Dynamic Layer 3 VPNs with Multipoint GRE Tunnels

Last Updated: December 15, 2011

The Dynamic Layer 3 VPNs with Multipoint GRE Tunnels feature provides a Layer 3 (L3) transport mechanism based on an enhanced multipoint generic routing encapsulation (mGRE) tunneling technology for use in IP networks. The dynamic Layer 3 tunneling transport can also be used within IP networks to transport Virtual Private Network (VPN) traffic across service provider and enterprise networks, and to provide interoperability for packet transport between IP and Multiprotocol Label Switching (MPLS) VPNs. This feature provides support for RFC 2547, which defines the outsourcing of IP backbone services for enterprise networks.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the Feature Information for Dynamic L3 VPNs with mGRE Tunnels, page 22.

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

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

Prerequisites for Dynamic L3 VPNs with mGRE Tunnels

Before you configure the Dynamic Layer 3 VPNs with Multipoint GRE Tunnels feature, ensure that your MPLS VPN is configured and working properly. See the "Configuring MPLS Layer 3 VPNs" module for information about setting up MPLS VPNs.

Restrictions for Dynamic L3 VPNs with mGRE Tunnels

- The deployment of a MPLS VPN using both IP/GRE and MPLS encapsulation within a single network is not supported.
- Each provider edge (PE) router supports one tunnel configuration only.

Information About Dynamic L3 VPNs with mGRE Tunnels

You can configure mGRE tunnels to create a multipoint tunnel network that overlays an IP backbone. This overlay connects PE routers to transport VPN traffic. To deploy L3 VPN mGRE tunnels, you create a VRF instance, create the mGRE tunnel, redirect the VPN IP traffic to the tunnel, and set up the BGP VPNv4 exchange so that updates are filtered through a route map and interesting prefixes are resolved in the VRF table.

In addition, when MPLS VPNs are configured over mGRE, you can deploy L3 PE-based VPN services using a standards-based IP core. This allows you to provision the VPN services without using the overlay method. When an MPLS VPN over mGRE is configured, the system uses IPv4-based mGRE tunnels to encapsulate VPN-labeled IPv4 and IPv6 packets between PEs.

Layer 3 mGRE Tunnels

By configuring mGRE tunnels, you create a multipoint tunnel network as an overlay to the IP backbone. This overlay interconnects the PE routers to transport VPN traffic through the backbone. This multipoint tunnel network uses Border Gateway Protocol (BGP) to distribute VPNv4 routing information between PE routers, maintaining the peer relationship between the service provider or enterprise network and customer sites. The advertised next hop in BGP VPNv4 triggers tunnel endpoint discovery. This feature provides the ability for multiple service providers to cooperate and offer a joint VPN service with traffic tunneled directly from the ingress PE router at one service provider directly to the egress PE router at a different service provider site.

In addition to providing the VPN transport capability, the mGRE tunnels create a full-mesh topology and reduce the administrative and operational overhead previously associated with a full mesh of point-to-point tunnels used to interconnect multiple customer sites. The configuration requirements are greatly reduced and enable the network to grow with minimal additional configuration.

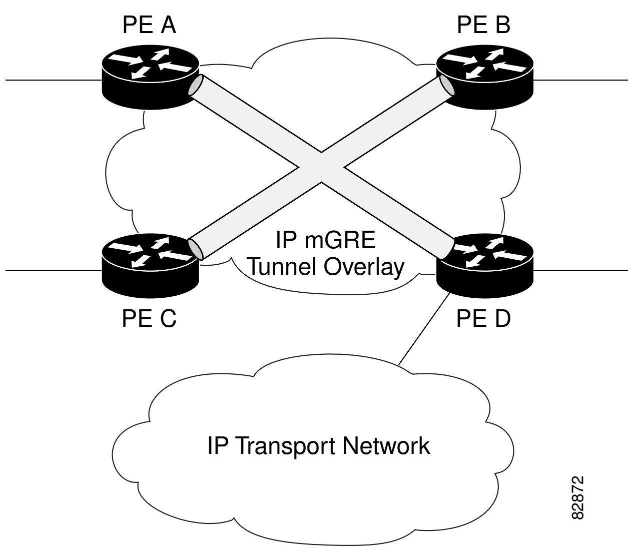
Dynamic L3 tunnels provide for better scaling when creating partial-mesh or full-mesh VPNs. Adding new remote VPN peers is simplified because only the new router needs to be configured. The new address is learned dynamically and propagated to the nodes in the network. The dynamic routing capability dramatically reduces the size of configuration needed on all routers in the VPN, such that with the use of multipoint tunnels, only one tunnel interface needs to be configured on a PE that services many VPNs. The L3 mGRE tunnels need to be configured only on the PE router. Features available with GRE are still available with mGRE, including dynamic IP routing and IP multicast and Cisco Express Forwarding (CEF) switching of mGRE/Next Hop Routing Protocol (NHRP) tunnel traffic.

