Dynamic Multipoint VPN Configuration Guide, Cisco IOS XE Release 2

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS

Dynamic Multipoint VPN

The Dynamic Multipoint VPN feature allows users to better scale large and small IP Security (IPsec) Virtual Private Networks (VPNs) by combining generic routing encapsulation (GRE) tunnels, IPsec encryption, and Next Hop Resolution Protocol (NHRP).

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 Table at the end of this document.

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.

Prerequisites for Dynamic Multipoint VPN

- Before a multipoint GRE (mGRE) and IPsec tunnel can be established, you must define an Internet Key Exchange (IKE) policy by using the **crypto isakmp policy** command.
- To use the 2547oDMPVN--Traffic Segmentation Within DMVPN feature you must configure Multiprotocol Label Switching (MPLS) by using the **mpls ip** command.

Restrictions for Dynamic Multipoint VPN

- If you use the Restrictions for Dynamic Multipoint VPN, page 1 benefit of this feature, you must use IKE certificates or wildcard preshared keys for Internet Security Association Key Management Protocol (ISAKMP) authentication.

Note It is highly recommended that you do not use wildcard preshared keys because an attacker will have access to the VPN if one spoke router is compromised.

- GRE tunnel keepalives (that is, the **keepalive** command under a GRE interface) are not supported on point-to-point or multipoint GRE tunnels in a DMVPN network.
- If one spoke is behind one Network Address Translation (NAT) device and a different spoke is behind another NAT device, and Port Address Translation (PAT) is the type of NAT used on both NAT devices, then a session initiated between the two spokes cannot be established.

One example of a PAT configuration on a NAT interface is:

```
ip nat inside source list nat_acl interface FastEthernet0/0/1 overload
```

Information About Dynamic Multipoint VPN

Benefits of Dynamic Multipoint VPN

Hub Router Configuration Reduction

- For each spoke router, there is a separate block of configuration lines on the hub router that define the crypto map characteristics, the crypto access list, and the GRE tunnel interface. This feature allows users to configure a single mGRE tunnel interface, a single IPsec profile, and no crypto access lists on the hub router to handle all spoke routers. Thus, the size of the configuration on the hub router remains constant even if spoke routers are added to the network.
- DMVPN architecture can group many spokes into a single multipoint GRE interface, removing the need for a distinct physical or logical interface for each spoke in a native IPsec installation.

Automatic IPsec Encryption Initiation

- GRE has the peer source and destination address configured or resolved with NHRP. Thus, this feature allows IPsec to be immediately triggered for the point-to-point GRE tunneling or when the GRE peer address is resolved via NHRP for the multipoint GRE tunnel.

Support for Dynamically Addressed Spoke Routers

- When using point-to-point GRE and IPsec hub-and-spoke VPN networks, the physical interface IP address of the spoke routers must be known when configuring the hub router because the IP address must be configured as the GRE tunnel destination address. This feature allows spoke routers to have dynamic physical interface IP addresses (common for cable and DSL connections). When the spoke router comes online, it will send registration packets to the hub router: within these registration packets is the current physical interface IP address of this spoke.

Dynamic Creation for Spoke-to-Spoke Tunnels

- This feature eliminates the need for spoke-to-spoke configuration for direct tunnels. When a spoke router wants to transmit a packet to another spoke router, it can now use NHRP to dynamically determine the required destination address of the target spoke router. (The hub router acts as the NHRP server, handling the request for the source spoke router.) The two spoke routers dynamically create an IPsec tunnel between them so data can be directly transferred.

Feature Design of Dynamic Multipoint VPN

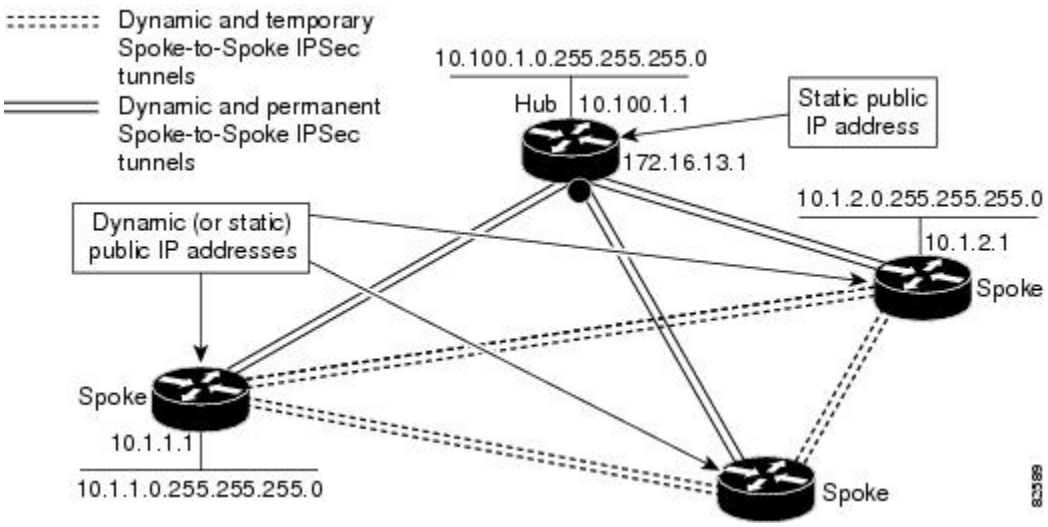
The Dynamic Multipoint VPN feature combines GRE tunnels, IPsec encryption, and NHRP routing to provide users an ease of configuration via crypto profiles--which override the requirement for defining static crypto maps--and dynamic discovery of tunnel endpoints.

This feature relies on the following two Cisco enhanced standard technologies:

- NHRP--A client and server protocol where the hub is the server and the spokes are the clients. The hub maintains an NHRP database of the public interface addresses of each spoke. Each spoke registers its real address when it boots and queries the NHRP database for real addresses of the destination spokes to build direct tunnels.
- mGRE tunnel interface --Allows a single GRE interface to support multiple IPsec tunnels and simplifies the size and complexity of the configuration.

The topology shown in the figure below and the corresponding bullets explain how this feature works.

Figure 1 Sample mGRE and IPsec Integration Topology



- Each spoke has a permanent IPsec tunnel to the hub, not to the other spokes within the network. Each spoke registers as clients of the NHRP server.
- When a spoke needs to send a packet to a destination (private) subnet on another spoke, it queries the NHRP server for the real (outside) address of the destination (target) spoke.
- After the originating spoke "learns" the peer address of the target spoke, it can initiate a dynamic IPsec tunnel to the target spoke.
- The spoke-to-spoke tunnel is built over the multipoint GRE interface.
- The spoke-to-spoke links are established on demand whenever there is traffic between the spokes. Thereafter, packets can bypass the hub and use the spoke-to-spoke tunnel.

Note After a preconfigured amount of inactivity on the spoke-to-spoke tunnels, the router will tear down those tunnels to save resources (IPsec security associations [SAs]).

IPsec Profiles

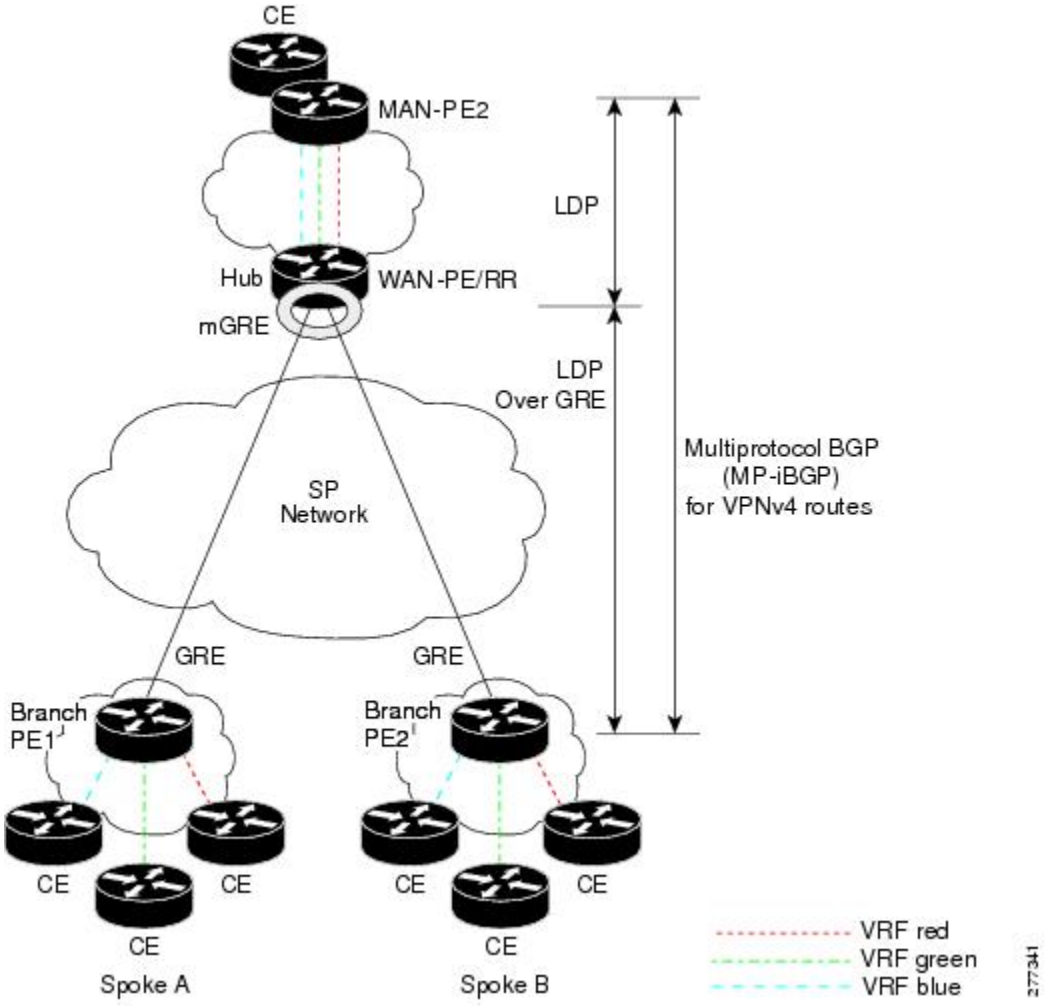
IPsec profiles abstract IPsec policy information into a single configuration entity, which can be referenced by name from other parts of the configuration. Therefore, users can configure functionality such as GRE tunnel protection with a single line of configuration. By referencing an IPsec profile, the user need not configure an entire crypto map configuration. An IPsec profile contains only IPsec information; that is, it does not contain any access list information or peering information.

Enabling Traffic Segmentation Within DMVPN

Cisco IOS XE Release 2.5 provides an enhancement that allows you to segment VPN traffic within a DMVPN tunnel by using a PE-PE mGRE tunnel. This secured mGRE tunnel can be used to transport all (or a set of) VPN traffic.

The diagram below and the corresponding bullets explain how traffic segmentation within DMVPN works.

Figure 2 Traffic Segmentation with DMVPN



- The hub shown in the diagram is a WAN-PE and a Route Reflector, and the spokes (PE routers) are clients.
- There are three VRFs, designated “red,” “green,” and “blue.”
- Each spoke has both a neighbor relationship with the hub (multiprotocol internal Border Gateway Protocol [MP-iBGP] peering) and a GRE tunnel to the hub.
- Each spoke advertises its routes and VPN-IPv4 (VPNv4) prefixes to the hub.
- The hub sets its own IP address as the next-hop route for all the VPNv4 addresses it learns from the spokes and assigns a local MPLS label for each VPN when it advertises routes back to the spokes. As a result, traffic from Spoke A to Spoke B is routed via the hub.

An example illustrates the process:

1. Spoke A advertises a VPNv4 route to the hub, and applies the label *x* to the VPN.
2. The hub changes the label to *y* when the hub advertises the route to Spoke B.
3. When Spoke B has traffic to send to Spoke A, it applies the *y* label, and the traffic goes to the hub.

4 The hub swaps the VPN label, by removing the y label and applying an x label, and sends the traffic to Spoke A.

NAT-Transparency Aware DMVPN

DMVPN spokes are often situated behind a NAT router (which is often controlled by the Internet Service Provider [ISP] for the spoke site) with the outside interface address of the spoke router being dynamically assigned by the ISP using a private IP address (per Internet Engineering Task Force [IETF] RFC 1918).

With the NAT-Transparency Aware DMVPN enhancement, NHRP can learn and use the NAT public address for its mappings as long as IPsec transport mode is used (which is the recommended IPsec mode for DMVPN networks). It is recommended that all DMVPN routers be upgraded to the new code before you try to use the NAT-Transparency Aware DMVPN functionality even though spoke routers that are not behind NAT need not be upgraded. In addition, you cannot convert upgraded spoke routers that are behind NAT to the new configuration (IPsec transport mode) until the hub routers have been upgraded.

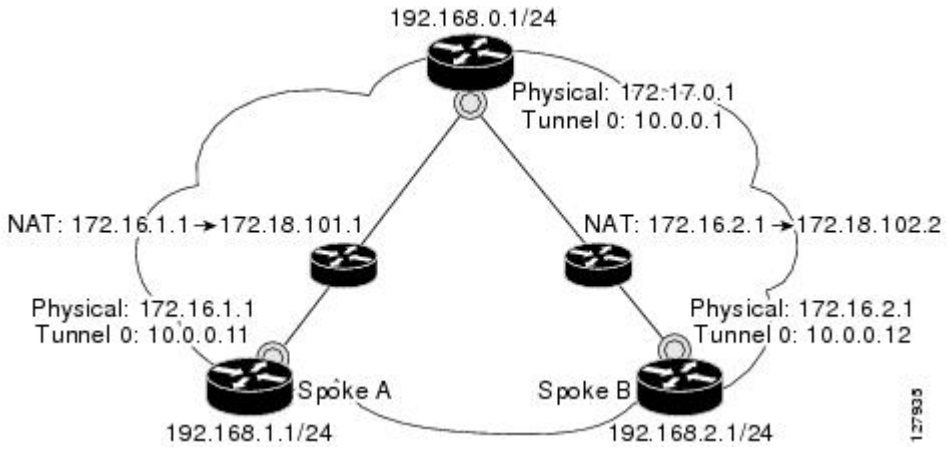
With this NAT Transparency enhancement, the hub DMVPN router can be behind the static NAT. For this functionality to be used, all the DMVPN spoke routers and hub routers must be upgraded, and IPsec must use transport mode.

For these NAT-Transparency Aware enhancements to work, you must use IPsec transport mode on the transform set. Also, even though NAT-Transparency (IKE and IPsec) can support two peers (IKE and IPsec) being translated to the same IP address (using the UDP ports to differentiate them), this functionality is not supported for DMVPN. All DMVPN spokes must have a unique IP address after they have been NAT translated. They can have the same IP address before they are NAT translated.

The figure below illustrates a NAT-Transparency Aware DMVPN scenario.

Note DMVPN spokes behind NAT will participate in dynamic direct spoke-to-spoke tunnels. The spokes must be behind NAT boxes that are preforming NAT, not PAT. The NAT box must translate the spoke to the same outside NAT IP address for the spoke-to-spoke connections as the NAT box does for the spoke-to-hub connection. If there is more than one DMVPN spoke behind the same NAT box, the NAT box must translate the DMVPN spokes to different outside NAT IP addresses. It is also likely that you may not be able to build a direct spoke-to-spoke tunnel between these spokes. If a spoke-to-spoke tunnel fails to form, the spoke-to-spoke packets will continue to be forwarded via the spoke-to-hub-spoke path.

Figure 3 NAT-Transparency Aware DMVPN



Call Admission Control with DMVPN

In a DMVPN network, it is easy for a DMVPN router to become “overwhelmed” with the number of tunnels it is trying to build. Call Admission Control can be used to limit the number of tunnels that can be built at any one time, thus protecting the memory of the router and CPU resources.

It is most likely that Call Admission Control will be used on a DMVPN spoke to limit the total number of ISAKMP sessions (DMVPN tunnels) that a spoke router will attempt to initiate or accept. This limiting is accomplished by configuring an IKE SA limit under Call Admission Control, which configures the router to drop new ISAKMP session requests (inbound and outbound) if the current number of ISAKMP SAs exceeds the limit.

It is most likely that Call Admission Control will be used on a DMVPN hub to rate limit the number of DMVPN tunnels that are attempting to be built at the same time. The rate limiting is accomplished by configuring a system resource limit under Call Admission Control, which configures the router to drop new ISAKMP session requests (new DMVPN tunnels) when the system utilization is above a specified percentage. The dropped session requests allow the DMVPN hub router to complete the current ISAKMP session requests, and when the system utilization drops, it can process the previously dropped sessions when they are reattempted.

No special configuration is required to use Call Admission Control with DMVPN. For information about configuring Call Admission Control, see the “ Call Admission Control for IKE ” module in the *Cisco IOS XE Security Configuration Guide: Secure Connectivity* .

NHRP Rate-Limiting Mechanism

NHRP has a rate-limiting mechanism that restricts the total number of NHRP packets from any given interface. The default values, which are set using the **ip nhrp max-send** command, are 100 packets every 10 seconds per interface. If the limit is exceeded, you will get the following system message:

```
%NHRP-4-QUOTA: Max-send quota of [int]pkts/[int]Sec. exceeded on [chars]
```

For more information about this system message, see the document System Messages for Cisco IOS XE Software .

How to Configure Dynamic Multipoint VPN

To enable mGRE and IPsec tunneling for hub and spoke routers, you must configure an IPsec profile that uses a global IPsec policy template and configure your mGRE tunnel for IPsec encryption. This section contains the following procedures:

Configuring an IPsec Profile

The IPsec profile shares most of the same commands with the crypto map configuration, but only a subset of the commands are valid in an IPsec profile. Only commands that pertain to an IPsec policy can be issued under an IPsec profile; you cannot specify the IPsec peer address or the Access Control List (ACL) to match the packets that are to be encrypted.

