IPsec Virtual Tunnel Interfaces

IPsec virtual tunnel interfaces (VTIs) provide a routable interface type for terminating IPsec tunnels and an easy way to define protection between sites to form an overlay network. IPsec VTIs simplify the configuration of IPsec for protection of remote links, support multicast, and simplify network management and load balancing.

Note Security threats, as well as the cryptographic technologies to help protect against them, are constantly changing. For more information about the latest Cisco cryptographic recommendations, see the Next Generation Encryption (NGE) white paper.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and 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.

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 IPsec Virtual Tunnel Interfaces

Fragmentation

Fragmentation is not supported over IPsec tunnel. You can choose to set the lower MTU on hosts to avoid packet fragments or choose to fragment the packets on any device before it reaches ASR 920.

IPsec Transform Set

The IPsec transform set must be configured in tunnel mode only.

IKE Security Association

The Internet Key Exchange (IKE) security association (SA) is bound to the VTI.

IPsec SA Traffic Selectors

Static VTIs (SVTIs) support only a single IPsec SA that is attached to the VTI interface. The traffic selector for the IPsec SA is always “IP any any.”

IPv4

This feature supports SVTIs that are configured to encapsulate IPv4 packets for 15.5(3)M and earlier releases.

Quality of Service (QoS) Traffic Shaping

The shaped traffic is process switched.

Tunnel Protection

Do not configure the **shared** keyword when using the **tunnel mode ipsec ipv4** command for IPsec IPv4 mode.

Traceroute

The traceroute function with crypto offload on VTIs is not supported.

Information About IPsec Virtual Tunnel Interfaces

The use of IPsec VTIs can simplify the configuration process when you need to provide protection for remote access and it provides an alternative to using generic routing encapsulation (GRE) or Layer 2 Tunneling Protocol (L2TP) tunnels for encapsulation. A benefit of using IPsec VTIs is that the configuration does not require static mapping of IPsec sessions to a physical interface. The IPsec tunnel endpoint is associated with an actual (virtual) interface. Because there is a routable interface at the tunnel endpoint, many common interface capabilities can be applied to the IPsec tunnel.

The IPsec VTI allows for the flexibility of sending and receiving both IP unicast and multicast encrypted traffic on any physical interface, such as in the case of multiple paths. Traffic is encrypted or decrypted when it is forwarded from or to the tunnel interface and is managed by the IP routing table. Using IP routing to forward the traffic to the tunnel interface simplifies the IPsec VPN configuration . Because DVTIs function like any other real interface you can apply quality of service (QoS), firewall, and other security services as soon as the tunnel is active.

Without VPN Acceleration Module2+ (VAM2+) accelerating virtual interfaces, the packet traversing an IPsec virtual interface is directed to the Router Processor (RP) for encapsulation. This method tends to be slow and has limited scalability. In hardware crypto mode, all the IPsec VTIs are accelerated by the VAM2+ crypto engine, and all traffic going through the tunnel is encrypted and decrypted by the VAM2+.

The following sections provide details about the IPSec VTI:

Benefits of Using IPsec Virtual Tunnel Interfaces

IPsec VTIs allow you to configure a virtual interface to which you can apply features. Features for clear-text packets are configured on the VTI. Features for encrypted packets are applied on the physical outside interface. When IPsec VTIs are used, you can separate the application of features such as Network Address Translation (NAT), ACLs, and QoS and apply them to clear-text, or encrypted text, or both.

There are two types of VTI interfaces: static VTIs (SVTIs) and dynamic VTIs (DVTIs).

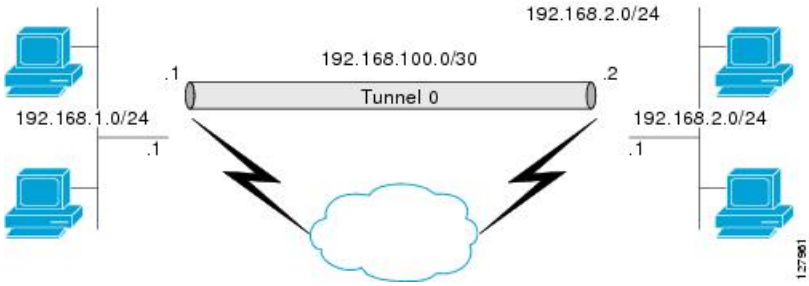
Static Virtual Tunnel Interfaces

SVTI configurations can be used for site-to-site connectivity in which a tunnel provides always-on access between two sites.