The following sections describe how the mGRE tunnels are used:

Interconnecting Provider Edge Routers Within an IP Network

The Dynamic Layer 3 VPNs with Multipoint GRE Tunnels feature allows you to create a multiaccess tunnel network to interconnect the PE routers that service your IP network. This tunnel network transports IP VPN traffic to all of the PE routers. The figure below illustrates the tunnel overlay network used in an IP network to transport VPN traffic between the PE routers.

Figure 1 ***mGRE Tunnel Overlay Connecting PE Routers Within an IP Network***

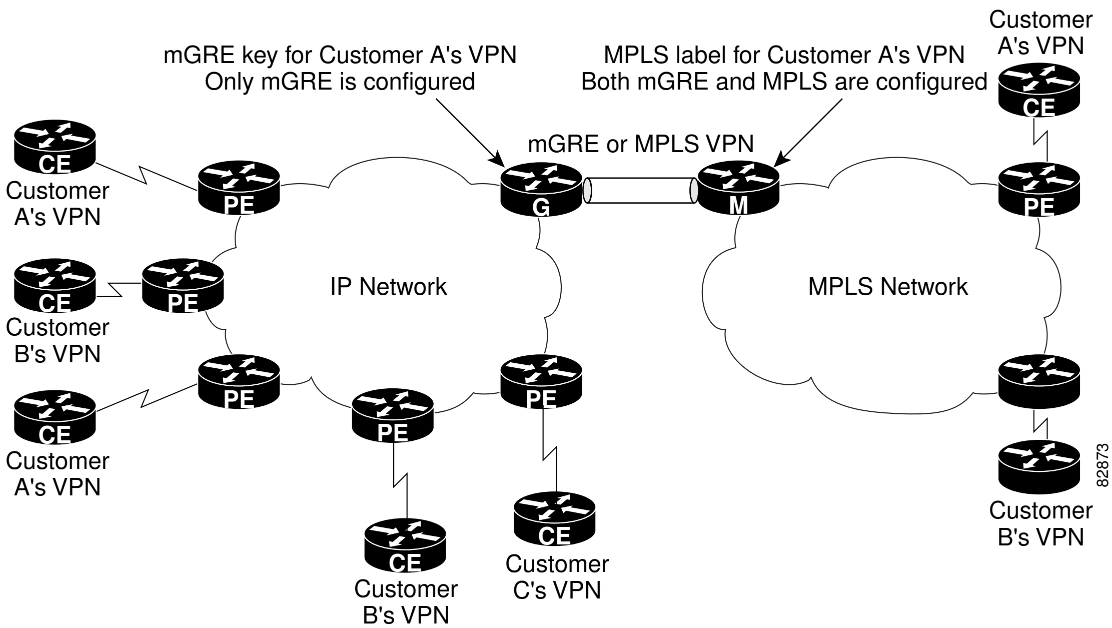


The multiaccess tunnel overlay network provides full connectivity between PE routers. The PE routers exchange VPN routes by using BGP as defined in RFC 2547. IP traffic is redirected through the multipoint tunnel overlay network using distinct IP address spaces for the overlay and transport networks and by changing the address space instead of changing the numerical value of the address.