Before configuring an IPsec profile, you must define a transform set by using the **crypto ipsec transform-set** command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto ipsec profile** *name*
4. **set transform-set** *transform-set-name*
5. **set identity**
6. **set security association lifetime** {**seconds** *seconds* | **kilobytes** *kilobytes*}
7. **set pfs** [**group1** | **group2**]

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	**crypto ipsec profile** *name* **Example:** `Router(config)# crypto ipsec profile vpnprof`	Defines the IPsec parameters that are to be used for IPsec encryption between "spoke and hub" and "spoke and spoke" routers. • This command enters crypto map configuration mode. • The *name* argument specifies the name of the IPsec profile.
Step 4	**set transform-set** *transform-set-name* **Example:** `Router(config-crypto-map)# set transform-set trans2`	Specifies which transform sets can be used with the IPsec profile. • The *transform-set-name* argument specifies the name of the transform set.

	Command or Action	Purpose
Step 5	**set identity** **Example:** `Router(config-crypto-map)# set identity`	(Optional) Specifies identity restrictions to be used with the IPsec profile.
Step 6	**set security association lifetime** {**seconds** *seconds* | **kilobytes** *kilobytes*} **Example:** `Router(config-crypto-map)# set security association lifetime seconds 1800`	(Optional) Overrides the global lifetime value for the IPsec profile. • The **seconds** *seconds* option specifies the number of seconds a security association will live before expiring; the **kilobytes** *kilobytes*option specifies the volume of traffic (in kilobytes) that can pass between IPsec peers using a given security association before that security association expires. • The default for the *seconds* argument is 3600 seconds.
Step 7	**set pfs** [**group1** | **group2**] **Example:** `Router(config-crypto-map)# set pfs group2`	(Optional) Specifies that IPsec should ask for perfect forward secrecy (PFS) when requesting new security associations for this IPsec profile. • If this command is not specified, the default (**group1**) is enabled. • The **group1** keyword specifies that IPsec should use the 768-bit Diffie-Hellman (DH) prime modulus group when performing the new DH exchange; the **group2** keyword specifies the 1024-bit DH prime modulus group.

Configuring the Hub for DMVPN

To configure the hub router for mGRE and IPsec integration (that is, associate the tunnel with the IPsec profile configured in the previous procedure), use the following commands.

Note NHRP network IDs are locally significant and can be different. It makes sense from a deployment and maintenance perspective to use unique **network ID**numbers (using the **ip nhrp network-id** command) across all routers in a DMVPN network, but it is not necessary that they be the same.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *number*
4. **ip address** *ip-address mask secondary*
5. **ip mtu** *bytes*
6. **ip nhrp authentication** *string*
7. **ip nhrp map multicast dynamic**
8. **ip nhrp network-id** *number*
9. **tunnel source** {*ip-address* | *type number*}
10. **tunnel key** *key-number*
11. **tunnel mode gre multipoint**
12. **tunnel protection ipsec profile** *name*
13. **bandwidth** *kbps*
14. **ip tcp adjust-mss** *max-segment-size*
15. **ip nhrp holdtime** *seconds*
16. **delay** *number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel** *number* **Example:** `Router(config)# interface tunnel 5`	Configures a tunnel interface and enters interface configuration mode • The *number* argument specifies the number of the tunnel interface that you want to create or configure. There is no limit on the number of tunnel interfaces you can create.

	Command or Action	Purpose
Step 4	**ip address** *ip-address mask secondary* **Example:** `Router(config-if)# ip address 10.0.0.1 255.255.255.0`	Sets a primary or secondary IP address for the tunnel interface. **Note** All hubs and spokes that are in the same DMVPN network must be addressed in the same IP subnet.
Step 5	**ip mtu** *bytes* **Example:** `Router(config-if)# ip mtu 1400`	Sets the maximum transmission unit (MTU) size, in bytes, of IP packets sent on an interface.
Step 6	**ip nhrp authentication** *string* **Example:** `Router(config-if)# ip nhrp authentication donttell`	Configures the authentication string for an interface using NHRP. **Note** The NHRP authentication string must be set to the same value on all hubs and spokes that are in the same DMVPN network.
Step 7	**ip nhrp map multicast dynamic** **Example:** `Router(config-if)# ip nhrp map multicast dynamic`	Allows NHRP to automatically add spoke routers to the multicast NHRP mappings.
Step 8	**ip nhrp network-id** *number* **Example:** `Router(config-if)# ip nhrp network-id 99`	Enables NHRP on an interface. • The *number* argument specifies a globally unique 32-bit network identifier from a nonbroadcast multiaccess (NBMA) network. The range is from 1 to 4294967295.
Step 9	**tunnel source** {*ip-address* | *type number*} **Example:** `Router(config-if)# tunnel source Gigabitethernet 0/0/0`	Sets the source address for a tunnel interface.
Step 10	**tunnel key** *key-number* **Example:** `Router(config-if)# tunnel key 100000`	(Optional) Enables an ID key for a tunnel interface. • The *key-number* argument specifies a number from 0 to 4,294,967,295 that identifies the tunnel key. **Note** The key number must be set to the same value on all hubs and spokes that are in the same DMVPN network.

	Command or Action	Purpose
Step 11	**tunnel mode gre multipoint** **Example:** `Router(config-if)# tunnel mode gre multipoint`	Sets the encapsulation mode to mGRE for the tunnel interface.
Step 12	**tunnel protection ipsec profile** *name* **Example:** `Router(config-if)# tunnel protection ipsec profile vpnprof`	Associates a tunnel interface with an IPsec profile. • The *name* argument specifies the name of the IPsec profile; this value must match the *name* specified in the **crypto ipsec profile** *name*command.
Step 13	**bandwidth** *kbps* **Example:** `Router(config-if)# bandwidth 1000`	Sets the current bandwidth value for an interface to higher-level protocols. • The *kbps* argument specifies the bandwidth in kilobits per second. The default value is 9. The recommended bandwidth value is 1000 or greater. • Setting the bandwidth value to at least 1000 is critical if EIGRP is used over the tunnel interface. Higher bandwidth values may be necessary depending on the number of spokes supported by a hub.
Step 14	**ip tcp adjust-mss** *max-segment-size* **Example:** `Router(config-if)# ip tcp adjust-mss 1360`	Adjusts the maximum segment size (MSS) value of TCP packets going through a router. • The *max-segment-size* argument specifies the maximum segment size, in bytes. The range is from 500 to 1460. • The recommended value is 1360 when the number of IP MTU bytes is set to 1400. With these recommended settings, TCP sessions quickly scale back to 1400-byte IP packets so the packets will “fit” in the tunnel.
Step 15	**ip nhrp holdtime** *seconds* **Example:** `Router(config-if)# ip nhrp holdtime 450`	Changes the number of seconds that NHRP NBMA addresses are advertised as valid in authoritative NHRP responses. • The *seconds* argument specifies the time in seconds that NBMA addresses are advertised as valid in positive authoritative NHRP responses. The recommended value ranges from 300 seconds to 600 seconds.
Step 16	**delay** *number* **Example:** `Router(config-if)# delay 1000`	(Optional) Changes the EIGRP routing metric for routes learned over the tunnel interface. • The *number* argument specifies the delay time in seconds. The recommended value is 1000.

Configuring the Spoke for DMVPN

To configure spoke routers for mGRE and IPsec integration, use the following commands.

Note NHRP network IDs are locally significant and can be different. It makes sense from a deployment and maintenance perspective to use unique **network ID**numbers (using the **ip nhrp network-id** command) across all routers in a DMVPN network, but it is not necessary that they be the same.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *number*
4. **ip address** *ip-address mask secondary*
5. **ip mtu** *bytes*
6. **ip nhrp authentication** *string*
7. **ip nhrp map** *hub-tunnel-ip-address hub-physical-ip-address*
8. **ip nhrp map multicast** *hub-physical-ip-address*
9. **ip nhrp nhs** *hub-tunnel-ip-address*
10. **ip nhrp network-id** *number*
11. **tunnel source** {*ip-address* | *type number*}
12. **tunnel key** *key-number*
13. Do one of the following:
    - **tunnel mode gre multipoint**
    -
    -
    -
    -
    - **tunnel destination** *hub-physical-ip-address*
14. **tunnel protection ipsec profile** *name*
15. **bandwidth** *kbps*
16. **ip tcp adjust-mss** *max-segment-size*
17. **ip nhrp holdtime** *seconds*
18. **delay** *number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel** *number* **Example:** `Router(config)# interface tunnel 5`	Configures a tunnel interface and enters interface configuration mode. • The *number* argument specifies the number of the tunnel interface that you want to create or configure. There is no limit on the number of tunnel interfaces you can create.
Step 4	**ip address** *ip-address mask secondary* **Example:** `Router(config-if)# ip address 10.0.0.2 255.255.255.0`	Sets a primary or secondary IP address for the tunnel interface. **Note** All hubs and spokes that are in the same DMVPN network must be addressed in the same IP subnet.
Step 5	**ip mtu** *bytes* **Example:** `Router(config-if)# ip mtu 1400`	Sets the MTU size, in bytes, of IP packets sent on an interface.
Step 6	**ip nhrp authentication** *string* **Example:** `Router(config-if)# ip nhrp authentication donttell`	Configures the authentication string for an interface using NHRP. **Note** The NHRP authentication string must be set to the same value on all hubs and spokes that are in the same DMVPN network.

	Command or Action	Purpose
Step 7	**ip nhrp map** *hub-tunnel-ip-address hub-physical-ip-address* **Example:** `Router(config-if)# ip nhrp map 10.0.0.1 172.17.0.1`	Statically configures the IP-to-NBMA address mapping of IP destinations connected to an NBMA network. • *hub-tunnel-ip-address* --Defines the NHRP server at the hub, which is permanently mapped to the static public IP address of the hub. • *hub-physical-ip-address* --Defines the static public IP address of the hub.
Step 8	**ip nhrp map multicast** *hub-physical-ip-address* **Example:** `Router(config-if)# ip nhrp map multicast 172.17.0.1`	Enables the use of a dynamic routing protocol between the spoke and hub, and sends multicast packets to the hub router.
Step 9	**ip nhrp nhs** *hub-tunnel-ip-address* **Example:** `Router(config-if)# ip nhrp nhs 10.0.0.1`	Configures the hub router as the NHRP next-hop server.
Step 10	**ip nhrp network-id** *number* **Example:** `Router(config-if)# ip nhrp network-id 99`	Enables NHRP on an interface. • The *number* argument specifies a globally unique 32-bit network identifier from a NBMA network. The range is from 1 to 4294967295.
Step 11	**tunnel source** {*ip-address* | *type number*} **Example:** `Router(config-if)# tunnel source Gigabitethernet 0/0/0`	Sets the source address for a tunnel interface.
Step 12	**tunnel key** *key-number* **Example:** `Router(config-if)# tunnel key 100000`	(Optional) Enables an ID key for a tunnel interface. • The *key-number* argument specifies a number from 0 to 4,294,967,295 that identifies the tunnel key. • The key number must be set to the same value on all hubs and spokes that are in the same DMVPN network.

Command or Action	Purpose

Step 13 Do one of the following:
• **tunnel mode gre multipoint**
•
•
•
•
• **tunnel destination** *hub-physical-ip-address*
 	
Example:
`Router(config-if)# tunnel mode gre multipoint`
 	
Example:
 	
Example:
`Router(config-if)# tunnel destination 172.17.0.1`
	Sets the encapsulation mode to mGRE for the tunnel interface.
	• Use this command if data traffic can use dynamic spoke-to-spoke traffic.
	
	Specifies the destination for a tunnel interface.
	• Use this command if data traffic can use hub-and-spoke tunnels.

Step 14 **tunnel protection ipsec profile** *name* **Example:** `Router(config-if)# tunnel protection ipsec profile vpnprof`	Associates a tunnel interface with an IPsec profile. • The *name* argument specifies the name of the IPsec profile; this value must match the *name* specified in the **crypto ipsec profile** *name*command.
Step 15 **bandwidth** *kbps* **Example:** `Router(config-if)# bandwidth 1000`	Sets the current bandwidth value for an interface to higher-level protocols. • The *kbps* argument specifies the bandwidth in kilobits per second. The default value is 9. The recommended bandwidth value is 1000 or greater. • The bandwidth setting for the spoke need not equal the bandwidth setting for the DMVPN hub. It is usually easier if all of the spokes use the same or similar value.

	Command or Action	Purpose
Step 16	**ip tcp adjust-mss** *max-segment-size* **Example:** `Router(config-if)# ip tcp adjust-mss 1360`	Adjusts the MSS value of TCP packets going through a router. • The *max-segment-size* argument specifies the maximum segment size, in bytes. The range is from 500 to 1460. • The recommended number value is 1360 when the number of IP MTU bytes is set to 1400. With these recommended settings, TCP sessions quickly scale back to 1400-byte IP packets so the packets will "fit" in the tunnel.
Step 17	**ip nhrp holdtime** *seconds* **Example:** `Router(config-if)# ip nhrp holdtime 450`	Changes the number of seconds that NHRP NBMA addresses are advertised as valid in authoritative NHRP responses. • The *seconds* argument specifies the time in seconds that NBMA addresses are advertised as valid in positive authoritative NHRP responses. The recommended value ranges from 300 seconds to 600 seconds.
Step 18	**delay** *number* **Example:** `Router(config-if)# delay 1000`	(Optional) Changes the EIGRP routing metric for routes learned over the tunnel interface. • The *number* argument specifies the delay time in seconds. The recommended value is 1000.

Configuring the Forwarding of Clear-Text Data IP Packets into a VRF

To configure the forwarding of clear-text data IP packets into a VRF, perform the following steps. This configuration assumes that the VRF Blue has already been configured.

Note To configure VRF Blue, use the **ip vrf** *vrf-name* command in global configuration mode.

SUMMARY STEPS

1. **enable**
2. configure terminal
3. **interface** *type number*
4. **ip vrf forwarding** *vrf-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	configure terminal **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface tunnel 0`	Configures an interface type and enters interface configuration mode.
Step 4	**ip vrf forwarding** *vrf-name* **Example:** `Router(config-if)# ip vrf forwarding Blue`	Allows the forwarding of clear-text data IP packets into a VRF.

Configuring the Forwarding of Encrypted Tunnel Packets into a VRF

To configure the forwarding of encrypted tunnel packets into a VRF, perform the following steps. This configuration assumes that the VRF Red has already been configured.

Note To configure VRF Red, use the **ip vrf** *vrf-name* command in global configuration mode.

SUMMARY STEPS

1. **enable**
2. configure terminal
3. **interface** *type number*
4. **tunnel vrf** *vrf-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	configure terminal **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface tunnel 0`	Configures an interface type and enters interface configuration mode.
Step 4	**tunnel vrf** *vrf-name* **Example:** `Router(config-if)# tunnel vrf RED`	Associates a VPN VRF instance with a specific tunnel destination, interface, or subinterface and allows the forwarding of encrypted tunnel packets into a VRF.

Configuring Traffic Segmentation Within DMVPN

Cisco IOS XE Release 2.5 introduces no new commands to use when configuring traffic segmentation, but you must complete the tasks described in the following sections in order to segment traffic within a DMVPN tunnel:

Prerequisites

The tasks that follow assume that the DMVPN tunnel and the VRFs Red and Blue have already been configured.

To configure VRF Red or Blue, use the **ip vrf** *vrf-name* command in global configuration mode.

For information on configuring a DMVPN tunnel, see the Configuring the Hub for DMVPN, page 9 and the Configuring the Spoke for DMVPN, page 13. For details about VRF configuration, see the Configuring the Forwarding of Clear-Text Data IP Packets into a VRF, page 17 and the Configuring the Forwarding of Encrypted Tunnel Packets into a VRF, page 18.

Enabling MPLS on the VPN Tunnel

Because traffic segmentation within a DMVPN tunnel depends upon MPLS, you must configure MPLS for each VRF instance in which traffic will be segmented.

Note On the Cisco ASR 1000 Series Aggregation Services Routers, only distributed switching is supported. Use the following commands for distributed switching: **ip multicast-routing** [**vrf** *vrf-name*] [**distributed**], **debug ip bgp vpnv4 unicast**, and **ip cef distributed.**

SUMMARY STEPS

1. **enable**
2. configure terminal
3. **interface** *type number*
4. **mpls ip**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	configure terminal **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface tunnel 0`	Configures an interface type and enters interface configuration mode.
Step 4	**mpls ip** **Example:** `Router(config-if)# mpls ip`	Enables MPLS tagging of packets on the specified tunnel interface.

Configuring Multiprotocol BGP on the Hub Router

You must configure multiprotocol iBGP (MP-iBGP) to enable advertisement of VPNv4 prefixes and labels to be applied to the VPN traffic. Use BGP to configure the hub as a Route Reflector. To force all traffic to

be routed via the hub, configure the BGP Route Reflector to change the next hop to itself when it advertises VPNv4 prefixes to the route reflector clients (spokes).

For more information about the BGP routing protocol, see the " Cisco BGP Overview " module in the *Cisco IOS XE IP Routing: BGP Configuration Guide*.

SUMMARY STEPS

1. **enable**
2. configure terminal
3. **router bgp** *autonomous-system-number*
4. **neighbor** *ipaddress* **remote-as** *as - number*
5. **neighbor** *ipaddress* **update-source** *interface*
6. **address-family vpnv4**
7. **neighbor** *ipaddress* **activate**
8. **neighbor** *ipaddress* **send-community extended**
9. **neighbor** *ipaddress* **route-reflector-client**
10. **neighbor** *ipaddress* **route-map nexthop out**
11. **exit**
12. **address-family ipv4** *vrf-name*
13. **redistribute connected**
14. **route-map** *map-tag* [**permit**| **deny**] [*sequence-number*]
15. **set ip next-hop** *ipaddress*

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	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 1`	Enables configuration of the BGP routing process.

