CE extensions

**Multiple VRF Support**

OSPFv3 supports multiple VRFs in a single routing process that allows scaling to tens and hundreds of VRFs without consuming too much route processor (RP) resources.

Multiple OSPFv3 processes can be configured on a single router. In large-scale VRF deployments, this allows partition VRF processing across multiple RPs. It is also used to isolate default routing table or high impact VRFs from the regular VRFs. It is recommended to use a single process for all the VRFs. If needed, a second OSPFv3 process must be configured for IPv6 routing.

**Note**
The maximum of four OSPFv3 processes are supported.

**OSPFv3 PE-CE Extensions**

IPv6 protocol is being vastly deployed in today’s customer networks. Service Providers (SPs) need to be able to offer Virtual Private Network (VPN) services to their customers for supporting IPv6 protocol, in addition to the already offered VPN services for IPv4 protocol.

In order to support IPv6, routing protocols require additional extensions for operating in the VPN environment. Extensions to OSPFv3 are required in order for OSPFv3 to operate at the PE-CE links.

**VRF Lite**

VRF lite feature enables VRF deployment without BGP or MPLS based backbone. In VRF lite, the PE routers are directly connected using VRF interfaces. For OSPFv3, the following needs to operate differently in the VRF lite scenario, as opposed to the deployment with BGP or MPLS backbone:

- DN bit processing—In VRF lite environment, the DN bit processing is disabled.
- ABR status—In VRF context (except default VRF), OSPFv3 router is automatically set as an ABR, regardless to it’s connectivity to area 0. This automatic ABR status setting is disabled in the VRF lite environment.

**Note**
To enable VRF Lite, issue the `capability vrf-lite` command in the OSPFv3 VRF configuration submode.
How to Implement 6PE/VPE

This section includes these implementation procedures:

Configuring 6PE/VPE

This task describes how to configure 6PE/VPE on PE routers to transport the IPv6 prefixes across the IPv4 cloud.

Ensure that you configure 6PE/VPE on PE routers participating in both the IPv4 cloud and IPv6 clouds.

Note

For 6PE, you can use all routing protocols supported on Cisco IOS XR software such as BGP, OSPF, IS-IS, EIGRP, RIP, and Static to learn routes from both clouds. However, for 6VPE, you can use only the BGP, EIGRP and Static routing protocols to learn routes. Also, 6VPE supports OSPFv3 routing protocol between PE and CE routers.

SUMMARY STEPS

1. configure
2. router bgp as-number
3. neighbor ip-address
4. remote-as as-number
5. address-family ipv6 labeled-unicast
6. exit
7. exit
8. address-family ipv6 unicast
9. allocate-label [all | route-policy policy_name]
10. Use the commit or end command.

DETAILED STEPS

Step 1 configure

Example:

RP/0/RSP0/CPU0:router# configure

Enters the Global Configuration mode.

Step 2 router bgp as-number

Example:

RP/0/RSP0/CPU0:router(config)# router bgp 1

Enters the number that identifies the autonomous system (AS) in which the router resides.
Range for 2-byte numbers is 1 to 65535. Range for 4-byte numbers is 1.0 to 65535.65535.

**Step 3**

`neighbor ip-address`

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp)# neighbor 1.1.1.1
```

Enters neighbor configuration mode for configuring Border Gateway Protocol (BGP) routing sessions.

**Step 4**

`remote-as as-number`

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 100
```

Creates a neighbor and assigns a remote autonomous system number to it.

**Step 5**

`address-family ipv6 labeled-unicast`

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family ipv6 labeled-unicast
```

Specifies IPv6 labeled-unicast address prefixes.

**Note** This option is also available in IPv6 neighbor configuration mode and VRF neighbor configuration mode.

**Step 6**

`exit`

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# exit
```

Exits BGP address-family submode.

**Step 7**

`exit`

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-nbr)# exit
```

Exits BGP neighbor submode.

**Step 8**

`address-family ipv6 unicast`

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp)# address-family ipv6 unicast
```

Specifies IPv6 unicast address prefixes.

**Step 9**

`allocate-label [all | route-policy policy_name]`

**Example:**
RP/0/RSP0/CPU0:router(config-bgp-af)# allocate-label all

Allocates MPLS labels for specified IPv4 unicast routes.

**Note** The route-policy keyword provides finer control to filter out certain routes from being advertised to the neighbor.

**Step 10**
Use the commit or end command.

**commit** - Saves the configuration changes and remains within the configuration session.

**end** - Prompts user to take one of these actions:

- **Yes** - Saves configuration changes and exits the configuration session.
- **No** - Exits the configuration session without committing the configuration changes.
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.

---

**Configuring PE to PE Core**

This task describes how to configure a Provider Edge (PE) to PE Core.

For information on configuring VPN Routing and Forwarding (VRF), refer to the Implementing BGP module of the Routing Configuration Guide for Cisco ASR 9000 Series Routers.

**SUMMARY STEPS**

1. configure
2. router bgp
3. address-family vpnv6 unicast
4. bgp dampening [ half-life [ reuse suppress max-suppress-time ] | route-policy route-policy-name ]
5. bgp client-to-client reflection { cluster-id | disable }
6. neighbor ip-address
7. remote-as as-number
8. description text
9. password { clear | encrypted } password
10. shutdown
11. timers keepalive hold-time
12. update-source type interface-id
13. address-family vpnv6 unicast
14. route-policy route-policy-name { in | out }
15. exit
16. vrf vrf-name
17. rd { as-number : nn | ip-address : nn | auto }
18. Use the commit or end command.
**DETAILED STEPS**

**Step 1**  
**configure**  
**Example:**  
RP/0/RSP0/CPU0:router# configure

Enters the Global Configuration mode.

**Step 2**  
**router bgp**  
**Example:**  
RP/0/RSP0/CPU0:router(config)# router bgp 10

Specifies the BGP AS number and enters the BGP configuration mode, allowing you to configure the BGP routing process.

**Step 3**  
**address-family vpnv6 unicast**  
**Example:**  
RP/0/RSP0/CPU0:router(config-bgp)# address-family vpnv6 unicast

Specifies the vpnv6 address family and enters address family configuration submode.

**Step 4**  
**bgp dampening [ half-life [ reuse suppress max-suppress-time ] | route-policy route-policy-name ]**  
**Example:**  
RP/0/RSP0/CPU0:router(config-bgp-af)# bgp dampening 30 1500 10000 120

Configures BGP dampening for the specified address family.

**Step 5**  
**bgp client-to-client reflection { cluster-id | disable }**  
**Example:**  
RP/0/RSP0/CPU0:router(config-bgp-af)# bgp client-to-client reflection disable

Configures client to client route reflection.

**Step 6**  
**neighbor ip-address**  
**Example:**  
RP/0/RSP0/CPU0:router(config-bgp)# neighbor 10.1.1.1

Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address as a BGP peer.

**Step 7**  
**remote-as as-number**
**Example:**

RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 100

Creates a neighbor and assigns a remote autonomous system number to it.

**Step 8**  
**description text**

**Example:**

RP/0/RSP0/CPU0:router(config-bgp-nbr)# description neighbor 172.16.1.1

Provides a description of the neighbor. The description is used to save comments and does not affect software function.

**Step 9**  
**password { clear | encrypted } password**

**Example:**

RP/0/RSP0/CPU0:router(config-bgp-nbr)# password encrypted 123abc

Enables Message Digest 5 (MD5) authentication on the TCP connection between the two BGP neighbors.

**Step 10**  
**shutdown**

**Example:**

RP/0/RSP0/CPU0:router(config-bgp-nbr)# router bgp 1

Terminates any active sessions for the specified neighbor and removes all associated routing information.

**Step 11**  
**timers keepalive hold-time**

**Example:**

RP/0/RSP0/CPU0:router(config-bgp-nbr)# timers 12000 200

Set the timers for the BGP neighbor.

**Step 12**  
**update-source type interface-id**

**Example:**

RP/0/RSP0/CPU0:router(config-bgp-nbr)# update-source TenGigE 0/1/5/0

Allows iBGP sessions to use the primary IP address from a specific interface as the local address when forming an iBGP session with a neighbor.

**Step 13**  
**address-family vpnv6 unicast**

**Example:**

RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family vpnv6 unicast

Enters VPN neighbor address family configuration mode.
Step 14  route-policy route-policy-name { in | out }

Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# route-policy pe-pe-vpn-out out

Specifies a routing policy for an outbound route. The policy can be used to filter routes or modify route attributes.

Step 15  exit

Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# exit

Exits address family configuration and neighbor submode.

Step 16  vrf vrf-name

Example:
RP/0/RSP0/CPU0:router(config-bgp)# vrf vrf-pe

Configures a VRF instance.

Step 17  rd { as-number : nn | ip-address : nn | auto }

Example:
RP/0/RSP0/CPU0:router(config-bgp-vrf)# rd 345:567

Configures the route distinguisher.

Use the auto keyword if you want the router to automatically assign a unique RD to the VRF.

Step 18  Use the commit or end command.

commit - Saves the configuration changes and remains within the configuration session.

end - Prompts user to take one of these actions:

• Yes - Saves configuration changes and exits the configuration session.
• No - Exits the configuration session without committing the configuration changes.
• Cancel - Remains in the configuration mode, without committing the configuration changes.

Configuring OSPFv3 as the Routing Protocol Between the PE and CE Routers

Perform this task to configure provider edge (PE)-to-customer edge (CE) routing sessions that use Open Shortest Path First version 3 (OSPFv3).

SUMMARY STEPS

1. configure
2. router ospf process-name
3. vrf vrf-name
4. capability vrf-lite
5. router-id {router-id | type interface-path-id}
6. domain-id type {0005 | 0105 | 0205 | 8005} value domain-id
7. Do one of the following:
   • redistribute bgp process-id [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]
   • redistribute connected [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]
   • redistribute ospf process-id [match {external {1 | 2} | internal | nssa-external {1 | 2}}] [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]
   • redistribute static [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]
   • redistribute eigrp process-id [match {external {1 | 2} | internal | nssa-external {1 | 2}}] [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]
   • redistribute rip [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]
8. area area-id
9. interface {type interface-path-id}
10. Use the commit or end command.

DETAILED STEPS

Step 1 configure
   Example:
   RP/0/RSP0/CPU0:router# configure

   Enters the Global Configuration mode.

Step 2 router ospf process-name
   Example:
   RP/0/RSP0/CPU0:router(config)# router ospf 109

   Enters OSPF configuration mode allowing you to configure the OSPF routing process.

Step 3 vrf vrf-name
   Example:
   RP/0/RSP0/CPU0:router(config-ospf)# vrf vrf_1

   Configures a VPN routing and forwarding (VRF) instance and enters VRF configuration mode for OSPF routing.

Step 4 capability vrf-lite
   Example:
Enables VRF Lite feature.

**Step 5**

`router-id` 

Example:

```
RP/0/RSP0/CPU0:router(config-ospf-vrf)# router-id 172.20.10.10
```

Configures the router ID for the OSPF routing process.

*Note* Router ID configuration is required for each VRF.

**Step 6**

`domain-id` 

Example:

```
RP/0/RSP0/CPU0:router(config-ospf-vrf)# domain-id type 0005 value CAFE00112233
```

Specifies the domain ID.

**Step 7**

Do one of the following:

- `redistribute bgp` 
  - `process-id` 
  - `metric metric-value` 
  - `metric-type {1 | 2}` 
  - `route-policy policy-name` 
  - `tag tag-value`

- `redistribute connected` 
  - `metric metric-value` 
  - `metric-type {1 | 2}` 
  - `route-policy policy-name` 
  - `tag tag-value`

- `redistribute ospf` 
  - `process-id` 
  - `match {external [1 | 2] | internal | nssa-external [1 | 2]} [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]`

- `redistribute static` 
  - `metric metric-value` 
  - `metric-type {1 | 2}` 
  - `route-policy policy-name` 
  - `tag tag-value`

- `redistribute eigrp` 
  - `process-id` 
  - `match {external [1 | 2] | internal | nssa-external [1 | 2]} [metric metric-value] [metric-type {1 | 2}] [route-policy policy-name] [tag tag-value]`

- `redistribute rip` 
  - `metric metric-value` 
  - `metric-type {1 | 2}` 
  - `route-policy policy-name` 
  - `tag tag-value`

Example:

```
RP/0/RSP0/CPU0:router(config-ospf-vrf)# redistribute connected
```

Causes routes to be redistributed into OSPF. The routes that can be redistributed into OSPF are:

- Border Gateway Protocol (BGP)
- Connected
- Enhanced Interior Gateway Routing Protocol (EIGRP)
- OSPF
- Static
- Routing Information Protocol (RIP)

**Step 8**

`area area-id`

Example:
Step 9

**interface** \{type interface-path-id\}

**Example:**

```
RP/0/RSP0/CPU0:router(config-ospf-vrf-ar)# interface GigabitEthernet 0/3/0/0
```

Associates interface GigabitEthernet 0/3/0/0 with area 0.

Step 10

Use the **commit** or **end** command.

- **commit** - Saves the configuration changes and remains within the configuration session.
- **end** - Prompts user to take one of these actions:
  - **Yes** - Saves configuration changes and exits the configuration session.
  - **No** - Exits the configuration session without committing the configuration changes.
  - **Cancel** - Remains in the configuration mode, without committing the configuration changes.

---

**Configuration Examples for 6PE/VPE**

This section includes the following configuration example:

**Configuring 6PE on a PE Router: Example**

This sample configuration shows the configuration of 6PE on a PE router:

```
interface TenGigE0/3/0/0
  ipv6 address 2001::1/64

router isis ipv6-cloud
  net 49.0000.0000.0001.00
  address-family ipv6 unicast
    single-topology
  interface TenGigE0/3/0/0
    address-family ipv6 unicast
    !

router bgp 55400
  bgp router-id 54.6.1.1
  address-family ipv4 unicast
  !
  address-family ipv6 unicast
  network 55:5::/64
  redistribute connected
  redistribute isis ipv6-cloud
  allocate-label all
  !
```
Configuring 6VPE on a PE Router: Example

This sample configuration shows the configuration of 6VPE on a PE router:

```plaintext
neighbor 34.4.3.3
remote-as 55400
address-family ipv4 unicast
! address-family ipv6 labeled-unicast

Configuring 6VPE on a PE Router: Example

This sample configuration shows the configuration of 6VPE on a PE router:

```
```
CHAPTER 4

Implementing Generic Routing Encapsulation

Generic Routing Encapsulation (GRE) is a tunneling protocol developed by Cisco Systems that encapsulates a wide variety of network layer protocols inside virtual point-to-point links over an Internet Protocol internetwork.

Feature History for Configuring Link Bundling on Cisco IOS XR Software

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release 4.3.0</td>
<td>These feature were supported on the Cisco ASR 9000 Series Aggregation Services Routers:</td>
</tr>
<tr>
<td></td>
<td>• MPLS/L3VPNNoGRE on ASR 9000 Enhanced Ethernet Line Card and Cisco ASR 9000 Series SPA Interface Processor-700</td>
</tr>
<tr>
<td></td>
<td>• RSVP/TEoGRE on ASR 9000 Enhanced Ethernet Line Card and Cisco ASR 9000 Series SPA Interface Processor-700</td>
</tr>
<tr>
<td></td>
<td>• VRF aware GRE on ASR 9000 Enhanced Ethernet Line Card and Cisco ASR 9000 Series SPA Interface Processor-700</td>
</tr>
<tr>
<td></td>
<td>• L2VPN (VPWS and VPLS) on GRE for ASR 9000 Enhanced Ethernet Line Card only</td>
</tr>
<tr>
<td>Release 5.1.1</td>
<td>Support for GRE Tunnel Key and Tunnel Key-Ignore was introduced.</td>
</tr>
<tr>
<td>Release 5.2.2</td>
<td>Support for GRE tunnel on an IPv6 transport network.</td>
</tr>
<tr>
<td>Release 5.3.2</td>
<td>Support for GRE IPv4 Transport Over MPLS was introduced.</td>
</tr>
<tr>
<td>Release 6.0.1</td>
<td>Support for GRE IPv6 Transport Over MPLS was introduced.</td>
</tr>
</tbody>
</table>

Prerequisites for Configuring Generic Routing Encapsulation

Before configuring Link Bundling, be sure that these tasks and conditions are met:
Information About Generic Routing Encapsulation

To implement the GRE feature, you must understand these concepts:

GRE Overview

Generic Routing Encapsulation (GRE) tunneling protocol provides a simple generic approach to transport packets of one protocol over another protocol by means of encapsulation.

GRE encapsulates a payload, that is, an inner packet that needs to be delivered to a destination network inside an outer IP packet. GRE tunnel endpoints send payloads through GRE tunnels by routing encapsulated packets through intervening IP networks. Other IP routers along the way do not parse the payload (the inner packet); they only parse the outer IP packet as they forward it towards the GRE tunnel endpoint. Upon reaching the tunnel endpoint, GRE encapsulation is removed and the payload is forwarded to it’s ultimate destination.

MPLS networks provide VPN functionality by tunneling customer data through public networks using routing labels. Service Providers (SP) provide MPLS L3VPN, 6PE/6VPE and L2VPN services to their customers who have interconnected private networks.

MPLS and L3VPN are supported over regular interfaces on Cisco ASR 9000 Series Aggregation Services Routers through GRE tunnels over an IPv4 transport network. MPLS support is extended over IPv4 GRE tunnels between routers as the provider core may not be fully MPLS aware.

GRE Features

The following sections list the GRE features:

Note

An IPv6 GRE tunnel does not support features that involve transport of MPLS packets through a GRE tunnel.

MPLS/L3VPN over GRE

The MPLS VPN over GRE feature provides a mechanism for tunneling Multiprotocol Label Switching (MPLS) packets over a non-MPLS network. This feature utilizes MPLS over generic routing encapsulation (MPLSoGRE) to encapsulate MPLS packets inside IP tunnels. The encapsulation of MPLS packets inside IP tunnels creates a virtual point-to-point link across non-MPLS networks.

L3VPN over GRE basically means encapsulating L3VPN traffic in GRE header and its outer IPv4 header with tunnel destination and source IP addresses after imposing zero or more MPLS labels, and transporting it across the tunnel over to the remote tunnel end point. The incoming packet can be a pure IPv4 packet or an MPLS packet. If the incoming packet is IPv4, the packet enters the tunnel through a VRF interface, and if the incoming packet is MPLS, then the packet enters through an MPLS interface. In the IPv4 case, before encapsulating in the outer IPv4 and GRE headers, a VPN label corresponding to the VRF prefix and any IGP
label corresponding to the IGP prefix of the GRE tunnel destination is imposed on the packet. In the case of MPLS, the top IGP label is swapped with any label corresponding to the GRE tunnel destination address.

**PE-to-PE Tunneling**

The provider-edge-to-provider-edge (PE-to-PE) tunneling configuration provides a scalable way to connect multiple customer networks across a non-MPLS network. With this configuration, traffic that is destined to multiple customer networks is multiplexed through a single GRE tunnel.

A similar nonscalable alternative is to connect each customer network through separate GRE tunnels (for example, connecting one customer network to each GRE tunnel).

As shown in the following figure, the PE devices assign VPN routing and forwarding (VRF) numbers to the customer edge (CE) devices on each side of the non-MPLS network.

The PE devices use routing protocols such as Border Gateway Protocol (BGP), Open Shortest Path First (OSPF), or Routing Information Protocol (RIP) to learn about the IP networks behind the CE devices. The routes to the IP networks behind the CE devices are stored in the associated CE device's VRF routing table.

The PE device on one side of the non-MPLS network uses the routing protocols (that operate within the non-MPLS network) to learn about the PE device on the other side of the non-MPLS network. The learned routes that are established between the PE devices are then stored in the main or default routing table.

The opposing PE device uses BGP to learn about the routes that are associated with the customer networks that are behind the PE devices. These learned routes are not known to the non-MPLS network.

The following figure shows BGP defining a static route to the BGP neighbor (the opposing PE device) through the GRE tunnel that spans the non-MPLS network. Because routes that are learned by the BGP neighbor include the GRE tunnel next hop, all customer network traffic is sent using the GRE tunnel.

**Figure 8: PE-to-PE Tunneling**

**P-to-PE Tunneling**

As shown in the following figure, the provider-to-provider-edge (P-to-PE) tunneling configuration provides a way to connect a PE device (P1) to an MPLS segment (PE-2) across a non-MPLS network. In this configuration, MPLS traffic that is destined to the other side of the non-MPLS network is sent through a single GRE tunnel.
Figure 9: P-to-PE Tunneling

As shown in the following figure, the provider-to-provider (P-to-P) configuration provides a method of connecting two MPLS segments (P1 to P2) across a non-MPLS network. In this configuration, MPLS traffic that is destined to the other side of the non-MPLS network is sent through a single GRE tunnel.

Figure 10: P-to-P Tunneling

6PE/6VPE

Service Providers (SPs) use a stable and established core with IPv4/MPLS backbone for providing IPv4 VPN services. The 6PE/6VPE feature facilitates SPs to offer IPv6 VPN services over this backbone without an IPv6 core. The provide edge (PE) routers run MP-iBGP (Multi-Protocol iBGP) to advertise v6 reachability and v6 label distribution. For 6PE, the labels are allocated per IPv6 prefix learnt from connected customer edge (CE) routers and for 6VPE, the PE router can be configured to allocate labels on a per-prefix or per-CE/VRF level.

6PE/6VPE over GRE

While IPv4/MPLS allows SPs to transport IPv6 traffic across IPv4 core (IPv6 unaware), MPLS over GRE allows MPLS traffic to be tunneled through MPLS unaware networks. These two features together facilitate IPv6 traffic to be transported across IPv6 as well as MPLS unaware core segments. Only the PE routers need to be aware of MPLS and IPv6 (Dual stack).

The 6PE/6VPE over GRE feature allows the use of IPv4 GRE tunnels to provide IPv6 VPN over MPLS functionality to reach the destination v6 prefixes via the BGP next hop through MPLS & IPv6 unaware core.

MPLS Forwarding

When IPv6 traffic is received from one customer site, the ingress PE device uses MPLS to tunnel IPv6 VPN packets over the backbone toward the egress PE device identified as the BGP next hop. The ingress PE device prefixes the IPv6 packets with the outer and inner labels before placing the packet on the egress interface.

Under normal operation, a P device along the forwarding path does not lookup the frame beyond the first label. The P device either swaps the incoming label with an outgoing one or removes the incoming label if the next device is a PE device. Removing the incoming label is called penultimate hop popping. The remaining
label (BGP label) is used to identify the egress PE interface toward the customer site. The label also hides the protocol version (IPv6) from the last P device, which it would otherwise need to forward an IPv6 packet.

A P device is ignorant of the IPv6 VPN routes. The IPv6 header remains hidden under one or more MPLS labels. When the P device receives an MPLS-encapsulated IPv6 packet that cannot be delivered, it has two options. If the P device is IPv6 aware, it exposes the IPv6 header, builds an Internet Control Message Protocol (ICMP) for IPv6 message, and sends the message, which is MPLS encapsulated, to the source of the original packet. If the P device is not IPv6 aware, it drops the packet.

6PE/6VPE over GRE

As discussed earlier, 6PE/6VPE over GRE basically means enabling IPv6/IPv6 VPN over MPLS over GRE.

The ingress PE device uses IPv4 generic routing encapsulation (GRE) tunnels combined with 6PE/6VPE over MPLS to tunnel IPv6 VPN packets over the backbone toward the egress PE device identified as the BGP next hop.

The PE devices establish MP-iBGP sessions and MPLS LDP sessions just as in the case of 6PE/6VPE. The difference here is that these sessions are established over GRE tunnels, which also means that the PEs are just one IGP hop away. The P routers in the tunnel path only need to forward the traffic to the tunnel destination, which is an IPv4 address.

This is how the IPv6 LSP is setup for label switching the IPv6 traffic:

- After the LDP and BGP sessions are established, the PEs exchange IPv6 prefixes that they learn from the CEs and the corresponding IPv6 labels, just as in the case of IPv4 VPN.
- The IPv6 labels occupy the inner most position in the label stack.
- The IPv4 labels corresponding to the PE IPv4 addresses occupy the outer position in the stack.
- When IPv6 traffic needs to be forwarded from PE1 to PE2, the outer PE2 IPv4 label is used to label switch the traffic to PE2, and the inner IPv6 label is used to send the packet out of the interface connected to the CE.

GRE Tunnel Key

The GRE Tunnel Key feature enables the encapsulation router to add a four-byte key, as part of the GRE header, during encapsulation. In the decapsulation router, the GRE key of an incoming packet should match the key value configured under the GRE tunnel. During decapsulation, if a mismatch between the key value of the incoming GRE packet and the key value configured under the GRE tunnel is identified, the incoming packet is dropped.

- GRE tunnel key feature is supported only on Cisco ASR 9000 Enhanced Ethernet line cards. It is mandatory to have ingress and egress line cards as Enhanced Ethernet line cards.
- Either the same key or different keys can be configured under multiple GRE tunnels for a given router. However, more than one tunnel, having the same tunnel source and destination but a different tunnel key is not supported because the source and destination pair for various configured tunnels must be unique irrespective of the key value. Also, two tunnels with the same tunnel source and destination, but one tunnel being with key and the other tunnel being without key is not supported.
- Different traffic streams passing through the same GRE tunnel contains the same GRE key configured for that tunnel.
- Use the `tunnel key` command to configure the key value at both ends of a GRE tunnel.
The following figure shows a simple representation of the GRE tunnel key configuration:

**Figure 11: GRE Tunnel with Key**

![GRE Tunnel with Key](image)

The following figure shows the complete format of the GRE header with the key field:

**Figure 12: GRE Header**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>Control plane packet</td>
</tr>
<tr>
<td>K/S</td>
<td>Key/Service protocol</td>
</tr>
<tr>
<td>Reserved</td>
<td>Optional</td>
</tr>
<tr>
<td>Ver</td>
<td>Version</td>
</tr>
<tr>
<td>Protocol Type</td>
<td>Protocol type</td>
</tr>
<tr>
<td>Checksum</td>
<td>(Optional)</td>
</tr>
<tr>
<td>Reserved</td>
<td>(Optional)</td>
</tr>
<tr>
<td>Key</td>
<td>(Optional)</td>
</tr>
</tbody>
</table>

### GRE Tunnel Key-Ignore

If a GRE key is configured on only one endpoint router of a GRE tunnel, the other router that has no GRE key configured discards any incoming tunnel packet that has a GRE key. To enable this router to ignore GRE keys and accept incoming data plane packets on the GRE tunnel, run the `tunnel key-ignore` command. Control plane packets over a GRE tunnel are accepted only if there is no GRE tunnel key configured on both the tunnel endpoints or both the endpoints are configured with a GRE key and the control plane packet passes the GRE key validation. Hence, in the above scenario, both the routers discard any incoming control plane packets from the GRE tunnel.

---

**Note**

Do not configure a GRE key on the GRE tunnel endpoint router if you have configured the router to ignore GRE keys. Configuring a GRE key overrides the `tunnel key-ignore` command and thus cancels the skipping of GRE key validation. This results in the router accepting from the incoming tunnel traffic only those packets that have the matching GRE key.

### GRE tunnel in VRF domains

You can configure an IPv4/IPv6 GRE tunnel between two interfaces that belong to a Virtual Forwarding and Routing (VRF) instance. This contains or limits the tunnel path within this specific VRF instance. For example, packets can be sent internally within a default or non-default VRF instance separated through an intermediate VRF that contains the GRE tunnel.
In the above topology, a GRE tunnel is configured in the core network, which is an IPv4 cloud. For packets entering through Interface1, the provider edge (PE) devices PEi and PEe are the tunnel head and tunnel exit respectively.

The VRF configured on Interface1 is the customer VRF. Packets entering this interface are routed using this customer VRF to the tunnel. The routing by the customer VRF is called inner IP packet routing. You can configure the tunnel to be visible to the customer VRF instance using the `vrf vrf-name` command. This enables only the configured VRF instance to use the tunnel, that is, forward traffic from PEi into this tunnel and also receive all incoming PEi tunnel packets.

The VRF configured on the tunnel using the `tunnel vrf` command is the transport VRF. The packet entering the tunnel is encapsulated with the tunnel source and destination addresses. The transport VRF routes this encapsulated payload between the tunnel endpoints. The routing by the transport VRF is the outer IP packet routing. If no transport VRF is configured for the tunnel, the PEi device looks up the tunnel endpoint addresses in the default VRF instance, that is, the global routing table.

### Restrictions on a GRE tunnel

The following restrictions are applicable for a GRE tunnel:

- MPLS packets cannot be transported within an IPv6 GRE tunnel. Therefore, the following features are not supported on an IPv6 GRE tunnel:
  - MPLS/L3VPN over GRE
  - 6PE/6VPE
  - 6PE/6VPE over GRE

- Multicast packets cannot be transported within an IPv6 GRE tunnel.

- Multicast packets cannot be transported within an IPv4 GRE tunnel that is configured in a transport VRF.

- Keep-Alive packets are not supported on an IPv6 GRE tunnel. You can use the Bidirectional Forwarding Detection (BFD) protocol to detect link failures in an IPv6 GRE tunnel.

- The IPv4 addresses are mandatory for configuring GRE tunnels under the VRF, as this would ensure the traffic flows through the tunnel in an expected manner. Use either an IP unnumbered interface or a loopback interface belonging to that VRF for establishing the GRE tunnels under a VRF. Though the tunnel may come up without the aforementioned configuration, the traffic may not pass over the GRE tunnel, since the IP information on the tunnel interface is not available for forwarding the traffic correctly. Also, for the VRF information to be written in hardware database the IP information is required. Therefore, the IP unnumbered GRE tunnels may not work as expected as they may not forward traffic on the device.
GRE IPv4/IPv6 Transport Over MPLS

The Generic Routing Encapsulation (GRE) IPv4/IPv6 transport over Multiprotocol Label Switching (MPLS) feature provides a mechanism to configure GRE tunnels, where the tunnel destination IPv4/IPv6 address is reachable through an MPLS label switched path (LSP). With this feature, IPv4, IPv6, routing protocols - OSPF, ISIS, and L2VPN and L3VPN packets are accepted as payload packets for GRE encapsulation. IPv4/IPv6 is supported as the GRE delivery protocol.

This feature overcomes the restriction of not being able to configure the tunnel destination endpoint through an MPLS LSP during tunnel configuration.

The GRE IPv4/IPv6 transport over MPLS feature facilitates creation of GRE tunnels over LSPs, through L3VPN inter-AS (autonomous system) options:

- External Border Gateway Protocol (EBGP) redistribution of labeled VPN IPv4/IPv6 routes from an AS to a neighboring AS.
- Multi-hop EBGP redistribution of labeled VPN IPv4/IPv6 routes between source and destination ASs, with EBGP redistribution of labeled IPv4/IPv6 routes from an AS to a neighboring AS.

Multipoint GRE IPv4/IPv6 transport over MPLS is also supported.

The GRE IPv4/IPv6 transport over MPLS feature is supported on the following types of Cisco ASR 9000 line cards:

- Cisco ASR 9000 Enhanced Ethernet line card
- Cisco ASR 9000 High Density 100GE Ethernet line card

Limitations

- GRE IPv4/IPv6 transport over MPLS-TE tunnels is not supported.
- GREoMPLS with IP Fast Reroute (IPFRR).

How to Configure Generic Routing Encapsulation

Configuring a GRE Tunnel

Perform this task to configure a GRE tunnel.

SUMMARY STEPS

1. configure
2. interface tunnel-ip number
3. vrf vrf-name
4. ipv4 address ipv4-address mask
5. tunnel mode gre {ipv4 | ipv6}
6. tunnel source {ip-address | type path-id}
7. **tunnel destination ip-address**
8. **tunnel vrf transport-vrf-name**
9. Use the **commit** or **end** command.

### DETAILED STEPS

**Step 1**  
**configure**

**Example:**

```
RP/0/RSP0/CPU0:router# configure
```

Enters the Global Configuration mode.

**Step 2**  
**interface tunnel-ip number**

**Example:**

```
RP/0/RSP0/CPU0:router(config)# interface tunnel-ip 4000
```

Enters tunnel interface configuration mode.
- **number** is the number associated with the tunnel interface.

**Step 3**  
**vrf vrf-name**

**Example:**

```
RP/0/RSP0/CPU0:router(config-if)# vrf vrf1
```

(Optional) Specifies the VRF domain that can route packets into and from the tunnel.

**Note**  
This step is not required if the tunnel is available for global routing and therefore, is not specific to a VRF.

**Step 4**  
**ipv4 address ipv4-address mask**

**Example:**

```
RP/0/RSP0/CPU0:router(config-if)# ipv4 address 10.1.1.1 255.255.255.0
```

Specifies the IPv4 address and subnet mask for the interface.
- **ipv4-address** specifies the IP address of the interface.
- **subnet-mask** specifies the subnet mask of the interface.

**Step 5**  
**tunnel mode gre {ipv4 | ipv6}**

**Example:**

```
Configuring a GRE Tunnel

RP/0/RSP0/CPU0:router(config-if)# tunnel mode gre ipv4

Specify whether the transport network is an IPv4 or IPv6 network. The default GRE tunnel mode is IPv4.

**Note**  The tunnel source and destination addresses should match the tunnel mode. A mismatch in configuration causes the tunnel to fail without any error message.

**Step 6**

**tunnel source** \{ ip-address | type path-id \}

**Example:**

RP/0/RSP0/CPU0:router(config-if)# tunnel source TenGigE0/2/0/1

Specifies the source of the tunnel interface.

**Note**  It is recommended that the tunnel source is identified using the interface ID and not the IP address. Using the interface ID enables the router to mark the tunnel as down when the interface is down and the routing protocol tries to find and use an alternate route to the tunnel route.

**Step 7**

**tunnel destination** ip-address

**Example:**

RP/0/RSP0/CPU0:router(config-if)# tunnel destination 145.12.5.2

Defines the tunnel destination.

**Step 8**

**tunnel vrf** transport-vrf-name

**Example:**

RP/0/RSP0/CPU0:router(config-if)# tunnel vrf vrf99

(Optional) Associates the transport VRF with the tunnel. The transport VRF contains the interfaces over which the tunnel sends as well as receives packets (outer IP packet routing).

**Note**  This step is not required if the tunnel endpoints belong to the global routing table.

**Step 9**

Use the `commit` or `end` command.

- **commit** - Saves the configuration changes and remains within the configuration session.
- **end** - Prompts user to take one of these actions:
  - **Yes** - Saves configuration changes and exits the configuration session.
  - **No** - Exits the configuration session without committing the configuration changes.
  - **Cancel** - Remains in the configuration mode, without committing the configuration changes.
Configuring the Tunnel Key

Perform this task to configure the tunnel key for the GRE encapsulated packets. You need to perform same configuration steps on the other endpoint router of the tunnel ensuring that the key value is the same at both the local and remote GRE interfaces.

**SUMMARY STEPS**

1. configure
2. interface tunnel-ip number
3. ipv4 address ipv4-address subnet-mask
4. tunnel key value
5. (Optional) tunnel tos tos-value
6. tunnel source type path-id
7. tunnel destination ip-address
8. Use the commit or end command.

**DETAILED STEPS**

**Step 1**

configure

Example:

```
RP/0/RSP0/CPU0:router# configure
```

Enters the Global Configuration mode.

**Step 2**

interface tunnel-ip number

Example:

```
RP/0/RSP0/CPU0:router(config)# interface tunnel-ip 10
```

Enters tunnel interface configuration mode.

- number is the number associated with the tunnel interface.

**Step 3**

ipv4 address ipv4-address subnet-mask

Example:

```
RP/0/RSP0/CPU0:router(config-if)# ipv4 address 101.0.9.1 255.255.255.0
```

Specifies the IPv4 address and subnet mask for the interface.

- ipv4-address specifies the IP address of the interface.
- subnet-mask specifies the subnet mask of the interface.

**Step 4**

tunnel key value
Example:

RP/0/RSP0/CPU0:router(config-if)# tunnel key 10

Enables tunnel key.

Step 5  
(Optional) **tunnel tos tos-value**

Example:

RP/0/RSP0/CPU0:router(config-if)# tunnel tos 96

Specifies the value of the TOS field in the tunnel encapsulating packets.

Step 6  
**tunnel source** *type path-id*

Example:

RP/0/RSP0/CPU0:router(config-if)# tunnel source Loopback10

Specifies the source of the tunnel interface.

Step 7  
**tunnel destination** *ip-address*

Example:

RP/0/RSP0/CPU0:router(config-if)# tunnel destination 33.0.9.33

Defines the tunnel destination.

Step 8  
Use the **commit** or **end** command.

**commit** - Saves the configuration changes and remains within the configuration session.

**end** - Prompts user to take one of these actions:

- **Yes** - Saves configuration changes and exits the configuration session.
- **No** - Exits the configuration session without committing the configuration changes.
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.

---

**Configuring the Tunnel Key-Ignore**

Perform this task to configure the tunnel key-ignore for the GRE encapsulated packets. You need to perform same configuration steps on the other endpoint router of the tunnel.

**SUMMARY STEPS**

1. configure
2. interface tunnel-ip *number*
3. ipv4 address *ipv4-address subnet-mask*
4. **tunnel key-ignore**

5. *(Optional) tunnel tos tos-value*

6. **tunnel source** type path-id

7. **tunnel destination** ip-address

8. Use the **commit** or **end** command.

**DETAILED STEPS**

**Step 1**  
**configure**

*Example:*

```
RP/0/RSP0/CPU0:router# configure
```

Enters the Global Configuration mode.

**Step 2**  
**interface tunnel-ip number**

*Example:*

```
RP/0/RSP0/CPU0:router(config)# interface tunnel-ip 10
```

Enters tunnel interface configuration mode.

- *number* is the number associated with the tunnel interface.

**Step 3**  
**ipv4 address ipv4-address subnet-mask**

*Example:*

```
RP/0/RSP0/CPU0:router(config-if)# ipv4 address 101.0.9.1 255.255.255.0
```

Specifies the IPv4 address and subnet mask for the interface.

- *ipv4-address* specifies the IP address of the interface.
- *subnet-mask* specifies the subnet mask of the interface.

**Step 4**  
**tunnel key-ignore**

*Example:*

```
RP/0/RSP0/CPU0:router(config-if)# tunnel key-ignore
```

Enables tunnel key-ignore.

**Step 5**  
*(Optional) tunnel tos tos-value*

*Example:*

```
RP/0/RSP0/CPU0:router(config-if)# tunnel tos 96
```

Specifies the value of the TOS field in the tunnel encapsulating packets.
Configuring a VRF Interface

Perform this task to configure a VRF interface.

SUMMARY STEPS

1. configure
2. interface type interface-path-id
3. vrf vrf-name
4. ipv4 address ipv4-address mask
5. Use the commit or end command.

DETAILED STEPS

Step 1  configure
Example:
RP/0/RSP0/CPU0:router# configure

Enters the Global Configuration mode.

Step 2  interface type interface-path-id
Example:
Configuring VRF Routing Protocol

Perform this task to configure the VRF routing protocol.

**SUMMARY STEPS**

1. **configure**
2. **router ospf** `process-name`
3. **vrf** `vrf-name`
4. **router-id** `{router-id | type interface-path-id}`
5. **area** `area-id`
6. **interface** `type interface-path-id`
7. Use the **commit** or **end** command.

**DETAILED STEPS**

**Step 1** **configure**

Example:
Step 2  
**router ospf** *process-name*

**Example:**

```
RP/0/RSP0/CPU0:router(config)# router ospf 109
```

Enters OSPF configuration mode allowing you to configure the OSPF routing process.

Step 3  
**vrf** *vrf-name*

**Example:**

```
RP/0/RSP0/CPU0:router(config-ospf)# vrf vrf_1
```

Configures a VPN routing and forwarding (VRF) instance and enters VRF configuration mode for OSPF routing.

Step 4  
**router-id** `{router-id | type interface-path-id}`

**Example:**

```
RP/0/RSP0/CPU0:router(config-ospf-vrf)# router-id 172.20.10.10
```

Configures the router ID for the OSPF routing process.

Step 5  
**area** *area-id*

**Example:**

```
RP/0/RSP0/CPU0:router(config-ospf-vrf)# area 0
```

Configures the OSPF area as area 0.

Step 6  
**interface** *type interface-path-id*

**Example:**

```
RP/0/RSP0/CPU0:router(config-ospf-vrf-ar)# interface GigabitEthernet 0/3/0/0
```

Associates interface GigabitEthernet 0/3/0/0 with area 0.