Packet Transport Between IP and MPLS Networks

Layer 3 mGRE tunnels can be used as a packet transport mechanism between IP and MPLS networks. To enable the packet transport between the two different protocols, one PE router on one side of the connection between the two networks must run MPLS. The figure below shows how mGRE tunnels can be used to transport VPN traffic between PE routers.

Figure 2 mGRE Used to Transport VPN Traffic Between IP and MPLS Network



For the packet transport to occur between the IP and MPLS network, the MPLS VPN label is mapped to the GRE key. The mapping takes place on the router where both mGRE and MPLS are configured. In the figure above the mapping of the label to the key occurs on Router M, which sits on the MPLS network.

BGP Next Hop Verification

BGP performs the BGP path selection, or next hop verification, at the PE. For a BGP path to a network to be considered in the path selection process, the next hop for the path must be reachable in the Interior Gateway Protocol (IGP). When an IP prefix is received and advertised as the next hop IP address, the IP traffic is tunneled from the source to the destination by switching the address space of the next hop.

How to Configure L3 VPN mGRE Tunnels

- Creating the VRF and mGRE Tunnel, page 5
- Setting Up BGP VPN Exchange, page 7
- Enabling the MPLS VPN over mGRE Tunnels and Configuring an L3VPN Encapsulation Profile, page 9
- Defining the Address Space and Specifying Address Resolution for MPLS VPNs over mGRE, page 12

Creating the VRF and mGRE Tunnel

The tunnel that transports the VPN traffic across the service provider network resides in its own address space. A special VRF instance must be created called Resolve in VRF (RiV). This section describes how to create the VRF and GRE tunnel.

The IP address on the interface should be the same as that of the source interface specified in the configuration. The source interface specified should match that used by BGP as a source for the VPNv4 update.

Note Tunnel mode IPSec is not supported on MPLS over GRE Tunnel.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip vrf** *vrf-name*
4. **rd 1:1**
5. **interface tunnel** *tunnel-name*
6. **ip address** *ip-address subnet-id*
7. **tunnel source loopback** *n*
8. **tunnel mode gre multipoint l3vpn**
9. **tunnel key** *gre-ke* y
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip vrf** *vrf-name* **Example:** `Router(config)# ip vrf customer a riv`	Creates the special Resolve in VRF (RiV) VRF instance and table that will be used for the tunnel and redirection of the IP address.

	Command or Action	Purpose
Step 4	**rd 1:1** **Example:** `Router(config-vrf)# rd 1:1`	Enters the VRF configuration mode and specifies a route distinguisher (RD) for a VPN VRF instance.
Step 5	**interface tunnel** *tunnel-name* **Example:** `Router(config-vrf)# interface tunnel 1`	Enters interface configuration mode to create the tunnel.
Step 6	**ip address** *ip-address subnet-id* **Example:** `Router(config-if)# ipaddress 209.165.200.225 255.255.255.224`	Specifies the IP address for the tunnel.
Step 7	**tunnel source loopback** *n* **Example:** `Router(config-if)# tunnel source loopback test1`	Creates the loopback interface.
Step 8	**tunnel mode gre multipoint l3vpn** **Example:** `Router(config-if)# tunnel mode gre multipoint l3vpn`	Sets the mode for the tunnel as "gre multipoint l3vpn".
Step 9	**tunnel key** *gre-ke* y **Example:** `Router(config-if)# tunnel key 18`	Specifies the GRE key for the tunnel.
Step 10	**end** **Example:** `Router(config-if)# end`	Exits the current configuration mode and returns to privileged EXEC mode.

Setting Up BGP VPN Exchange

The configuration task described in this section sets up the BGP VPNv4 exchange so that the updates are filtered through a route map and interesting prefixes are resolved in the VRF table.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *tunnel-name*
4. **ip route vrf** *riv-vrf-name ip-address subnet- mask* **tunnel** *n*
5. **router bgp** *as-number*
6. **network** *network-id*
7. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *as-number*
8. **neighbor** {*ip-address* | *peer-group-name*} **update-source** *interface-type*
9. **address-family vpnv4** [**unicast**]
10. **neighbor** {*ip-address* | *peer-group-name*} **activate**
11. **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
12. **set ip next-hop resolve-in-vrf** *vrf-name*
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel** *tunnel-name* **Example:** `Router(config)# interface tunnel 1`	Enters interface configuration mode for the tunnel.