	Command or Action	Purpose
Step 4	**neighbor** *ipaddress* **remote-as** *as - number* **Example:** `Router(config-router)# neighbor 10.0.0.11 remote-as 1`	Adds an entry to the BGP or multiprotocol BGP neighbor table.
Step 5	**neighbor** *ipaddress* **update-source** *interface* **Example:** `Router(config-router)# neighbor 10.10.10.11 update-source Tunnel1`	Configures the Cisco IOS XE software to allow BGP sessions to use any operational interface for TCP connections.
Step 6	**address-family vpnv4** **Example:** `Router(config)# address-family vpnv4`	Enters address family configuration mode to configure a routing session using VPNv4 address prefixes.
Step 7	**neighbor** *ipaddress* **activate** **Example:** `Router(config-router-af)# neighbor 10.0.0.11 activate`	Enables the exchange of information with a BGP neighbor.
Step 8	**neighbor** *ipaddress* **send-community extended** **Example:** `Router(config-router-af)# neighbor 10.0.0.11 send-community extended`	Specifies that extended community attributes should be sent to a BGP neighbor.
Step 9	**neighbor** *ipaddress* **route-reflector-client** **Example:** `Router(config-router-af)# neighbor 10.0.0.11 route-reflector-client`	Configures the router as a BGP Route Reflector and configures the specified neighbor as its client.
Step 10	**neighbor** *ipaddress* **route-map nexthop out** **Example:** `Router(config-router-af)# neighbor 10.0.0.11 route-map nexthop out`	Forces all traffic to be routed via the hub.

	Command or Action	Purpose
Step 11	**exit** **Example:** `Router(config-router-af)# exit`	Exits the address family configuration mode for VPNv4.
Step 12	**address-family ipv4** *vrf-name* **Example:** `Router(config)# address-family ipv4 red`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.
Step 13	**redistribute connected** **Example:** `Router(config-router-af)# redistribute connected`	Redistributes routes that are established automatically by virtue of having enabled IP on an interface from one routing domain into another routing domain.
Step 14	**route-map** *map-tag* [**permit**| **deny**] [*sequence-number*] **Example:** `Router(config-router-af)# route-map cisco permit 10`	Enters route map configuration mode to configure the next-hop that will be advertised to the spokes.
Step 15	**set ip next-hop** *ipaddress* **Example:** `Router(config-route-map)# set ip next-hop 10.0.0.1`	Sets the next hop to be the hub.

Configuring Multiprotocol BGP on the Spoke Routers

In order to segment traffic within a DMVPN tunnel, Multiprotocol-iBGP (MP-iBGP) must be configured on both the spoke routers and the hub. Perform the following task for each spoke router in the DMVPN.

SUMMARY STEPS

1. **enable**
2. configure terminal
3. **router bgp** *autonomous-system-number*
4. **neighbor** *ipaddress* **remote-as** *as - number*
5. **neighbor** *ipaddress* **update-source** *interface*
6. **address-family vpnv4**
7. **neighbor** *ipaddress* **activate**
8. **neighbor** *ipaddress* **send-community extended**
9. **exit**
10. **address-family ipv4** *vrf-name*
11. **redistribute connected**
12. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	configure terminal **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 1`	Enters BGP configuration mode.
Step 4	**neighbor** *ipaddress* **remote-as** *as - number* **Example:** `Router(config-router)# neighbor 10.0.0.1 remote-as 1`	Adds an entry to the BGP or multiprotocol BGP neighbor table.

	Command or Action	Purpose
Step 5	**neighbor** *ipaddress* **update-source** *interface* **Example:** `Router(config-router)# neighbor 10.10.10.1 update-source Tunnel1`	Configures the Cisco IOS XE software to allow BGP sessions to use any operational interface for TCP connections.
Step 6	**address-family vpnv4** **Example:** `Router(config)# address-family vpnv4`	Enters address family configuration mode to configure a routing session using VPNv4 address prefixes.
Step 7	**neighbor** *ipaddress* **activate** **Example:** `Router(config-router-af)# neighbor 10.0.0.1 activate`	Enables the exchange of information with a BGP neighbor.
Step 8	**neighbor** *ipaddress* **send-community extended** **Example:** `Router(config-router-af)# neighbor 10.0.0.1 send-community extended`	Specifies that extended community attributes should be sent to a BGP neighbor.
Step 9	**exit** **Example:** `Router(config-router-af)# exit`	Exits address family configuration mode.
Step 10	**address-family ipv4** *vrf-name* **Example:** `Router(config)# address-family ipv4 red`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.
Step 11	**redistribute connected** **Example:** `Router(config-router-af)# redistribute connected`	Redistributes routes that are established automatically by virtue of having enabled IP on an interface from one routing domain into another routing domain.

Command or Action	Purpose
Step 12 **exit** **Example:** `Router(config-router-af)# exit`	Exits address family configuration mode. **Note** Repeat Steps 10 through 12 for each VRF.

Troubleshooting Dynamic Multipoint VPN

After configuring DMVPN, perform the following optional steps in this task to verify that DMVPN is operating correctly, to clear DMVPN statistics or sessions, or to debug DMVPN. These commands may be used in any order.

SUMMARY STEPS

1. **clear dmvpn session**
2. **clear dmvpn statistics**
3. **debug dmvpn**
4. **debug dmvpn condition**
5. **debug nhrp condition**
6. **debug nhrp error**
7. **logging dmvpn**
8. **show crypto ipsec sa**
9. **show crypto isakmp sa**
10. **show crypto map**
11. **show dmvpn**
12. **show ip nhrp traffic**

DETAILED STEPS

Step 1 **clear dmvpn session**

This command clears DMVPN sessions. The following example clears only dynamic DMVPN sessions, for the specified tunnel:

Example:

```
Router# clear dmvpn session interface tunnel 5
```

The following example clears all DMVPN sessions, both static and dynamic, for the specified tunnel:

Example:

```
Router# clear dmvpn session interface tunnel 5 static
```

Step 2 **clear dmvpn statistics**

This command is used to clear DMVPN-related counters. The following example shows how to clear DMVPN-related session counters for the specified tunnel interface:

Example:

```
Router#
 clear dmvpn statistics interface tunnel 5
```

Step 3 **debug dmvpn**

This command is used to debug DMVPN sessions. You can enable or disable DMVPN debugging based on a specific condition. There are three levels of DMVPN debugging, listed in the order of details from lowest to highest:

- Error level
- Detail level
- Packet level

The following example shows how to enable conditional DMVPN debugging that displays all error debugs for NHRP, sockets, tunnel protection, and crypto information:

Example:

```
Router# debug dmvpn error all
```

Step 4 **debug dmvpn condition**

This command displays conditional debug DMVPN session information. The following example shows how to enable conditional debugging for a specific tunnel interface:

Example:

```
Router# debug dmvpn condition interface tunnel 5
```

Step 5 **debug nhrp condition**

This command enables or disables debugging based on a specific condition. The following example shows how to enable conditional NHRP debugging:

Example:

```
Router#
debug nhrp condition
```

Step 6 **debug nhrp error**

This command displays information about NHRP error activity. The following example shows how to enable debugging for NHRP error messages:

Example:

```
Router#
debug nhrp error
```

Step 7 **logging dmvpn**

This command is used to enable DMVPN system logging. The following example shows how to enable DMVPN system logging at the rate of 1 message every 20 seconds:

Example:

```
Router(config)#
logging dmvpn rate-limit 20
```

The following example shows a sample system log with DMVPN messages:

Example:

```
%DMVPN-7-CRYPTO_SS: Tunnel101-192.0.2.1 socket is UP
%DMVPN-5-NHRP_NHS: Tunnel101 192.0.2.251 is UP
%DMVPN-5-NHRP_CACHE: Client 192.0.2.2 on Tunnel1 Registered.
%DMVPN-5-NHRP_CACHE: Client 192.0.2.2 on Tunnel101 came UP.
%DMVPN-3-NHRP_ERROR: Registration Request failed for 192.0.2.251 on Tunnel101
```

Step 8 **show crypto ipsec sa**

This command displays the settings used by the current SAs. The following example output shows the IPsec SA status of only the active device:

Example:

```
Router#
show crypto ipsec sa active
interface: gigabitethernet0/0/0
    Crypto map tag: to-peer-outside, local addr 209.165.201.3
   protected vrf: (none
   local  ident (addr/mask/prot/port): (192.168.0.1/255.255.255.255/0/0)
   remote ident (addr/mask/prot/port): (172.16.0.1/255.255.255.255/0/0)
   current_peer 209.165.200.225 port 500
     PERMIT, flags={origin_is_acl,}
    #pkts encaps: 3, #pkts encrypt: 3, #pkts digest: 3
    #pkts decaps: 4, #pkts decrypt: 4, #pkts verify: 4
    #pkts compressed: 0, #pkts decompressed: 0
    #pkts not compressed: 0, #pkts compr. failed: 0
    #pkts not decompressed: 0, #pkts decompress failed: 0
    #send errors 0, #recv errors 0
     local crypto endpt.: 209.165.201.3, remote crypto endpt.: 209.165.200.225
     path mtu 1500, media mtu 1500
     current outbound spi: 0xD42904F0(3559458032)
     inbound esp sas:
      spi: 0xD3E9ABD0(3555306448)
        transform: esp-3des ,
        in use settings ={Tunnel, }
        conn id: 2006, flow_id: 6, crypto map: to-peer-outside
        sa timing: remaining key lifetime (k/sec): (4586265/3542)
        HA last key lifetime sent(k): (4586267)
        ike_cookies: 9263635C CA4B4E99 C14E908E 8EE2D79C
        IV size: 8 bytes
        replay detection support: Y
        Status: ACTIVE
```

Step 9 **show crypto isakmp sa**

This command displays all current IKE SAs at a peer. For example, the following sample output is displayed after IKE negotiations have successfully completed between two peers:

Example:

```
Router# show crypto isakmp sa
dst             src             state          conn-id    slot
172.17.63.19    172.16.175.76   QM_IDLE              2       0
```

```
172.17.63.19    172.17.63.20    QM_IDLE              1       0
172.16.175.75   172.17.63.19    QM_IDLE              3       0
```

Step 10 **show crypto map**

This command displays the crypto map configuration. The following sample output is displayed after a crypto map has been configured:

Example:

```
Router# show crypto map
Crypto Map "Tunnel5-head-0" 10 ipsec-isakmp
        Profile name: vpnprof
        Security association lifetime: 4608000 kilobytes/3600 seconds
        PFS (Y/N): N
        Transform sets={trans2, }
Crypto Map "Tunnel5-head-0" 20 ipsec-isakmp
        Map is a PROFILE INSTANCE.
        Peer = 172.16.175.75
        Extended IP access list
            access-list permit gre host 172.17.63.19 host 172.16.175.75
        Current peer: 172.16.175.75
        Security association lifetime: 4608000 kilobytes/3600 seconds
        PFS (Y/N): N
        Transform sets={trans2, }
Crypto Map "Tunnel5-head-0" 30 ipsec-isakmp
        Map is a PROFILE INSTANCE.
        Peer = 172.17.63.20
        Extended IP access list
            access-list permit gre host 172.17.63.19 host 172.17.63.20
        Current peer: 172.17.63.20
        Security association lifetime: 4608000 kilobytes/3600 seconds
        PFS (Y/N): N
        Transform sets={trans2, }
Crypto Map "Tunnel5-head-0" 40 ipsec-isakmp
        Map is a PROFILE INSTANCE.
        Peer = 172.16.175.76
        Extended IP access list
            access-list permit gre host 172.17.63.19 host 172.16.175.76
        Current peer: 172.16.175.76
        Security association lifetime: 4608000 kilobytes/3600 seconds
        PFS (Y/N): N
        Transform sets={trans2, }
        Interfaces using crypto map Tunnel5-head-0:
```

Tunnel5

Step 11 **show dmvpn**

This command displays DMVPN-specific session information. The following sample shows example summary output:

Example:

```
Router# show dmvpn
Legend: Attrb --> S - Static, D - Dynamic, I - Incomplete
        N - NATed, L - Local, X - No Socket
        # Ent --> Number of NHRP entries with same NBMA peer
! The line below indicates that the sessions are being displayed for Tunnel1.
! Tunnel1 is acting as a spoke and is a peer with three other NBMA peers.
Tunnel1, Type: Spoke, NBMA Peers: 3,
 # Ent  Peer NBMA Addr Peer Tunnel Add State  UpDn Tm Attrb
 ----- --------------- --------------- ----- -------- -----
     2      192.0.2.21     192.0.2.116   IKE     3w0d D
     1     192.0.2.102      192.0.2.11  NHRP 02:40:51 S
     1     192.0.2.225      192.0.2.10    UP     3w0d S
Tunnel2, Type: Spoke, NBMA Peers: 1,
 # Ent  Peer NBMA Addr Peer Tunnel Add State  UpDn Tm Attrb
```

```
 ----- --------------- --------------- ----- -------- -----
     1        192.0.2.25      192.0.2.171   IKE    never S
```

Step 12 **show ip nhrp traffic**

This command displays NHRP statistics. The following example shows output for a specific tunnel (tunnel7):

Example:

```
Router# s
how ip nhrp traffic interface tunnel7
Tunnel7: Max-send limit:100Pkts/10Sec, Usage:0%
   Sent: Total 79
         18 Resolution Request  10 Resolution Reply  42 Registration Request
         0 Registration Reply    3 Purge Request      6 Purge Reply
         0 Error Indication      0 Traffic Indication
   Rcvd: Total 69
         10 Resolution Request  15 Resolution Reply  0 Registration Request
         36 Registration Reply   6 Purge Request     2 Purge Reply
         0 Error Indication      0 Traffic Indication
```

- What to Do Next, page 30

What to Do Next

If you still have a problem after troubleshooting your DMVPN configuration, you may use the **show tech-support** command. This command includes information for DMVPN sessions. For more information, see the **show tech-support**command in the Cisco IOS Configuration Fundamentals Command Reference .

Configuration Examples for Dynamic Multipoint VPN Feature

Example Hub Configuration for DMVPN

In the following example, which configures the hub router for multipoint GRE and IPsec integration, no explicit configuration lines are needed for each spoke; that is, the hub is configured with a global IPsec policy template that all spoke routers can talk to. In this example, EIGRP is configured to run over the private physical interface and the tunnel interface.

```
crypto isakmp policy 1
 authentication pre-share
crypto isakmp key cisco47 address 0.0.0.0
!
crypto ipsec transform-set trans2 esp-des esp-md5-hmac
mode transport
!
crypto ipsec profile vpnprof
 set transform-set trans2
!
```

```
interface Tunnel0
 bandwidth 1000
 ip address 10.0.0.1 255.255.255.0
! Ensures longer packets are fragmented before they are encrypted; otherwise, the
receiving router would have to do the reassembly.
 ip mtu 1400
! The following line must match on all nodes that “want to use” this mGRE tunnel:
 ip nhrp authentication donttell
! Note that the next line is required only on the hub.
 ip nhrp map multicast dynamic
! The following line must match on all nodes that want to use this mGRE tunnel:
 ip nhrp network-id 99
 ip nhrp holdtime 300
! Turns off split horizon on the mGRE tunnel interface; otherwise, EIGRP will not
advertise routes that are learned via the mGRE interface back out that interface.
 no ip split-horizon eigrp 1
! Enables dynamic, direct spoke-to-spoke tunnels when using EIGRP.
 no ip next-hop-self eigrp 1
 ip tcp adjust-mss 1360
 delay 1000
! Sets IPsec peer address to Ethernet interface’s public address.
 tunnel source Gigabitethernet 0/0/0
 tunnel mode gre multipoint
! The following line must match on all nodes that want to use this mGRE tunnel.
 tunnel key 100000
 tunnel protection ipsec profile vpnprof
!
interface FastEthernet0/0/0
 ip address 172.17.0.1 255.255.255.0
!
interface FastEthernet0/0/1
 ip address 192.168.0.1 255.255.255.0
!
router eigrp 1
 network 10.0.0.0 0.0.0.255
 network 192.168.0.0 0.0.0.255
!
```

For information about defining and configuring ISAKMP profiles, see the “ Certificate to ISAKMP Profile Mapping ” module in the *Cisco IOS XE Security Configuration Guide: Secure Connectivity* .