Additionally, multiple Cisco IOS software features can be configured directly on the tunnel interface and on the physical egress interface of the tunnel interface. This direct configuration allows users to have solid control on the application of the features in the pre- or post-encryption path.

The figure below illustrates how a SVTI is used.

Figure 1: IPsec SVTI



The IPsec VTI supports native IPsec tunneling and exhibits most of the properties of a physical interface.

Dynamic Virtual Tunnel Interfaces

DVTIs can provide highly secure and scalable connectivity for remote-access VPNs. The DVTI technology replaces dynamic crypto maps and the dynamic hub-and-spoke method for establishing tunnels.

Note You can configure DVTIs with IKEv1 or IKEv2. The legacy crypto map based configuration supports DVTIs with IKEv1 only. A DVTI configuration with IKEv2 is supported only in FlexVPN.

DVTIs can be used for both the server and the remote configuration. The tunnels provide an on-demand separate virtual access interface for each VPN session. The configuration of the virtual access interfaces is cloned from a virtual template configuration, which includes the IPsec configuration and any Cisco IOS software feature configured on the virtual template interface, such as QoS, NetFlow, or ACLs.

DVTIs function like any other real interface, so you can apply QoS, firewall, or other security services as soon as the tunnel is active. QoS features can be used to improve the performance of various applications across the network. Any combination of QoS features offered in Cisco IOS software can be used to support voice, video, or data applications.

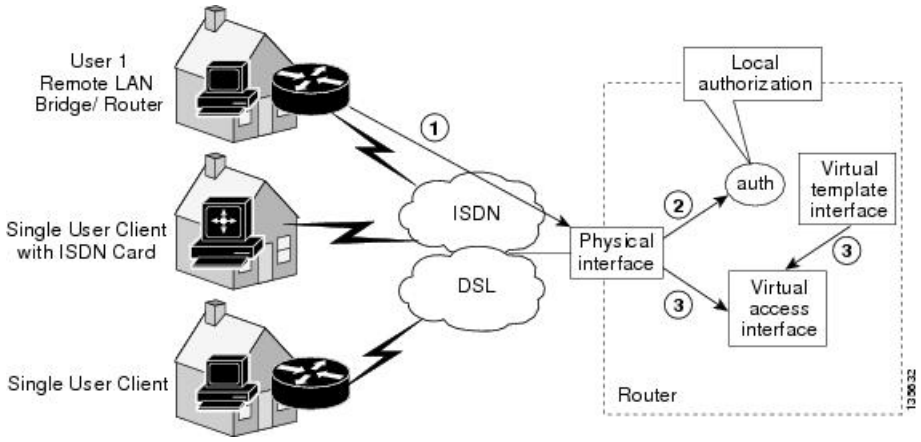
DVTIs provide efficiency in the use of IP addresses and provide secure connectivity. DVTIs allow dynamically downloadable per-group and per-user policies to be configured on a RADIUS server. The per-group or per-user definition can be created using an extended authentication (Xauth) User or Unity group, or can be derived from a certificate. DVTIs are standards based, so interoperability in a multiple-vendor environment is supported. IPsec DVTIs allow you to create highly secure connectivity for remote access VPNs and can be combined with Cisco Architecture for Voice, Video, and Integrated Data (AVVID) to deliver converged voice, video, and data over IP networks. The DVTI simplifies VPN routing and forwarding- (VRF-) aware IPsec deployment. The VRF is configured on the interface.

A DVTI requires minimal configuration on the router. A single virtual template can be configured and cloned.