	Command or Action	Purpose
Step 4	**ip route vrf** *riv-vrf-name ip-address subnet- mask* **tunnel** *n* **Example:** `Router(config-if)# ip route vrf vrf1 209.165.200.226 255.255.255.224 tunnel 1`	Sets the packet forwarding to the special RiV VRF.
Step 5	**router bgp** *as-number* **Example:** `Router(config)# router bgp 100`	Specifies the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along.
Step 6	**network** *network-id* **Example:** `Router(config)# network 209.165.200.255`	Specifies the network ID for the networks to be advertised by the BGP and multiprotocol BGP routing processes.
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Router(config)# neighbor 209.165.200.227 remote-as 100`	Adds an entry to the BGP or multiprotocol BGP neighbor table.
Step 8	**neighbor** {*ip-address* | *peer-group-name*} **update-source** *interface-type* **Example:** `Router(config)# neighbor 209.165.200.228 update-source FastEthernet0/1`	Specifies a specific operational interface that BGP sessions use for TCP connections.
Step 9	**address-family vpnv4** [**unicast**] **Example:** `Router(config)# address-family vpnv4`	Specifies address family configuration mode for configuring routing sessions, such as BGP, that use standard VPN4 address prefixes.
Step 10	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Router(config)# neighbor 209.165.200.229 activate`	Enables the exchange of information with a neighboring router.

	Command or Action	Purpose
Step 11	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**} **Example:** `Router(config)# neighbor 209.165.200.230 route-map mpt in`	Applies a route map to incoming or outgoing routes. • Use once for each inbound route.
Step 12	**set ip next-hop resolve-in-vrf** *vrf-name* **Example:** `Router(config)# set ip next-hop resolve-in-vrf vrft`	Specifies that the next hop is to be resolved in the VRF table for the specified VRF.
Step 13	**end** **Example:** `Router(config)# end`	Exits the current configuration mode and returns to privileged EXEC mode.

Enabling the MPLS VPN over mGRE Tunnels and Configuring an L3VPN Encapsulation Profile

This section describes how to define the VRF, enable MPLS VPN over mGRE, and configure an L3VPN encapsulation profile.

Note Transport protocols such as IPv6, MPLS, IP, and Layer 2 Tunneling Protocol version 3 (L2TPv3) can also be used in this configuration.

To enable and configure MPLS VPN over mGRE, you must first define the VRF for tunnel encapsulation and enable L3VPN encapsulation in the system.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **vrf definition** *vrf-name*
4. **rd 1:1**
5. **exit**
6. **ip cef**
7. **ipv6** *unicast-routing*
8. **ipv6 cef**
9. **l3vpn encapsulation ip** *profile-name*
10. **transport ipv4 source** *interface n*
11. **protocol gre [key** *gre-key* **]**
12. **exit**
13. **interface** *type number*
14. **ip address** *ip-address mask*
15. **ip router isis**
16. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**vrf definition** *vrf-name* **Example:** `Router(config)# vrf definition tunnel encap`	Configures a VPN VRF routing table instance and enters VRF configuration mode.

	Command or Action	Purpose
Step 4	**rd 1:1** **Example:** `Router(config-vrf)# rd 1:1`	Specifies an RD for a VPN VRF instance.
Step 5	**exit** **Example:** `Router(config-vrf)# exit`	Exits VRF configuration mode.
Step 6	**ip cef** **Example:** `Router(config)# ip cef`	Enables Cisco Express Forwarding on the router.
Step 7	**ipv6** *unicast-routing* **Example:** `Router(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 8	**ipv6 cef** **Example:** `Router(config)# ipv6 cef`	Enables Cisco Express Forwarding for IPv6 on the router.
Step 9	**l3vpn encapsulation ip** *profile-name* **Example:** `Router(config)# l3vpn encapsulation ip tunnel encap`	Enters L3 VPN encapsulation configuration mode to create the tunnel.
Step 10	**transport ipv4 source** *interface n* **Example:** `Router(config-l3vpn-encap-ip)# transport ipv4 source loopback 0`	Specifies IPv4 transport source mode and defines the transport source interface.