Example Spoke Configuration for DMVPN

In the following example, all spokes are configured the same except for tunnel and local interface address, thereby reducing necessary configurations for the user:

```
crypto isakmp policy 1
 authentication pre-share
crypto isakmp key cisco47 address 0.0.0.0
!
crypto ipsec transform-set trans2 esp-des esp-md5-hmac
 mode transport
!
crypto ipsec profile vpnprof
 set transform-set trans2
!
interface Tunnel0
 bandwidth 1000
 ip address 10.0.0.2 255.255.255.0
 ip mtu 1400
! The following line must match on all nodes that want to use this mGRE tunnel:
 ip nhrp authentication donttell
! Definition of NHRP server at the hub (10.0.0.1), which is permanently mapped to the
static public address of the hub (172.17.0.1).
 ip nhrp map 10.0.0.1 172.17.0.1
! Sends multicast packets to the hub router, and enables the use of a dynamic routing
protocol between the spoke and the hub.
 ip nhrp map multicast 172.17.0.1
! The following line must match on all nodes that want to use this mGRE tunnel:
 ip nhrp network-id 99
```

```
 ip nhrp holdtime 300
! Configures the hub router as the NHRP next-hop server.
 ip nhrp nhs 10.0.0.1
 ip tcp adjust-mss 1360
 delay 1000
 tunnel source Gigabitethernet 0/0/0
 tunnel mode gre multipoint
! The following line must match on all nodes that want to use this mGRE tunnel:
 tunnel key 100000
 tunnel protection ipsec profile vpnprof
!
! This is a spoke, so the public address might be dynamically assigned via DHCP.
interface FastEthernet0/0/0
 ip address dhcp hostname Spoke1
!
interface FastEthernet0/0/1
 ip address 192.168.1.1 255.255.255.0
!
! EIGRP is configured to run over the inside physical interface and the tunnel.
router eigrp 1
 network 10.0.0.0 0.0.0.255
 network 192.168.1.0 0.0.0.255
```

Example 2547oDMVPN with BGP Only Traffic Segmentation

The following example show a traffic segmentation configuration in which traffic is segmented between two spokes that serve as PE devices:

Hub Configuration

```
hostname hub-pe1
boot-start-marker
boot-end-marker
no aaa new-model
resource policy
clock timezone EST 0
ip cef
no ip domain lookup
!This section refers to the forwarding table for VRF blue:
ip vrf blue
 rd 2:2
 route-target export 2:2
 route-target import 2:2
!This section refers to the forwarding table for VRF red:
ip vrf red
 rd 1:1
 route-target export 1:1
 route-target import 1:1
mpls label protocol ldp
crypto isakmp policy 1
 authentication pre-share
crypto isakmp key cisco address 0.0.0.0 0.0.0.0
crypto ipsec transform-set t1 esp-des
 mode transport
crypto ipsec profile prof
 set transform-set t1
interface Tunnel1
 ip address 10.9.9.1 255.255.255.0
 no ip redirects
 ip nhrp authentication cisco
 ip nhrp map multicast dynamic
 ip nhrp network-id 1
!The command below enables MPLS on the DMVPN network:
mpls ip
 tunnel source Gigabitethernet 0/0/0
 tunnel mode gre multipoint
 tunnel protection ipsec profile prof
interface Loopback0
 ip address 10.0.0.1 255.255.255.255
```

```
interface Ethernet0/0/0
 ip address 172.0.0.1 255.255.255.0
!The multiprotocol BGP route reflector (the hub) configuration changes the next-hop
information to set itself as the next-hop and assigns a new VPN label for the prefixes
learned from the spokes and advertises the VPN prefix:
router bgp 1
 no synchronization
 bgp log-neighbor-changes
 neighbor 10.0.0.11 remote-as 1
 neighbor 10.0.0.11 update-source Tunnel1
 neighbor 10.0.0.12 remote-as 1
 neighbor 10.0.0.12 update-source Tunnel1
 no auto-summary
 address-family vpnv4
 neighbor 10.0.0.11 activate
 neighbor 10.0.0.11 send-community extended
 neighbor 10.0.0.11 route-reflector-client
 neighbor 10.0.0.11 route-map nexthop out
 neighbor 10.0.0.12 activate
 neighbor 10.0.0.12 send-community extended
 neighbor 10.0.0.12 route-reflector-client
 neighbor 10.0.0.12 route-map nexthop out
 exit
 address-family ipv4 vrf red
 redistribute connected
 no synchronization
 exit
 address-family ipv4 vrf blue
 redistribute connected
 no synchronization
 exit
no ip http server
no ip http secure-server
!In this route map information, the hub sets the next hop to itself, and the VPN prefixes
are advertised:
route-map cisco permit 10
 set ip next-hop 10.0.0.1
control-plane
line con 0
 logging synchronous
line aux 0
line vty 0 4
 no login
end
```

Spoke Configurations

Spoke 2

```
hostname spoke-pe2
boot-start-marker
boot-end-marker
no aaa new-model
resource policy
clock timezone EST 0
ip cef
no ip domain lookup
!This section refers to the forwarding table for VRF blue:
ip vrf blue
 rd 2:2
 route-target export 2:2
 route-target import 2:2
!This section refers to the forwarding table for VRF red:
ip vrf red
 rd 1:1
 route-target export 1:1
 route-target import 1:1
mpls label protocol ldp
crypto isakmp policy 1
 authentication pre-share
crypto isakmp key cisco address 0.0.0.0 0.0.0.0
```

```
crypto ipsec transform-set t1 esp-des
 mode transport
crypto ipsec profile prof
 set transform-set t1
interface Tunnel1
 ip address 10.0.0.11 255.255.255.0
 no ip redirects
 ip nhrp authentication cisco
 ip nhrp map multicast dynamic
 ip nhrp map 10.0.0.1 172.0.0.1
 ip nhrp map multicast 172.0.0.1
 ip nhrp network-id 1
 ip nhrp nhs 10.0.0.1
!The command below enables MPLS on the DMVPN network:
 mpls ip
 tunnel source Gigabitethernet 0/0/0
 tunnel mode gre multipoint
 tunnel protection ipsec profile prof
interface Loopback0
 ip address 10.9.9.11 255.255.255.255
interface FastEthernet0/0/0
 ip address 172.0.0.11 255.255.255.0
!
!
interface FastEthernet1/0/0
 ip vrf forwarding red
 ip address 192.168.11.2 255.255.255.0
interface FastEthernet2/0/0
 ip vrf forwarding blue
 ip address 192.168.11.2 255.255.255.0
!The multiprotocol BGP route reflector (the hub) configuration changes the next-hop
information to set itself as the next-hop and assigns a new VPN label for the prefixes
learned from the spokes and advertises the VPN prefix:
router bgp 1
 no synchronization
 bgp log-neighbor-changes
 neighbor 10.0.0.1 remote-as 1
 neighbor 10.0.0.1 update-source Tunnel1
 no auto-summary
 address-family vpnv4
 neighbor 10.0.0.1 activate
 neighbor 10.0.0.1 send-community extended
 exit
!
 address-family ipv4 vrf red
 redistribute connected
 no synchronization
 exit
 !
 address-family ipv4 vrf blue
 redistribute connected
 no synchronization
 exit
no ip http server
no ip http secure-server
control-plane
line con 0
 logging synchronous
line aux 0
line vty 0 4
 no login
end
```

Spoke 3

```
hostname spoke-PE3
boot-start-marker
boot-end-marker
no aaa new-model
resource policy
clock timezone EST 0
ip cef
```

```
no ip domain lookup
!This section refers to the forwarding table for VRF blue:
ip vrf blue
 rd 2:2
 route-target export 2:2
 route-target import 2:2
!This section refers to the forwarding table for VRF red:
ip vrf red
 rd 1:1
 route-target export 1:1
 route-target import 1:1
mpls label protocol ldp
crypto isakmp policy 1
 authentication pre-share
crypto isakmp key cisco address 0.0.0.0 0.0.0.0
crypto ipsec transform-set t1 esp-des
 mode transport
crypto ipsec profile prof
 set transform-set t1
interface Tunnel1
 ip address 10.0.0.12 255.255.255.0
 no ip redirects
 ip nhrp authentication cisco
 ip nhrp map multicast dynamic
 ip nhrp map 10.0.0.1 172.0.0.1
 ip nhrp map multicast 172.0.0.1
 ip nhrp network-id 1
 ip nhrp nhs 10.0.0.1
!The command below enables MPLS on the DMVPN network:
 mpls ip
 tunnel source Gigabitethernet 0/0/0
 tunnel mode gre multipoint
 tunnel protection ipsec profile prof
!
interface Loopback0
 ip address 10.9.9.12 255.255.255.255
interface FastEthernet0/0/0
 ip address 172.0.0.12 255.255.255.0
interface FastEthernet1/0/0
 ip vrf forwarding red
 ip address 192.168.12.2 255.255.255.0
interface FastEthernet2/0/0
 ip vrf forwarding blue
 ip address 192.168.12.2 255.255.255.0
!The multiprotocol BGP route reflector (the hub) configuration changes the next-hop
information to set itself as the next-hop and assigns a new VPN label for the prefixes
learned from the spokes and advertises the VPN prefix:
router bgp 1
 no synchronization
 bgp log-neighbor-changes
 neighbor 10.0.0.1 remote-as 1
 neighbor 10.0.0.1 update-source Tunnel1
 no auto-summary
 address-family vpnv4
 neighbor 10.0.0.1 activate
 neighbor 10.0.0.1 send-community extended
 exit
 address-family ipv4 vrf red
 redistribute connected
 no synchronization
 exit
 address-family ipv4 vrf blue
 redistribute connected
 no synchronization
 exit
no ip http server
no ip http secure-server
control-plane
line con 0
 logging synchronous
line aux 0
line vty 0 4
```

```
 no login
end
```

Example 2547oDMVPN with Enterprise Branch Traffic Segmentation

The following example shows a configuration for segmenting traffic between two spokes located at branch offices of an enterprise. In this example, EIGRP is configured to learn routes to reach BGP neighbors within the DMVPN.

Hub Configuration

```
hostname HUB
boot-start-marker
boot-end-marker
no aaa new-model
resource policy
clock timezone EST 0
ip cef
no ip domain lookup
!This section refers to the forwarding table for VRF blue:
ip vrf blue
 rd 2:2
 route-target export 2:2
 route-target import 2:2
!This refers to the forwarding table for VRF red:
ip vrf red
 rd 1:1
 route-target export 1:1
 route-target import 1:1
mpls label protocol ldp
crypto isakmp policy 1
 authentication pre-share
crypto isakmp key cisco address 0.0.0.0 0.0.0.0
crypto ipsec transform-set t1 esp-des
 mode transport
crypto ipsec profile prof
 set transform-set t1
interface Tunnel1
 ip address 10.0.0.1 255.255.255.0
 no ip redirects
 ip nhrp authentication cisco
 ip nhrp map multicast dynamic
 ip nhrp network-id 1
!EIGRP is enabled on the DMVPN network to learn the IGP prefixes:
no ip split-horizon eigrp 1
!The command below enables MPLS on the DMVPN network:
 mpls ip
 tunnel source Gigabitethernet 0/0/0
 tunnel mode gre multipoint
 tunnel protection ipsec profile prof
!This address is advertised by EIGRP and used as the BGP endpoint:
interface Loopback0
 ip address 10.9.9.1 255.255.255.255
interface FastEthernet0/0/0
 ip address 172.0.0.1 255.255.255.0
!EIGRP is configured to learn the BGP peer addresses (10.9.9.x networks)
router eigrp 1
 network 10.9.9.1 0.0.0.0
 network 10.0.0.0 0.0.0.255
 no auto-summary
!The multiprotocol BGP route reflector (the hub) configuration changes the next-hop
information to set itself as the next-hop and assigns a new VPN label for the prefixes
learned from the spokes and advertises the VPN prefix:
router bgp 1
 no synchronization
 bgp router-id 10.9.9.1
 bgp log-neighbor-changes
 neighbor 10.9.9.11 remote-as 1
 neighbor 10.9.9.11 update-source Loopback0
```

```
 neighbor 10.9.9.12 remote-as 1
 neighbor 10.9.9.12 update-source Loopback0
 no auto-summary
 address-family vpnv4
 neighbor 10.9.9.11 activate
 neighbor 10.9.9.11 send-community extended
 neighbor 10.9.9.11 route-reflector-client
 neighbor 10.9.9.12 activate
 neighbor 10.9.9.12 send-community extended
 neighbor 10.9.9.12 route-reflector-client
 exit
 address-family ipv4 vrf red
 redistribute connected
 no synchronization
 exit
 address-family ipv4 vrf blue
 redistribute connected
 no synchronization
 exit
no ip http server
no ip http secure-server
control-plane
line con 0
 logging synchronous
line aux 0
line vty 0 4
 no login
end
```

Spoke Configurations

Spoke 2

```
hostname Spoke2
boot-start-marker
boot-end-marker
no aaa new-model
resource policy
clock timezone EST 0
ip cef
no ip domain lookup
!This section refers to the forwarding table for VRF blue:
ip vrf blue
 rd 2:2
 route-target export 2:2
 route-target import 2:2
!This section refers to the forwarding table for VRF red:
ip vrf red
 rd 1:1
 route-target export 1:1
 route-target import 1:1
mpls label protocol ldp
crypto isakmp policy 1
 authentication pre-share
crypto isakmp key cisco address 0.0.0.0 0.0.0.0
crypto ipsec transform-set t1 esp-des
 mode transport
crypto ipsec profile prof
 set transform-set t1
interface Tunnel1
 ip address 10.0.0.11 255.255.255.0
 no ip redirects
 ip nhrp authentication cisco
 ip nhrp map multicast dynamic
 ip nhrp map 10.0.0.1 172.0.0.1
 ip nhrp map multicast 172.0.0.1
 ip nhrp network-id 1
 ip nhrp nhs 10.0.0.1
!The command below enables MPLS on the DMVPN network:
 mpls ip
 tunnel source Gigabitethernet 0/0/0
```

```
 tunnel mode gre multipoint
 tunnel protection ipsec profile prof
!This address is advertised by EIGRP and used as the BGP endpoint:
interface Loopback0
 ip address 10.9.9.11 255.255.255.255
interface FastEthernet0/0/0
 ip address 172.0.0.11 255.255.255.0
interface FastEthernet1/0/0
 ip vrf forwarding red
 ip address 192.168.11.2 255.255.255.0
interface FastEthernet2/0/0
 ip vrf forwarding blue
 ip address 192.168.11.2 255.255.255.0
!EIGRP is enabled on the DMVPN network to learn the IGP prefixes:
router eigrp 1
 network 10.9.9.11 0.0.0.0
 network 10.0.0.0 0.0.0.255
 no auto-summary
!The multiprotocol BGP route reflector (the hub) configuration changes the next-hop
information to set itself as the next-hop and assigns a new VPN label for the prefixes
learned from the spokes and advertises the VPN prefix:
router bgp 1
 no synchronization
 bgp router-id 10.9.9.11
 bgp log-neighbor-changes
 neighbor 10.9.9.1 remote-as 1
 neighbor 10.9.9.1 update-source Loopback0
 no auto-summary
 address-family vpnv4
 neighbor 10.9.9.1 activate
 neighbor 10.9.9.1 send-community extended
 exit
 address-family ipv4 vrf red
 redistribute connected
 no synchronization
 exit
 address-family ipv4 vrf blue
 redistribute connected
 no synchronization
 exit
no ip http server
no ip http secure-server
control-plane
line con 0
 logging synchronous
line aux 0
line vty 0 4
 no login
end
```

Spoke 3

```
hostname Spoke3
boot-start-marker
boot-end-marker
no aaa new-model
resource policy
clock timezone EST 0
ip cef
no ip domain lookup
!This section refers to the forwarding table for VRF blue:
ip vrf blue
 rd 2:2
 route-target export 2:2
 route-target import 2:2
!This section refers to the forwarding table for VRF red:
ip vrf red
 rd 1:1
 route-target export 1:1
 route-target import 1:1
mpls label protocol ldp
crypto isakmp policy 1
```

```
 authentication pre-share
crypto isakmp key cisco address 0.0.0.0 0.0.0.0
crypto ipsec transform-set t1 esp-des
 mode transport
crypto ipsec profile prof
 set transform-set t1
interface Tunnel1
 ip address 10.0.0.12 255.255.255.0
 no ip redirects
 ip nhrp authentication cisco
 ip nhrp map multicast dynamic
 ip nhrp map 10.0.0.1 172.0.0.1
 ip nhrp map multicast 172.0.0.1
 ip nhrp network-id 1
 ip nhrp nhs 10.0.0.1
!The command below enables MPLS on the DMVPN network:
 mpls ip
 tunnel source Gigabitethernet 0/0/0
 tunnel mode gre multipoint
 tunnel protection ipsec profile prof
!This address is advertised by EIGRP and used as the BGP endpoint:
interface Loopback0
 ip address 10.9.9.12 255.255.255.255
interface FastEthernet0/0/0
 ip address 172.0.0.12 255.255.255.0
interface FastEthernet1/0/0
 ip vrf forwarding red
 ip address 192.168.12.2 255.255.255.0
interface FastEthernet2/0/0
 ip vrf forwarding blue
 ip address 192.168.12.2 255.255.255.0
!EIGRP is enabled on the DMVPN network to learn the IGP prefixes:
router eigrp 1
 network 10.9.9.12 0.0.0.0
 network 10.0.0.0 0.0.0.255
 no auto-summary
!The multiprotocol BGP route reflector (the hub) configuration changes the next-hop
information to set itself as the next-hop and assigns a new VPN label for the prefixes
learned from the spokes and advertises the VPN prefix:
router bgp 1
 no synchronization
 bgp router-id 10.9.9.12
 bgp log-neighbor-changes
 neighbor 10.9.9.1 remote-as 1
 neighbor 10.9.9.1 update-source Loopback0
 no auto-summary
 address-family vpnv4
 neighbor 10.9.9.1 activate
 neighbor 10.9.9.1 send-community extended
 exit
 address-family ipv4 vrf red
 redistribute connected
 no synchronization
 exit
 address-family ipv4 vrf blue
 redistribute connected
 no synchronization
 exit
no ip http server
no ip http secure-server
control-plane
line con 0
 logging synchronous
line aux 0
line vty 0 4
 no login
end
```

Sample Command Output: show mpls ldp bindings

```
Spoke2# show mpls ldp bindings
  tib entry: 10.9.9.1/32, rev 8
```

```
        local binding:  tag: 16
        remote binding: tsr: 10.9.9.1:0, tag: imp-null
  tib entry: 10.9.9.11/32, rev 4
        local binding:  tag: imp-null
        remote binding: tsr: 10.9.9.1:0, tag: 16
  tib entry: 10.9.9.12/32, rev 10
        local binding:  tag: 17
        remote binding: tsr: 10.9.9.1:0, tag: 17
  tib entry: 10.0.0.0/24, rev 6
        local binding:  tag: imp-null
        remote binding: tsr: 10.9.9.1:0, tag: imp-null
  tib entry: 172.0.0.0/24, rev 3
        local binding:  tag: imp-null
        remote binding: tsr: 10.9.9.1:0, tag: imp-null
Spoke2#
```