The DVTI creates an interface for IPsec sessions and uses the virtual template infrastructure for dynamic instantiation and management of dynamic IPsec VTIs. The virtual template infrastructure is extended to create dynamic virtual-access tunnel interfaces. DVTIs are used in hub-and-spoke configurations. A single DVTI can support several static VTIs.

The figure below illustrates the DVTI authentication path.

Figure 2: Dynamic IPsec VTI



The authentication shown in the figure above follows this path:

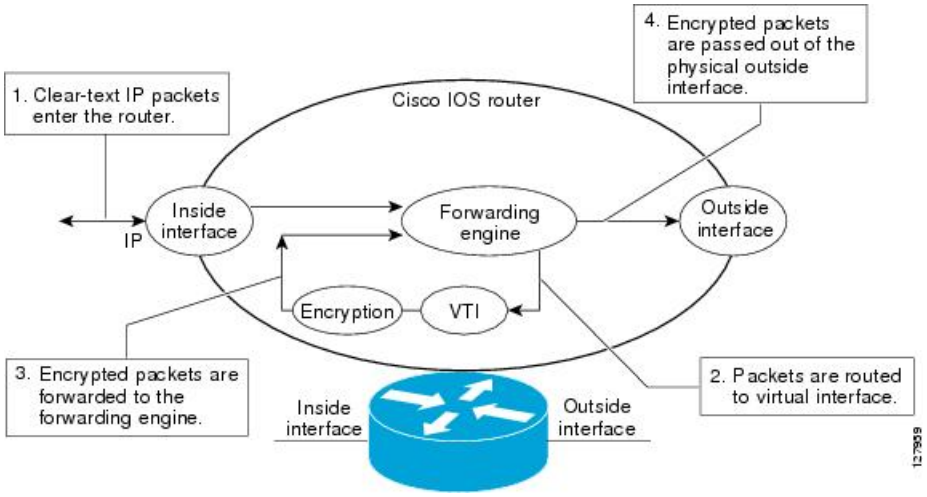
1. User 1 calls the router.
2. Router 1 authenticates User 1.
3. IPsec clones the virtual access interface from the virtual template interface.

Traffic Encryption with the IPsec Virtual Tunnel Interface

When an IPsec VTI is configured, encryption occurs in the tunnel. Traffic is encrypted when it is forwarded to the tunnel interface. Traffic forwarding is handled by the IP routing table, and dynamic or static routing can be used to route traffic to the SVTI. DVTI uses reverse route injection to further simplify the routing configurations. Using IP routing to forward the traffic to encryption simplifies the IPsec VPN configuration . The IPsec virtual tunnel also allows you to encrypt multicast traffic with IPsec.

IPsec packet flow into the IPSec tunnel is illustrated in the figure below.

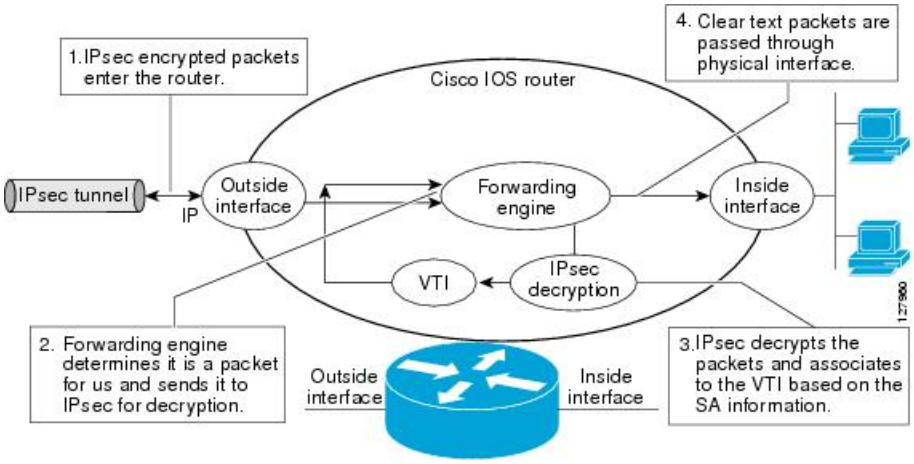
Figure 3: Packet Flow into the IPsec Tunnel



After packets arrive on the inside interface, the forwarding engine switches the packets to the VTI, where they are encrypted. The encrypted packets are handed back to the forwarding engine, where they are switched through the outside interface.