	Command or Action	Purpose
Step 11	**protocol gre [key** *gre-key* **]** **Example:** `Router(config-l3vpn-encap-ip)# protocol gre key 1234`	Specifies GRE as the tunnel mode and sets the GRE key.
Step 12	**exit** **Example:** `Router(config-l3vpn-encap-ip)# exit`	Exits L3 VPN encapsulation configuration mode.
Step 13	**interface** *type number* **Example:** `Router(config)# interface loopback 0`	Enters interface configuration mode to configure the interface type.
Step 14	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.10.10.4 255.255.255.255`	Specifies the primary IP address and mask for the interface.
Step 15	**ip router isis** **Example:** `Router(config-if)# ip router isis`	Configures an Intermediate System-to-Intermediate System (IS-IS) routing process for IP on the interface and attaches a null area designator to the routing process.
Step 16	**end** **Example:** `Router(config-if)#end`	Exits the current configuration mode and returns to privileged EXEC mode.

Defining the Address Space and Specifying Address Resolution for MPLS VPNs over mGRE

This section describes how to define the address space and specify the address resolution for MPLS VPNs over mGRE. The following steps also enable you to link the route map to the application template and set up the BGP VPNv4 and VPNv6 exchange so that updates are filtered through the route map.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **bgp log-neighbor-changes**
5. **neighbor** *ip-address* **remote-as** *as-number*
6. **neighbor** *ip-address* **update-source** *interface-type interface-name*
7. **address-family vpn4**
8. **no synchronization**
9. **redistribute connected**
10. **neighbor** *ip-address* **activate**
11. **no auto-summary**
12. **exit**
13. **address-family vpnv4**
14. **neighbor** *ip-address* **activate**
15. **neighbor** *ip-address* **send-community both**
16. **neighbor** *ip-address* **route-map** *map-name* **in**
17. **exit**
18. **address-family vpnv6**
19. **neighbor** *ip-address* **activate**
20. **neighbor** *ip-address* **send-community both**
21. **neighbor** *ip-address* **route-map** *ip-address* **in**
22. **exit**
23. **route-map** *map-tag* **permit** *position*
24. **set ip next-hop encapsulate l3vpn** *tunnel encap*
25. **set ipv6 next-hop encapsulate l3vpn** *profile name*
26. **end**
27. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Router (config)# router bgp 100`	Specifies the number of an autonomous system that identifies the router to other BGP routers, tags the routing information passed along, and enters router configuration mode.
Step 4	**bgp log-neighbor-changes** **Example:** `Router (config-router)# bgp log-neighbor-changes`	Enables logging of BGP neighbor resets.
Step 5	**neighbor** *ip-address* **remote-as** *as-number* **Example:** `Router (config-router)# neighbor 10.10.10.6 remote-as 100`	Adds an entry to the BGP or multiprotocol BGP neighbor table.
Step 6	**neighbor** *ip-address* **update-source** *interface-type interface-name* **Example:** `Router (config-router)# neighbor 10.10.10.6 update-source loopback 0`	Allows BGP sessions to use any operational interface for TCP connections.
Step 7	**address-family vpn4** **Example:** `Router (config-router)# address-family vpnv4`	Enters address family configuration mode to configure routing sessions, that use IPv4 address prefixes.
Step 8	**no synchronization** **Example:** `Router (config-router-af)# no synchronization`	Enables the Cisco IOS software to advertise a network route without waiting for an IGP.