Sample Command Output: show mpls forwarding-table

```
Spoke2# show mpls forwarding-table

Local  Outgoing    Prefix            Bytes tag  Outgoing   Next Hop
tag    tag or VC   or Tunnel Id      switched   interface
16     Pop tag     10.9.9.1/32        0          Tu1        10.0.0.1
17     17          10.9.9.12/32       0          Tu1        10.0.0.1
18     Aggregate   192.168.11.0/24[V]   \
                                     0
19     Aggregate   192.168.11.0/24[V]   \
                                     0
Spoke2#
```

Sample Command Output: show ip route vrf red

```
Spoke2# show ip route vrf red
Routing Table: red
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route
Gateway of last resort is not set
B    192.168.12.0/24 [200/0] via 10.9.9.12, 00:00:02
C    192.168.11.0/24 is directly connected, FastEthernet1/0/0
Spoke2#
```

Sample Command Output: show ip route vrf blue

```
Spoke2# show ip route vrf blue
Routing Table: blue
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route
Gateway of last resort is not set
B    192.168.12.0/24 [200/0] via 10.9.9.12, 00:00:08
C    192.168.11.0/24 is directly connected, FastEthernet2/0/0
Spoke2#
Spoke2# show ip cef vrf red 192.168.12.0
192.168.12.0/24, version 5, epoch 0
0 packets, 0 bytes
  tag information set
    local tag: VPN-route-head
    fast tag rewrite with Tu1, 10.0.0.1, tags imposed: {17 18}
  via 10.9.9.12, 0 dependencies, recursive
    next hop 10.0.0.1, Tunnel1 via 10.9.9.12/32
```

```
    valid adjacency
    tag rewrite with Tu1, 10.0.0.1, tags imposed: {17 18}
Spoke2#
```

Sample Command Output: show ip bgp neighbors

```
Spoke2# show ip bgp neighbors

BGP neighbor is 10.9.9.1,  remote AS 1, internal link
  BGP version 4, remote router ID 10.9.9.1
  BGP state = Established, up for 00:02:09
  Last read 00:00:08, last write 00:00:08, hold time is 180, keepalive interval is 60
seconds
  Neighbor capabilities:
    Route refresh: advertised and received(old & new)
    Address family IPv4 Unicast: advertised and received
    Address family VPNv4 Unicast: advertised and received
  Message statistics:
    InQ depth is 0
    OutQ depth is 0
                         Sent       Rcvd
    Opens:                  1          1
    Notifications:          0          0
    Updates:                4          4
    Keepalives:             4          4
    Route Refresh:          0          0
    Total:                  9          9
  Default minimum time between advertisement runs is 0 seconds
 For address family: IPv4 Unicast
  BGP table version 1, neighbor version 1/0
 Output queue size : 0
  Index 1, Offset 0, Mask 0x2
  1 update-group member
                               Sent       Rcvd
  Prefix activity:             ----       ----
    Prefixes Current:             0          0
    Prefixes Total:               0          0
    Implicit Withdraw:            0          0
    Explicit Withdraw:            0          0
    Used as bestpath:           n/a          0
    Used as multipath:          n/a          0
                                 Outbound    Inbound
  Local Policy Denied Prefixes:  --------    -------
    Total:                              0          0
  Number of NLRIs in the update sent: max 0, min 0
 For address family: VPNv4 Unicast
  BGP table version 9, neighbor version 9/0
 Output queue size : 0
  Index 1, Offset 0, Mask 0x2
  1 update-group member
                               Sent       Rcvd
  Prefix activity:             ----       ----
    Prefixes Current:             2          2 (Consumes 136 bytes)
    Prefixes Total:               4          2
    Implicit Withdraw:            2          0
    Explicit Withdraw:            0          0
    Used as bestpath:           n/a          2
    Used as multipath:          n/a          0
                                 Outbound    Inbound
  Local Policy Denied Prefixes:  --------    -------
    ORIGINATOR loop:                 n/a          2
    Bestpath from this peer:           4        n/a
    Total:                             4          2
  Number of NLRIs in the update sent: max 1, min 1
  Connections established 1; dropped 0
  Last reset never
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Connection is ECN Disabled
Local host: 10.9.9.11, Local port: 179
Foreign host: 10.9.9.1, Foreign port: 12365
Enqueued packets for retransmit: 0, input: 0  mis-ordered: 0 (0 bytes)
Event Timers (current time is 0x2D0F0):
```

```
Timer          Starts    Wakeups            Next
Retrans             6          0             0x0
TimeWait            0          0             0x0
AckHold             7          3             0x0
SendWnd             0          0             0x0
KeepAlive           0          0             0x0
GiveUp              0          0             0x0
PmtuAger            0          0             0x0
DeadWait            0          0             0x0
iss: 3328307266  snduna: 3328307756  sndnxt: 3328307756     sndwnd:  15895
irs: 4023050141  rcvnxt: 4023050687  rcvwnd:      16384  delrcvwnd:      0
SRTT: 165 ms, RTTO: 1457 ms, RTV: 1292 ms, KRTT: 0 ms
minRTT: 0 ms, maxRTT: 300 ms, ACK hold: 200 ms
Flags: passive open, nagle, gen tcbs
IP Precedence value : 6
Datagrams (max data segment is 536 bytes):
Rcvd: 13 (out of order: 0), with data: 7, total data bytes: 545
Sent: 11 (retransmit: 0, fastretransmit: 0, partialack: 0, Second Congestion: 0), with
data: 6, total data bytes: 489
Spoke2#
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Call Admission Control	" Call Admission Control for IKE " module in the *Cisco IOS XE Security Configuration Guide: Secure Connectivity*
IKE configuration tasks such as defining an IKE policy	" Configuring Internet Key Exchange for IPSec VPNs " module in the *Cisco IOS XE Security Configuration Guide: Secure Connectivity*
IPsec configuration tasks	" Configuring Security for VPNs with IPsec " module in the *Cisco IOS XE Security Configuration Guide: Secure Connectivity*
Configuring VRF-aware IPsec	" VRF-Aware IPsec " module in the *Cisco IOS XE Security Configuration Guide: Secure Connectivity*
Configuring MPLS	" Multiprotocol Label Switching (MPLS) on Cisco Routers " module in the *Cisco IOS XE Multiprotocol Label Switching Configuration Guide*
Configuring BGP	" Cisco BGP Overview " module in the *Cisco IOS XE IP Routing: BGP Configuration Guide*
System messages	System Messages for Cisco IOS XE Software
Defining and configuring ISAKMP profiles	" Certificate to ISAKMP Profile Mapping " module in the *Cisco IOS XE Security Configuration Guide: Secure Connectivity*

Related Topic	Document Title
Security commands	Cisco IOS Security Command Reference

Standards

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

MIBs

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

RFCs

RFCs	Title
RFC 2547	BGP/MPLS 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 Multipoint VPN

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 Multipoint VPN

Feature Name	Releases	Feature Information
Dynamic Multipoint VPN (DMVPN) Phase 1	Cisco IOS XE Release 2.1	The Dynamic Multipoint VPN (DMVPN) feature allows users to better scale large and small IPsec Virtual Private Networks (VPNs) by combining generic routing encapsulation (GRE) tunnels, IP security (IPsec) encryption, and Next Hop Resolution Protocol (NHRP).
DMVPN Phase 2	Cisco IOS XE Release 2.1	DMVPN Spoke-to-Spoke functionality was made more production ready.
NAT-Transparency Aware DMVPN	Cisco IOS XE Release 2.1	The Network Address Translation-Transparency (NAT-T) Aware DMVPN enhancement was added. In addition, DMVPN hub-to-spoke functionality was made more production ready.
Manageability Enhancements for DMVPN	Cisco IOS XE Release 2.5	DMVPN session manageability was expanded with DMVPN-specific commands for debugging, show output, session and counter control, and system log information. The following section provides information about this feature: • Troubleshooting Dynamic Multipoint VPN The following commands were introduced or modified by this feature: **clear dmvpn session**, **clear dmvpn statistics**, **debug dmvpn**, **debug dmvpn condition**, **debug nhrp condition**, **debug nhrp error**, **logging dmvpn**, **show dmvpn**, **show ip nhrp traffic**

Feature Name	Releases	Feature Information
DMVPN--Enabling Traffic Segmentation Within DMVPN	Cisco IOS XE Release 2.5	The 2547oDMVPN feature allows users to segment VPN traffic within a DMVPN tunnel by applying MPLS labels to VRF instances to indicate the source and destination of each VRF.

Glossary

AM --aggressive mode. A mode during IKE negotiation. Compared to MM, AM eliminates several steps, making it faster but less secure than MM. Cisco IOS XE software will respond in aggressive mode to an IKE peer that initiates aggressive mode.

GRE --generic routing encapsulation. Tunnels that provide a specific pathway across the shared WAN and encapsulate traffic with new packet headers to ensure delivery to specific destinations. The network is private because traffic can enter a tunnel only at an endpoint. Tunnels do not provide true confidentiality (encryption does) but can carry encrypted traffic.

GRE tunneling can also be used to encapsulate non-IP traffic into IP and send it over the Internet or IP network. The Internet Package Exchange (IPX) and AppleTalk protocols are examples of non-IP traffic.

IKE --Internet Key Exchange. A hybrid protocol that implements Oakley key exchange and Skeme key exchange inside the ISAKMP framework. Although IKE can be used with other protocols, its initial implementation is with IPsec. IKE provides authentication of the IPsec peers, negotiates IPsec keys, and negotiates IPsec security associations.

IPsec --IP security. A framework of open standards developed by the Internet Engineering Task Force (IETF). IPsec provides security for transmission of sensitive information over unprotected networks such as the Internet. IPsec acts at the network layer, protecting and authenticating IP packets between participating IPsec devices (“peers”), such as Cisco routers.

ISAKMP--Internet Security Association Key Management Protocol. A protocol framework that defines payload formats, the mechanics of implementing a key exchange protocol, and the negotiation of a security association.

MM--main mode. Mode that is slower than aggressive mode but more secure and more flexible than aggressive mode because it can offer an IKE peer more security proposals. The default action for IKE authentication (rsa-sig, rsa-encr, or preshared) is to initiate main mode.

NHRP --Next Hop Resolution Protocol. Routers, access servers, and hosts can use NHRP to discover the addresses of other routers and hosts connected to an NBMA network.

The Cisco implementation of NHRP supports the IETF draft version 11 of NBMA Next Hop Resolution Protocol (NHRP).

The Cisco implementation of NHRP supports IP Version 4, Internet Packet Exchange (IPX) network layers, and, at the link layer, ATM, FastEthernet, SMDS, and multipoint tunnel networks. Although NHRP is available on FastEthernet, NHRP need not be implemented over FastEthernet media because FastEthernet is capable of broadcasting. FastEthernet support is unnecessary (and not provided) for IPX.

PFS--perfect forward secrecy. A cryptographic characteristic associated with a derived shared secret value. With PFS, if one key is compromised, previous and subsequent keys are not compromised, because subsequent keys are not derived from previous keys.

SA--security association. Describes how two or more entities will utilize security services to communicate securely. For example, an IPsec SA defines the encryption algorithm (if used), the authentication algorithm, and the shared session key to be used during the IPsec connection.

Both IPsec and IKE require and use SAs to identify the parameters of their connections. IKE can negotiate and establish its own SA. The IPsec SA is established either by IKE or by manual user configuration.

transform--The list of operations done on a data flow to provide data authentication, data confidentiality, and data compression. For example, one transform is the ESP protocol with the HMAC-MD5 authentication algorithm; another transform is the AH protocol with the 56-bit DES encryption algorithm and the ESP protocol with the HMAC-SHA authentication algorithm.

VPN--Virtual Private Network. A framework that consists of multiple peers transmitting private data securely to one another over an otherwise public infrastructure. In this framework, inbound and outbound network traffic is protected using protocols that tunnel and encrypt all data. This framework permits networks to extend beyond their local topology, while remote users are provided with the appearance and functionality of a direct network connection.

NHRP MIB

The Cisco NHRP MIB feature introduces support for the NHRP MIB, which helps to manage and monitor the Next Hop Resolution Protocol (NHRP) via Simple Network Management Protocol (SNMP). Statistics can be collected and monitored via standards-based SNMP techniques (get operations) to query objects defined in the NHRP MIB. The NHRP MIB is VPN Routing and Forwarding (VRF) aware and supports VRF-aware queries.

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 Table at the end of this document.

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.

Prerequisites for NHRP MIB

- You should be familiar with configuring SNMP.

Restrictions for NHRP MIB

- Cisco does not support all the MIB variables defined in RFC 2677, Definitions of Managed Objects for the NBMA Next Hop Resolution Protocol (NHRP). For a list of variables supported and other caveats of this feature, see the Agent Capabilities file. Cisco does not support the set operations defined in RFC 2677.

Information About NHRP MIB

CISCO-NHRP-MIB

CISCO-NHRP-MIB provides NHRP MIB information on managed objects relating to clients only, servers only, and clients and servers.

The NHRP MIB module contains ten tables of objects as follows:

- NHRP Cache Table
- NHRP Purge Request Table
- NHRP Client Table
- NHRP Client Registration Table
- NHRP Client NHS Table
- NHRP Client Statistics Table
- NHRP Server Table
- NHRP Server Cache Table
- NHRP Server NHC Table
- NHRP Server Statistics Table

The Cisco implementation supports all of the tables except the NHRP Purge Request Table.

RFC-2677

RFC-2677 - Definitions of Managed Objects for the NBMA Next Hop Resolution Protocol (NHRP), describes managed objects that can be used to remotely monitor NHRP using SNMP and provide management information on the performance of NHRP.

How to Use NHRP MIB

No special configuration is needed to implement the NHRP MIB feature. The SNMP framework can be used to manage NHRP MIB. See the section “Configuration Examples for NHRP MIB” for an example of how to manage a VRF-aware NHRP MIB.

This section contains the following task:

Verifying NHRP MIB Status

Use this task to verify the NHRP MIB status.

SUMMARY STEPS

1. **enable**
2. **show snmp mib nhrp status**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables EXEC mode. • Enter your password if prompted.
Step 2	**show snmp mib nhrp status** **Example:** `Router# show snmp mib nhrp status`	Displays the status of the NHRP MIB.

Configuration Examples for NHRP MIB

Example Verifying NHRP MIB Status

The following output is from the show snmp mib nhrp status command:

```
Router# show snmp mib nhrp status
NHRP-SNMP Agent Feature: Enabled
NHRP-SNMP Tree State: Good
ListEnqueue Count = 0 Node Malloc Counts = 1
Spoke_103#
```

The “Enabled” status of “NHRP-SNMP Agent Feature:” indicates that the NHRP MIB is enabled. If the NHRP MIB was disabled, it would display “Disabled.” “ListEnqueue Count” and “Node Malloc Counts” counts are internal counts. “ListEnqueue Count” indicates how many nodes are queued for freeing. “Node Malloc Counts” displays how many nodes are allocated.

Example VRF-Aware NHRP MIB Configuration

The following is an example of how to configure a VRF table with the name Vrf1, for monitoring by SNMP:

```
ip vrf Vrf1
 rd 198102
! Name of the SNMP VPN context
 context Vrf1-context
```

```
!
crypto isakmp policy 1
 authentication pre-share
crypto isakmp key cisco47 address 0.0.0.0
!
crypto ipsec transform-set trans2 esp-des esp-md5-hmac
!
crypto ipsec profile vpnprof
 set transform-set trans2
!
interface Tunnel0
 bandwidth 1000
! DMVPN tunnel for Vrf1 VPN
 ip vrf forwarding Vrf1
 ip address 10.0.0.1 255.255.255.0
 ip mtu 1400
 ip nhrp authentication sample
 ip nhrp map multicast dynamic
 ip nhrp network-id 99
 ip nhrp holdtime 300
 no ip split-horizon eigrp 1
 ip tcp adjust-mss 1360
 delay 1000
 tunnel source Ethernet0
 tunnel mode gre multipoint
 tunnel key 100000
 tunnel protection ipsec profile vpnprof
!
interface Ethernet0
 ip address 172.17.0.1 255.255.255.0
!
interface Ethernet1
 ip address 192.168.0.1 255.255.255.0
!
router eigrp 1
address-family ipv4 vrf Vrf1
 network 10.0.0.0 0.0.0.255
 network 192.168.0.0 0.0.0.255
 no auto-summary
 autonomous-system 1
exit-address-family
!
! V2C Community ABC for VRF Vrf1
snmp-server group abc v2c context V3red_context read view_V3
snmp-server view view_V3 iso included
snmp-server community abc RO
snmp-server community public RO
snmp-server context Vrf1_context
!
!
snmp mib community-map abc context Vrf1-context
Spoke Configuration for DMVPN Example
crypto isakmp policy 1
 authentication pre-share
crypto isakmp key cisco47 address 0.0.0.0
!
crypto ipsec transform-set trans2 esp-des esp-md5-hmac
!
crypto ipsec profile vpnprof
 set transform-set trans2
!
interface Tunnel0
 bandwidth 1000
 ip address 10.0.0.2 255.255.255.0
 ip mtu 1400
 ip nhrp authentication sample
 ip nhrp map 10.0.0.1 172.17.0.1
 ip nhrp map multicast 172.17.0.1
 ip nhrp network-id 99
 ip nhrp holdtime 300
 ip nhrp nhs 10.0.0.1
 ip tcp adjust-mss 1360
 delay 1000
```

```
 tunnel source Ethernet0
 tunnel mode gre multipoint
 tunnel key 100000
 tunnel protection ipsec profile vpnprof
!
interface Ethernet0
 ip address dhcp hostname Spoke1
!
interface Ethernet1
 ip address 192.168.1.1 255.255.255.0
!
router eigrp 1
 network 10.0.0.0 0.0.0.255
 network 192.168.1.0 0.0.0.255
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Description of SNMP, SNMP MIBs, and how to configure SNMP on Cisco devices	"Configuring SNMP Support" chapter in the *Cisco IOS Network Management Configuration Guide*
Security commands	*Cisco IOS Security Command Reference*

Standards

Standard	Title
None	--

MIBs

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

RFCs

RFC	Title
RFC 2677	Definitions of Managed Objects for the NBMA Next Hop Resolution Protocol (NHRP)

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 NHRP MIB

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 2 Feature Information for NHRP MIB

Feature Name	Releases	Feature Information
NHRP MIB for DMVPN Networks	Cisco IOS XE Release 2.5	The Cisco NHRP MIB feature introduces support for the NHRP MIB, which helps to manage and monitor Next Hop Resolution Protocol (NHRP) via Simple Network Management Protocol (SNMP). Statistics can be collected and monitored via standards-based SNMP techniques (get operations) to query objects defined in the NHRP MIB. The following commands were introduced or modified: debug snmp mib nhrp**, show snmp mib nhrp status**.

trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1005R)

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

DMVPN Dynamic Tunnels Between Spokes Behind a NAT Device

The DMVPN Dynamic Tunnels Between Spokes Behind a NAT Device feature allows Next Hop Resolution Protocol (NHRP) spoke-to-spoke tunnels to be built in Dynamic Multipoint Virtual Private Networks (DMVPNs), even if one or more spokes is behind a Network Address Translation (NAT) device.