The figure below shows the packet flow out of the IPsec tunnel.

Figure 4: Packet Flow out of the IPsec Tunnel



Multi-SA Support for Dynamic Virtual Tunnel Interfaces for IKEv1

DVTI supports multiple IPsec SAs. The DVTI can accept multiple IPsec selectors that are proposed by the initiator.

The DVTIs allow per peer features to be applied on a dedicated interface. You can order features in such way that all features that are applied on the virtual access interfaces are applied before applying crypto. Additionally, all the features that are applied on the physical interfaces are applied after applying crypto. Clean routing is available across all VRFs so that there are no traffic leaks from one VRF to another before encrypting.

Multi-SA VTIs ensure interoperation with third-party devices and provide a flexible, clean, and modular feature set.

Multi-SA VTIs enable a clean Cisco IOS infrastructure, even when the Cisco IOS software interoperates with third-party devices that implement only crypto maps.

VRF and Scalability of the Baseline Configuration for IKEv1

Virtual access instances inherit the Inside-VRF (IVRF) from the template configuration. Users must configure several templates to enforce an appropriate IVRF for each customer. The number of templates must be equal to the number of customers connecting to the headend. Such a configuration is cumbersome and undesirable.

This complication can be avoided by allowing the IKE profile to override the virtual access VRF with the VRF configured on the IKE profile. An even better solution will be to allow the IKE profile to override the virtual access VRF using AAA, but this method is supported only for IKEv2.

This complication can be avoided by allowing the IKE profile to override the virtual access VRF with the VRF configured on the IKE profile. A better solution is to allow the IKE profile to override the virtual access VRF using AAA, but this method is supported only for IKEv2.

The VRF configured in the ISAKMP profile is applied to the virtual access first. Then the configuration from virtual template is applied to the virtual access. If your virtual template contains **ip vrf forwarding** command configuration, the VRF from the template overrides the VRF from the ISAKMP profile.

Rules for Initial Configuration of a VRF

The following rules must be applied during the initial configuration of VRF:

- If you configure IVRF in the IKE profile without configuring it in the virtual template, then you must apply the VRF from the IKE profile on each virtual access derived from this IKE profile.
- If you configure VRF in an IKE profile and virtual template, then the virtual template IVRF gets precedence.

Rules for Changing the VRF

If you change the VRF configured in an IKE profile, all the IKE SAs, IPsec SAs, and the virtual access identifier derived from this profile will get deleted. The same rule applies when the VRF is configured on the IKE profile for the first time.

Multi-SA Support for Dynamic Virtual Tunnel Interfaces for IKEv2

The configuration of an IKEv2 profile in an IPsec profile on an IKEv2 responder is not mandatory. The IPsec DVTI sessions using the same virtual template can use different IKEv2 profiles, thus avoiding the need for a separate virtual template for each DVTI session that needs a different IKEv2 profile. Such an arrangement helps reduce the configuration size and save virtual template Interface Descriptor Block (IDB).

The IKEv2 authorization policy, which is a container of IKEv2 local AAA group authorization parameters, contains an AAA attribute AAA_AT_IPSEC_FLOW_LIMIT and the **ipsec flow- limit** command. This attribute limits the number of IPsec flows that can terminate on an IPsec DVTI virtual access interface.

The value specified by the **ipsec flow- limit** command from the AAA overrides the value set by the **set security-policy limit** command from the IPsec profile. Any change to the value set by the **set security-policy limit** command in the IPSec profile is not applied to the current session but is applied to subsequent sessions.

If the value set by the **set security-policy limit** command is overridden by AAA, then the value from the IPsec profile is ignored, and any change to the value set by the **set security-policy limit** command in the IPsec profile does not affect the virtual access.