Step 7  
Use the **commit** or **end** command.

- **commit** - Saves the configuration changes and remains within the configuration session.
- **end** - Prompts user to take one of these actions:
  - **Yes** - Saves configuration changes and exits the configuration session.
  - **No** - Exits the configuration session without committing the configuration changes.
• **Cancel** - Remains in the configuration mode, without committing the configuration changes.

---

# Configuring IGP for Remote PE Reachability

Perform this task to configure IGP for remote PE reachability.

## SUMMARY STEPS

1. `configure`
2. `router ospf process-name`
3. `router-id {router-id}`
4. `area area-id`
5. `interface tunnel-ip number`
6. Use the `commit` or `end` command.

## DETAILED STEPS

### Step 1  configure

**Example:**
```
RP/0/RSP0/CPU0:router# configure
```

Enters the Global Configuration mode.

### Step 2  router ospf process-name

**Example:**
```
RP/0/RSP0/CPU0:router(config)# router ospf 1
```

Enables OSPF routing for the specified routing process and places the router in router configuration mode.

### Step 3  router-id {router-id}

**Example:**
```
RP/0/RSP0/CPU0:router(config-ospf)# router-id 1.1.1.1
```

Configures a router ID for the OSPF process.

**Note** We recommend using a stable IP address as the router ID.

### Step 4  area area-id

**Example:**
```
RP/0/RSP0/CPU0:router(config-ospf)# area 0
```

Enters area configuration mode and configures an area for the OSPF process.
Configuring LDP on GRE Tunnel

Perform this task to configure LDP on a GRE tunnel.

SUMMARY STEPS

1. configure
2. mpls ldp
3. router-id \{router-id\}
4. interface tunnel-ip \number\n5. Use the commit or end command.

DETAILED STEPS

Step 1    configure
Example:
RP/0/RSP0/CPU0:router# configure

Enters the Global Configuration mode.

Step 2    mpls ldp
Example:
RP/0/RSP0/CPU0:router(config)# mpls ldp

Enables MPLS LDP configuration mode.

Step 3    router-id \{router-id\}
Example:

RP/0/RSP0/CPU0:router(config-ldp)# router-id 1.1.1.1

Configures a router ID for the OSPF process.

Note: We recommend using a stable IP address as the router ID.

Step 4  interface tunnel-ip number
Example:

RP/0/RSP0/CPU0:router(config-ldp)# interface tunnel-ip 4

Enters tunnel interface configuration mode.
  • number is the number associated with the tunnel interface.

Step 5  Use the commit or end command.

commit - Saves the configuration changes and remains within the configuration session.
end - Prompts user to take one of these actions:
  • Yes - Saves configuration changes and exits the configuration session.
  • No - Exits the configuration session without committing the configuration changes.
  • Cancel - Remains in the configuration mode, without committing the configuration changes.

Configuring MP-iBGP to Exchange VPN-IPv4 Routes

Perform this task to configure MP-iBGP to exchange VPN-IPv4 routes.

SUMMARY STEPS

1. configure
2. router bgp as-number
3. router-id ip-address
4. neighbor ip-address
5. remote-as as-number
6. update-source type interface-path-id
7. address-family { vpnv4 | vpnv6 unicast }
8. Use the commit or end command.

DETAILED STEPS

Step 1  configure
Example:

RP/0/RSP0/CPU0:router# configure
Enters the Global Configuration mode.

**Step 2**  
**router bgp** `as-number`

**Example:**

```
RP/0/RSP0/CPU0:router(config)# router bgp 1
```

Specifies the autonomous system number and enters the BGP configuration mode, allowing you to configure the BGP routing process.

**Step 3**  
**router-id** `ip-address`

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp)# router-id 1.1.1.1
```

Configures the local router with a specified router ID.

**Step 4**  
**neighbor** `ip-address`

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp)# neighbor 4.4.4.4
```

Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address as a BGP peer.

**Step 5**  
**remote-as** `as-number`

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 1
```

Creates a neighbor and assigns a remote autonomous system number to it.

**Step 6**  
**update-source** `type interface-path-id`

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-nbr)# update-source Loopback0
```

Allows sessions to use the primary IP address from a specific interface as the local address when forming a session with a neighbor.

**Step 7**  
**address-family** `{ vpnv4 | vpnv6 unicast }

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family vpnv4 unicast
```

Enters address family configuration submode for the specified address family.

**Step 8**  
Use the **commit** or **end** command.

- **commit** - Saves the configuration changes and remains within the configuration session.
end - Prompts user to take one of these actions:

• Yes - Saves configuration changes and exits the configuration session.
• No - Exits the configuration session without committing the configuration changes.
• Cancel - Remains in the configuration mode, without committing the configuration changes.

Configuration Examples for Generic Routing Encapsulation

This section provides examples to configure GRE:

Configuring an IPv4 GRE Tunnel: Example

This example shows how to configure an IPv4 GRE tunnel:

```
configure
  interface tunnel-ip1
  ipv4 address 12.0.0.1 255.255.255.0
  tunnel source Loopback0
  tunnel destination 200.200.200.1
end
```

Configuring an IPv6 GRE Tunnel: Example

```
interface tunnel-ip 1
  vrf RED
  ipv4 address 10.1.1.2/24
  ipv6 address 10::2/64
  tunnel mode gre ipv6
  tunnel source GigabitEthernet 0/0/0/0
  tunnel destination 100::1
  tunnel vrf BLUE

vrf blue
  description connected to IXIA in blue VRF
  address-family ipv4 unicast
    import route-target 100:1
    export route-target 100:1

vrf red
  description connected to core interface in red VRF
  address-family ipv4 unicast
    import route-target 200:1
    export route-target 200:1
```
Configuring Global VRF: Example

This example shows how to configure global VRF:

```
configure
vrf VRF1
  address-family ipv4 unicast
  import route-target 120:1
  export route-target 120:2
  exit
exit
router bgp 120
  vrf VRF1
  rd auto
end
```

Configuring a VRF Interface: Example

This example shows how to configure a VRF interface:

```
configure
  interface tunnel-ip 100
  vrf VRF1
  ipv4 address 1.1.1.1 255.255.255.0
  ipv6 address 100::2/64
end
```

Configuring VRF Routing Protocol: Example

This example shows how to configure VRF routing protocol:
Configuring IGP for Remote PE Reachability: Example

This example shows how to configure IGP for remote provider edge (PE) reachability:

```
configure
 router ospf109
  vrf VRF1
  router-id 172.20.10.10
  area0
    interface GigabitEthernet0/3/0/0
end
```

Configuring LDP on GRE Tunnel: Example

This example shows how to configure LDP on a GRE tunnel:

```
configure
 mpls ldp
 router-id 172.20.10.10
  interface tunnel-ip1
end
```

Configuring MP-iBGP to Exchange VPN-IPv4 Routes: Example

This example shows how to configure MP-iBGP to exchange VPN-IPv4 routes:

```
configure
 router bgp100
  router-id 172.20.10.10
  neighbor 2.2.2.2 remote-as 100
  update-source Loopback0
  address-family vpnv4 unicast
end
```
Configuring MP-iBGP to Exchange VPN-IPv4 Routes: Example
Implementing VXLAN

This module provides configuration information for layer 3 VXLAN on Cisco ASR 9000 Series Router. For conceptual information on VXLAN, see Implementing VXLAN chapter in the L2VPN and Ethernet Services Configuration Guide for Cisco ASR 9000 Series Routers.

Table 2: Feature History for VXLAN

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release 5.2.0</td>
<td>This feature was introduced on Cisco ASR 9000 Series Router.</td>
</tr>
</tbody>
</table>

- Configuring a Layer 3 VXLAN gateway, on page 119
- Configuration Example for Implementing Layer 3 VXLAN Gateway, on page 123

Configuring a Layer 3 VXLAN gateway

A layer 3 VXLAN gateway provides routing between VXLAN segment and any other network segment such as VXLAN, VLAN or L3VPN. The following sections show how to configure an ASR 9000 series router as a Layer 3 VXLAN gateway between a VLAN and a VXLAN segment in different networks.

Prerequisites

The following are the prerequisites to configuring a Cisco ASR 9000 series router as a VXLAN Layer 2 gateway:

- Configure a loopback interface. It serves as a source interface for the local VTEP.
- Configure unicast reachability to remote VTEPs.
- Configure Bidirectional Protocol Independent Multicast (Bidir PIM) or PIM Sparse Mode. For more information, see the Multicast Configuration Guide for Cisco ASR 9000 Series Routers.

Restrictions

Consider the following restrictions while configuring VXLAN:

- You configure VXLAN only on Overlay Transport Virtualization (OTV) and VXLAN UDP ports.
- The source interface can only be a loopback interface.
- You cannot share a VNI or a multicast group or a source interface across multiple NVE interfaces.
- The VNI range and the multicast range both can only be specified contiguously. A non-contiguous range with comma separated values is not supported.
- The VNI to multicast group mapping can be only either 1:1 or N:1. For example,
  - The "member vni 5000 mcast-group 239.1.1.1" command configures a valid 1:1 mapping.
  - The "member vni 5000-5005 mcast-group 239.1.1.1" command configures a valid N:1 mapping.
- When a VNI is configured as a part of a VNI range, it can be modified or deleted only as part of the same range. For example, if the "member vni 5000-5002 mcast-group 239.1.1.1" command is configured, you cannot disassociate just the VNI 5001 from the NVE interface with a "no member vni 5001" command.
- Static MAC configuration is not supported.
- You can configure a maximum of 128k Layer 2 and Layer 3 sub-interfaces per system. The configuration can be a combination of both Layer 2 sub-interfaces and Layer 3 sub-interfaces; or either fully Layer 2 sub-interfaces or Layer 3 sub-interfaces.
  Though the system allows you to configure more than 128k sub-interfaces per system, you cannot use this configuration for services. Though the system displays a warning message on reaching the threshold of 128k sub-interfaces, the configuration is still applied. However, you cannot use this configuration for services.

Creating and configuring the Network Virtualization Endpoint (NVE) interface

Perform this task to create an NVE interface and configure it as a VXLAN Tunnel EndPoint (VTEP) for VxLAN.

SUMMARY STEPS

1. interface nve nve-identifier
2. source-interface loopback loopback-interface-identifier
3. member vni vni_number [-end_vni_range] mcast-group ip_address [-end_ip_address_range]
4. Use the commit or end command.

DETAILED STEPS

Step 1  
interface nve nve-identifier  

Example:

RP/0/RSP0/CPU0:router(config)# interface nve 1  

Creates the NVE interface and enters the NVE interface configuration sub-mode.

Step 2  
source-interface loopback loopback-interface-identifier  

Example:

RP/0/RSP0/CPU0:router(config-if)# source-interface loopback 1
Sets a loopback interface as the source interface for the VTEP.

**Step 3**

```
member vni vni_number [ -end_vni_range ] mcast-group ip_address [ end_ip_address_range ]
```

**Example:**

```
RP/0/RSP0/CPU0:router(config-if)# member vni 1-10 mcast-group 224.2.2.2
```

Associates a single VxLAN or a contiguous range of VxLANs with the NVE interface using their VxLAN Network Identifiers (VNIs) and specifies a multicast address or a contiguous multicast address range associated with these VNIs.

**Note**  The mapping between the VNIs and the multicast groups is either one-to-one or many-to-one.

**Step 4**

Use the `commit` or `end` command.

- **commit** - Saves the configuration changes and remains within the configuration session.
- **end** - Prompts user to take one of these actions:
  - **Yes** - Saves configuration changes and exits the configuration session.
  - **No** - Exits the configuration session without committing the configuration changes.
  - **Cancel** - Remains in the configuration mode, without committing the configuration changes.

---

**Configuring the L3 bridge virtual interface**

Perform this task to configure the IPv4 address for a bridge virtual interface for L3 routing.

**SUMMARY STEPS**

1. `interface BVI BVI-identifier`
2. `ipv4 address ip-address (/prefix | subnet mask)`
3. Use the `commit` or `end` command.

**DETAILED STEPS**

**Step 1**

`interface BVI BVI-identifier`

**Example:**

```
RP/0/RSP0/CPU0:router(config)# interface BVI 1
```

Enters the bridge virtual interface configuration mode.

**Step 2**

`ipv4 address ip-address (/prefix | subnet mask)`

**Example:**

```
RP/0/RSP0/CPU0:router(config-if)# ipv4 address 1.1.1.1 255.0.0.0
```

Sets the IPv4 address for the bridge virtual interface.

**Step 3**

Use the `commit` or `end` command.

- **commit** - Saves the configuration changes and remains within the configuration session.
- **end** - Prompts user to take one of these actions:
• Yes - Saves configuration changes and exits the configuration session.
• No - Exits the configuration session without committing the configuration changes.
• Cancel - Remains in the configuration mode, without committing the configuration changes.

Configuring a bridge domain

Perform this task to configure a bridge domain.

SUMMARY STEPS

1. `l2vpn`
2. `bridge group bridge-group-name`
3. `bridge-domain bridge-domain-name`
4. `member vni vxlan-id`
5. `routed interface BVI BVI-id`
6. Use the `commit` or `end` command.

DETAILED STEPS

Step 1 `l2vpn`
Example:
```
RP/0/RSP0/CPU0:router(config)# l2vpn
```
Enters the l2vpn configuration mode.

Step 2 `bridge group bridge-group-name`
Example:
```
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group bgroup1
```
Enters the bridge group configuration mode.

Step 3 `bridge-domain bridge-domain-name`
Example:
```
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain bdomain1
```
Enters the bridge domain configuration mode.

Step 4 `member vni vxlan-id`
Example:
```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# member vni 10
```
Associates a member VNI with the bridge domain.

Step 5 `routed interface BVI BVI-id`
Example:
```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# routed interface BVI 1
```
Sets the bridge virtual interface for the bridge domain.
Step 6  Use the **commit** or **end** command.

**commit** - Saves the configuration changes and remains within the configuration session.

**end** - Prompts user to take one of these actions:

- **Yes** - Saves configuration changes and exits the configuration session.
- **No** - Exits the configuration session without committing the configuration changes.
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.

Configuration Example for Implementing Layer 3 VXLAN Gateway

The following example shows layer 3 VXLAN gateway configuration on two Provider Edge (PE) routers, R1 and R2, from a sample network topology that has the core network simplified as a bundle link connection between the PE routers.

*Figure 14: Network with Layer 3 VXLAN Gateways*

Configuration at R1:

```plaintext
interface Bundle-Ether10
    ipv4 address 192.168.1.1/24
!
interface Loopback0
    ipv4 address 1.1.1.1/32
!
interface T0/2/0/1
    no shut
!
interface T0/2/0/1.100
    encapsulation dot1q 100
    ipv4 address 19.19.19.1/24
!
interface BVI1
    ipv4 address 100.1.1.1 255.255.255.0
    ipv6 address 100::1/64
!
router ospf underlay
    router-id 1.1.1.1
    area 0
        interface Bundle-Ether10
        interface Loopback0
!
interface nve 1
    member vni 1 mcast-group 224.2.2.2 0.0.0.0
    overlay-encapsulation vxlan
```
source-interface Loopback0
!
router ospf overlay
  area 0
    interface bvi1
    interface T0/2/0/1.100
!
l2vpn
  bridge group vxlan
  bridge-domain vxlan
    routed interface BVI1
    member vni 1
!
multicast-routing
  address-family ipv4
    interface loopback0
      enable
    interface Bundle-Ether10
      enable
!
router pim
  address-family ipv4
    rp-address 1.1.1.1 bidir

Configuration at R2:

interface Bundle-Ether10
  ipv4 address 192.168.1.2/24
!
interface Loopback0
  ipv4 address 2.2.2.2/32
!
interface T0/3/0/23
  no shut
!
interface T0/3/0/23.100
  encapsulation dot1q 100
  ipv4 address 20.20.20.1/24
!
interface BVI1
  ipv4 address 100.1.1.2 255.255.255.0
  ipv6 address 100::2/64
router ospf underlay
  router-id 2.2.2.2
  area 0
    interface Bundle-Ether10
    interface Loopback0
!
interface nve 1
  member vni 1 mcast-group 224.2.2.2 0.0.0.0
  overlay-encapsulation vxlan
  source-interface Loopback0
!
router ospf overlay
  area 0
    interface bvi1
    interface T0/3/0/23.100
!
l2vpn
  bridge group vxlan
  bridge-domain vxlan
    routed interface BVI1
    member vni 1
!
multicast-routing
address-family ipv4
interface loopback0
    enable
interface Bundle-Ether10
    enable
!
router pim
address-family ipv4
    rp-address 1.1.1.1 bidir
Configuration Example for Implementing Layer 3 VXLAN Gateway
CHAPTER 6

Implementing IP in IP Tunnel

This chapter module provides conceptual and configuration information for IP in IP tunnels on Cisco ASR 9000 Series Router.

Note

For a complete description of the IP in IP tunnel commands listed in this chapter, see the VPN and Ethernet Services Command Reference for Cisco ASR 9000 Series Routers. To locate documentation of other commands that appear in this chapter, use the command reference master index, or search online.

Table 3: Feature History for IP in IP tunnel

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release 5.3.1</td>
<td>This feature was introduced on Cisco ASR 9000 Series Router.</td>
</tr>
</tbody>
</table>

• IP in IP Tunneling, on page 127
• Configuring IP in IP Tunnel, on page 128
• IP in IP Tunneling: Examples, on page 129

IP in IP Tunneling

IP in IP tunneling refers to the encapsulation of an IP packet as a payload in another IP packet. ASR9K routers support IP in IP tunnels with all possible combinations of IPv4 and IPv6; that is, IPv4 over IPv4, IPv6 over IPv4, IPv4 over IPv6, and IPv6 over IPv6. For example, an IPv4 over IPv6 refers to an IPv4 packet as a payload encapsulated within an IPv6 packet and routed across an IPv6 network to reach the destination IPv4 network, where it is decapsulated.

IP in IP tunneling does not require any additional header such as a GRE header used in the GRE tunnels. So, IP in IP tunneling is preferred over GRE tunnels if both the networks are IP networks.

Restrictions

The following are not supported in IP in IP tunnels:

• MPLS
• Multicast packets
• Keep-Alive packets
• Path MTU (Maximum Transmission Unit) discovery
• DF (Do not Fragment) bit configuration in IPv6 tunnel mode.

Note If DF bit is configured for the tunnel interface, you cannot enable IPv6 tunnel mode.

# Configuring IP in IP Tunnel

Perform the following steps to configure an IP in IP tunnel.

## SUMMARY STEPS

1. configure
2. interface tunnel-ip tunnel-id
3. {ipv4 | ipv6} address ip-address
4. tunnel mode {ipv4 | ipv6}
5. tunnel source {interface-id | ipv4/v6-address}
6. tunnel destination ipv4/v6-address
7. (Optional) tunnel df-bit {copy | disable}
8. (Optional) tunnel tos tos-value
9. commit

## DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters the Global Configuration mode.</td>
</tr>
<tr>
<td>Example: RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> interface tunnel-ip tunnel-id</td>
<td>Creates a tunnel interface and enters the tunnel configuration sub-mode. The tunnel-id is the numeric identifier for the tunnel interface.</td>
</tr>
<tr>
<td>Example: RP/0/RSP0/CPU0:router(config)# interface tunnel-ip 1</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> {ipv4</td>
<td>ipv6} address ip-address</td>
</tr>
<tr>
<td>Example: RP/0/RSP0/CPU0:router(config-if)# ipv6 address 10::1/64</td>
<td></td>
</tr>
</tbody>
</table>
### Command or Action

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 4</td>
<td>`tunnel mode {ipv4</td>
<td>ipv6}`</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-if)# tunnel mode ipv6</td>
<td></td>
</tr>
<tr>
<td>Step 5</td>
<td>`tunnel source {interface-id</td>
<td>ipv4/v6-address}`</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-if)# tunnel source GigabitEthernet0/0/0/0</td>
<td></td>
</tr>
<tr>
<td>Step 6</td>
<td><code>tunnel destination ipv4/v6-address</code></td>
<td>Specifies the destination IP address for the tunnel. The encapsulated packet uses this IP address as the destination address.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-if)# tunnel destination 100::2</td>
<td></td>
</tr>
<tr>
<td>Step 7</td>
<td>`(Optional) tunnel df-bit {copy</td>
<td>disable}`</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-if)# tunnel df-bit disable</td>
<td></td>
</tr>
<tr>
<td>Step 8</td>
<td><code>(Optional) tunnel tos tos-value</code></td>
<td>Sets the TOS value for the outer IP packet in the tunnel. For details on this <code>tunnel tos</code> command, see the <code>VPN and Ethernet Services Command Reference for Cisco ASR 9000 Series Routers</code>.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-if)# tunnel tos 1</td>
<td></td>
</tr>
<tr>
<td>Step 9</td>
<td>commit</td>
<td></td>
</tr>
</tbody>
</table>

### IP in IP Tunneling: Examples

The following examples provide configurations for an IPv4 or IPv6 tunnel, with the transport VRF as the default VRF for the following simplified network topology.

*Figure 15: IP in IP tunnel network topology*

![IP in IP tunnel network topology](image)

**Configuration example for an IPv4 tunnel**

1. Set the tunnel mode as IPv4.
2. Specify the tunnel source as `GigabitEthernet0/0/0/0`.
3. Specify the tunnel destination as `100::2`.
4. Disable the DF bit for the outer IP packet.
5. Commit the configuration.

---

**Example for an IPv6 tunnel:**

1. Set the tunnel mode as IPv6.
2. Specify the tunnel source as `GigabitEthernet0/0/0/0`.
3. Specify the tunnel destination as `100::2`.
4. Disable the DF bit for the outer IP packet.
5. Commit the configuration.

---

**Note:**

- This is valid only for a tunnel that uses an IPv4 transport network.
- For details on this `tunnel df-bit` command, see the `VPN and Ethernet Services Command Reference for Cisco ASR 9000 Series Routers`.
- For details on this `tunnel tos` command, see the `VPN and Ethernet Services Command Reference for Cisco ASR 9000 Series Routers`.

---

**Example (IPv4 tunnel):**