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 Table at the end of this document.

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.

Restrictions for DMVPN Dynamic Tunnels Between Spokes Behind a NAT Device

In order for two spokes to build tunnels between them, they need to know the post-NAT address of the other spoke.

Consider the following restrictions when using spoke-to-spoke tunneling in NAT environments:

- **Multiple NAT translations** --A packet can go across multiple NAT devices in a nonbroadcast multiaccess (NBMA) DMVPN cloud and make several (unimportant) translations before it reaches its destination. The last translation is the important translation because it is used to create the NAT translation for all devices that reach a spoke through the last NAT device.
- **Hub or spoke can be reached through pre-NAT addresses** --It is possible for two or more spokes to be behind the same NAT device, which can be reached through a pre-NAT IP address. Only the post-

NAT IP address is relied on even if it means that a tunnel may take a less desirable path. If both spokes use NAT through the same device, then a packet may not travel inside-out or outside-in as expected by the NAT device and translations may not occur correctly.

- **Interoperability between NAT and non-NAT capable devices** --In networks that are deployed with DMVPN, it is important that a device with NHRP NAT functionality operate together with non-NAT supported devices. A capability bit in the NHRP packet header indicates to any receiver whether a sending device understands a NAT extension.
- **Same NAT translation** --A spoke's post-NAT IP address must be the same when the spoke is communicating with its hubs and when it is communicating with other spokes. For example, a spoke must have the same post-NAT IP address no matter where it is sending tunnel packets within the DMVPN network.
- If one spoke is behind one NAT device and another different spoke is behind another NAT device, and Peer Address Translation (PAT) is the type of NAT used on both NAT devices, then a session initiated between the two spokes cannot be established.

One example of a PAT configuration on a NAT interface is:

```
ip nat inside source list nat_acl interface FastEthernet0/1 overload
```

Information About DMVPN Dynamic Tunnels Between Spokes Behind a NAT Device

The following sections describe how the DMVPN Dynamic Tunnels Between Spokes Behind a NAT Device feature allows spoke-to-spoke tunnels to be built even if one or both spoke devices are behind a NAT device:

- DMVPN Spoke-to-Spoke Tunneling Limited to Spokes Not Behind a NAT Device, page 56
- NHRP Spoke-to-Spoke Tunnel with a NAT Device, page 58

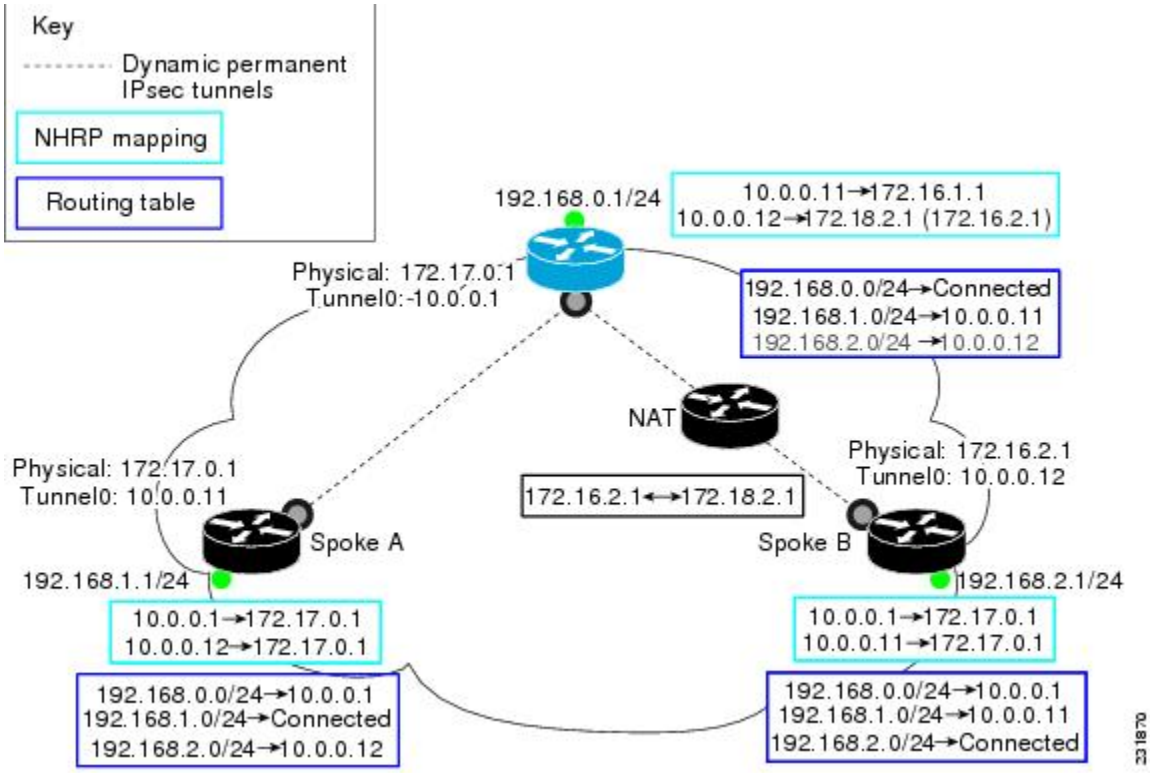
DMVPN Spoke-to-Spoke Tunneling Limited to Spokes Not Behind a NAT Device

NAT allows a single device, such as a router, to act as agent between the Internet (or "public network") and a local (or "private") network, and is often used because of the scarcity of available IP addresses. A single unique IP address is required to represent an entire group of devices to anything outside the NAT device. NAT is also deployed for security and administration purposes.

In DMVPN networks, spoke-to-spoke tunneling is limited to spokes that are not behind the NAT device. If one or both spokes are behind a NAT device, a spoke-to-spoke tunnel cannot be built to or from the NAT device because it is possible for the spoke-to-spoke tunnel traffic to fail or be lost "black-holed" for an extended period.

The figure below and the following sections describe how DMVPN works when spoke-to-spoke tunneling is limited to spokes that are not behind a NAT device.

Figure 4 Implementation of DMVPN Spoke-to-Spoke Tunneling Limited to Spokes Not Behind a NAT Device



- NHRP Registration, page 57
- NHRP Resolution, page 58

NHRP Registration

When an NHRP registration is received, the hub checks the source IP address on the encapsulating GRE/IP header of the NHRP packet with the source NBMA IP address, which is contained in the NHRP registration packet. If these IP addresses are different, then NHRP knows that NAT is changing the outer IP header source address. The hub preserves both the pre- and post-NAT address of the registered spoke.

Note If encryption is used, then IPsec transport mode must be used to enable NHRP.

The following **show ip nhrp** command output example shows the source IP address of the NHRP packet and tunnel information for Spoke B in the figure above:

Note The NBMA (post-NAT) address for Spoke B is 172.18.2.1 (the claimed NBMA (pre-NAT) source address is 172.16.2.1).

```
Router# show ip nhrp
10.0.0.11/32 via 10.0.0.11, Tunnel0 created 00:00:21, expire 00:05:38
  Type: dynamic, Flags: authoritative unique registered used
NBMA address: 172.18.2.1
      (Claimed NBMA address: 172.16.2.1)
```

NHRP Resolution

The following describes the NHRP resolution process between Spoke A and Spoke B shown in the figure above, where Spoke B is behind a NAT device with pre-NAT address of 172.16.2.1 and a post-NAT address of 172.18.2.1:

- The NHRP table entry for Spoke B on the hub contains both the post-NAT and pre-NAT addresses. When the hub receives an NHRP resolution request for the VPN address (tunnel address) of Spoke B, it answers with its own NBMA address instead of Spoke B's NBMA address.
- When the hub receives an NHRP resolution request sourced from Spoke B for any other spoke, the hub also answers with its own NBMA address. This ensures that any attempt to build a spoke-to-spoke tunnel with Spoke B results in the data packets being sent through the hub rather than through a spoke-to-spoke tunnel.

For example:

- ◦ Data traffic from source IP address 192.168.1.1 (behind Spoke A) to destination IP address 192.168.2.1 (behind Spoke B) triggers Spoke A to send a resolution request for Spoke B (10.0.0.12) to the next hop router (hub).
  - ◦ The hub receives the resolution request and finds a mapping entry for Spoke B (10.0.0.12). Because Spoke B is behind a NAT device, it acts as a proxy and replies with its own NBMA address (172.17.0.1).
  - ◦ The hub also receives a resolution request from Spoke B for Spoke A (10.0.0.11). Because Spoke B is behind a NAT device, it acts as a proxy and replies with its own NBMA address (172.17.0.1). This restricts any spoke-to-spoke traffic to or from Spoke B to travel through the hub router, which is done rather than having a tunnel between the spokes.

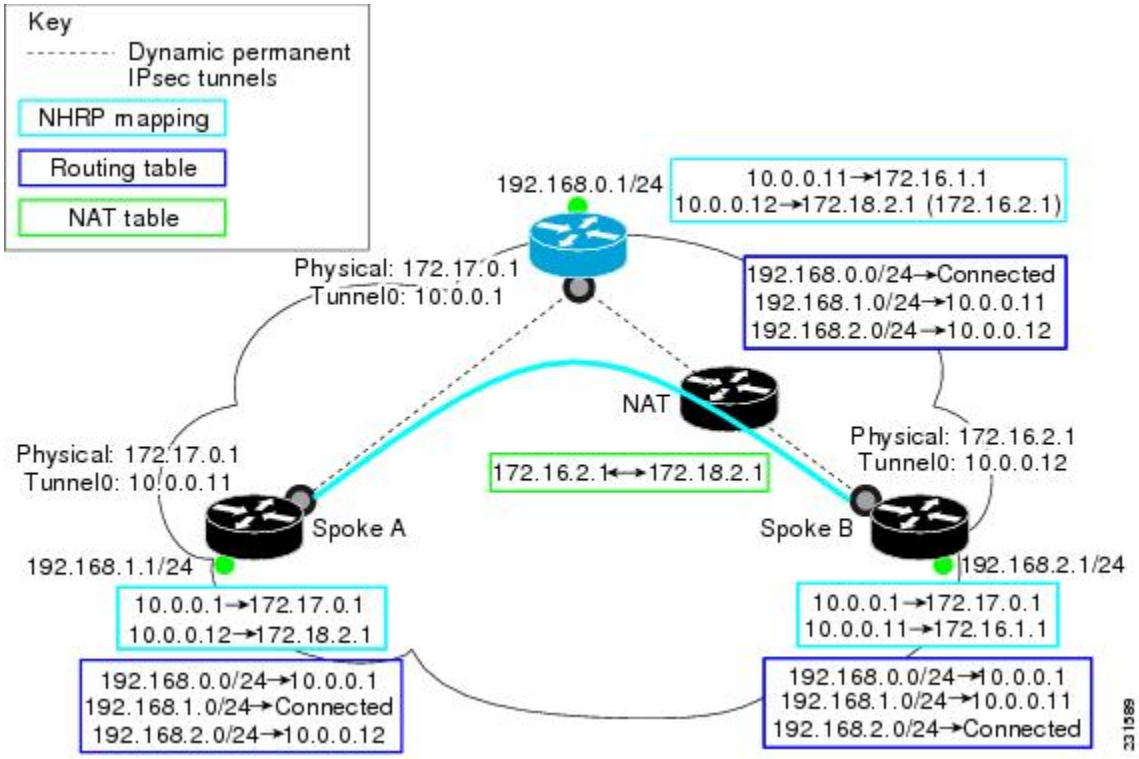
NHRP Spoke-to-Spoke Tunnel with a NAT Device

The NHRP Spoke-to-Spoke Tunnel with NAT feature introduces NAT extension in the NHRP protocol and is enabled automatically. The NHRP NAT extension is a Client Information Entry (CIE) entry with information about the protocol and post-NAT NBMA address. This additional information allows the support of spoke-to-spoke tunnels between spokes where one or both are behind a NAT device without the problem of losing (black-holing) traffic for an extended period.

Note The spoke-to-spoke tunnel may fail to come up, but it is detected and the data traffic flows through the hub, rather than being lost (black-holed).

The figure below shows how the NHRP spoke-to-spoke tunnel works with NAT.

Figure 5 NHRP Between Spoke-to-Spoke Tunnels



- NHRP Registration Process, page 59
- NHRP Resolution and Purge Process, page 59

NHRP Registration Process

The following steps describe the NHRP registration process:

1. A spoke sends a registration request with the NAT-Capability=1 parameter and a NAT NHRP extension of the NBMA address of the hub as configured on the spoke.
2. The hub compares the NHRP (NAT) extension with its configured NBMA address and determines whether it is or is not behind a NAT device. The hub also makes a note of whether the spoke is behind a NAT device by comparing the incoming GRE/IP source address with the spoke's NBMA address in the NHRP packet.
3. The registration reply from the hub to the spoke includes a NAT NHRP extension with the post-NAT address of the spoke, if the hub detects if it is behind a NAT device.
4. If the spokes get a NAT NHRP extension in the NHRP registration reply, it then records its post-NAT IP address for possible use later.

NHRP Resolution and Purge Process

The following steps describe the NHRP resolution and purge process:

1. When a spoke is behind a NAT device, it includes a NAT NHRP extension when it sends NHRP resolution requests.
2. The hub receives the resolution request. If the spoke is behind a NAT device and there is no NAT extension, then the hub adds a NAT extension before forwarding this extension to the next node (spoke or next hop server) along the path. However, if the hub is forwarding the request to a non-NAT extension capable node, it rewrites the source-NBMA inside the packet to be the post-NAT IP address for the requesting spoke rather than its pre-NAT IP address.
3. The receiver (spoke) uses a NAT NHRP extension record (NAT capable) or the source NBMA address (non-NAT capable information) to build the tunnel. This spoke's reply includes its own NAT extension if it is behind a NAT device.

Note Hubs do not answer NHRP resolution requests on behalf of spokes. Hubs always forward NHRP resolution requests to the end spoke that has the requested tunnel IP address or services the requested data from the host IP address.

The following describes the NHRP resolution process between Spoke A and Spoke B shown in the figure above, where Spoke B is behind a NAT device with pre-NAT address 172.16.2.1 and post-NAT address of 172.18.2.1:

- Data traffic to the 192.168.2.0/24 network from hosts behind Spoke A triggers an NHRP resolution request for Spoke B's tunnel IP address (10.0.0.12) to be sent through the hub. The hub receives a resolution request and forwards it to Spoke B. Spoke B creates a dynamic spoke-to-spoke tunnel using the source NBMA IP address for Spoke A from the NHRP resolution request and sends an NHRP resolution reply directly to Spoke A. It includes its post-NAT address in the NAT NHRP-extension header.
- Alternatively, traffic to the192.168.1.0/24 network from hosts behind the NAT device on Spoke B triggers an NHRP resolution request for Spoke A's tunnel IP address (10.0.0.11). Spoke B adds its own post-NAT IP address in the NHRP NAT-extension in the resolution request. The hub receives a resolution request and forwards it to Spoke A. Spoke A parses the NHRP NAT-extension and builds a tunnel using Spoke B's post-NAT address and replies directly to Spoke B.

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
NHRP commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS IP Addressing Services Command Reference*
Dynamic Multipoint VPN	"Dynamic Multipoint VPN (DMVPN)" module in the *Cisco IOS XE Security Configuration Guide: Secure Connectivity*

Standards

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

MIBs

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

RFCs

RFC	Title
No new or modified RFCs are supported by this release.	--

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 DMVPN Dynamic Tunnels Between Spokes Behind a NAT Device

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 3 Feature Information for DMVPN Dynamic Tunnels Between Spokes Behind a NAT Device

Feature Name	Releases	Feature Information
DMVPN Dynamic Tunnels Between Spokes Behind a NAT Device	Cisco IOS XE Release 2.5	The DMVPN Dynamic Tunnels Between Spokes Behind a NAT Device feature allows NHRP spoke-to-spoke tunnels to be built in DMVPN networks, even if one or more spokes is behind a Network Address Translation (NAT) device. In Cisco IOS XE Release 2.5, this feature was introduced on the Cisco ASR 1000 Series Aggregation Routers.

Sharing IPsec with Tunnel Protection

The Sharing IPsec with Tunnel Protection feature allows an IP Security (IPsec) Security Association Database (SADB) to be shared between two or more generic routing encapsulation (GRE) tunnel interfaces, when tunnel protection is used. When these tunnel interfaces are shared, they have a single underlying cryptographic SADB, cryptographic map, and IPsec profile in the Dynamic Multipoint Virtual Private Network (DMVPN) configuration.