	Command or Action	Purpose
Step 9	**redistribute connected** **Example:** `Router (config-router-af)# redistribute connected`	Redistributes routes from one routing domain into another routing domain and allows the target protocol to redistribute routes learned by the source protocol and connected prefixes on those interfaces over which the source protocol is running.
Step 10	**neighbor** *ip-address* **activate** **Example:** `Router (config-router-af)# neighbor 10.10.10.6 activate`	Enables the exchange of information with a BGP neighbor.
Step 11	**no auto-summary** **Example:** `Router (config-router-af)# no auto-summary`	Disables automatic summarization and sends subprefix routing information across classful network boundaries
Step 12	**exit** **Example:** `Router (config-router-af)# exit`	Exits address family configuration mode.
Step 13	**address-family vpnv4** **Example:** `Router (config-router)# address-family vpnv4`	Enters address family configuration mode to configure routing sessions, such as BGP, that use standard VPNv4 address prefixes.
Step 14	**neighbor** *ip-address* **activate** **Example:** `Router (config-router-af)# neighbor 10.10.10.6 activate`	Enables the exchange of information with a BGP neighbor.
Step 15	**neighbor** *ip-address* **send-community both** **Example:** `Router (config-router-af)# neighbor 10.10.10.6 send-community both`	Specifies that a community attribute, for both standard and extended communities, should be sent to a BGP neighbor.

	Command or Action	Purpose
Step 16	**neighbor** *ip-address* **route-map** *map-name* **in** **Example:** `Router (config-router-af)# neighbor 10.10.10.6 route-map SELECT UPDATE FOR L3VPN in`	Applies the named route map to the incoming route.
Step 17	**exit** **Example:** `Router (config-router-af)# exit`	Exits address family configuration mode.
Step 18	**address-family vpnv6** **Example:** `6Router (config-router)# address-family vpnv4`	Enters address family configuration mode to configure routing sessions, such as BGP, that use VPNv6 address prefixes.
Step 19	**neighbor** *ip-address* **activate** **Example:** `Router (config-router-af)# neighbor 209.165.200.252 activate`	Enables the exchange of information with a BGP neighbor.
Step 20	**neighbor** *ip-address* **send-community both** **Example:** `Router (config-router-af)# neighbor 209.165.200.252 send-community both`	Specifies that a communities attribute, for both standard and extended communities, should be sent to a BGP neighbor.
Step 21	**neighbor** *ip-address* **route-map** *ip-address* **in** **Example:** `Router (config-router-af)# neighbor 209.165.200.252 route-map SELECT UPDATE FOR L3VPN in`	Applies the named route map to the incoming route.
Step 22	**exit** **Example:** `Router (config-router-af)# exit`	Exits address family configuration mode.

	Command or Action	Purpose
Step 23	**route-map** *map-tag* **permit** *position* **Example:** `Router (config-router)# route-map 192.168.10.1 permit 10`	Enters route-map configuration mode and defines the conditions for redistributing routes from one routing protocol into another. • The **redistribute** router configuration command uses the specified map tag to reference this route map. Multiple route maps may share the same map tag name. • If the match criteria are met for this route map, the route is redistributed as controlled by the set actions. • If the match criteria are not met, the next route map with the same map tag is tested. If a route passes none of the match criteria for the set of route maps sharing the same name, it is not redistributed by that set. • The *position* argument indicates the position that new route map will have in the list of route maps already configured with the same name.
Step 24	**set ip next-hop encapsulate l3vpn** *tunnel encap* **Example:** `Router (config-route-map)# set ip next-hop encapsulate l3vpn my profile`	Indicates that output IPv4 packets that pass a match clause of the route map are sent to the VRF for tunnel encapsulation.
Step 25	**set ipv6 next-hop encapsulate l3vpn** *profile name* **Example:** `Router (config-route-map)# set ip next-hop encapsulate l3vpn tunnel encap` **Example:**	Indicates that output IPv6 packets that pass a match clause of the route map are sent to the VRF for tunnel encapsulation.
Step 26	**end** **Example:** `Router (config-route-map)# exit`	Exits route-map configuration mode and enters global configuration mode.
Step 27	**end** **Example:** `Router (config)# exit`	Exits global configuration mode.

• What to Do Next, page 18

What to Do Next

You can perform the following to make sure that the configuration is working properly.