```plaintext
Step 4
RP/0/RSP0/CPU0:router(config-if)# tunnel mode ipv4

Step 5
RP/0/RSP0/CPU0:router(config-if)# tunnel source GigabitEthernet0/0/0/0

Step 6
RP/0/RSP0/CPU0:router(config-if)# tunnel destination 100::2

Step 7
RP/0/RSP0/CPU0:router(config-if)# tunnel df-bit disable

Step 8
RP/0/RSP0/CPU0:router(config-if)# tunnel tos 1

Step 9
commit
```
### PE1 Router Configuration

```plaintext
interface GigabitEthernet0/0/0
  !! Link between PE1-PE2
  ipv4 address 100.1.1.1/64
!
interface GigabitEthernet0/0/1
  !! Link between CE1-PE1
  ipv4 address 20.1.1.1/24
  ipv6 address 20::1/64
  
interface tunnel-ip 1
  ipv4 address 10.1.1.1/24
  ipv6 address 10::1/64
  tunnel mode ipv4
  tunnel source GigabitEthernet0/0/0
  tunnel destination 100.1.1.2
!

router static
  address-family ipv4 unicast
    30.1.1.0/24 tunnel-ip1
  address-family ipv6 unicast
    30::0/64 tunnel-ip1
!
```

### PE2 Router Configuration

```plaintext
interface GigabitEthernet0/0/0
  !! Link between PE1-PE2
  ipv4 address 100.1.1.2/64
!
interface GigabitEthernet0/0/1
  !! Link between PE2-CE2
  ipv4 address 30.1.1.1/24
  ipv6 address 30::1/64
!
interface tunnel-ip 1
  ipv4 address 10.1.1.1/24
  ipv6 address 10::1/64
  tunnel mode ipv4
  tunnel source GigabitEthernet0/0/0
  tunnel destination 100.1.1.1
!

router static
  address-family ipv4 unicast
    20.1.1.0/24 tunnel-ip1
  address-family ipv6 unicast
    20::0/64 tunnel-ip1
!
```

### CE1 Router Configuration

```plaintext
interface GigabitEthernet0/0/1
  !! Link between CE1-PE1
  ipv4 address 20.1.1.2 255.255.255.0
  ipv6 address 20::2/64
!
router static
  address-family ipv4 unicast
    30.1.1.0/24 20.1.1.1
  address-family ipv6 unicast
    30::0/64 20::1
!
```

### CE2 Router Configuration

```plaintext
interface GigabitEthernet0/0/1
  !! Link between CE2-PE2
  ipv4 address 30.1.1.2 255.255.255.0
  ipv6 address 30::2/64
!
router static
  address-family ipv4 unicast
    20.1.1.0/24 30.1.1.1
  address-family ipv6 unicast
    20::0/64 30::1
!
```

### Configuration example for an IPv6 tunnel

```plaintext
PE1 Router Configuration

interface GigabitEthernet0/0/0
  !! Link between PE1-PE2
  ipv4 address 100.1.1.1/64
!
interface GigabitEthernet0/0/1
  !! Link between CE1-PE1
  ipv4 address 20.1.1.1/24
  ipv6 address 20::1/64
  
interface tunnel-ip 1
  ipv4 address 10.1.1.1/24
  ipv6 address 10::1/64
  tunnel mode ipv4
  tunnel source GigabitEthernet0/0/0
  tunnel destination 100.1.1.2
!

router static
  address-family ipv4 unicast
    30.1.1.0/24 tunnel-ip1
  address-family ipv6 unicast
    30::0/64 tunnel-ip1
!
```

### PE2 Router Configuration

```
interface GigabitEthernet0/0/0
  !! Link between PE1-PE2
  ipv4 address 100.1.1.2/64
!
interface GigabitEthernet0/0/1
  !! Link between PE2-CE2
  ipv4 address 30.1.1.1/24
  ipv6 address 30::1/64
!
interface tunnel-ip 1
  ipv4 address 10.1.1.1/24
  ipv6 address 10::1/64
  tunnel mode ipv4
  tunnel source GigabitEthernet0/0/0
  tunnel destination 100.1.1.1
!

router static
  address-family ipv4 unicast
    20.1.1.0/24 tunnel-ip1
  address-family ipv6 unicast
    20::0/64 tunnel-ip1
!
```

### CE1 Router Configuration

```
interface GigabitEthernet0/0/1
  !! Link between CE1-PE1
  ipv4 address 20.1.1.2 255.255.255.0
  ipv6 address 20::2/64
!
router static
  address-family ipv4 unicast
    30.1.1.0/24 20.1.1.1
  address-family ipv6 unicast
    30::0/64 20::1
!
```

### CE2 Router Configuration

```
interface GigabitEthernet0/0/1
  !! Link between CE2-PE2
  ipv4 address 30.1.1.2 255.255.255.0
  ipv6 address 30::2/64
!
router static
  address-family ipv4 unicast
    20.1.1.0/24 30.1.1.1
  address-family ipv6 unicast
    20::0/64 30::1
!
```
### CE1 Router Configuration

```plaintext
interface GigabitEthernet0/0/0/1
   !! Link between CE1-PE1
   vrf RED
   ipv4 address 20.1.1.2 255.255.255.0
   ipv6 address 20::2/64

router static
   address-family ipv4 unicast
      30.1.1.0/24 20.1.1.1
   address-family ipv6 unicast
      30::0/64 20::1
   !
```

### CE2 Router Configuration

```plaintext
interface GigabitEthernet0/0/0/1
   !! Link between CE2-PE2
   ipv4 address 30.1.1.2 255.255.255.0
   ipv6 address 30::2/64

router static
   address-family ipv4 unicast
      20.1.1.0/24 30.1.1.1
   address-family ipv6 unicast
      20::0/64 30::1
   !
```
Implementing IP in IP Tunnel

IP in IP Tunneling: Examples
CHAPTER 7

Implementing DCI VXLAN Layer 3 Gateway

This chapter module provides conceptual and configuration information for Data Center Interconnect (DCI) VXLAN Layer 3 Gateway on Cisco ASR 9000 Series Router.

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release 5.3.2</td>
<td>This feature was introduced.</td>
</tr>
<tr>
<td>Release 6.1.x</td>
<td>• OpFlex</td>
</tr>
</tbody>
</table>

• Prerequisites for Implementing Data Center Interconnect Layer 3 Gateway, on page 133
• Data Center Interconnect VXLAN Layer 3 Gateway, on page 134
• Configure Data Center Interconnect Router, on page 136
• Example: Data Center Interconnection Layer 3 Gateway Configuration, on page 153
• OpFlex, on page 155
• OpFlex Topology, on page 155
• Restrictions, on page 156
• Configure OpFlex, on page 156
• EVPN Default VRF Route Leaking, on page 168
• EVPN Service VRF Route Leaking, on page 175

Prerequisites for Implementing Data Center Interconnect Layer 3 Gateway

• You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

• You need to have understanding of the following features:
  • VxLAN: For detailed conceptual and configuration information, see the chapters Implementing Layer 2 VxLAN Gateway and Implementing Layer 3 VxLAN Gateway in Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Configuration Guide and Cisco ASR 9000 Series Aggregation Services Router MPLS Layer 3 VPN Configuration Guide.
• MP-BGP: For detailed conceptual and configuration information, see the chapter Implementing BGP in the Cisco ASR 9000 Series Aggregation Services Router Routing Configuration Guide.

• MPLS L3VPN: For detailed conceptual and configuration information, see the Cisco ASR 9000 Series Aggregation Services Router MPLS Layer 3 VPN Configuration Guide.

Data Center Interconnect VXLAN Layer 3 Gateway

The Cisco ASR 9000 Series Router can serve as a Data Center Interconnect (DCI) L3 Gateway to provide IP connectivity between multi-tenant remote Data Center sites. Consider the following network topology, which has two Data Center sites connected through the intermediate Service Provider network. The multi-tenant Data Centers use VXLAN encapsulation to carry separate tenant IP traffic. The VXLAN-enabled Data Center sites use MP-BGP EVPN control plane for distributing both Layer-2 and Layer-3 forwarding information within the site. The RFC 5512 and draft-ietf-bess-evpn-inter-subnet-forwarding-00 define how MP-BGP Network Layer Reachability Information (NRLI) carries VXLAN encapsulation as well as L2/L3 forwarding information details to provide an integrated routing and bridging solution within the Data Center site. The Cisco ASR 9000 Series Router uses MPLS L3VPN application service over the Service Provider network to provide L3 connectivity between the two Data Center sites.

Note
DCI Gateway does not provide layer 2 inter-connectivity across Data Centers.

Figure 16: Data Center Interconnect Layer 3 Gateway

The Cisco ASR 9000 Series Router functions as a Data Center Interconnect (DCI) gateway by intermediating between the two MP-BGP control planes, one on the Data Center site and the other on the MPLS L3VPN network. To enable this exchange of forwarding information between the two MP-BGP control planes, the DCI router has a VRF instance configured with two sets of import and export route-targets. One set of import/export route targets is associated with the Data Center BGP neighbor router that uses MP-BGP EVPN with route type 5 NRLI to exchange VXLAN encapsulation and L3 routing information with the DCI router. The other set of import/export route-targets is associated with the L3VPN BGP neighbor in the service provider network that uses VPNv4 or VPNv6 address-family to exchange L3 and MPLS information. The DCI router exchanges the IP prefixes in VRF instance as L3VPN NLRIs with the L3VPN BGP neighbor and as EVPN NLRIs with the EVPN BGP neighbor and thus, effectively stitches these two sets of route targets. This enables the DCI router to convert the received Data Center EVPN forwarding information into VPNv4 or VPNv6 routes that, in turn, is to be forwarded to the remote DCI router and vice versa. The remote DCI router connected to the remote Data Center performs same functions. This enables L3 connectivity between two hosts located across remote Data Center sites. The DCI Gateway enables tenant Layer 3 data traffic movement across remote Data Centers by stitching the per-tenant VXLAN encapsulation in the DCI Gateway router to the per-tenant MPLS encapsulation in L3VPN service provider network.
The DCI L3 Gateway can be configured on the Provider Edge (PE) router or a Data Center router that connects to the WAN. The WAN network can be any VRF-deployed network configured with any control plane protocol. For example, a VRF-lite WAN network configured with an IGP.

In a DCI deployment, for route reoriginate with stitching-rt for a particular VRF, using the same Route Distinguisher (RD) between ASR9K DCI and MPLS-VPN PE or same RD between ASR9K DCI and VxLAN Top of Rack (ToR) is not supported.

Route Targets

For each VRF on the DCI router, there are two sets of manually configured import and export route-targets. One set of import and export route-targets is associated with the Data Center BGP neighbor that uses EVPN address-family to exchange L3 information; the other set of import and export route-targets is associated with the L3VPN BGP neighbor that use VPNv4 or VPNv6 unicast address-family to exchange L3 information. This separation of route targets (RTs) enables the two sets of RTs to be independently configured. The DCI router effectively stitches the two set of RTs. The RTs associated with the EVPN BGP neighbor are labelled as stitching RTs. The RTs associated with the L3VPN BGP neighbor are normal RTs.

Route Re-origination

Consider the case of control plane information propagation by the DCI from the L3VPN side to the Data Center side. Here, instead of advertising the remote Data Center's original BGP EVPN routes, you can configure the DCI router to advertise to its BGP EVPN neighbor the routes that are re-originated after importing them from the L3VPN BGP neighbor. For this case of VPNv4 or VPNv6 routes being propagated to the BGP EVPN neighbors (Data Center neighbors), re-originating the routes refers to replacing the normal route-targets with the local route-target values associated with the BGP EVPN neighbors. The converse holds true for the routing information traffic propagation from the BGP EVPN control plane to BGP L3VPN control plane. You can configure this re-origination by using the re-originate keyword in the `import re-originate` command.

Configuring this command, by default, also enables advertisement of L2VPN EVPN prefixes to the EVPN BGP neighbors. You can suppress native L2VPN EVPN address-family NLRI advertisements towards the EVPN Neighbor using the `advertise l2vpn evpn disable` command under the EVPN BGP address-family configuration mode.

Route Address-Family and Encoded Address-Family

When an address-family is configured for a BGP neighbor, it means that the specified address-family routes encoded with the NLRI for that address-family is advertised to the neighbor. This does not hold for data center BGP neighbors because they use only EVPN address-family. Here, BGP neighbors advertise VPNv4 or VPNv6 unicast routes using the EVPN NLRI encoding. Thus, here the encoded address-family and route address family can be possibly different. You can advertise the VPNv4 or VPNv6 address-family using the `advertise vpnv4 unicast` or `advertise vpnv6 unicast` command. For example, a EVPN address-family BGP neighbor configured with the `advertise vpnv4 unicast` command sends VPNv4 unicast routes in an EVPN encoded NLRI.
Local VPNv4 or VPNv6 Routes Advertisement

On the DCI router, the locally sourced VPNv4 or VPNv6 routes can be advertised to the BGP EVPN neighbors with the normal route targets (RTs) configured for the VRF or the stitching RTs associated with the BGP EVPN neighbors. By default, these routes are advertised with the normal route targets. You can configure these local VPNv4 or VPNv6 route advertisements to be advertised with stitching RTs to the BGP EVPN neighbors by using the `advertise vpnv4 unicast local stitching-rt` or `advertise vpnv6 unicast local stitching-rt` command as required.

Data Center VXLAN with Support for MP-BGP

The Data Center VXLAN uses MP-BGP for control-plane learning of end-host Layer 2 and Layer 3 reachability information. The DCI router is configured with a VXLAN Tunnel EndPoint (VTEP). For VTEP configuration details, see the chapter Implementing Layer 3 VXLAN Gateway. You also need to run the `host-reachability protocol bgp` command to specify that control-plane learning within Data center site is through BGP routing protocol.

The DCI Gateway router and the EVPN BGP neighbor (Data Center BGP neighbor) exchange BGP EVPN NLRIs of route type 5 that carry L3 routing information and associated VXLAN encapsulation information. Some of the VXLAN information is carried in the EVPN NLRI and the rest is carried in RFC 5512 Tunnel Type Encapsulation EXTCOMM and Router MAC EXTCOMM defined in `draft-ietf-bess-evpn-inter-subnet-forwarding-00`. BGP downloads VXLAN encapsulation as `RIB remote next hop opaque` attribute to L3RIB.

Default-Originate Forwarding to BGP EVPN Neighbor

Instead of advertising the specific networks available in the remote Data Center, you can configure the DCI gateway to advertise a default route to the directly connected Data Center neighbor. To send the default route for a VRF instance to the Data Center BGP EVPN neighbor, the VPN default-originate information that is typically forwarded to the L3VPN BGP neighbor, is also configured to be forwarded to the BGP EVPN neighbor in the Data Center. To do so, you need to configure `allow vpn default-originate` command in the BGP VRF configuration mode and also configure `default-originate` command under EVPN BGP neighbor in L2VPN EVPN address-family configuration mode. This configures BGP to forward only one default route information for a VRF instance from the DCI Gateway to the BGP neighbor that has L2VPN EVPN address-family. This default route information is encoded in the EVPN "IP Prefix Route" NLRI.

With the advertisement of a default route to the connected Data Center, the DCI Gateway should not advertise specific prefixes of the remote Data Center to the BGP EVPN neighbor. To prevent forwarding of VRF prefixes, you need to configure the DCI gateway with a EVPN BGP neighbor policy that drops forwarding of all prefixes.

Configure Data Center Interconnect Router

This section describes tasks to configure the Data Center Interconnect (DCI) router. Perform the following tasks to complete the configuration:
Configure VRF and route targets import/export rules

Perform the following steps to configure VRF and define route targets to be used for import and export of forwarding information.

**SUMMARY STEPS**

1. configure
2. vrf **vrf-name**
3. address-family \{ ipv4 \ipv6 \} unicast
4. import route-target **route-target-id**
5. export route-target **route-target-id**
6. import route-target **route-target-id** stitching
7. export route-target **route-target-id** stitching
8. commit

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Defines a VPN routing and forwarding (VRF) instance and enters VRF configuration mode.</td>
</tr>
</tbody>
</table>
| **Step 2** vrf **vrf-name**  
**Example:**  
RP/0/RSP0/CPU0:router(config)# vrf data-center-10 | Specifies either the IPv4 or IPv6 address family for the VRF instance and enters address family configuration submode. |
| **Step 3** address-family \{ ipv4 \ipv6 \} unicast  
**Example:**  
RP/0/RSP0/CPU0:router(config-vrf-af)# address-family ipv4 unicast | Configures importing of routes to the VRF from the L3VPN BGP NLRIs that have the matching route-target value. |
| **Step 4** import route-target **route-target-id**  
**Example:**  
RP/0/RSP0/CPU0:router(config-vrf-af)# import route-target 1:1 | Configures exporting of routes from the VRF to the L3VPN BGP NLRIs and assigns the specified route-target identifiers to the L3VPN BGP NLRIs. |
| **Step 5** export route-target **route-target-id**  
**Example:**  
RP/0/RSP0/CPU0:router(config-vrf-af)# export route-target 1:2 | Configures importing of routes from the EVVPN BGP NLRI that have the matching route-target value. |
| **Step 6** import route-target **route-target-id** stitching  
**Example:**  
RP/0/RSP0/CPU0:router(config-vrf-af)# import route-target 10:1 stitching | |

Implementing DCI VXLAN Layer 3 Gateway

Configure VRF and route targets import/export rules
### Configure Bridge Domain for DCI Gateway

Perform the following steps to configure the bridge domain on the DCI Gateway.

**SUMMARY STEPS**

1. configure
2. interface bvi bvi-id
3. vrf vrf-id
4. ipv4 address ip-address subnet-mask
5. exit
6. l2vpn
7. bridge group bridge-group-identifier
8. bridge-domain bridge-group-identifier
9. routed interface interface-identifier
10. member vni vni-id
11. commit

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td>configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td>interface bvi bvi-id</td>
<td>Creates a BVI (Bridge Virtual Interface) with the user-defined identifier and enters the interface configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config)# interface bvi 1</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td>vrf vrf-id</td>
<td>Associates a VRF (Virtual Routing and Forwarding) instance to the BVI.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-if)# vrf cust1</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td>ipv4 address ip-address subnet-mask</td>
<td>Creates an IPv4 address for the BVI.</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-if)# ipv4 address 40.1.1.1 255.255.255.255</td>
<td></td>
</tr>
</tbody>
</table>
### Purpose

**Command or Action**

**Step 5**

**Purpose:** Exits the interface configuration mode and returns to Global Configuration mode.

**Example:**

```
RP/0/RSP0/CPU0:router(config)# exit
```

**Step 6**

**Purpose:** Enters the L2VPN configuration mode.

**Example:**

```
RP/0/RSP0/CPU0:router(config)# l2vpn
```

**Step 7**

**Purpose:** Configures a bridge group and enters the Bridge Group configuration mode.

**Example:**

```
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group bg1
```

**Step 8**

**Purpose:** Configures a bridge domain and enters Bridge Domain configuration mode.

**Example:**

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain bd1
```

**Step 9**

**Purpose:** Configures a BVI interface as a routing interface for the bridge domain.

**Example:**

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# routed interface BVI1
```

**Step 10**

**Purpose:** Assigns the bridge domain to a VxLAN segment.

**Example:**

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# member vni 5001
```

**Step 11**

**Purpose:**

### Configure VTEP

Perform the following steps to configure VTEP (VxLAN Terminal EndPoint) on the DCI Gateway.