VRF and Scalability of Baseline Configuration for IKEv2

The IKEv2 multi-SA does not allow simultaneous configuration of a VRF and a template on the IKEv2 profile. Instead, the VRF can be configured on AAA and applied to the virtual access interface at the time of its creation.

You can use the AAA attribute INTERFACE_CONFIG to specify the **ip vrf forwarding**, **ip unnumbered** commands, and other interface configuration mode commands that are applied on the virtual access interface.

Note If you override VRF using AAA, you must also specify the **ip unnumbered** command using AAA because the **ip vrf forwarding** command removes the **ip unnumbered** command configuration from the interface.

Dynamic Virtual Tunnel Interface Life Cycle

IPsec profiles define the policy for DVTIs. The dynamic interface is created at the end of IKE Phase 1 and IKE Phase 1.5. The interface is deleted when the IPsec session to the peer is closed. The IPsec session is closed when both IKE and IPsec SAs to the peer are deleted.

Routing with IPsec Virtual Tunnel Interfaces

Because VTIs are routable interfaces, routing plays an important role in the encryption process. Traffic is encrypted only if it is forwarded out of the VTI, and traffic arriving on the VTI is decrypted and routed accordingly. VTIs allow you to establish an encryption tunnel using a real interface as the tunnel endpoint. You can route to the interface or apply services such as QoS, firewalls, network address translation (NAT), and NetFlow statistics as you would to any other interface. You can monitor the interface and route to it, and the interface provides benefits similar to other Cisco IOS interface.

FlexVPN Mixed Mode Support

The FlexVPN Mixed Mode feature provides support for carrying IPv4 traffic over IPsec IPv6 transport. This is the first phase towards providing dual stack support on the IPsec stack. This implementation does not support using a single IPsec security association (SA) pair for both IPv4 and IPv6 traffic.

This feature is only supported for Remote Access VPN with IKEv2 and Dynamic VTI.

The FlexVPN Mixed Mode feature provides support for carrying IPv6 traffic over IPsec IPv4 transport from Cisco IOS XE Everest 16.4.1.

IKE Profile Based Tunnel Selection

The IKE Profile Based Tunnel Selection feature uses the Internet Key Exchange (IKE) or Internet Key Exchange version 2 (IKEv2) profile to select a tunnel interface for an IPsec session. Use keywords **isakmp-profile** or **ikev2-profile** keyword in the **tunnel protection** command to specify an IKE profile or IKEv2 profile respectively.

The IKE Profile Based Tunnel Selection feature allows tunnel interfaces to share the tunnel source IP address and IPsec transform set without sharing the IPsec security association databases (SADBs) among tunnel interfaces thereby providing the following benefits:

- Tunnels are secure and there is no traffic leak.
- All tunnel types are supported.
- Seamless migration from IKEv1 to IKEv2 by accommodating configurations from legacy VPN technologies to coexist and share the local address with newer VPN technologies.
- Ability to set up multiple IKE and IPsec tunnels between peers sharing the same local or remote addresses.

Auto Tunnel Mode Support in IPsec

When configuring a VPN headend in a multiple vendor scenario, you must be aware of the technical details of the peer or responder. For example, some devices may use IPsec tunnels while others may use generic routing encapsulation (GRE) or IPsec tunnel, and sometimes, a tunnel may be IPv4 or IPv6. In the last case, you must configure an Internet Key Exchange (IKE) profile and a virtual template.

The Tunnel Mode Auto Selection feature eases the configuration and spares you about knowing the responder's details. This feature automatically applies the tunneling protocol (GRE or IPsec) and transport protocol (IPv4 or IPv6) on the virtual template as soon as the IKE profile creates the virtual access interface. This feature is useful on dual stack hubs aggregating multivendor remote access, such as Cisco AnyConnect VPN Client, Microsoft Windows7 Client, and so on.

Note The Tunnel Mode Auto Selection feature eases the configuration for a responder only. The tunnel must be statically configured for an initiator.