Check the VRF Prefix

Verify that the specified VRF prefix has been received by BGP. The BGP table entry should show that the route map has worked and that the next hop is showing in the RiV. Use the **show ip bgp vpnv4** command as shown in this example.

```
Router# show ip bgp vpnv4 vrf customer 209.165.200.250
BGP routing table entry for 100:1:209.165.200.250/24, version 12
Paths: (1 available, best #1)
  Not advertised to any peer
  Local
209.165.200.251 in "my riv" from 209.165.200.251 (209.165.200.251)
      Origin incomplete, metric 0, localpref 100, valid, internal, best
      Extended Community: RT:100:1
```

Confirm that the same information has been propagated to the routing table:

```
Router# show ip route vrf customer 209.165.200.250

Routing entry for 209.165.200.250
/24
  Known via "bgp 100", distance 200, metric 0, type internal
  Last update from 209.165.200.251 00:23:07 ago
  Routing Descriptor Blocks:
  * 209.165.200.251 (my riv), from 209.165.200.251, 00:23:07 ago
      Route metric is 0, traffic share count is 1
      AS Hops 0
```

CEF Switching

You can also verify that CEF switching is working as expected:

```
Router# show ip cef vrf customer
 209.165.200.250

209.165.200.250
/24, version 6, epoch 0
0 packets, 0 bytes
  tag information set
    local tag: VPN-route-head
    fast tag rewrite with Tu1, 123.1.1.2, tags imposed: {17}
  via 209.165.200.251, 0 dependencies, recursive
    next hop 209.165.200.251, Tunnel1 via 209.165.200.251/32 (my riv)
    valid adjacency
    tag rewrite with Tu1, 209.165.200.251, tags imposed: {17}
```

Endpoint Creation

Note that in this example display the tunnel endpoint has been created correctly:

```
Router# show tunnel endpoint tunnel 1
Tunnel1 running in multi-GRE/IP mode
  RFC2547/L3VPN Tunnel endpoint discovery is active on Tu1
  Transporting l3vpn traffic to all routes recursing through "my riv"
 Endpoint 209.165.200.251 via destination 209.165.200.251
 Endpoint 209.165.200.254 via destination 209.165.200.254
```

Adjacency

Confirm that the corresponding adjacency has been created.

```
Router# show adjacency Tunnel 1 interface
Protocol Interface                  Address
TAG      Tunnel1                    209.165.200.251(4)
                                    15 packets, 1980 bytes
                                    4500000000000000FF2FC3C77B010103
                                    7B01010200008847
                                    Epoch: 0
                                    Fast adjacency disabled
                                    IP redirect disabled
                                    IP mtu 1472 (0x0)
                                    Fixup enabled (0x2)
                                          GRE tunnel
                                    Adjacency pointer 0x624A1580, refCount 4
                                    Connection Id 0x0
                                    Bucket 121
```

Note that because MPLS is being transported over mGRE, the LINK_TAG adjacency is the relevant adjacency. The MTU reported in the adjacency is the payload length (including the MPLS label) that the packet will accept. The MAC string shown in the adjacency display can be interpreted as follows:

```
45000000 -> Beginning of IP Header (Partially populated, tl & chksum
00000000    are fixed up per packet)
FF2FC3C7
7B010103 -> Source IP Address in transport network 209.165.200.253
7B010102 -> Destination IP address in transport network 209.165.200.252
00008847 -> GRE Header
```

Refer to the Cisco IOS Multiprotocol Label Switching Configuration Guide for information about configuring MPLS Layer 3 VPNs.

You can use the **show l3vpn encapsulation** *profile-name* command to get information on the basic state of the application. The output of this command provides you details on the references to the tunnel and VRF.