**SUMMARY STEPS**

1. `configure`
2. `interface loopback loopback-id`
3. `ipv4 address ip-address subnet-mask`
4. `exit`
5. `interface nve nve-id`
### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td>configure</td>
</tr>
</tbody>
</table>
| **Step 2** | interface loopback  loopback-id  
Example:  
RP/0/RSP0/CPU0:router(config)# interface loopback 0 |
| | Creates a loopback interface with the user-defined loopback identifier and enters the interface configuration mode. |
| **Step 3** | ipv4 address  ip-address subnet-mask  
Example:  
RP/0/RSP0/CPU0:router(config-if)# ipv4 address 40.1.1.1 255.255.255.255 |
| | Creates an IPv4 address for the loopback interface. |
| **Step 4** | exit  
Example:  
RP/0/RSP0/CPU0:router(config)# exit |
| | Exits the interface configuration mode and returns to Global Configuration mode. |
| **Step 5** | interface nve  nve-id  
Example:  
RP/0/RSP0/CPU0:router(config)# interface nve 1 |
| | Creates an NVE (Network Virtualization Endpoint) interface with the identifier and enters the interface configuration mode. |
| **Step 6** | source interface loopback  loopback-interface-identifier  
Example:  
RP/0/RSP0/CPU0:router(config-if)# source interface loopback 0 |
| | Configures the loopback interface IP address as the source IP address for the NVE interface. |
| **Step 7** | member vni  vni-id  
Example:  
RP/0/RSP0/CPU0:router(config-if)# member vni 5001 |
| | Configures the NVE interface under a VxLAN with VNI (VxLAN Network Identifier) 5001 and enters the NVE VNI configuration mode. |
| **Step 8** | vrf  vrf-id  
Example:  
RP/0/RSP0/CPU0:router(config-nve-vni)# vrf cust1 |
| | Associates a VRF (Virtual Routing and Forwarding) instance to the VxLAN segment. |
Configure EVVPN BGP neighbor and route advertisements

Perform this task on the DCI router to configure BGP neighbor relationship and route advertisements with the EVPN BGP neighbor.

SUMMARY STEPS

1. configure
2. router bgp asn-id
3. address-family l2vpn evpn
4. exit
5. neighbor neighbor-ip-address
6. remote-as remote-as-id
7. address-family l2vpn evpn
8. (Optional) default-originate
9. import stitching-rt reoriginate
10. advertise { vpnv4 | vpnv6 } unicast re-originated
11. advertise { vpnv4 | vpnv6 } unicast local stitching-rt
12. (Optional) advertise l2vpn evpn disable
13. commit

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1 configure</td>
<td></td>
</tr>
<tr>
<td>Step 2 router bgp asn-id</td>
<td>Specifies the BGP AS number and enters the BGP configuration mode, allowing you to configure the BGP routing process.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config)# router bgp 100</td>
</tr>
<tr>
<td>Step 3 address-family l2vpn evpn</td>
<td>Enables EVVPN address family globally under BGP routing process and enters EVVPN address family configuration submode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router(config-bgp)# address-family l2vpn evpn</td>
</tr>
<tr>
<td>Step</td>
<td>Command or Action</td>
</tr>
<tr>
<td>------</td>
<td>------------------</td>
</tr>
<tr>
<td><strong>4</strong></td>
<td>exit</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CP00:router(config-bgp-af)# exit</td>
</tr>
<tr>
<td><strong>5</strong></td>
<td>neighbor  neighbor-ip-address</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CP00:router(config-bgp)# neighbor 1.1.1.1</td>
</tr>
<tr>
<td><strong>6</strong></td>
<td>remote-as  remote-as-id</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CP00:router(config-bgp-nbr)# remote-as 100</td>
</tr>
<tr>
<td><strong>7</strong></td>
<td>address-family l2vpn evpn</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CP00:router(config-bgp-nbr)# address-family family l2vpn evpn</td>
</tr>
<tr>
<td><strong>8</strong></td>
<td>(Optional) default-originate</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CP00:router(config-bgp-nbr)# default-originate</td>
</tr>
<tr>
<td><strong>9</strong></td>
<td>import stitching-rt reoriginate</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CP00:router(config-bgp-nbr-af)# import stitching-rt reoriginate</td>
</tr>
<tr>
<td><strong>10</strong></td>
<td>advertise { vvpn4</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CP00:router(config-bgp-nbr-af)# advertise vvpn4 unicast re-originated</td>
</tr>
<tr>
<td><strong>11</strong></td>
<td>advertise { vvpn4</td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CP00:router(config-bgp-nbr-af)# advertise vvpn4 unicast local stitching-rt</td>
</tr>
</tbody>
</table>
Configure L3VPN BGP neighbor relationship and route advertisements

Perform the following steps to configure BGP neighbor relationship and route advertisements with the L3VPN BGP neighbor.

SUMMARY STEPS

1. configure
2. router bgp  asn-id
3. address-family { vpnv4 | vpnv6 }
4. exit
5. neighbor neighbor-ipv4/6-address
6. remote-as remote-as-id
7. address-family { vpnv4 | vpnv6 }
8. import reoriginate stitching-rt
9. advertise { vpnv4 | vpnv6 } unicast re-originated
10. commit

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td>configure</td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td>router bgp  asn-id</td>
</tr>
<tr>
<td>Example: RP/0/RSP0/CPU0:router(config)# router bgp 100</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Specifies the BGP AS number and enters the BGP configuration mode, allowing you to configure the BGP routing process.</td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td>address-family { vpnv4</td>
</tr>
<tr>
<td>Example: RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# address-family vpnv4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Configures VPNv4 or VPNv6 address family for the global BGP routing process and enters VPNv4 or VPNv6 address family configuration mode.</td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td>exit</td>
</tr>
<tr>
<td>Example: RP/0/RSP0/CPU0:router(config-bgp-af)# exit</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Exits the VPNv4 or VPNv6 address family configuration and returns to the BGP router configuration mode.</td>
</tr>
<tr>
<td>Step</td>
<td>Command or Action</td>
</tr>
<tr>
<td>------</td>
<td>------------------</td>
</tr>
<tr>
<td>5</td>
<td>neighbor neighbor-ipv4/6-address</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config-bgp)# neighbor 1.1.1.1</td>
</tr>
<tr>
<td>6</td>
<td>remote-as remote-as-id</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 100</td>
</tr>
<tr>
<td>7</td>
<td>address-family { vpnv4</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family vpnv4</td>
</tr>
<tr>
<td>8</td>
<td>import reoriginate stitching-rt</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# import reoriginate stitching-rt</td>
</tr>
<tr>
<td>9</td>
<td>advertise { vpnv4</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# advertise vpnv4 unicast re-originated</td>
</tr>
<tr>
<td>10</td>
<td>commit</td>
</tr>
</tbody>
</table>

**Verification of Data Center Gateway Configuration**

These show commands can be used to verify the DCI Gateway configurations:

**SUMMARY STEPS**

1. show bgp l2vpn evpn
2. show bgp l2vpn evpn rd rd-id l2vpn-evpn-prefix detail
3. show bgp l2vpn evpn neighbors neighbor-ip-address detail
4. show bgp sessions
5. show bgp vpnv4 unicast
6. show bgp vpnv4 unicast rd-id vpnv4-prefix detail
7. show bgp vrf poo
8. show bgp vrf poo ipv4 unicast 100.1.1.1/32 detail
9. show bgp vpnv4 unicast update-group
10. show bgp l2vpn evpn update-group

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> show bgp l2vpn evpn</td>
<td>Displays a summary of the BGP information advertised from the EVPN BGP neighbor.</td>
</tr>
</tbody>
</table>

**Example:**

```
RP/0/RSP0/CPU0:router# show bgp l2vpn evpn
Fri Aug 21 00:24:10.773 PDT
BGP router identifier 30.30.30.30, local AS number 100
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0x0   RD version: 0
BGP main routing table version 16
BGP NSR Initial initsync version 1 (Reached)
BGP NSR/ISSU Sync-Group versions 16/0
BGP scan interval 60 secs

Status codes: s suppressed, d damped, h history,
  * valid, > best
  i - internal, r RIB-failure, S stale, N Nexthop-discard
Origin codes: i - IGP, e - EGP, ? - incomplete
Network   Next Hop   Metric   LocPrf Weight Path
Route Distinguisher: 100:1
*>i[2][10000][48][0226.51bd.c81c][32][200::1001]/232
   11.0.0.1
  100 0 i
*>i[2][10000][48][0226.51bd.c81c][32][200::1001]/232
   11.0.0.1
  100 0 i
*>i[2][10000][48][0226.51bd.c81c][32][200.1.1.1]/136
   11.0.0.1
  100 0 i
*>i[2][10000][48][0226.51bd.c81c][32][200.1.1.2]/136
   11.0.0.1
  100 0 i
*>i[5][4231][32][100.1.1.1]/80
   11.0.0.1
  100 0 i
*>i[5][4231][32][100.1.1.2]/80
   11.0.0.1
  100 0 i
*>i[5][4231][112][fec0::1001]/176
    11.0.0.1
  100 0 i
*>i[5][4232][112][fec0::1:1001]/176
    11.0.0.1
  100 0 i
Processed 8 prefixes, 8 paths
```

MPLS Layer 3 VPN Configuration Guide for Cisco ASR 9000 Series Routers, IOS XR Release 6.3.x
### Verification of Data Center Gateway Configuration

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 2</strong>&lt;br&gt;<code>show bgp l2vpn evpn rd rd-id l2vpn-evpn-prefix detail</code></td>
<td>Displays detailed information for a specific prefix advertised from an EVPN BGP neighbor.</td>
</tr>
</tbody>
</table>

**Example:**

```
RP/0/RSP0/CPU0:router# show bgp l2vpn evpn rd 100:1 [5][4231][112][fec0::1001]/176 detail
Fri Aug 21 00:34:43.747 PDT
BGP routing table entry for [5][4231][112][fec0::1001]/176, Route Distinguisher: 100:1
Versions:
  Process  brRIB/RIB  SendTblVer
  Speaker  5  5
Flags: 0x04040001+0x00000000;
Last Modified: Aug 21 00:16:58.000 for 00:17:46
Paths: (1 available, best #1)
  Not advertised to any peer
  Path #1: Received by speaker 0
Flags: 0x4000600025060005, import: 0x3f
  Not advertised to any peer
Local 11.0.0.1 (metric 2) from 20.0.0.1 (11.0.0.1)
  Received Label 16001
  Origin IGP, localpref 100, valid, internal,
  best, group-best, import-candidate, reoriginate,
  not-in-vrf
  Received Path ID 0, Local Path ID 1, version 5
  Extended community: Flags 0x2: Encapsulation
  Type:8 Router MAC:aabb.ccdd.eeff RT:65540:1
  RT:40.40.40.40.1 RT:100:1
  Originator: 11.0.0.1, Cluster list:
  20.20.20.20
  EVPN ESI: ffff.ffff.ffff.ffff.ff01, Gateway Address: fec0::254
```

| **Step 3**<br>`show bgp l2vpn evpn neighbors neighbor-ip-address detail` | Displays a detailed information of the specified L2VPN EVPN BGP neighbor. |

**Example:**

```
RP/0/RSP0/CPU0:router# show bgp l2vpn evpn neighbors 20.0.0.1 detail
Fri Aug 21 00:25:37.383 PDT
BGP neighbor is 20.0.0.1
Remote AS 100, local AS 100, internal link
Remote router ID 20.20.20.20
BGP state = Established, up for 00:08:58
NSR State: NSR Ready
Last read 00:00:34, Last read before reset 00:00:00
  Hold time is 180, keepalive interval is 60 seconds
  Configured hold time: 180, keepalive: 60, min acceptable hold time: 3
  Last write 00:00:36, attempted 19, written 19
```
Purpose
Command or Action

Second last write 00:01:36, attempted 143, written 143
Last write before reset 00:00:00, attempted 0, written 0
Second last write before reset 00:00:00, attempted 0, written 0
Last write pulse rcvd Aug 21 00:25:03.667 last full not set pulse count 33
Last write pulse rcvd before reset 00:00:00
Socket not armed for io, armed for read, armed for write
Last write thread event before reset 00:00:00, second last 00:00:00
Last KA expiry before reset 00:00:00, second last 00:00:00
Last KA error before reset 00:00:00, second last 00:00:00
Last KA start before reset 00:00:00, second last 00:00:00
Precedence: internet
Non-stop routing is enabled
Entered Neighbor NSR TCP mode:
  TCP Initial Sync : Aug 21 00:18:07.291
  TCP Initial Sync Phase Two : Aug 21 00:18:07.319
  TCP Initial Sync Done : Aug 21 00:18:08.334
Multi-protocol capability received
Neighbor capabilities: Adv Rcvd
  Route refresh: Yes
  4-byte AS: Yes
Yes
Address family VPNv4 Unicast: Yes
Yes
Address family VPNv6 Unicast: Yes
No
Address family L2VPN EVPN: Yes
Yes
Message stats:
  InQ depth: 0, OutQ depth: 0
  Last_Sent Rcvd Sent
  Last_Rcvd
  Open: Aug 21 00:16:38.087 1
  Aug 21 00:16:40.123 1
  Notification: *** 0
  Update: Aug 21 00:24:01.421 9
  Aug 21 00:24:03.652 13
  Keepalive: Aug 21 00:25:01.434 8
  Aug 21 00:25:03.667 9
  Route_Refresh: Aug 21 00:24:01.377 3
  *** 0
  Total: 21
Minimum time between advertisement runs is 0 secs
Inbound message logging enabled, 3 messages buffered
Outbound message logging enabled, 3 messages

Verifying of Data Center Gateway Configuration

MPLS Layer 3 VPN Configuration Guide for Cisco ASR 9000 Series Routers, IOS XR Release 6.3.x
<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>buffered</td>
<td></td>
</tr>
<tr>
<td>For Address Family: VPNv4 Unicast</td>
<td></td>
</tr>
<tr>
<td>BGP neighbor version 35</td>
<td></td>
</tr>
<tr>
<td>Update group: 0.3 Filter-group: 0.1 No Refresh request being processed</td>
<td></td>
</tr>
<tr>
<td>Advertise Reorigination Enabled</td>
<td></td>
</tr>
<tr>
<td>Advertise AFI EoR can be sent</td>
<td></td>
</tr>
<tr>
<td>Route refresh request: received 0, sent 0</td>
<td></td>
</tr>
<tr>
<td>0 accepted prefixes, 0 are bestpaths</td>
<td></td>
</tr>
<tr>
<td>Cumulative no. of prefixes denied: 0.</td>
<td></td>
</tr>
<tr>
<td>Prefix advertised 4, suppressed 0, withdrawn 0</td>
<td></td>
</tr>
<tr>
<td>Maximum prefixes allowed 2097152</td>
<td></td>
</tr>
<tr>
<td>Threshold for warning message 75%, restart interval 0 min</td>
<td></td>
</tr>
<tr>
<td>AIGP is enabled</td>
<td></td>
</tr>
<tr>
<td>An EoR was not received during read-only mode</td>
<td></td>
</tr>
<tr>
<td>Last ack version 35, Last synced ack version 35</td>
<td></td>
</tr>
<tr>
<td>Outstanding version objects: current 0, max 1</td>
<td></td>
</tr>
<tr>
<td>Additional-paths operation: None</td>
<td></td>
</tr>
<tr>
<td>Send Multicast Attributes</td>
<td></td>
</tr>
<tr>
<td>For Address Family: VPNv6 Unicast</td>
<td></td>
</tr>
<tr>
<td>BGP neighbor version 29</td>
<td></td>
</tr>
<tr>
<td>Update group: 0.3 Filter-group: 0.1 No Refresh request being processed</td>
<td></td>
</tr>
<tr>
<td>Advertise Reorigination Enabled</td>
<td></td>
</tr>
<tr>
<td>Advertise AFI EoR can be sent</td>
<td></td>
</tr>
<tr>
<td>Route refresh request: received 0, sent 0</td>
<td></td>
</tr>
<tr>
<td>0 accepted prefixes, 0 are bestpaths</td>
<td></td>
</tr>
<tr>
<td>Cumulative no. of prefixes denied: 0.</td>
<td></td>
</tr>
<tr>
<td>Prefix advertised 0, suppressed 0, withdrawn 0</td>
<td></td>
</tr>
<tr>
<td>Maximum prefixes allowed 1048576</td>
<td></td>
</tr>
<tr>
<td>Threshold for warning message 75%, restart interval 0 min</td>
<td></td>
</tr>
<tr>
<td>AIGP is enabled</td>
<td></td>
</tr>
<tr>
<td>An EoR was not received during read-only mode</td>
<td></td>
</tr>
<tr>
<td>Last ack version 29, Last synced ack version 29</td>
<td></td>
</tr>
<tr>
<td>Outstanding version objects: current 0, max 0</td>
<td></td>
</tr>
<tr>
<td>Additional-paths operation: None</td>
<td></td>
</tr>
<tr>
<td>Send Multicast Attributes</td>
<td></td>
</tr>
<tr>
<td>Advertise VPNv4 routes enabled with Reoriginate,Local with stitching-RT option</td>
<td></td>
</tr>
<tr>
<td>For Address Family: L2VPN EVPN</td>
<td></td>
</tr>
<tr>
<td>BGP neighbor version 18</td>
<td></td>
</tr>
<tr>
<td>Update group: 0.2 Filter-group: 0.1 No Refresh request being processed</td>
<td></td>
</tr>
<tr>
<td>Route refresh request: received 0, sent 3</td>
<td></td>
</tr>
<tr>
<td>8 accepted prefixes, 8 are bestpaths</td>
<td></td>
</tr>
<tr>
<td>Cumulative no. of prefixes denied: 0.</td>
<td></td>
</tr>
<tr>
<td>Prefix advertised 4, suppressed 0, withdrawn 6</td>
<td></td>
</tr>
<tr>
<td>Maximum prefixes allowed 2097152</td>
<td></td>
</tr>
<tr>
<td>Threshold for warning message 75%, restart interval 0 min</td>
<td></td>
</tr>
<tr>
<td>AIGP is enabled</td>
<td></td>
</tr>
<tr>
<td>An EoR was received during read-only mode</td>
<td></td>
</tr>
<tr>
<td>Last ack version 18, Last synced ack version 18</td>
<td></td>
</tr>
<tr>
<td>Outstanding version objects: current 0, max 2</td>
<td></td>
</tr>
</tbody>
</table>
Purpose

Command or Action

| Additional-paths operation: None |
| Send Multicast Attributes |
| Advertise VPNv4 routes enabled with Reoriginate, option |
| Advertise VPNv6 routes is enabled with Reoriginate, option |
| Import Stitching is enabled for this neighbor address-family |
| Import Reoriginate is enabled for this neighbor address-family |
| Connections established 1; dropped 0 |
| Local host: 30.0.0.1, Local port: 59405, IF Handle: 0x00000000 |
| Foreign host: 20.0.0.1, Foreign port: 179 |
| Last reset 00:00:00 |

Step 4

**show bgp sessions**

Example:

```
RP/0/RSP0/CPU0:router# show bgp sessions
Fri Aug 21 00:25:57.216 PDT

Neighbor   VRF     Spk AS
InQ OutQ  NBRState  NSRState
20.0.0.1 default 0 100
   0 0 Established NSR Ready[PP]
32.0.0.2 default 0 200
   0 0 Established NSR Ready
```

Displays current BGP sessions.