IPSec Mixed Mode Support for VTI

The IPSec Mixed Mode feature provides support for carrying IPv4 traffic over IPsec IPv6 transport. This is the first phase towards providing dual stack support on the IPsec stack. This implementation does not support using a single IPsec security association (SA) pair for both IPv4 and IPv6 traffic.

This feature is supported for SVTI as well as DVTI and IKEv1 as well as IKEv2.

How to Configure IPsec Virtual Tunnel Interfaces

Configuring Static IPsec Virtual Tunnel Interfaces

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto IPsec profile** *profile-name*
4. **set transform-set** *transform-set-name* [*transform-set-name2...transform-set-name6*]
5. **exit**
6. **interface** *type number*
7. **ip address** *address mask*

8. **tunnel mode ipsec ipv4**
9. **tunnel source** *interface-type interface-number*
10. **tunnel destination** *ip-address*
11. **tunnel protection IPsec profile** *profile-name*
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto IPsec profile** *profile-name* **Example:** `Device(config)# crypto IPsec profile PROF`	Defines the IPsec parameters that are to be used for IPsec encryption between two IPsec devices, and enters IPsec profile configuration mode.
Step 4	**set transform-set** *transform-set-name* [*transform-set-name2...transform-set-name6*] **Example:** `Device(ipsec-profile)# set transform-set tset`	Specifies which transform sets can be used .
Step 5	**exit** **Example:** `Device(ipsec-profile)# exit`	Exits IPsec profile configuration mode, and enters global configuration mode.
Step 6	**interface** *type number* **Example:** `Device(config)# interface tunnel 0`	Specifies the interface on which the tunnel will be configured and enters interface configuration mode.
Step 7	**ip address** *address mask* **Example:** `Device(config-if)# ip address 10.1.1.1 255.255.255.0`	Specifies the IP address and mask.
Step 8	**tunnel mode ipsec ipv4** **Example:** `Device(config-if)# tunnel mode ipsec ipv4`	Defines the mode for the tunnel.

	Command or Action	Purpose
Step 9	**tunnel source** *interface-type interface-number* **Example:** `Device(config-if)# tunnel source loopback 0`	Specifies the tunnel source as a loopback interface.
Step 10	**tunnel destination** *ip-address* **Example:** `Device(config-if)# tunnel destination 172.16.1.1`	Identifies the IP address of the tunnel destination.
Step 11	**tunnel protection IPsec profile** *profile-name* **Example:** `Device(config-if)# tunnel protection IPsec profile PROF`	Associates a tunnel interface with an IPsec profile.
Step 12	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring BGP over IPsec Virtual Tunnel Interfaces

Perform this task to optionally configure BGP over the virtual tunnel interfaces of two routers.

Before you begin

Perform steps in Configuring Static IPsec Virtual Tunnel Interfaces, on page 8.

SUMMARY STEPS

1. **router bgp** *autonomous-system-number*
2. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
3. **network** *network-ip-address* **mask** *subnet-mask*
4. **exit**
5. Enter the following commands on the second router.
6. **router bgp** *autonomous-system-number*
7. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
8. **network** *network-ip-address* **mask** *subnet-mask*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**router bgp** *autonomous-system-number* **Example:**	Enters router configuration mode and creates a BGP routing process.