Configuration Examples for Dynamic L3 VPNs Support Using mGRE Tunnels

Configuring Layer 3 VPN mGRE Tunnels Example

This example shows the configuration sequence for creating mGRE tunnels. It includes the definition of the special VRF instance.

```
ip vrf my riv
 rd 1:1
interface Tunnel1
 ip vrf forwarding my_riv
 ip address 209.165.200.250 255.255.255.224
 tunnel source Loopback0
 tunnel mode gre multipoint l3vpn
 tunnel key 123
end
ip route vrf my riv ip address subnet mask Tunnel1
router bgp 100
 network 209.165.200.251
 neighbor 209.165.200.250 remote-as 100
```

```
 neighbor 209.165.200.250 update-source Loopback0
 !
 address-family vpnv4
 neighbor 209.165.200.250 activate
 neighbor 209.165.200.250 route-map SELECT_UPDATES_FOR_L3VPN_OVER_MGRE in
!
route-map SELECT UPDATES FOR L3VPN OVER MGRE permit 10
 set ip next-hop in-vrf my riv
```

This example shows the configuration to link a route map to the application:

```
vrf definition Customer A
 rd 100:110
 route-target export 100:1000
 route-target import 100:1000
 !
 address-family ipv4
 exit-address-family
 !
 address-family ipv6
 exit-address-family
!
vrf definition tunnel encap
 rd 1:1
!
 address-family ipv4
 exit-address-family
 !
 address-family ipv6
 exit-address-family
!
!
ip cef
!
ipv6 unicast-routing
ipv6 cef
!
!
l3vpn encapsulation ip profile name
 transport source loopback 0
 protocol gre key 1234
!
!
 interface Loopback0
  ip address 209.165.200.252 255.255.255.224
  ip router isis
!
interface Serial2/0
 vrf forwarding Customer A
 ip address 209.165.200.253 255.255.255.224
 ipv6 address 3FFE:1001::/64 eui-64
 no fair-queue
 serial restart-delay 0
!
router bgp 100
 bgp log-neighbor-changes
 neighbor 209.165.200.254 remote-as 100
 neighbor 209.165.200.254 update-source Loopback0
 !
 address-family ipv4
  no synchronization
  redistribute connected
  neighbor 209.165.200.254 activate
  no auto-summary
 exit-address-family
 !
 address-family vpnv4
  neighbor 209.165.200.254 activate
  neighbor 209.165.200.254 send-community both
  neighbor 209.165.200.254 route-map SELECT UPDATE FOR L3VPN in
 exit-address-family
 !
 address-family vpnv6
```

```
  neighbor 209.165.200.254 activate
  neighbor 209.165.200.254 send-community both
  neighbor 209.165.200.254 route-map SELECT UPDATE FOR L3VPN in
 exit-address-family
 !
 address-family ipv4 vrf Customer A
  no synchronization
  redistribute connected
 exit-address-family
 !
 address-family ipv6 vrf Customer A
  redistribute connected
  no synchronization
 exit-address-family
!
!
route-map SELECT UPDATE FOR L3VPN permit 10
set ip next-hop encapulate <profile_name>
set ipv6 next-hop encapsulate <profile_name>
```

Additional References

For additional information related to dynamic L3 VPN mGRE tunnels, refer to the following references:

Related Documents

Related Topic	Document Title
Configuring MPLS Layer 3 VPNs	*Cisco IOS Multiprotocol Label Switching Configuration Guide*
MPLS VPN Over mGRE	*Cisco IOS Interface and Hardware Component Configuration Guide*
Cisco Express Forwarding	*Cisco IOS IP Switching Configuration Guide*
Generic Routing Encapsulation	*Cisco IOS Interface and Hardware Component Configuration Guide*

Standards

Standard	Title
None	--

MIBs

MIB	MIBs Link
IETF-PPVPN-MPLS-VPN-MIB	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2547	*BGP/MPLS VPNs*
RFC 2784	*Generic Routing Encapsulation (GRE)*
RFC 2890	Key Sequence Number Extensions to GRE
RFC 4023	Encapsulating MPLS in IP or Generic Routing Encapsulation
RFC 4364	*BGP/MPLS IP Virtual Private Networks (VPNs)*

Technical Assistance

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

Feature Information for Dynamic L3 VPNs with mGRE Tunnels

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

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

Table 1 Feature Information for Dynamic L3 VPNs with mGRE Tunnels

Feature Name	Releases	Feature Information
Dynamic Layer 3 VPNs with Multipoint GRE Tunnels	12.0(23)S	This feature provides an L3 transport mechanism based on an enhanced mGRE tunneling technology for use in IP networks.