Step 5

**show bgp vpnv4 unicast**

Example:

```
RP/0/RSP0/CPU0:router# show bgp vpnv4 unicast
Fri Aug 21 00:28:41.253 PDT
BGP router identifier 30.30.30.30, local AS number 100
BGP generic scan interval 60 secs
Non-stop routing is enabled
BGP table state: Active
Table ID: 0x0 RD version: 0
BGP main routing table version 39
BGP NSR Initial initsync version 4 (Reached)
BGP NSR/ISSU Sync-Group versions 39/0
BGP scan interval 60 secs

Status codes: s suppressed, d damped, h history,
   * valid, > best
   i - internal, r RIB-failure, S stale, N Nexthop-discard
Origin codes: i - IGP, e - EGP, ? - incomplete
Network Next Hop Metric
LocPrf Weight Path
Route Distinguisher: 1:1
*> 1.1.1.0/24 32.0.0.2
```

Displays summary information of unicast BGP prefixes. The **show bgp vpnv6 unicast** provides similar output information for IPv6 prefixes.
<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>show bgp vpnv4 unicast rd-id vpnv4-prefix detail</td>
<td>Displays detailed information of specified VPNv4 prefixes. The <code>show bgp vpnv6 unicast</code> provides similar output information for VPNv6 prefixes.</td>
</tr>
<tr>
<td>show bgp vrf foo</td>
<td>Displays summary information of prefixes in the specified VRF instance BGP table.</td>
</tr>
</tbody>
</table>

### Step 6

**Example:**

```
RP/0/RSP0/CPU0:router# show bgp vpnv4 unicast rd 30.30.30.0 1.1.1.0/24 detail
```

Fri Aug 21 00:28:57.824 PDT
BGP routing table entry for 1.1.1.0/24, Route Distinguisher: 30.30.30.0
Variations:
- Process bRIB/RIB SendTblVer
- Speaker 26 26
- Flags: 0x04103001+0x00000000;
- Last Modified: Aug 21 00:24:01.000 for 00:04:58
- Paths: (1 available, best #1)
- Advertised to peers (in unique update groups):
  - 20.0.0.1
- Path #1: Received by speaker 0
- Flags: 0x4103001+0x00000000, import: 0x80
- Advertised to peers (in unique update groups):
  - 20.0.0.1
  - 32.0.0.2 from 32.0.0.2 (40.40.40.40)
  - Received Label 24001
  - Origin IGP, localpref 100, valid, external, best, group-best, import-candidate, imported, reoriginated with stitching-rt
  - Received Path ID 0, Local Path ID 1, version 26
  - Extended community: RT:100:2
- Source AFI: VPNv4 Unicast, Source VRF: default, Source Route Distinguisher: 1:1

### Step 7

**Example:**

```
RP/0/RSP0/CPU0:router# show bgp vrf foo
```

Fri Aug 21 00:24:36.523 PDT
### Command or Action

- BGP VRF foo, state: Active
- BGP Route Distinguisher: 30.30.30.30
- VRF ID: 0x60000002
- BGP router identifier 30.30.30.30, local AS number 100
- Non-stop routing is enabled
- BGP table state: Active
- Table ID: 0xe0000011  RD version: 3
- BGP main routing table version 35
- BGP NSR Initial initSync version 4 ( Reached)
- BGP NSR/ISSU Sync-Group versions 31/0
- Status codes: s suppressed, d damped, h history, * valid, > best
  
  - i - internal, r RIB-failure, S stale, N Nexthop-discard
- Origin codes: i - IGP, e - EGP, ? - incomplete

### Purpose

Displays detailed information for a specified prefix in the specified VRF instance BGP table.

#### Step 8

**show bgp vrf foo ipv4 unicast 100.1.1.1/32 detail**

**Example:**

```
RP/0/RSP0/CPU0:router# show bgp vrf foo ipv4 unicast 100.1.1.1/32 detail

Mon Dec 8 23:24:50.243 PST
BGP routing table entry for 100.1.1.1/32, Route Distinguisher: 30.30.30.30:0
Versions:
  Process   bRIB/RIB SendTblVer
  Speaker  43  43
     Local Label: 24001 (with rewrite);
      Flags: 0x05081001+0x00000200;
  Last Modified: Dec 8 18:04:21.000 for 05:20:30
  Paths: (1 available, best #1)
     Advertised to PE peers (in unique update groups):
       32.0.0.2
     Path #1: Received by speaker 0
      Flags: 0x400061000d060005, import: 0x80
     Advertised to PE peers (in unique update groups):
```

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>show bgp vpnv4 unicast update-group</code></td>
<td>Displays update-group details for BGP VPNv4 unicast address-family.</td>
</tr>
</tbody>
</table>

**Example:**

```plaintext
RP/0/RSP0/CPU0:router# show bgp vpnv4 unicast update-group
Fri Aug 21 00:27:57.910 PDT

Update group for VPNv4 Unicast, index 0.1:
Attributes:
- Outbound policy: pass
- First neighbor AS: 200
- Send communities
- Send GSHUT community if originated
- Send extended communities
- 4-byte AS capable
- Send Re-originated VPN routes
- Send multicast attributes
- Minimum advertisement interval: 30 secs
- Update group desynchronized: 0
- Sub-groups merged: 0
- Number of refresh subgroups: 0
- Messages formatted: 8, replicated: 8
- All neighbors are assigned to sub-group(s)
- Neighbors in sub-group: 0.2, Filter-Groups num: 1
  - Neighbors in filter-group: 0.2(RT num: 0)
  - 32.0.0.2

Update group for VPNv4 Unicast, index 0.3:
Attributes:
- Neighbor sessions are IPv4
- Internal
- Common admin
- First neighbor AS: 100
- Send communities
- Send GSHUT community if originated
- Send extended communities
- 4-byte AS capable
- Send AIGP
```
## Purpose

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send Re-originated VPN routes</td>
<td></td>
</tr>
<tr>
<td>Send multicast attributes</td>
<td></td>
</tr>
<tr>
<td>Minimum advertisement interval: 0 secs</td>
<td></td>
</tr>
<tr>
<td>Update group desynchronized: 0</td>
<td></td>
</tr>
<tr>
<td>Sub-groups merged: 0</td>
<td></td>
</tr>
<tr>
<td>Number of refresh subgroups: 0</td>
<td></td>
</tr>
<tr>
<td>Messages formatted: 2, replicated: 2</td>
<td></td>
</tr>
<tr>
<td>All neighbors are assigned to sub-group(s)</td>
<td></td>
</tr>
<tr>
<td>Neighbors in sub-group: 0.1, Filter-Groups num:1</td>
<td></td>
</tr>
<tr>
<td>Neighbors in filter-group: 0.1(RT num: 0) 20.0.0.1</td>
<td></td>
</tr>
</tbody>
</table>

### Step 10

**Command:**

```
show bgp l2vpn evpn update-group
```

**Example:**

```
RP/0/RSP0/CPU0:router# show bgp l2vpn evpn update-group
Fri Aug 21 00:27:42.786 PDT
Update group for L2VPN EVPN, index 0.2:
Attributes:
   Neighbor sessions are IPv4
   Internal
   Common admin
   First neighbor AS: 100
   Send communities
   Send GSHUT community if originated
   Send extended communities
   4-byte AS capable
   Send AIGP
   Send multicast attributes
   Minimum advertisement interval: 0 secs
   Update group desynchronized: 0
   Sub-groups merged: 0
   Number of refresh subgroups: 0
   Messages formatted: 4, replicated: 4
   All neighbors are assigned to sub-group(s)
   Neighbors in sub-group: 0.1, Filter-Groups num:1
   Neighbors in filter-group: 0.1(RT num: 0) 20.0.0.1
```

### Example: Data Center Interconnection Layer 3 Gateway Configuration

The following configurations provide an example Data Center Interconnection (DCI) Layer 3 Gateway configuration.

**VTEP-related configuration**

```
interface Loopback1
 ipv4 address 40.1.1.1 255.255.255.255
```
Example: Data Center Interconnection Layer 3 Gateway Configuration

interface nve1
  source-interface Loopback1
  member vni 1
  vrf cust1
  host-reachability protocol bgp
! interface BVI1
  vrf cust1
  ipv4 address 10.99.1.30 255.255.255.0
  ipv6 address 10:99:1::30/64
!

l2vpn
  bridge group bg1
  bridge-domain bd1
  routed interface BVI1
  member vni 1
!

VRF-related configuration

vrf data-center-10
  import route-target 1:1
  export route-target 1:2
  import route-target 10:10 stitching
  export route-target 10:20 stitching

Data Center EVPN BGP neighbor-related configuration

router bgp 1
  neighbor 1.1.1.1
    address-family l2vpn evpn
      import stitching-rt reoriginate
      advertise vpnv4 unicast reoriginated
      advertise vpnv6 unicast reoriginated
      advertise vpnv4 unicast local stitching-rt
      advertise vpnv6 unicast local stitching-rt
      advertise l2vpn evpn disable

L3VPN BGP neighbor-related configuration

router bgp 2
  neighbor 10.10.10.10
    address-family vpnv4
      import reoriginate stitching-rt
      advertise vpnv4 unicast reoriginated

The following example configuration shows how to configure the DCI router to forward default route to its Data Center neighbor.

router bgp 1
  address-family vpnv4 unicast
  address-family vpnv6 unicast
  address-family l2vpn evpn
  exit
  neighbor 1.1.1.1
    address-family l2vpn evpn
    default-originate
    exit

Implementing DCI VXLAN Layer 3 Gateway

Example: DCI VXLAN Layer 3 Gateway Configuration
OpFlex

OpFlex is an open and extensible policy protocol used for transferring the policy information between a network policy controller such as the Cisco Application Policy Infrastructure Controller (APIC) and network elements such as routers that are configured as Data Center Interconnect (DCI) gateway. The policies are distributed using the Cisco® Application Centric Infrastructure (ACI) infrastructure within the fabric to the spine nodes. The spine nodes send policies to the DCI gateway through the OpFlex framework. An OpFlex framework resides between the spines and the DCIs. It enables the distribution of the DCI policy model from the fabric to the DCI gateways. DCI gateway acts as an OpFlex agent and the spine acts a policy repository. Fabric tenant interconnect (FTI) is the OpFlex agent application that runs on the DCI to generate and apply the tenant device configuration on the DCI. Policies configure the DCI service for a given tenant on the DCI gateway.

OpFlex Topology

Consider the topology where OpFlex framework is used between the DCI gateway and the Cisco ACI spine switches to automate fabric-facing tenant provisioning on the DCI gateway. When you configure a new external Layer 3 outside (L3Out) policy for a tenant on the Cisco Application Policy Infrastructure Controller (APIC), the controller programs all related information associated with that tenant, such as VRF instance name and BGP extended community route-target attributes for the Cisco ACI spine switches. The OpFlex framework running on the spine switches reads the L3Out managed object and converts it to the OpFlex model. This information is then pushed to the DCI gateway, which acts as a policy repository for the OpFlex framework. On the DCI, the fabric facing configuration for the tenant VFR is auto-generated.
Restrictions

The OpFlex feature is supported with the following restrictions:

- OpFlex feature is not supported on ASR9K with power PC based route-processor.
- FTI cannot generate configuration for multiple RTs of one address family in a tenant VRF provisioned in one fabric.
- On exhaustion of FTI configuration pools, the OpFlex notifications to add tenants are ignored. If existing tenants are deleted, the new tenants must be added again to enable OpFlex notifications to be re-sent to the DCI.
- FTI supports only Type 0 RT format: 2 byte ASN + 4 byte value. Type 1 and Type 2 RT formats are not supported.
- XML configuration and oper schema are not supported for FTI configuration and show commands.

Configure OpFlex

Perform the following tasks to configure the OpFlex session to automate fabric-facing tenant provisioning on the DCI gateway. This includes the one-time configuration that must be done on the DCI to enable DCI hand-off from an ACI fabric.

Configure BGP

Perform this task to enable address-family under BGP routing process for fabric and WAN peering.

**SUMMARY STEPS**

1. `configure`
2. `router bgp as-number`
3. `bgp router-id ip-address`
4. `address-family {vpnv4 | vpnv6} unicast`
5. `address-family l2vpn evpn`
6. Use the `commit` or `end` command.

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td></td>
</tr>
<tr>
<td><code>configure</code></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Example: <code>RP/0/RSP0/CPU0:router# configure</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td></td>
</tr>
<tr>
<td><code>router bgp as-number</code></td>
<td>Specifies the autonomous system number and enters the BGP configuration mode, allowing you to configure the BGP routing process.</td>
</tr>
<tr>
<td>Example: <code>RP/0/RSP0/CPU0:router(config)# router bgp 1234</code></td>
<td></td>
</tr>
</tbody>
</table>
### Configure BGP Session on the Fabric Side

Perform this task to configure BGP session on the fabric side.

**SUMMARY STEPS**

1. `configure`
2. `router bgp asn_id`
3. `neighbor ip-address`
4. `remote-as autonomous-system-number`
5. `update-source loopback`
6. `address-family l2vpn evpn`
7. `import stitching-rt reoriginate`
8. `advertise {vpnv4 | vpnv6} unicast re-originated`
9. Use the `commit` or `end` command.
### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| **Step 1** configure  
Example:  
RP/0/RSP0/CP0:router# configure | Enters global configuration mode. |
| **Step 2** router bgp asn_id  
Example:  
RP/0/RSP0/CP0:router(config)# router bgp 200 | Specifies the BGP AS number and enters the BGP configuration mode, allowing you to configure the BGP routing process. |
| **Step 3** neighbor ip-address  
Example:  
RP/0/RSP0/CP0:router(config-bgp)# neighbor 209.165.201.1 | Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address 209.165.201.1 as a BGP peer. |
| **Step 4** remote-as autonomous-system-number  
Example:  
RP/0/RSP0/CP0:router(config-bgp-nbr)# remote-as 100 | Creates a neighbor and assigns it a remote autonomous system number. |
| **Step 5** update-source loopback  
Example:  
RP/0/RSP0/CP0:router(config-bgp-nbr)# update-source loopback2 | Allows BGP sessions to use the primary IP address from a particular interface as the local address. |
| **Step 6** address-family l2vpn evpn  
Example:  
RP/0/RSP0/CP0:router(config-bgp-nbr)# address-family l2vpn evpn | Configures EVPN address family. |
| **Step 7** import stitching-rt reoriginate  
Example:  
RP/0/RSP0/CP0:router(config-bgp-nbr-af)# import stitching-rt reoriginate | Enables import of routing information from BGP EVPN NLRIs that has route target identifier matching the stitching route target identifier and exports this routing information after re-origination to the L2VPN BGP neighbor. |
| **Step 8** advertise {vpnv4 | vpnv6} unicast re-originated  
Example:  
RP/0/RSP0/CP0:router(config-bgp-nbr-af)# advertise vpnv4 unicast re-originated | Configures advertisement of VVPNv4 or VVPNv6 unicast routes that are redistributed from the L2VPN BGP neighbor, to the EVPN BGP neighbor. The route targets are changed to the stitching route targets before advertising onto the EVPN BGP neighbor. |
| **Step 9** Use the **commit** or **end** command. | **commit** - Saves the configuration changes and remains within the configuration session.  
**end** - Prompts user to take one of these actions:  
  * **Yes** - Saves configuration changes and exits the configuration session. |
Configure BGP Session on the WAN Side

Perform this task to configure BGP session on the WAN side.

SUMMARY STEPS

1. `configure`
2. `router bgp asn_id`
3. `neighbor ip-address`
4. `remote-as autonomous-system-number`
5. `update-source loopback`
6. `address-family vpnv4 unicast`
7. `import re-originate stitching-rt`
8. `advertise {vpnv4 | vpnv6} unicast re-originated`
9. Use the `commit` or `end` command.

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
<td>RP/0/RSP0/CPU0:router# configure</td>
</tr>
<tr>
<td><strong>Step 2</strong> router bgp asn_id</td>
<td>Specifies the BGP AS number and enters the BGP configuration mode, allowing you to configure the BGP routing process.</td>
<td>RP/0/RSP0/CPU0:router(config)# router bgp 200</td>
</tr>
<tr>
<td><strong>Step 3</strong> neighbor ip-address</td>
<td>Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address 209.165.200.226 as a BGP peer.</td>
<td>RP/0/RSP0/CPU0:router(config-bgp)# neighbor 209.165.200.226</td>
</tr>
<tr>
<td><strong>Step 4</strong> remote-as autonomous-system-number</td>
<td>Creates a neighbor and assigns it a remote autonomous system number.</td>
<td>RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 100</td>
</tr>
<tr>
<td><strong>Step 5</strong> update-source loopback</td>
<td>Allows BGP sessions to use the primary IP address from a particular interface as the local address.</td>
<td></td>
</tr>
<tr>
<td>Command or Action</td>
<td>Purpose</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-bgp-nbr)# update-source loopback2</td>
<td>Enters VNPv4 address family configuration mode for the VNPv4 address family.</td>
<td></td>
</tr>
</tbody>
</table>

**Step 6**

**address-family vpng4 unicast**

*Example:*

```
RP/0/RSP0/CPU0:router(config-bgp-nbr)\# address-family vpng4 unicast
```

**Step 7**

**import re-originate stitching-rt**

*Example:*

```
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# import re-originate stitching-rt
```

**Step 8**

**advertise {vpng4 | vpng6} unicast re-originated**

*Example:*

```
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# advertise vpng4 unicast re-originated
```

**Step 9**

Use the **commit** or **end** command.

- **commit** - Saves the configuration changes and remains within the configuration session.
- **end** - Prompts user to take one of these actions:
  - **Yes** - Saves configuration changes and exits the configuration session.
  - **No** - Exits the configuration session without committing the configuration changes.
  - **Cancel** - Remains in the configuration mode, without committing the configuration changes.

---

**Configure DCI Underlay for Fabric and WAN Interfaces**

Perform this task to configure DCI underlay for fabric facing interface and WAN facing interface. Perform this task on both the interfaces.

**SUMMARY STEPS**

1. **configure**
2. **interface type interface-path-id**
3. **ipv4 address ipv4-address mask**
4. Use the **commit** or **end** command.
### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> interface type interface-path-id</td>
<td>Configures Gigabit Ethernet interface.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config)# interface GigabitEthernet 0/0/0</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> ipv4 address ipv4-address mask</td>
<td>Specifies the IPv4 address and subnet mask for the interface.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>RP/0/RSP0/CPU0:router(config-if)# ipv4 address 209.165.200.226 255.255.255.224</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> Use the commit or end command.</td>
<td>commit - Saves the configuration changes and remains within the configuration session.</td>
</tr>
<tr>
<td></td>
<td>end - Prompts user to take one of these actions:</td>
</tr>
<tr>
<td></td>
<td>• Yes - Saves configuration changes and exits the configuration session.</td>
</tr>
<tr>
<td></td>
<td>• No - Exits the configuration session without committing the configuration changes.</td>
</tr>
<tr>
<td></td>
<td>• Cancel - Remains in the configuration mode, without committing the configuration changes.</td>
</tr>
</tbody>
</table>

### Configure IGP for ACI and WAN Reachability

**Perform this task to configure IGP for ACI and WAN reachability.**

**SUMMARY STEPS**

1. configure  
2. router ospf process-name  
3. area area-id  
4. interface type interface-path-id  
5. exit  
6. exit  
7. area area-id  
8. nssa  
9. interface loopback loopback-id  
10. exit  
11. interface type interface-path-id  
12. Use the commit or end command.
## DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>configure</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router# configure</td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td>Enables OSPF routing for the specified routing process and places the router in router configuration mode.</td>
</tr>
<tr>
<td>router ospf process-name</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config)# router ospf 100</td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td>Enters area configuration mode and configures an area for the OSPF process.</td>
</tr>
<tr>
<td>area area-id</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-ospf)# area 0</td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td>Configures Gigabit Ethernet interface. Enables reachability to WAN.</td>
</tr>
<tr>
<td>interface type interface-path-id</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-ospf-ar)# interface GigabitEthernet 0/0/0/1</td>
</tr>
<tr>
<td><strong>Step 5</strong></td>
<td>Exits the interface submode and returns to area submode.</td>
</tr>
<tr>
<td>exit</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-ospf-ar-if)# exit</td>
</tr>
<tr>
<td><strong>Step 6</strong></td>
<td>Exits the area submode and returns to router configuration mode.</td>
</tr>
<tr>
<td>exit</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-ospf-ar)# exit</td>
</tr>
<tr>
<td><strong>Step 7</strong></td>
<td>Enters area configuration mode and configures an area for the OSPF process.</td>
</tr>
<tr>
<td>area area-id</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-ospf)# area 100</td>
</tr>
<tr>
<td><strong>Step 8</strong></td>
<td>Specifies area as a NSSA area</td>
</tr>
<tr>
<td>nssa</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-ospf-ar)# nssa</td>
</tr>
<tr>
<td><strong>Step 9</strong></td>
<td>Creates a loopback interface with the user-defined loopback identifier and enters the interface configuration mode.</td>
</tr>
<tr>
<td>interface loopback loopback-id</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-ospf-ar)# interface loopback0</td>
</tr>
<tr>
<td><strong>Step 10</strong></td>
<td>Exits the interface submode and returns to area submode.</td>
</tr>
<tr>
<td>exit</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td>RP/0/RSP0/CPU0:router(config-ospf-ar-if)# exit</td>
</tr>
<tr>
<td><strong>Step 11</strong></td>
<td>Configures Gigabit Ethernet interface. Enables reachability to ACI.</td>
</tr>
<tr>
<td>interface type interface-path-id</td>
<td></td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
</tbody>
</table>
Configure MPLS towards WAN

Perform this task to configure MPLS on the DCI.