The Sharing IPsec with Tunnel Protection feature is required by some DMVPN configurations. If IPsec security association (SA) sessions are not shared within the same IPsec SADB, then an IPsec SA may get associated with the wrong IPsec SADB and therefore the wrong tunnel interface, causing duplication of IPsec SAs and tunnel interfaces to flap. If the tunnel interfaces flap (change rapidly and repeatedly between online and offline states), then network connectivity problems occur.

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 Table at the end of this document.

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.

Restrictions for Sharing IPsec with Tunnel Protection

Consider the following restrictions when sharing IPsec with tunnel protection:

- The **tunnel source** command on all tunnel interfaces using the same tunnel source *must* be configured using the interface type and number, not its IP address.
- All tunnels with the same **tunnel source** interface must use the same IPsec profile and the **shared** keyword with the **tunnel protection** command on all such tunnels. The only exception is a scenario

when there are only peer-to-peer (P2P) GRE tunnel interfaces configured with the same tunnel source in the system, all with unique tunnel destination IP addresses.

- Different IPsec profile names must be used for shared and unshared tunnels.

For example, if “tunnel 1” is configured with the **tunnel source loopback0** command, and “tunnel 2” and “tunnel 3” are shared using the **tunnel source loopback1** command, then use **ipsec_profile_1** for tunnel 1 and **ipsec_profile_2** for tunnels 2 and 3.

- A different IPsec profile must be used for each set of shared tunnels.

For example, if tunnels 1 through 5 use **loopback0** as their tunnel source and tunnels 6 through 10 use loopback1, then define **ipsec_profile_1** for tunnels 1 through 5 and **ipsec_profile_2** for tunnels 6 to 10.

- Sometimes it may be desirable to not share an IPsec session between two or more tunnel interfaces using the same tunnel source.

For example, in a service provider environment, each DMVPN cloud can represent a different customer. It is desirable to lock the connections from a customer to a tunnel interface and not share or allow IPsec sessions from other customers. For such scenarios, Internet Security Association and Key Management Protocol (ISAKMP) profiles can be used to identify and bind customer connections to an ISAKMP profile and through that to an IPsec profile. This ISAKMP profile limits the IPsec profile to accept only those connections that matched the corresponding ISAKMP profile. Separate ISAKMP and IPsec profiles can be obtained for each DMVPN cloud (tunnel interface) without sharing the same IPsec SADB.

- Sharing IPsec is not desired and not supported for a virtual tunnel interface (VTI). A VTI provides a routable interface type for terminating IPsec tunnels and a way to define protection between sites to form an overlay network.

Information About Sharing IPsec with Tunnel Protection

The following section describes how the Sharing IPsec with Tunnel Protection feature allows an IPsec SADB to be shared between two or more GRE tunnel interfaces:

- Single IPsec SAs and GRE Tunnel Sessions, page 64

Single IPsec SAs and GRE Tunnel Sessions

In a dual-hub dual-DMVPN topology, it is possible to have two or more GRE tunnel sessions (same tunnel source and destination, but different tunnel keys) between the same two endpoints. In this case, it is desirable to use a single IPsec SA to secure both GRE tunnel sessions. Also, it is not possible to decide under which tunnel interface an IPsec Quick Mode (QM) request must be processed and bound when two tunnel interfaces use the same tunnel source.

The tunnel protection ipsec profile shared command is used to create a single IPsec SADB for all the tunnel interfaces that use the same profile and tunnel source interface. This allows a single IPsec SA to be used for all GRE tunnels (same tunnel source and destination, but different tunnel keys) between the same two endpoints. It also makes IPsec QM processing unambiguous because there is one SADB under which to process the incoming IPsec QM request for all shared tunnel interfaces as opposed to multiple SADBs, one for each tunnel interface when not shared.

The SA of a QM proposal to a tunnel interface is processed by using the shared SADB and crypto map parameters. On the cryptodata plane, the decrypted and GRE decapsulated packets are demultiplexed to the appropriate tunnel interface by the GRE module using a local address, remote address, and optional tunnel key information.

Note The tunnel source, tunnel destination, and tunnel key (triplet) must be unique for all tunnel interfaces on a router. For a multipoint GRE interface where the tunnel destination is not configured, the pair (tunnel source and tunnel key) must be unique. Incoming GRE packets are also matched to P2P GRE tunnels first; if there is not a match, then they are matched to mGRE tunnels.

How to Share an IPsec Session Between Multiple Tunnels

Sharing an IPsec SADB Between Multiple Tunnel Interfaces in a DMVPN

Use the following commands to configure a Cisco IOS router to share an IPsec SADB between multiple tunnel interfaces in a DMVPN.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *number*
4. *tunnel source {ip-address | interface-type interface-number}*
5. **tunnel protection ipsec profile** *name* [**shared]**
6. **exit**
7. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface tunnel** *number* **Example:** `Router(config)# interface tunnel 5`	Configures a tunnel interface and enters interface configuration mode. • The *number* argument specifies the number of the tunnel interface that you want to create or configure. There is no limit on the number of tunnel interfaces you can create.
Step 4	*tunnel source {ip-address | interface-type interface-number}* **Example:** `Router(config-if)# tunnel source GigabitEthernet 0`	Sets the source IP address or source interface type number for a tunnel interface. • When you are using the **tunnel protection ipsec profile**command, you must specify an interface, not an IP address for the tunnel source.
Step 5	**tunnel protection ipsec profile** *name* [**shared]** **Example:** `Router(config-if)# tunnel protection ipsec profile vpnprof shared`	Associates a tunnel interface with an IPsec profile. • The *name* argument specifies the name of the IPsec profile; this value must match the *name* specified in the **crypto ipsec profile***name* command. • The **shared** keyword allows IPsec sessions to be shared between multiple tunnel interfaces configured with the same tunnel source IP.
Step 6	**exit** **Example:** `Router(config-if)# exit`	Exits the tunnel interface.
Step 7	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.

What to Do Next

If your configuration requires more spoke routers in a dual-hub, dual DMVPN topology, repeat the steps in GUID-13F2F7BA-6999-4B7D-90A2-285ABF75EFE3 to configure additional spokes.

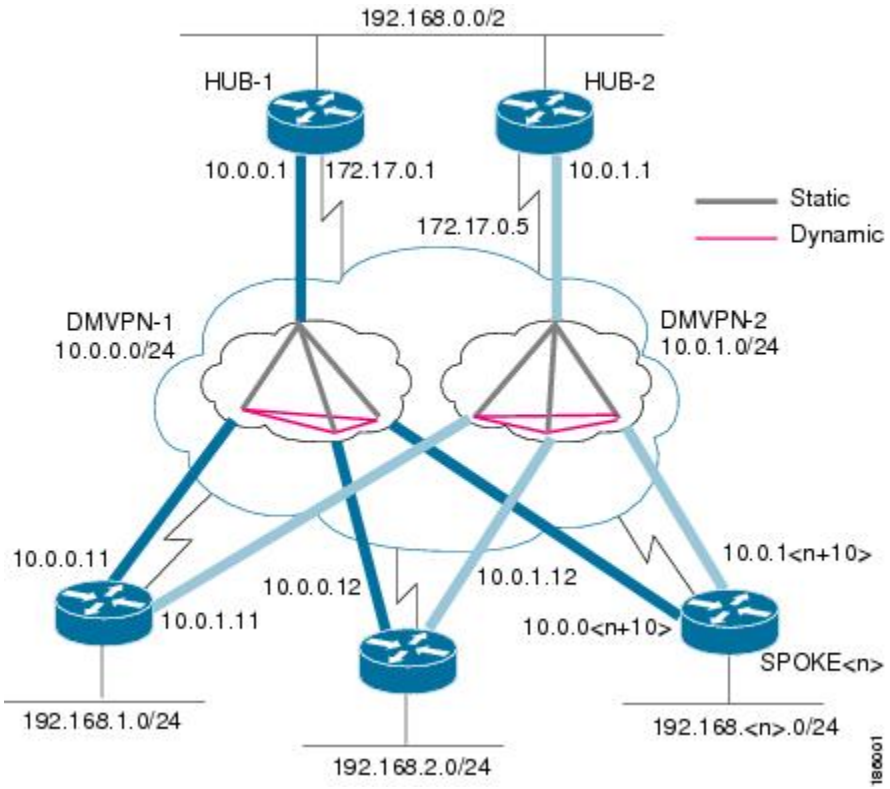
Configuration Examples for Sharing IPsec with Tunnel Protection

Dual-Hub Router Dual-DMVPN Topology

The dual-hub router, dual-DMVPN topology, shown in the figure below, has the following attributes:

- Each hub router is configured with a single mGRE tunnel interface.
- Each hub router is connected to one DMVPN subnet (blue cloud), and the spokes are connected to both DMVPN 1 and DMVPN 2.
- Each spoke router is configured with two mGRE tunnel interfaces.
- One mGRE tunnel interface belongs to DMVPN 1 and the other mGRE tunnel interface belongs to DMVPN 2.
- Each mGRE tunnel interface is configured with a same tunnel source IP address and uses shared tunnel protection between them.

Figure 6 Dual-Hub Router, Dual-DMVPN Topology.



Configuring an IPsec SADB Between Multiple Tunnel Interfaces in a DMVPN Example

The following configuration examples are given when configuring an IPsec SADB between multiple tunnel interfaces in a DMVPN:

- Hub 1 Configuration Example, page 68
- Hub 2 Configuration Example, page 68
- Spoke 1 Configuration Example, page 69

- Spoke 2 Configuration Example, page 70
- Results on Spoke 1 Example, page 71

Hub 1 Configuration Example

Hub 1 and Hub 2 configurations are similar, except that each hub belongs to a different DMVPN.

Hub 1 has the following DMVPN configuration:

- IP subnet: 10.0.0.0/24
- Next Hop Address Resolution Protocol (NHRP) network ID: 100000
- Tunnel key: 100000
- Dynamic routing protocol: Enhanced Interior Gateway Routing Protocol (EIGRP)

```
!
hostname Hub1
 !
 crypto isakmp policy 1
  authentication pre-share
 crypto isakmp key cisco47 address 0.0.0.0 0.0.0.0
 !
 crypto IPsec transform-set trans2 esp-des esp-md5-hmac
  mode transport
 !
 crypto IPsec profile vpnprof
  set transform-set trans2
 !
 interface Tunnel 5
  bandwidth 1000
  ip address 10.0.0.1 255.255.255.0
  ip mtu 1400
  no ip next-hop-self eigrp 1
  ip nhrp authentication test
  ip nhrp map multicast dynamic
  ip nhrp network-id 100000
  ip nhrp holdtime 600
no ip split-horizon eigrp 1
ip tcp adjust-mss 1360
  delay 1000
  tunnel source GigabitEthernet 0/0/0
  tunnel mode gre multipoint
  tunnel key 100000
  tunnel protection IPsec profile vpnprof
 !
 interface GigabitEthernet 0/0/0
  ip address 172.17.0.1 255.255.255.252
 !
 interface GigabitEthernet 0/0/1
  ip address 192.168.0.1 255.255.255.0
 !
 router eigrp 1
  network 10.0.0.0 0.0.0.255
  network 192.168.0.0 0.0.0.255
  no auto-summary
 !
```

Hub 2 Configuration Example

Hub 2 has the following DMVPN configuration:

- IP subnet: 10.0.1.0/24
- NHRP network ID: 100001
- Tunnel key: 100001

- Dynamic routing protocol: EIGRP

```
!
hostname Hub2
 !
 crypto isakmp policy 1
  authentication pre-share
 crypto isakmp key cisco47 address 0.0.0.0 0.0.0.0
 !
 crypto ipsec transform-set trans2 esp-des esp-md5-hmac
  mode transport
 !
 crypto ipsec profile vpnprof
  set transform-set trans2
 !
 interface Tunnel 5
  bandwidth 1000
  ip address 10.0.1.1 255.255.255.0
  ip mtu 1400
  no ip next-hop-self eigrp 1
  ip nhrp authentication test
  ip nhrp map multicast dynamic
  ip nhrp network-id 100001
  ip nhrp holdtime 600
no ip split-horizon eigrp 1
ip tcp adjust-mss 1360
  delay 1000
  tunnel source GigabitEthernet 0/0/0
  tunnel mode gre multipoint
  tunnel key 100001
  tunnel protection ipsec profile vpnprof
 !
 interface GigabitEthernet 0/0/0
  ip address 172.17.0.5 255.255.255.252
 !
 interface GigabitEthernet 0/0/1
  ip address 192.168.0.2 255.255.255.0
 !
 router eigrp 1
  network 10.0.1.0 0.0.0.255
  network 192.168.0.0 0.0.0.255
  no auto-summary
 !
```

Spoke 1 Configuration Example

Spoke 1 has the following DMVPN configuration:

```
!
 hostname Spoke1
 !
 crypto isakmp policy 1
  authentication pre-share
 crypto isakmp key cisco47 address 0.0.0.0 0.0.0.0
 !
 crypto ipsec transform-set trans2 esp-des esp-md5-hmac
  mode transport
 !
 crypto ipsec profile vpnprof
  set transform-set trans2
 !
interface Tunnel 5
 bandwidth 1000
.
.
.
 ip nhrp authentication test
 ip nhrp map 10.0.0.1 172.17.0.1
 ip nhrp map multicast 172.17.0.1
 ip nhrp network-id 100000
```

```
 ip nhrp holdtime 300|
 ip nhrp nhs 10.0.0.1
 ip tcp adjust-mss 1360
 delay 1000
.
.
.
 tunnel protection ipsec profile vpnprof shared
!
interface Tunnel 5
 bandwidth 1000
.
.
.
 ip nhrp authentication test
 ip nhrp map 10.0.1.1 172.17.0.5
 ip nhrp map multicast 172.17.0.5
 ip nhrp network-id 100001
 ip nhrp holdtime 300
 ip nhrp nhs 10.0.1.1
 ip tcp adjust-mss 1360
 delay 1000
.
.
.
 tunnel protection ipsec profile vpnprof shared
!
 interface GigabitEthernet 0/0/0
  ip address dhcp hostname Spoke1
 !
 interface GigabitEthernet 0/0/1
  ip address 192.168.1.1 255.255.255.0
 !
 router eigrp 1
  network 10.0.0.0 0.0.0.255
  network 10.0.1.0 0.0.0.255
  network 192.168.1.0 0.0.0.255
  no auto-summary
 !
```

Spoke 2 Configuration Example

Spoke 2 has the following DMVPN configuration:

```
!
 hostname Spoke2
 !
 crypto isakmp policy 1
  authentication pre-share
 crypto isakmp key cisco47 address 0.0.0.0 0.0.0.0
 !
 crypto ipsec transform-set trans2 esp-des esp-md5-hmac
  mode transport
 !
 crypto ipsec profile vpnprof
  set transform-set trans2
 !
interface Tunnel 5
 bandwidth 1000
.
.
.
 ip nhrp authentication test
 ip nhrp map 10.0.0.1 172.17.0.1
 ip nhrp map multicast 172.17.0.1
 ip nhrp network-id 100000
 ip nhrp holdtime 300|
 ip nhrp nhs 10.0.0.1
 ip tcp adjust-mss 1360
 delay 1000
.
```

```
.
.
 tunnel protection ipsec profile vpnprof shared
!
interface Tunnel 5
 bandwidth 1000
.
.
.
 ip nhrp authentication test
 ip nhrp map 10.0.1.1 172.17.0.5
 ip nhrp map multicast 172.17.0.5
 ip nhrp network-id 100001
 ip nhrp holdtime 300
 ip nhrp nhs 10.0.1.1
 ip tcp adjust-mss 1360
 delay 1000
.
.
.
 tunnel protection ipsec profile vpnprof shared
 !
 interface GigabitEthernet 0/0/0
  ip address dhcp hostname Spoke2
 !
 interface GigabitEthernet 0/0/1
  ip address 192.168.2.1 255.255.255.0
 !
 router eigrp 1
  network 10.0.0.0 0.0.0.255
  network 10.0.1.0 0.0.0.255
  network 192.168.2.0 0.0.0.255
  no auto-summary
 !
```

Results on Spoke 1 Example

Spoke 1 has the following results for its DMVPN configuration:

```
Spoke1# show ip nhrp
10.0.0.1/32 via 10.0.0.1, Tunnel 0 created 00:06:52, never expire
  Type: static, Flags: used
  NBMA address: 172.17.0.1
10.0.0.12/32 via 10.0.0.12, Tunnel 0 created 00:03:17, expire 00:01:52
  Type: dynamic, Flags: router
  NBMA address: 172.17.0.12
10.0.1.1/32 via 10.0.1.1, Tunnel 1 created 00:13:45, never expire
  Type: static, Flags: used
  NBMA address: 172.17.0.5
10.0.1.12/32 via 10.0.1.12, Tunnel 1 created 00:00:02, expire 00:04:57
  Type: dynamic, Flags: router
  NBMA address: 172.17.0.12
Spoke1# show crypto socket
```