	Command or Action	Purpose
	`Device(config)# router bgp 65510`	*autonomous-system-number* —Number of an autonomous system that identifies the router to other BGP routers and tags the routing information that is passed along. Number in the range from 1 to 65535. In the example, the first router in this procedure is identified as "65510".
Step 2	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 10.1.1.2 remote-as 65511`	*ip-address* —IP address of the adjacent router's tunnel interface. *autonomous-system-number* —Number of an autonomous system that identifies the router of the second router. Number in the range from 1 to 65535.
Step 3	**network** *network-ip-address* **mask** *subnet-mask* **Example:** `Device(config-router)# network 2.2.2.0 mask 255.255.255.0`	*network-ip-address*—IP address of the network advertised in BGP. For example, the IP address of a loopback interface. *subnet-mask*—subnet mask of the network advertised in BGP. **Note** The BGP network command network and mask *must* exactly match a route that is already in the routing table for it to be brought into BGP and advertised to BGP neighbors. This is different from EIGRP, OSPF where the network statement just has to "cover" an interface network and it will pick up the network with mask from the interface.
Step 4	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode.
Step 5	Enter the following commands on the second router.	
Step 6	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 65511`	Enters router configuration mode and creates a BGP routing process. *autonomous-system-number* —Number of an autonomous system that identifies the router to other BGP routers and tags the routing information that is passed along. Number in the range from 1 to 65535. In the example, the second router in this procedure is identified as "65511".
Step 7	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 10.1.1.1 remote-as 65510`	*ip-address* —IP address of the adjacent router's tunnel interface.

	Command or Action	Purpose
Step 8	**network** *network-ip-address* **mask** *subnet-mask* **Example:** `Device(config-router)# network 1.1.1.0 mask 255.255.255.0`	*network-ip-address*—IP address of the network advertised in BGP. For example, the IP address of a loopback interface. *subnet-mask*—subnet mask of the network advertised in BGP. **Note** Use the exact network IP address and subnet mask.

Configuring Dynamic IPsec Virtual Tunnel Interfaces

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto ipsec profile** *profile-name*
4. **set transform-set** *transform-set-name* [*transform-set-name2...transform-set-name6*]
5. **exit**
6. **interface virtual-template** *number* **type tunnel**
7. **tunnel mode ipsec ipv4**
8. **tunnel protection IPsec profile** *profile-name*
9. **exit**
10. **crypto isakamp profile** *profile-name*
11. **match identity address** *ip-address mask*
12. **virtual template** *template-number*
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto ipsec profile** *profile-name* **Example:** `Device(config)# crypto ipsec profile PROF`	Defines the IPsec parameters that are to be used for IPsec encryption between two IPsec devices and enters IPsec profile configuration mode.

	Command or Action	Purpose
Step 4	**set transform-set** *transform-set-name* [*transform-set-name2...transform-set-name6*] **Example:** `Device(ipsec-profile)# set transform-set tset`	Specifies which transform sets can be used with the crypto map entry.
Step 5	**exit** **Example:** `Device(ipsec-profile)# exit`	Exits ipsec profile configuration mode and enters global configuration mode.
Step 6	**interface virtual-template** *number* **type tunnel** **Example:** `Device(config)# interface virtual-template 2 type tunnel`	Defines a virtual-template tunnel interface and enters interface configuration mode.
Step 7	**tunnel mode ipsec ipv4** **Example:** `Device(config-if)# tunnel mode ipsec ipv4`	Defines the mode for the tunnel.
Step 8	**tunnel protection IPsec profile** *profile-name* **Example:** `Device(config-if)# tunnel protection ipsec profile PROF`	Associates a tunnel interface with an IPsec profile.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 10	**crypto isakamp profile** *profile-name* **Example:** `Device(config)# crypto isakamp profile profile1`	Defines the ISAKMP profile to be used for the virtual template.
Step 11	**match identity address** *ip-address mask* **Example:** `Device(conf-isa-prof)# match identity address 10.1.1.0 255.255.255.0`	Matches an identity from the ISAKMP profile and enters isakmp-profile configuration mode.
Step 12	**virtual template** *template-number* **Example:** `Device(config)# virtual-template 1`	Specifies the virtual template attached to the ISAKMP profile.
Step 13	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.