SUMMARY STEPS

1. configure
2. mpls ldp
3. interface type interface-path-id
4. exit
5. exit
6. interface loopback instance
7. ipv4 address ipv4-address mask
8. exit
9. interface nve nve-identifier
10. source-interface loopback loopback-interface-identifier
11. Use the commit or end command.

DETAILED STEPS

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>mpls ldp</td>
<td>Enables MPLS LDP configuration mode.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RP/0/RSP0/CPU0:router# mpls ldp</td>
<td></td>
</tr>
<tr>
<td>Step</td>
<td>Command or Action</td>
<td>Purpose</td>
</tr>
<tr>
<td>------</td>
<td>------------------</td>
<td>---------</td>
</tr>
<tr>
<td>3</td>
<td><code>interface</code> type <code>interface-path-id</code></td>
<td>Configures Gigabit Ethernet interface.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-ldp)# interface GigabitEthernet 0/0/0/1</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td><code>exit</code></td>
<td>Exits the interface submode and returns to MPLS LDP submode.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-ldp-if)# exit</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td><code>exit</code></td>
<td>Exits the MPLS LDP submode and returns to global configuration mode.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-ldp)# exit</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td><code>interface</code> loopback <code>instance</code></td>
<td>Enters interface configuration mode and names the new loopback interface.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config)# interface Loopback0</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td><code>ipv4 address</code> <code>ipv4-address mask</code></td>
<td>Specifies the IPv4 address and subnet mask for the interface.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-if)# ipv4 address 209.165.200.227 255.255.255.224</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td><code>exit</code></td>
<td>Exits the interface submode and returns to global configuration mode.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-if)# exit</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td><code>interface</code> nve <code>nve-identifier</code></td>
<td>Creates the NVE interface and enters the NVE interface configuration sub-mode.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config)# interface nve 1</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td><code>source-interface</code> loopback <code>loopback-interface-identifier</code></td>
<td>Sets a loopback interface as the source interface for the VTEP.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> RP/0/RSP0/CPU0:router(config-if)# source-interface loopback 0</td>
<td></td>
</tr>
</tbody>
</table>
| 11   | Use the **commit** or **end** command. | commit - Saves the configuration changes and remains within the configuration session. end - Prompts user to take one of these actions:  
  • Yes - Saves configuration changes and exits the configuration session.  
  • No - Exits the configuration session without committing the configuration changes.  
  • Cancel - Remains in the configuration mode, without committing the configuration changes. |
# Configure FTI Auto-Configuration Parameters

Perform this task to configure FTI auto-configuration parameters.

## SUMMARY STEPS

1. `configure`
2. `dci-fabric-interconnect`
3. `auto-configuration-pool`
4. `bgp-as AS number`
5. `bridge group bridge-group-name`
6. `vrf vrf name ipv4-address ipv4 address`
7. `bd-pool bd range minimum bd range maximum`
8. `bvi-pool bvi range minimum bvi range maximum`
9. `vni-pool vni minimum range vni maximum range`
10. `local-vtep nve index`
11. Use the `commit` or `end` command.

## DETAILED STEPS

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td><code>configure</code></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td>Step 2</td>
<td><code>dci-fabric-interconnect</code></td>
<td>Enters the fabric tenant interconnect submode.</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config)# dci-fabric-interconnect</td>
<td></td>
</tr>
<tr>
<td>Step 3</td>
<td><code>auto-configuration-pool</code></td>
<td>Enters the auto configuration pool submode and enables to set the auto configuration pool parameters.</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config-fti)# auto-configuration-pool</td>
<td></td>
</tr>
<tr>
<td>Step 4</td>
<td><code>bgp-as AS number</code></td>
<td>Specifies the BGP AS number that is used when the configuration is generated. The BGP AS must be configured separately.</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config-fti-acp)# bgp-as 1234</td>
<td></td>
</tr>
<tr>
<td>Step 5</td>
<td><code>bridge group bridge-group-name</code></td>
<td>Specifies the L2VPN bridge group to be used for generation of configuration.</td>
</tr>
<tr>
<td></td>
<td>Example: RP/0/RSP0/CPU0:router(config-fti-acp)# bridge group bg1</td>
<td></td>
</tr>
<tr>
<td>Step 6</td>
<td><code>vrf vrf name ipv4-address ipv4 address</code></td>
<td>Configures per-VRF BVI interface IP address. If the default IPv4 address from link-local range is not acceptable for tenant addressing, this IP address must be configured.</td>
</tr>
<tr>
<td></td>
<td>Example: vrf vrf1 ipv4-address 198.51.100.1</td>
<td></td>
</tr>
<tr>
<td>Command or Action</td>
<td>Purpose</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>If configured, this must match the WAN-side tenant VRF configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Step 7</strong> bd-pool bd range minimum bd range maximum</td>
<td>Specifies the bridge domain range. The range is from 1 through 4000.</td>
<td></td>
</tr>
<tr>
<td>Example: RP/0/RSP0/CPU0:router(config-fti-acp)# bd-pool 1 1000</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Step 8</strong> bvi-pool bvi range minimum bvi range maximum</td>
<td>Specifies the bridge-group virtual interface (BVI) range. The range is from 1 through 4000.</td>
<td></td>
</tr>
<tr>
<td>Example: RP/0/RSP0/CPU0:router(config-fti-acp)# bvi-pool 1 1000</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Step 9</strong> vni-pool vni minimum range vni maximum range</td>
<td>Specifies the VNI range. The range is from 1 through 4000.</td>
<td></td>
</tr>
<tr>
<td>Example: RP/0/RSP0/CPU0:router(config-fti-acp)# vni-pool 1 1000</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Step 10</strong> local-vtep nve index</td>
<td>Specifies an NVE interface and configures it as VXLAN Tunnel EndPoint (VTEP) for the VXLAN.</td>
<td></td>
</tr>
<tr>
<td>Example: RP/0/RSP0/CPU0:router(config-fti-acp)# local-vtep nve 1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| **Step 11** Use the commit or end command. | commit - Saves the configuration changes and remains within the configuration session. end - Prompts user to take one of these actions:  
  • Yes - Saves configuration changes and exits the configuration session.  
  • No - Exits the configuration session without committing the configuration changes.  
  • Cancel - Remains in the configuration mode, without committing the configuration changes. |

**Configure OpFlex Session**

This task enables the fabric tenant interconnect to setup an OpFlex session with the spine.

**SUMMARY STEPS**

1. configure
2. dci-fabric-interconnect
3. fabric fabric identifier
4. opflex-peer spine IP address
5. exit
6. identity loopback IP address
7. Use the **commit** or **end** command.

### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td><strong>configure</strong>&lt;br&gt;<strong>Example:</strong>&lt;br&gt;RP/0/RSP0/CPU0:router# configure&lt;br&gt;Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td><strong>dci-fabric-interconnect</strong>&lt;br&gt;<strong>Example:</strong>&lt;br&gt;RP/0/RSP0/CPU0:router(config)# dci-fabric-interconnect&lt;br&gt;Enters the fabric tenant interconnect submode.</td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td><strong>fabric</strong>&lt;br&gt;<strong>Example:</strong>&lt;br&gt;RP/0/RSP0/CPU0:router(config-FTI)# fabric 1001&lt;br&gt;Enters the fabric submode and you can configure the fabric parameters. The fabric identifier range is from 1000 through 9999.</td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td><strong>opflex-peer</strong>&lt;br&gt;<strong>Example:</strong>&lt;br&gt;RP/0/RSP0/CPU0:router(config-FTI-fabric)# opflex-peer 192.0.2.1&lt;br&gt;FTI sets up an OpFlex session with the spine.</td>
</tr>
<tr>
<td><strong>Step 5</strong></td>
<td><strong>exit</strong>&lt;br&gt;<strong>Example:</strong>&lt;br&gt;RP/0/RSP0/CPU0:router(config-FTI-fabric)# exit&lt;br&gt;Exits the current configuration mode and returns to fti submode.</td>
</tr>
<tr>
<td><strong>Step 6</strong></td>
<td><strong>identity</strong>&lt;br&gt;<strong>Example:</strong>&lt;br&gt;RP/0/RSP0/CPU0:router(config-FTI-fabric)# identity 203.0.113.1&lt;br&gt;Specifies the DCI's BGP loopback IP address.</td>
</tr>
<tr>
<td><strong>Step 7</strong></td>
<td>Use the <strong>commit</strong> or <strong>end</strong> command. <strong>commit</strong> - Saves the configuration changes and remains within the configuration session. <strong>end</strong> - Prompts user to take one of these actions:&lt;br&gt;• <strong>Yes</strong> - Saves configuration changes and exits the configuration session.&lt;br&gt;• <strong>No</strong> - Exits the configuration session without committing the configuration changes.&lt;br&gt;• <strong>Cancel</strong> - Remains in the configuration mode, without committing the configuration changes.</td>
</tr>
</tbody>
</table>
EVPN Default VRF Route Leaking

The EVPN Default VRF Route Leaking feature leak routes between EVPN address-family and IPv4/IPv6 unicast address-family (Default-VRF), enabling the data center hosts to access the Internet. This feature is an extension of Border Gateway Protocol (BGP) VRF Dynamic route leaking feature that provides connectivity between non-default VRF hosts and Default VRF hosts by exchanging routes between the non-default VRF and Default VRF. EVPN Default VRF Route Leaking feature extends the BGP VRF Dynamic leaking feature, by allowing EVPN/L3VPN hosts to communicate with Default VRF hosts.

The import process installs the Internet route in a VRF table or a VRF route in the Internet table, providing connectivity.

The BGP VRF Dynamic route leaking feature is enabled by:

- Importing from default-VRF to non-default-VRF using the following command in VRF address-family configuration mode.

  import from default-vrf route-policy route-policy-name [advertise-as-vpn]

  If the advertise-as-vpn keyword is used, the paths imported from the default-VRF to the non-default-VRF are advertised to the (EVPN/L3VPN) PEs as well as to the CEs. If the advertise-as-vpn keyword is not used, the paths imported from the default-VRF to the non-default-VRF are not advertised to the PEs. However, the paths are still advertised to the CEs.

  The EVPN Default VRF Route Leaking feature with advertise-as-vpn keyword, enables to advertise the paths imported from default-VRF to non-default VRFs to EVPN PE peers as well.

  A new command advertise vpnv4/vpnv6 unicast imported-from-default-vrf disable is added under neighbor address-family configuration mode for EVPN and VPNv4/VPNv6 unicast to disable advertisement of Default-VRF leaked routes to that neighbor.

- Importing from non-default-VRF to default-VRF using the following command in VRF address-family configuration mode.

  export to default-vrf route-policy route-policy-name [advertise-as-vpn]

  The Dynamic Route Leaking feature enables leaking of local and CE routes to Default-VRF.

  A new optional keyword allow-imported-vpn is added to the above command, when configured, enables the leaking of EVPN and L3VPN imported/re-originated routes to the Default-VRF.

  A route-policy is mandatory to filter the imported routes. This reduces the risk of unintended import of routes between the Internet table and the VRF tables and the corresponding security issues. There is no hard limit on the number of prefixes that can be imported. The import creates a new prefix in the destination VRF, which increases the total number of prefixes and paths.

  **Note** Each VRF importing global routes adds workload equivalent to a neighbor receiving the global table. This is true even if the user filters out all but a few prefixes.
Scale Limitation of Default Route Leaking

Default VRF route leaking uses Dynamic Route Leaking feature to leak prefixes between the default VRF and the DC VRF. Do not use Dynamic Route Leaking feature to leak default VRF prefixes to large number of DC VRFs, even if you filter out all prefixes except a few that are to be leaked.

The following are the key factors that affect the performance:

- The default VRF prefix scale, which is approximately 0.7 million internet prefixes.
- The number of DC VRFs the default VRF prefixes that are to be imported.

To improve the scale, either the prefix scale or the number of VRFs whose prefixes that are to be imported must be reduced.

To manage the scale limitation, Cisco recommends you to do the following:

- Host the Internet prefixes on an adjacent PE with IPv4 unicast peering with DCI, and advertise a default route towards the DCI. On the DCI, import the default route from default VRF to DC VRFs.
- Host the Internet prefixes on an adjacent PE with IPv4 unicast peering with DCI. On the DCI, configure a static default route in the DC VRF with the next hop of the default VRF pointing to the adjacent PE address.
- Configure the static default route 0.0.0.0/0 on DC VRF with nexthop as “vrf default”.

Note

If the static routes are re-distributed to BGP, make sure it is not unintentionally advertised out.

EVPN Default VRF Route Leaking on the DCI for Internet Connectivity

The EVPN Default VRF Route Leaking feature leak routes between the Default-VRF and Data Center-VRF on the DCI to provide Internet access to data center hosts.

This feature is enabled by:

- Leaking routes from Default-VRF to Data Center-VRF
- Leaking routes to Default-VRF from Data Center-VRF

Leaking Routes from Default-VRF to Data Center-VRF

This section explains the process of leaking Default-VRF routes to Data Center-VRF.
Step 1
The Internet routes are present in the Default-VRF on the DCI.

Note A static default-route (0/0) can be configured under Default-VRF router static address-family configuration and redistributed to BGP.

Step 2
A route-policy is configured to select the routes to be leaked from Default-VRF to Data Center-VRF.

Example:

```
route-policy import-from-default-policy
  if destination in (100.10.0.0/16, 100.20.0.0/16) then
    pass
  endif
end-policy
!
route-policy import-from-default-policy-v6
  if destination in (100:10::0/64, 100:20::0/64) then
    pass
  endif
end-policy
!
```

Note Instead of leaking the internet routes, you can leak the default-route 0/0 from Default-VRF to Data Center-VRF using the following policy.

```
route-policy import-from-default-policy
  if destination in (0.0.0.0/0) then
    pass
  endif
end-policy
!
route-policy import-from-default-policy-v6
  if destination in (0::0/0) then
    pass
  endif
end-policy
!
```
Step 3  
Leak Default-VRF routes specified in the route-policy to Data Center-VRF by configuring `import from default-vrf route-policy import-from-default-policy(-v6)` under Data Center VRF address-family configuration mode.

**Example:**

```
vrf data-center-vrf
  address-family ipv4 unicast
    import from default-vrf route-policy import-from-default-policy !
  address-family ipv6 unicast
    import from default-vrf route-policy import-from-default-policy-v6 !
```

Step 4  
Advertise the leaked (Default-VRF) routes in the Data Center-VRF as EVPN routes towards Data Center routers by configuring `advertise-as-vpn` option.

**Example:**

```
vrf data-center-vrf
  address-family ipv4 unicast
    import from default-vrf route-policy import-from-default-policy advertise-as-vpn !
  address-family ipv6 unicast
    import from default-vrf route-policy import-from-default-policy-v6 advertise-as-vpn !
```

**Note**  
To advertise any routes from L3VPN address-family to EVPN peers, use `advertise vpnv4/vpnv6 unicast re-originated [stitching-rt]` command under neighbor address-family L2VPN EVPN.

**EVPN Default-originate**

Instead of advertising the Default-VRF routes towards Data Center routers, default-originate can be configured under the EVPN neighbor address-family to advertise the default route. When default-originate is configured under the neighbor address-family for EVPN/L3VPN, there is no need to advertise the Default-VRF leaked routes to the data center and `advertise-as-vpn` need not be configured.

**Example:**

```
router bgp 100
  neighbor 40.0.0.1
    address-family l2vpn evpn
default-originate

vrf data-center-vrf
  rd auto
  address-family ipv4 unicast
    allowvpn default-originate !
  address-family ipv6 unicast
    allowvpn default-originate !
```

Step 5  
To block advertisement of the Default-VRF leaked routes towards a particular EVPN/L3VPN peer, use `advertise vpnv4/vpnv6 unicast imported-from-default-vrf disable` command under respective neighbor address-family.

**Example:**

```
router bgp 100
  neighbor 40.0.0.1
    address-family l2vpn evpn
```
advertise vpnv4 unicast imported-from-default-vrf disable
advertise vpnv6 unicast imported-from-default-vrf disable

! router bgp 100
    neighbor 60.0.0.1
        address-family vpnv4 unicast
        advertise vpnv4 unicast imported-from-default-vrf disable
        address-family vpnv6 unicast
        advertise vpnv6 unicast imported-from-default-vrf disable

---

**Leaking Routes to Default-VRF from Data Center-VRF**

This section explains the process of leaking Data Center-VRF routes to Default-VRF.

*Figure 19: Leaking Routes to Default-VRF from Data Center-VRF*

---

**Step 1**  
Data Center routes are received on the DCI as EVPN Route-type 2 and Route-type 5 NLRI and imported to the Data Center VRFs.

**Step 2**  
A route-policy is configured to select the routes to be leaked from Data Center-VRF to Default-VRF.

**Example:**

```plaintext
route-policy export-to-default-policy
  if destination in (200.47.0.0/16, 200.168.0.0/16) then
    pass
  endif
end-policy
```

```plaintext
route-policy export-to-default-policy-v6
  if destination in (200:47::0/64, 200:168::0/64) then
    pass
  endif
end-policy
```
Step 3  Leak Data Center-VRF routes specified in the above policy to Default-VRF by configuring `export to default-vrf route-policy export-to-default-policy(-v6) [allow-imported-vpn]` under Data Center-VRF address-family configuration mode.

Normally only local and CE VRF routes are allowed to be leaked to the Default-VRF, but `allow-imported-vpn` configuration enables leaking of EVPN/L3VPN imported routes to the Default-VRF.

Example:

```
vrf data-center-vrf
  address-family ipv4 unicast
    export to default-vrf route-policy export-to-default-policy [allow-imported-vpn]

  address-family ipv6 unicast
    export to default-vrf route-policy export-to-default-policy-v6 [allow-imported-vpn]
```

Step 4  The Leaked routes in the Default VRF are advertised to the Internet.

Note  Instead of advertising the leaked routes to the Internet, an aggregate can be configured and advertised to the Internet.