Note There are only three crypto connections because the two NHRP sessions (10.0.0.12, Tunnel0) and (10.0.1.12, Tunnel1) are only one IPsec session, because they both have the same nonbroadcast multiaccess (NBMA) IPsec peer address.

```
Number of Crypto Socket connections 3
   Shd Peers (local/remote): 172.17.0.11
/172.17.0.12
       Local Ident  (addr/mask/port/prot): (172.17.0.11/255.255.255.255/0/47)
       Remote Ident (addr/mask/port/prot): (172.17.0.12/255.255.255.255/0/47)
       Flags: shared
       ipsec Profile: "vpnprof"
       Socket State: Open
```

```
       Client: "TUNNEL SEC" (Client State: Active)
   Shd Peers (local/remote): 172.17.0.11
/172.17.0.5
       Local Ident  (addr/mask/port/prot): (172.17.0.11/255.255.255.255/0/47)
       Remote Ident (addr/mask/port/prot): (172.17.0.5/255.255.255.255/0/47)
       Flags: shared
       ipsec Profile: "vpnprof"
       Socket State: Open
       Client: "TUNNEL SEC" (Client State: Active)
   Shd Peers (local/remote): 172.17.0.11
/172.17.0.1
       Local Ident  (addr/mask/port/prot): (172.17.0.11/255.255.255.255/0/47)
       Remote Ident (addr/mask/port/prot): (172.17.0.1/255.255.255.255/0/47)
       Flags: shared
       ipsec Profile: "vpnprof"
       Socket State: Open
       Client: "TUNNEL SEC" (Client State: Active)
Crypto Sockets in Listen state:
Client: "TUNNEL SEC" Profile: "vpnprof" Map-name: "vpnprof-head-1"
Spoke1# show crypto map
Crypto Map: "vpnprof-head-1" idb: FastEthernet0/0/0 local address: 172.17.0.11
Crypto Map "vpnprof-head-1" 65536 ipsec-isakmp
        Profile name: vpnprof
        Security association lifetime: 4608000 kilobytes/3600 seconds
        PFS (Y/N): N
        Transform sets={
                trans2,
        }
Crypto Map "vpnprof-head-1" 65537 ipsec-isakmp
        Map is a PROFILE INSTANCE.
        Peer = 172.17.0.5
        Extended IP access list
            access-list  permit gre host 172.17.0.11 host 172.17.0.5
        Current peer: 172.17.0.5
        Security association lifetime: 4608000 kilobytes/3600 seconds
        PFS (Y/N): N
        Transform sets={
                trans2,
        }
Crypto Map "vpnprof-head-1" 65538 ipsec-isakmp
        Map is a PROFILE INSTANCE.
        Peer = 172.17.0.1
        Extended IP access list
            access-list  permit gre host 172.17.0.11 host 172.17.0.1
        Current peer: 172.17.0.1
        Security association lifetime: 4608000 kilobytes/3600 seconds
        PFS (Y/N): N
        Transform sets={
                trans2,
        }
Crypto Map "vpnprof-head-1" 65539 ipsec-isakmp
        Map is a PROFILE INSTANCE.
        Peer = 172.17.0.12
        Extended IP access list
            access-list  permit gre host 172.17.0.11 host 172.17.0.12
        Current peer: 172.17.0.12
        Security association lifetime: 4608000 kilobytes/3600 seconds
        PFS (Y/N): N
        Transform sets={
                trans2,
        }
        Interfaces using crypto map vpnprof-head-1:
                Tunnel1
                Tunnel0
```

Note All three crypto sessions are shown under each tunnel interface (three entries, twice) in the **show crypto ipsec sa** output, because both interfaces are mapped to the same IPsec SADB, which has three entries. This duplication of output is expected in this case.

```
Spoke1# show crypto ipsec sa
interface: Tunnel 0
    Crypto map tag: vpnprof-head-1, local addr 172.17.0.11
   protected vrf: (none)
   local  ident (addr/mask/prot/port): (172.17.0.11/255.255.255.255/47/0)
   remote ident (addr/mask/prot/port): (172.17.0.1/255.255.255.255/47/0)
   current_peer 172.17.0.1 port 500
     PERMIT, flags={origin_is_acl,}
    #pkts encaps: 134, #pkts encrypt: 134, #pkts digest: 134
    #pkts decaps: 118, #pkts decrypt: 118, #pkts verify: 118
    #pkts compressed: 0, #pkts decompressed: 0
    #pkts not compressed: 0, #pkts compr. failed: 0
    #pkts not decompressed: 0, #pkts decompress failed: 0
    #send errors 22, #recv errors 0
     local crypto endpt.: 172.17.0.11, remote crypto endpt.: 172.17.0.1
     path mtu 1500, ip mtu 1500, ip mtu idb FastEthernet0/0/0
     current outbound spi: 0xA75421B1(2807308721)
     inbound esp sas:
      spi: 0x96185188(2518176136)
        transform: esp-des esp-md5-hmac ,
        in use settings ={Transport, }
        conn id: 3, flow_id: SW:3, crypto map: vpnprof-head-1
        sa timing: remaining key lifetime (k/sec): (4569747/3242)
        IV size: 8 bytes
        replay detection support: Y
        Status: ACTIVE
     inbound ah sas:
     inbound pcp sas:
     outbound esp sas:
      spi: 0xA75421B1(2807308721)
        transform: esp-des esp-md5-hmac ,
        in use settings ={Transport, }
        conn id: 4, flow_id: SW:4, crypto map: vpnprof-head-1
        sa timing: remaining key lifetime (k/sec): (4569745/3242)
        IV size: 8 bytes
        replay detection support: Y
        Status: ACTIVE
     outbound ah sas:
     outbound pcp sas:
   protected vrf: (none)
   local  ident (addr/mask/prot/port): (172.17.0.11/255.255.255.255/47/0)
   remote ident (addr/mask/prot/port): (172.17.0.5/255.255.255.255/47/0)
   current_peer 172.17.0.5 port 500
     PERMIT, flags={origin_is_acl,}
    #pkts encaps: 244, #pkts encrypt: 244, #pkts digest: 244
    #pkts decaps: 253, #pkts decrypt: 253, #pkts verify: 253
    #pkts compressed: 0, #pkts decompressed: 0
    #pkts not compressed: 0, #pkts compr. failed: 0
    #pkts not decompressed: 0, #pkts decompress failed: 0
    #send errors 1, #recv errors 0
     local crypto endpt.: 172.17.0.11, remote crypto endpt.: 172.17.0.5
     path mtu 1500, ip mtu 1500, ip mtu idb FastEthernet0/0/0
     current outbound spi: 0x3C50B3AB(1011921835)
     inbound esp sas:
      spi: 0x3EBE84EF(1052673263)
        transform: esp-des esp-md5-hmac ,
        in use settings ={Transport, }
        conn id: 1, flow_id: SW:1, crypto map: vpnprof-head-1
        sa timing: remaining key lifetime (k/sec): (4549326/2779)
        IV size: 8 bytes
        replay detection support: Y
        Status: ACTIVE
     inbound ah sas:
     inbound pcp sas:
```

```
      outbound esp sas:
       spi: 0x3C50B3AB(1011921835)
         transform: esp-des esp-md5-hmac ,
         in use settings ={Transport, }
         conn id: 2, flow_id: SW:2, crypto map: vpnprof-head-1
         sa timing: remaining key lifetime (k/sec): (4549327/2779)
         IV size: 8 bytes
         replay detection support: Y
         Status: ACTIVE
      outbound ah sas:
      outbound pcp sas:
    protected vrf: (none)
    local  ident (addr/mask/prot/port): (172.17.0.11/255.255.255.255/47/0)
    remote ident (addr/mask/prot/port): (172.17.0.12/255.255.255.255/47/0)
    current_peer 172.17.0.12 port 500
      PERMIT, flags={origin_is_acl,}
     #pkts encaps: 0, #pkts encrypt: 0, #pkts digest: 0
     #pkts decaps: 2, #pkts decrypt: 2, #pkts verify: 2
     #pkts compressed: 0, #pkts decompressed: 0
     #pkts not compressed: 0, #pkts compr. failed: 0
     #pkts not decompressed: 0, #pkts decompress failed: 0
     #send errors 0, #recv errors 0
      local crypto endpt.: 172.17.0.11, remote crypto endpt.: 172.17.0.12
      path mtu 1500, ip mtu 1500, ip mtu idb FastEthernet0/0/0
      current outbound spi: 0x38C04B36(952126262)
      inbound esp sas:
       spi: 0xA2EC557(170837335)
         transform: esp-des esp-md5-hmac ,
         in use settings ={Transport, }
         conn id: 5, flow_id: SW:5, crypto map: vpnprof-head-1
         sa timing: remaining key lifetime (k/sec): (4515510/3395)
         IV size: 8 bytes
         replay detection support: Y
         Status: ACTIVE
      inbound ah sas:
      inbound pcp sas:
      outbound esp sas:
       spi: 0x38C04B36(952126262)
         transform: esp-des esp-md5-hmac ,
         in use settings ={Transport, }
         conn id: 6, flow_id: SW:6, crypto map: vpnprof-head-1
         sa timing: remaining key lifetime (k/sec): (4515511/3395)
         IV size: 8 bytes
         replay detection support: Y
         Status: ACTIVE
      outbound ah sas:
      outbound pcp sas:
interface: Tunnel 1
    Crypto map tag: vpnprof-head-1, local addr 172.17.0.11
    protected vrf: (none)
    local  ident (addr/mask/prot/port): (172.17.0.11/255.255.255.255/47/0)
    remote ident (addr/mask/prot/port): (172.17.0.1/255.255.255.255/47/0)
    current_peer 172.17.0.1 port 500
      PERMIT, flags={origin_is_acl,}
     #pkts encaps: 134, #pkts encrypt: 134, #pkts digest: 134
     #pkts decaps: 118, #pkts decrypt: 118, #pkts verify: 118
     #pkts compressed: 0, #pkts decompressed: 0
     #pkts not compressed: 0, #pkts compr. failed: 0
     #pkts not decompressed: 0, #pkts decompress failed: 0
     #send errors 22, #recv errors 0
      local crypto endpt.: 172.17.0.11, remote crypto endpt.: 172.17.0.1
      path mtu 1500, ip mtu 1500, ip mtu idb FastEthernet0/0/0
      current outbound spi: 0xA75421B1(2807308721)
      inbound esp sas:
       spi: 0x96185188(2518176136)
         transform: esp-des esp-md5-hmac ,
         in use settings ={Transport, }
         conn id: 3, flow_id: SW:3, crypto map: vpnprof-head-1
         sa timing: remaining key lifetime (k/sec): (4569747/3242)
         IV size: 8 bytes
         replay detection support: Y
         Status: ACTIVE
      inbound ah sas:
```

```
  inbound pcp sas:
  outbound esp sas:
   spi: 0xA75421B1(2807308721)
     transform: esp-des esp-md5-hmac ,
     in use settings ={Transport, }
     conn id: 4, flow_id: SW:4, crypto map: vpnprof-head-1
     sa timing: remaining key lifetime (k/sec): (4569745/3242)
     IV size: 8 bytes
     replay detection support: Y
     Status: ACTIVE
  outbound ah sas:
  outbound pcp sas:
protected vrf: (none)
local  ident (addr/mask/prot/port): (172.17.0.11/255.255.255.255/47/0)
remote ident (addr/mask/prot/port): (172.17.0.5/255.255.255.255/47/0)
current_peer 172.17.0.5 port 500
  PERMIT, flags={origin_is_acl,}
 #pkts encaps: 244, #pkts encrypt: 244, #pkts digest: 244
 #pkts decaps: 253, #pkts decrypt: 253, #pkts verify: 253
 #pkts compressed: 0, #pkts decompressed: 0
 #pkts not compressed: 0, #pkts compr. failed: 0
 #pkts not decompressed: 0, #pkts decompress failed: 0
 #send errors 1, #recv errors 0
  local crypto endpt.: 172.17.0.11, remote crypto endpt.: 172.17.0.5
  path mtu 1500, ip mtu 1500, ip mtu idb FastEthernet0/0/0
  current outbound spi: 0x3C50B3AB(1011921835)
  inbound esp sas:
   spi: 0x3EBE84EF(1052673263)
     transform: esp-des esp-md5-hmac ,
     in use settings ={Transport, }
     conn id: 1, flow_id: SW:1, crypto map: vpnprof-head-1
     sa timing: remaining key lifetime (k/sec): (4549326/2779)
     IV size: 8 bytes
     replay detection support: Y
     Status: ACTIVE
  inbound ah sas:
  inbound pcp sas:
  outbound esp sas:
   spi: 0x3C50B3AB(1011921835)
     transform: esp-des esp-md5-hmac ,
     in use settings ={Transport, }
     conn id: 2, flow_id: SW:2, crypto map: vpnprof-head-1
     sa timing: remaining key lifetime (k/sec): (4549327/2779)
     IV size: 8 bytes
     replay detection support: Y
     Status: ACTIVE
  outbound ah sas:
  outbound pcp sas:
protected vrf: (none)
local  ident (addr/mask/prot/port): (172.17.0.11/255.255.255.255/47/0)
remote ident (addr/mask/prot/port): (172.17.0.12/255.255.255.255/47/0)
current_peer 172.17.0.12 port 500
  PERMIT, flags={origin_is_acl,}
 #pkts encaps: 0, #pkts encrypt: 0, #pkts digest: 0
 #pkts decaps: 2, #pkts decrypt: 2, #pkts verify: 2
 #pkts compressed: 0, #pkts decompressed: 0
 #pkts not compressed: 0, #pkts compr. failed: 0
 #pkts not decompressed: 0, #pkts decompress failed: 0
 #send errors 0, #recv errors 0
  local crypto endpt.: 172.17.0.11, remote crypto endpt.: 172.17.0.12
  path mtu 1500, ip mtu 1500, ip mtu idb FastEthernet0/0/0
  current outbound spi: 0x38C04B36(952126262)
  inbound esp sas:
   spi: 0xA2EC557(170837335)
     transform: esp-des esp-md5-hmac ,
     in use settings ={Transport, }
     conn id: 5, flow_id: SW:5, crypto map: vpnprof-head-1
     sa timing: remaining key lifetime (k/sec): (4515510/3395)
     IV size: 8 bytes
     replay detection support: Y
     Status: ACTIVE
  inbound ah sas:
  inbound pcp sas:
```

```
     outbound esp sas:
      spi: 0x38C04B36(952126262)
        transform: esp-des esp-md5-hmac ,
        in use settings ={Transport, }
        conn id: 6, flow_id: SW:6, crypto map: vpnprof-head-1
        sa timing: remaining key lifetime (k/sec): (4515511/3395)
        IV size: 8 bytes
        replay detection support: Y
        Status: ACTIVE
     outbound ah sas:
     outbound pcp sas:
Spoke1#
```

Additional References

The following sections provide references related to the Sharing IPSec with Tunnel Protection feature.

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Security commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS Security Command Reference*
Configuring a DMVPN	Dynamic Multipoint VPN (DMVPN)
Configuring basic IP Security (IPsec) Virtual Private Networks (VPNs)	Configuring Security for VPNs with IPsec

Standards

Standard	Title
None	--

MIBs

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

RFCs

RFC	Title
RFC 2401	*Security Architecture for the Internet Protocol*

RFC	Title
RFC 2547	*BGP/MPLS VPNs*
RFC 2784	*Generic Routing Encapsulation (GRE)*

Technical Assistance

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

Feature Information for Sharing IPsec with Tunnel Protection

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 4 Feature Information for Sharing IPsec with Tunnel Protection

Feature Name	Releases	Feature Information
Sharing IPSect with Tunnel Protection	Cisco IOS XE Release 2.5	The Sharing IPsec with Tunnel Protection feature allows an Internet Protocol Security (IPsec) session to be shared between two or more generic routing encapsulation (GRE) tunnel interfaces. In Cisco IOS XE Release 2.5, this feature was introduced on the Cisco ASR 1000 Series Aggregation Routers. The following command was introduced or modified by this feature: **tunnel protection ipsec profile shared.**

Glossary

GRE-- generic routing encapsulation. Tunnels that provide a specific pathway across the shared WAN and encapsulate traffic with new packet headers to ensure delivery to specific destinations. The network is private because traffic can enter a tunnel only at an endpoint. Tunnels do not provide true confidentiality (encryption does) but can carry encrypted traffic.

GRE tunneling can also be used to encapsulate non-IP traffic into IP and send it over the Internet or IP network. The Internet Package Exchange (IPX) and AppleTalk protocols are examples of non-IP traffic.

IKE-- Internet Key Exchange. A hybrid protocol that implements Oakley key exchange and Skeme key exchange inside the ISAKMP framework. Although IKE can be used with other protocols, its initial implementation is with IPsec. IKE provides authentication of the IPsec peers, negotiates IPsec keys, and negotiates IPsec security associations.

IPsec-- IP security. A framework of open standards developed by the Internet Engineering Task Force (IETF). IPsec provides security for transmission of sensitive information over unprotected networks such as the Internet. IPsec acts at the network layer, protecting and authenticating IP packets between participating IPsec peers, such as Cisco routers.

ISAKMP-- Internet Security Association Key Management Protocol. A protocol framework that defines payload formats, the mechanics of implementing a key exchange protocol, and the negotiation of a security association.

NHRP-- Next Hop Resolution Protocol. Protocol that routers, access servers, and hosts can use to discover the addresses of other routers and hosts connected to an NBMA network.

The Cisco implementation of NHRP supports the IETF draft version 11 of NBMA NHRP.

The Cisco implementation of NHRP supports IP Version 4, Internet Packet Exchange (IPX) network layers, and, at the link layer, ATM, Ethernet, SMDS, and multipoint tunnel networks. Although NHRP is available on Ethernet, NHRP need not be implemented over Ethernet media because Ethernet is capable of broadcasting. Ethernet support is unnecessary (and not provided) for IPX.

SA-- security association. Describes how two or more entities use security services to communicate securely. For example, an IPsec SA defines the encryption algorithm (if used), the authentication algorithm, and the shared session key to be used during the IPsec connection.

Both IPsec and IKE require and use SAs to identify the parameters of their connections. IKE can negotiate and establish its own SA. The IPsec SA is established either by IKE or by manual user configuration.

transform-- List of operations performed on a data flow to provide data authentication, data confidentiality, and data compression. For example, one transform is the ESP protocol with the HMAC-MD5 authentication algorithm; another transform is the AH protocol with the 56-bit DES encryption algorithm and the ESP protocol with the HMAC-SHA authentication algorithm.

tunnel-- A secure communication path between two peers, such as two routers. It does not refer to using IPsec in tunnel mode.

VPN-- Virtual Private Network. A framework that consists of multiple peers transmitting private data securely to one another over an otherwise public infrastructure. In this framework, inbound and outbound network traffic is protected using protocols that tunnel and encrypt all data. This framework permits networks to extend beyond their local topology, while remote users are provided with the appearance and functionality of a direct network connection.