---

### Sample Router Configuration

The following sample configuration specifies how EVPN Default VRF Route Leaking feature is configured on a DCI router to provide Internet access to the data center hosts.

```
vrf data-center-vrf
  address-family ipv4 unicast
    import from default-vrf route-policy import-from-default-policy advertise-as-vpn
    export to default-vrf route-policy export-to-default-policy allow-imported-vpn

  address-family ipv6 unicast
    import from default-vrf route-policy import-from-default-policy-v6 advertise-as-vpn
    export to default-vrf route-policy export-to-default-policy-v6 allow-imported-vpn

route-policy import-from-default-policy
  if destination in (100.10.0.0/16, 100.20.0.0/16) then
    pass
  endif
end-policy

route-policy import-from-default-policy-v6
  if destination in (100:10::0/64, 100:20::0/64) then
    pass
  endif
end-policy

route-policy export-to-default-policy
  if destination in (200.47.0.0/16, 200.168.0.0/16) then
```

The following sample configuration specifies how EVPN Default VRF Route Leaking feature is configured along with default-originate on a DCI router to provide Internet access to data center hosts.

```
Sample Router Configuration: with default-originate

vrf data-center-vrf
  address-family ipv4 unicast
    import from default-vrf route-policy import-from-default-policy
    advertise as-vpn
      export to default-vrf route-policy export-to-default-policy allow-imported-vpn
    !
  address-family ipv6 unicast
    import from default-vrf route-policy import-from-default-policy-v6
    advertise as-vpn
      export to default-vrf route-policy export-to-default-policy-v6 allow-imported-vpn
    !
  route-policy import-from-default-policy
    if destination in (100.10.0.0/16, 100.20.0.0/16) then
      pass
    endif
  end-policy
  !
  route-policy import-from-default-policy-v6
    if destination in (100:10::0/64, 100:20::0/64) then
      pass
    endif
  end-policy
  !
  route-policy export-to-default-policy
    if destination in (200.47.0.0/16, 200.168.0.0/16) then
      pass
    endif
  end-policy
```
EVPN Service VRF Route Leaking

The EVPN Service VRF Route Leaking feature enables connectivity to the services in the Service VRF to customers in EVPN Data Center VRF. The Service VRF and Data Center VRF routes can be IPv4 and/or IPv6 addresses. The Services VRF is any L3 VRF providing services reachable through connected, static, re-distributed IGP or BGP routes.

This feature leaks routes between Data Center VRF and Service VRF, enabling the EVPN/L3VPN hosts to access the Services in the Service VRF. This feature rely on Border Gateway Protocol (BGP) VRF extranet feature that imports routes between two VRFs.

The import process installs the Data Center VRF routes in a Service VRF table or a Service VRF routes in the Data Center VRF table, providing connectivity.

The BGP Service VRF route leaking feature is enabled by:

- Importing routes from Service VRF to Data Center VRF and advertising it as EVPN/L3VPN route from Data Center VRF.
- Importing Service VRF routes to Data Center VRF by attaching Data Center VRF import RTs to Service VRF routes.

This can be achieved by configuring one or more Data Center VRF import RTs as export RT of Service VRF, or configuring a Service VRF export route-policy to attach import RT EXTCOMM.
to Service VRF routes matching the import RTs of Data Center VRF using the following command in Service VRF address-family configuration mode.

```
export route-policy service-vrf-export-route-policy-name
```

Where the route-policy "service-vrf-export-route-policy-name" attaches the RT EXTCOMM matching the one or more import RTs of Data Center VRF to Service VRF routes.

- Advertising Data Center VRF imported routes that are exported from Service VRFs as EVPN/L3VPN NLRI from Data Center VRF using the following command in Data Center VRF address-family configuration mode.

```
import from vrf advertise-as-vpn
```

If the `advertise-as-vpn` keyword is used, the paths imported from the Service VRF to the Data Center VRF are advertised to the (EVPN/L3VPN) PEs as well as to the CEs. If the `advertise-as-vpn` keyword is not used, the paths imported from the Service VRF to the Data Center VRF are not advertised to the PEs. However, the paths are still advertised to the CEs.

- Block advertising Data Center VRF leaked routes from being advertised to a neighbor using the following command in neighbor address-family configuration mode.

```
advertise vpnv4/vpnv6 unicast imported-from-vrf disable
```

A new command `advertise vpnv4/vpnv6 unicast imported-from-vrf disable` is added under neighbor address-family configuration mode for EVPN and VPNv4/VPNv6 unicast to disable advertisement of VRF to VRF leaked routes to that neighbor.

- Importing EVPN/L3VPN routes from Data Center VRF to Service VRF

  - Importing EVPN/L3VPN routes from Data Center VRF to Service VRF by attaching Service VRF import RTs.

  This can be achieved by configuring one or more Service VRF import RTs as export RT of Data Center VRF, or configuring a Data Center VRF export route-policy to attach import RT EXTCOMM to Data Center VRF routes matching the import RTs of Service VRF using the following command in Data Center VRF address-family configuration mode.

```
export route-policy data-center-vrf-export-route-policy-name
```

The route-policy "data-center-vrf-export-route-policy-name" attaches the RT EXTCOMM matching one or more import RTs of Service VRF.

- Allow leaking of Data Center VRF routes to Service VRF by using the following command in Data Center VRF address-family configuration mode.

```
export to vrf allow-imported-vpn
```

**Note**

In order to prevent un-intended import of routes to VRFs, select unique RT's to import routes between Service VRF and Data Center VRF, which are not used for normal import of VPN/EVPN routes to Data Center VRFs.

The Extranet Route Leaking feature enables leaking of local and CE routes from one VRF to another VRF. A new command `export to vrf allow-imported-vpn` is added to enable the leaking of EVPN and L3VPN imported/re-originated Data Center VRF routes to the Service VRF.
A route-policy is preferred to filter the imported routes. This reduces the risk of unintended import of routes between the Data Center VRF and the Service VRF, and the corresponding security issues. There is no hard limit on the number of prefixes that can be imported. The import creates a new prefix in the destination VRF, which increases the total number of prefixes and paths.

Note
This feature does not advertise EVPN/L3VPN PE routes imported to Data Center VRF and leaked to Service VRF as EVPN/L3VPN PE route.

EVPN Service VRF Route Leaking on the DCI for Service Connectivity

The EVPN Service VRF Route Leaking feature leaks routes between the Service VRF and Data Center VRF on the DCI to provide access to Services to data center hosts.

This feature is enabled by:

- Leaking routes from Service VRF to Data Center VRF
- Leaking routes to Service VRF from Data Center VRF

Leaking Routes from Service VRF to Data Center VRF

This section explains the process of leaking Service VRF routes to Data Center VRF.

Figure 20: Leaking Routes from Service VRF to Data Center VRF

Step 1
The Service routes are present in the Service VRF on the DCI.
Step 2
A route-policy is configured to select the routes to be leaked from Service VRF to Data Center VRF.

Example:

```conf
route-policy service-vrf-export-policy
  if destination in (100.10.0.0/16, 100.20.0.0/16) then
    set extcommunity rt (1:1) additive <--- matches import RT of Data Center-VRF
  endif
end-policy
!
route-policy service-vrf-export-policy-v6
  if destination in (100:10::/64, 100:20::/64) then
    set extcommunity rt (1:1) additive <--- matches import RT of Data Center-VRF
  endif
end-policy
```

Step 3
Leak Service VRF routes specified in the route-policy to Data Center VRF by configuring `export route-policy service-vrf-export-policy(-v6)` under Service VRF address-family configuration mode.

Example:

```conf
vrf service-vrf
  address-family ipv4 unicast
    import route-target
    3:1
    4:1 stitching
    export route-policy service-vrf-export-policy
    export route-target
    3:1
    4:1 stitching
  !
  address-family ipv6 unicast
    import route-target
    3:1
    4:1 stitching
    export route-policy service-vrf-export-policy-v6
    export route-target
    3:1
    4:1 stitching
  !
```

Step 4
Advertise the leaked (Service VRF) routes in the Data Center VRF as EVPN/L3VPN routes towards Data Center routers by configuring `import from vrf advertise-as-vpn` under Data Center VRF address-family configuration mode.

Example:

```conf
vrf data-center-vrf
  address-family ipv4 unicast
    import from vrf advertise-as-vpn
    import route-target
    1:1
    100:1
    200:1 stitching
    export route-target
    100:1
    200:1 stitching
  !
  address-family ipv6 unicast
    import from vrf advertise-as-vpn
    import route-target
```
To advertise any routes from L3VPN address-family to EVPN peers, use `advertise vpnv4/vpnv6 unicast re-originated [stitching-rt]` command under neighbor address-family L2VPN EVPN.

**EVPN Default-originate**

Instead of advertising the Service VRF routes towards Data Center routers, default-originate can be configured under the EVPN neighbor address-family to advertise the default route. When `allow vpn default-originate` is configured under the Data Center VRF, there is no need to advertise the Service VRF leaked routes to the data center and `advertise-as-vpn` need not be configured.

**Example:**

```conf
router bgp 100
  neighbor 40.0.0.1
    address-family l2vpn evpn
      default-originate
  vrf data-center-vrf
    rd auto
    address-family ipv4 unicast
      allow vpn default-originate
  !
  address-family ipv6 unicast
    allow vpn default-originate
```

**Step 5**  
To block advertisement of the Service VRF leaked routes towards a particular EVPN/L3VPN peer, use `advertise vpnv4/vpnv6 unicast imported-from-vrf disable` command under respective neighbor address-family.

**Example:**

```conf
router bgp 100
  neighbor 40.0.0.1
    address-family l2vpn evpn
      import stitching-rt re-originate
      advertise vpnv4 unicast imported-from-vrf disable
      advertise vpnv6 unicast imported-from-vrf disable
  !
  router bgp 100
    neighbor 40.0.0.1
      address-family vpnv4 unicast
        import re-originate stitching-rt
        advertise vpnv4 unicast re-originated stitching-rt
        advertise vpnv4 unicast imported-from-vrf disable
        !
```

```conf
router bgp 100
  neighbor 60.0.0.1
    address-family vpnv6 unicast
      import re-originate stitching-rt
      advertise vpnv4 unicast re-originated
      advertise vpnv4 unicast imported-from-vrf disable
```

**Note**  
Export route-target 100:1 stitching

```conf
export route-target 100:1 stitching
```

**Note**  
EVPN Default-originate

Instead of advertising the Service VRF routes towards Data Center routers, default-originate can be configured under the EVPN neighbor address-family to advertise the default route. When `allow vpn default-originate` is configured under the Data Center VRF, there is no need to advertise the Service VRF leaked routes to the data center and `advertise-as-vpn` need not be configured.

**Example:**

```conf
router bgp 100
  neighbor 40.0.0.1
    address-family l2vpn evpn
      default-originate
  vrf data-center-vrf
    rd auto
    address-family ipv4 unicast
      allow vpn default-originate
  !
  address-family ipv6 unicast
    allow vpn default-originate
```

**Step 5**  
To block advertisement of the Service VRF leaked routes towards a particular EVPN/L3VPN peer, use `advertise vpnv4/vpnv6 unicast imported-from-vrf disable` command under respective neighbor address-family.

**Example:**

```conf
router bgp 100
  neighbor 40.0.0.1
    address-family l2vpn evpn
      import stitching-rt re-originate
      advertise vpnv4 unicast imported-from-vrf disable
      advertise vpnv6 unicast imported-from-vrf disable
  !
  router bgp 100
    neighbor 40.0.0.1
      address-family vpnv4 unicast
        import re-originate stitching-rt
        advertise vpnv4 unicast re-originated stitching-rt
        advertise vpnv4 unicast imported-from-vrf disable
        !
```

```conf
router bgp 100
  neighbor 60.0.0.1
    address-family vpnv6 unicast
      import re-originate stitching-rt
      advertise vpnv4 unicast re-originated
      advertise vpnv4 unicast imported-from-vrf disable
```

**Step 5**  
To block advertisement of the Service VRF leaked routes towards a particular EVPN/L3VPN peer, use `advertise vpnv4/vpnv6 unicast imported-from-vrf disable` command under respective neighbor address-family.

**Example:**

```conf
router bgp 100
  neighbor 40.0.0.1
    address-family l2vpn evpn
      import stitching-rt re-originate
      advertise vpnv4 unicast imported-from-vrf disable
      advertise vpnv6 unicast imported-from-vrf disable
  !
  router bgp 100
    neighbor 40.0.0.1
      address-family vpnv4 unicast
        import re-originate stitching-rt
        advertise vpnv4 unicast re-originated stitching-rt
        advertise vpnv4 unicast imported-from-vrf disable
        !
```

```conf
router bgp 100
  neighbor 60.0.0.1
    address-family vpnv6 unicast
      import re-originate stitching-rt
      advertise vpnv4 unicast re-originated
      advertise vpnv4 unicast imported-from-vrf disable
```
Leaking Routes to Service VRF from Data Center VRF

This section explains the process of leaking Data Center VRF routes to Service VRF.

**Figure 21: Leaking Routes to Service VRF from Data Center VRF**

**Step 1**
Data Center routes are received on the DCI as EVPN Route-type 2 and Route-type 5 NLRI and imported to the Data Center VRFs.

**Step 2**
A route-policy is configured to select the routes to be leaked from Data Center VRF to Service VRF. The policy attaches RT EXTCOMM to Data Center VRF routes matching one or more import RT of the Service VRF.

**Example:**

```plaintext
route-policy data-center-vrf-export-policy
  if destination in (200.47.0.0/16) then <--- EVPN PE route
    set extcommunity rt (4:1) additive <--- matches import stitching-RT of service-VRF
  if destination in (200.168.0.0/16) then <--- VPNv4 PE route
    set extcommunity rt (3:1) additive <--- matches import RT of service-VRF
  endif
end-policy
!
route-policy data-center-vrf-export-policy-v6
  if destination in (200:47::0/64) then <--- EVPN PE route
    set extcommunity rt (4:1) additive <--- matches import stitching-RT of service-VRF
  elseif destination in (200:168::0/64) then <--- VPNv6 PE route
    set extcommunity rt (3:1) additive <--- matches import RT of service-VRF
  endif
end-policy
!
```
An EVPN/L3VPN route received from a neighbor configured locally with "import stitching-rt re-originate" is imported to Data Center VRF if the route's RT EXTCOMM matches with one or more Data Center VRF import stitching RTs, and is leaked to Service VRF if the Data Center VRF route's RT EXTCOMM matches with one or more Service VRF import stitching RTs.

**Step 3**

Leak Data Center VRF routes specified in the above policy to Service VRF by configuring `export route-policy data-center-vrf-export-policy(-v6)` under Data Center VRF address-family configuration mode.

Normally only local and CE VRF routes are allowed to be leaked to the Service VRF, but `allow-imported-vpn` configuration enables leaking of EVPN/L3VPN imported routes to the Service VRF.

**Example:**

```conf
vrf data-center-vrf
  address-family ipv4 unicast
    import from vrf advertise-as-vpn
    import route-target
    1:1
    100:1
    200:1 stitching
    export route-policy data-center-vrf-export-policy
    export to vrf allow-imported-vpn
    export route-target
    100:1
    200:1 stitching
  !
  address-family ipv6 unicast
    import from vrf advertise-as-vpn
    import route-target
    1:1
    100:1
    200:1 stitching
    export route-policy data-center-vrf-export-policy-v6
    export to vrf allow-imported-vpn
    export route-target
    100:1
    200:1 stitching
  !
```

**Step 4**

The Data Center VRF leaked routes in the Service VRF are advertised to Service VRF CE peers.

---

**Sample Router Configuration**

The following sample configuration specifies how EVPN Service VRF Route Leaking feature is configured on a DCI router providing access to data center hosts to Services in the Service VRF.

```conf
vrf data-center-vrf
  address-family ipv4 unicast
    import from vrf advertise-as-vpn
    import route-target
    1:1
    100:1
    200:1 stitching
    export route-policy data-center-vrf-export-policy
    export to vrf allow-imported-vpn
    export route-target
    100:1
    200:1 stitching
```
address-family ipv6 unicast
import from vrf advertise-as-vpn
import route-target
  1:1
  100:1
  200:1 stitching
export route-policy data-center-vrf-export-policy-v6
export to vrf allow-imported-vpn
export route-target
  100:1
  200:1 stitching

vrf service-vrf
address-family ipv4 unicast
import route-target
  3:1
  4:1 stitching
export route-policy service-vrf-export-policy
export route-target
  3:1
  4:1 stitching

address-family ipv6 unicast
import route-target
  3:1
  4:1 stitching
export route-policy service-vrf-export-policy-v6
export route-target
  3:1
  4:1 stitching

route-policy data-center-vrf-export-policy
if destination in (200.47.0.0/16) then
  set extcommunity rt (4:1) additive
if destination in (200.168.0.0/16)
  set extcommunity rt (3:1) additive
endif
end-policy

route-policy data-center-vrf-export-policy-v6
if destination in (200:47::0/64) then
  set extcommunity rt (4:1) additive
elseif destination in (200:168::0/64)
  set extcommunity rt (3:1) additive
endif
end-policy

route-policy service-vrf-export-policy
if destination in (100.10.0.0/16, 100.20.0.0/16) then
  set extcommunity rt (1:1) additive
endif
end-policy

route-policy service-vrf-export-policy-v6
if destination in (100:10::0/64, 100:20::0/64) then
  set extcommunity rt (1:1) additive
endif
end-policy
route-policy pass-all
pass
end-policy
!

router bgp 100
neighbor 40.0.0.1
  remote-as 100
  address-family l2vpn evpn
    import stitching-rt re-originate
    advertise vpnv4 unicast re-originated stitching-rt
    advertise vpnv6 unicast re-originated stitching-rt
!
neighbor 60.0.0.1
  remote-as 200
  address-family vpnv4 unicast
    import re-originate stitching-rt
    route-policy pass-all in
    route-policy pass-all out
    advertise vpnv4 unicast re-originated
    advertise vpnv4 unicast imported-from-vrf disable
  address-family vpnv6 unicast
    import re-originate stitching-rt
    route-policy pass-all in
    route-policy pass-all out
    advertise vpnv6 unicast re-originated
    advertise vpnv6 unicast imported-from-vrf disable
!

Sample Router Configuration: with default-originate

The following sample configuration specifies how EVPN Service VRF Route Leaking feature is configured along with default-originate on a DCI router to provide data center hosts access to Services in the Service VRF.

vrf data-center-vrf
  address-family ipv4 unicast
    import from vrf advertise-as-vpn
    import route-target
    1:1
    100:1
    200:1 stitching
    export route-policy data-center-vrf-export-policy
    export to vrf allow-imported-vpn
    export route-target
    100:1
    200:1 stitching
!
  address-family ipv6 unicast
    import from vrf advertise-as-vpn
    import route-target
    1:1
    100:1
    200:1 stitching
    export route-policy data-center-vrf-export-policy-v6
    export to vrf allow-imported-vpn
    export route-target
    100:1
    200:1 stitching
!
vrf service-vrf
  address-family ipv4 unicast
    import route-target
      3:1
      4:1 stitching
    export route-policy service-vrf-export-policy
  export route-target
    3:1
    4:1 stitching
!
address-family ipv6 unicast
  import route-target
    3:1
    4:1 stitching
  export route-policy service-vrf-export-policy-v6
  export route-target
    3:1
    4:1 stitching
!
route-policy data-center-vrf-export-policy
  if destination in (200.47.0.0/16) then
    set extcommunity rt (4:1) additive
  if destination in (200.168.0.0/16) then
    set extcommunity rt (3:1) additive
  endif
end-policy
!
route-policy data-center-vrf-export-policy-v6
  if destination in (200:47::0/64) then
    set extcommunity rt (4:1) additive
  elseif destination in (200:168::0/64) then
    set extcommunity rt (3:1) additive
  endif
end-policy
!
route-policy service-vrf-export-policy
  if destination in (100.10.0.0/16, 100.20.0.0/16) then
    set extcommunity rt (1:1) additive
  endif
end-policy
!
route-policy service-vrf-export-policy-v6
  if destination in (100:10::0/64, 100:20::0/64) then
    set extcommunity rt (1:1) additive
  endif
end-policy
!
route-policy pass-all
  pass
end-policy
!
router bgp 100
  neighbor 40.0.0.1
    remote-as 100
    address-family l2vpn evpn
      import stitching-rt re-originate
      advertise vpnv4 unicast re-originated stitching-rt
      advertise vpnv4 unicast imported-from-vrf disable
advertise vpnv6 unicast re-originated stitching-rt
advertise vpnv6 unicast imported-from-vrf disable
default-originate <= Added=>

neighbor 60.0.0.1
remote-as 200
address-family vpnv4 unicast
  import re-originate stitching-rt
  route-policy pass-all in
  route-policy pass-all out
  advertise vpnv4 unicast re-originated
  advertise vpnv4 unicast imported-from-vrf disable
default-originate <= Added=>
address-family vpnv6 unicast
  import re-originate stitching-rt
  route-policy pass-all in
  route-policy pass-all out
  advertise vpnv6 unicast re-originated
  advertise vpnv6 unicast imported-from-vrf disable
default-originate <= Added=>

vrf data-center-vrf
  rd auto
  address-family ipv4 unicast
    allow vpn default-originate <= Added=>

vrf data-center-vrf
  address-family ipv6 unicast
    allow vpn default-originate <= Added=>
Sample Router Configuration: with default-originate