Configuring Multi-SA Support for Dynamic Virtual Tunnel Interfaces Using IKEv1

Note Security threats, as well as the cryptographic technologies to help protect against them, are constantly changing. For more information about the latest Cisco cryptographic recommendations, see the Next Generation Encryption (NGE) white paper.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip vrf** *vrf-name*
4. **rd** *route-distinguisher*
5. **exit**
6. **crypto keyring** *keyring-name*
7. **pre-shared-key** *address* **key** *key*
8. **exit**
9. **crypto isakmp profile** *profile-name*
10. **keyring** *keyring-name*
11. **match identity** *address mask*
12. **virtual-template** *template-number*
13. **exit**
14. **crypto ipsec transform-set** *transform-set-name transform1* [*transform2*] [*transform3*]
15. **exit**
16. **crypto ipsec profile** *name*
17. **set security-policy limit** *maximum-limit*
18. **set transform-set** *transform-set-name* [*transform-set-name2 .... transform-set-name6*]
19. **exit**
20. **interface virtual-template** *number type tunnel*
21. **ip vrf forwarding** *vrf-name*
22. **ip unnumbered** *type number*
23. **tunnel mode ipsec ipv4**
24. **tunnel protection profile ipsec** *profile-name*
25. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip vrf** *vrf-name* **Example:** `Device(config)# ip vrf VRF-100-1`	Defines the VRF instance and enters VRF configuration mode.
Step 4	**rd** *route-distinguisher* **Example:** `Device(config-vrf)# rd 100:21`	Creates routing and forwarding tables for a VRF.
Step 5	**exit** **Example:** `Device(config-vrf)# exit`	Exits VRF configuration mode and enters global configuration mode.
Step 6	**crypto keyring** *keyring-name* **Example:** `Device(config)# crypto keyring cisco-100-1`	Defines a crypto key ring and enters key ring configuration mode.
Step 7	**pre-shared-key** *address* **key** *key* **Example:** `Device(config-keyring)# pre-shared-key address 10.1.1.1 key cisco-100-1`	Defines the preshared key to be used for Internet Key Exchange (IKE) authentication.
Step 8	**exit** **Example:** `Device(config-keyring)# exit`	Exits keyring configuration mode and enters global configuration mode.
Step 9	**crypto isakmp profile** *profile-name* **Example:** `Device(config)# crypto isakmp profile cisco-isakmp-profile-100-1`	Defines an ISAKMP profile and enters ISAKMP configuration mode.
Step 10	**keyring** *keyring-name* **Example:** `Device(conf-isa-prof)# keyring cisco-100-1`	Configures a key ring in ISAKMP mode.
Step 11	**match identity** *address mask* **Example:** `Device(conf-isa-prof)# match identity address 10.1.1.0 255.255.255.0`	Matches an identity from the ISAKMP profile.
Step 12	**virtual-template** *template-number* **Example:**	Specifies the virtual template that will be used to clone virtual access interfaces.

	Command or Action	Purpose
	`Device(conf-isa-prof)# virtual-template 101`	
Step 13	**exit** **Example:** `Device(conf-isa-prof)# exit`	Exits ISAKMP profile configuration mode and enters global configuration mode.
Step 14	**crypto ipsec transform-set** *transform-set-name transform1* [*transform2*] [*transform3*] **Example:** `Device(config)# crypto ipsec transform-set cisco esp-aes esp-sha-hmac`	Defines the transform set and enters crypto transform configuration mode.
Step 15	**exit** **Example:** `Device(conf-crypto-trans)# exit`	Exits crypto transform configuration mode and enters global configuration mode.
Step 16	**crypto ipsec profile** *name* **Example:** `Device(config)# crypto ipsec profile cisco-ipsec-profile-101`	Defines the IPsec parameters used for IPsec encryption between two IPsec devices, and enters IPsec profile configuration mode.
Step 17	**set security-policy limit** *maximum-limit* **Example:** `Device(ipsec-profile)# set security-policy limit 3`	Defines an upper limit to the number of flows that can be created for an individual virtual access interface.
Step 18	**set transform-set** *transform-set-name* [*transform-set-name2 .... transform-set-name6*] **Example:** `Device(ipsec-profile)# set transform-set cisco`	Specifies the transform sets to be used with the crypto map entry.
Step 19	**exit** **Example:** `Device(ipsec-profile)# exit`	Exits IPsec profile and enters global configuration mode.
Step 20	**interface virtual-template** *number type tunnel* **Example:** `Device(config)# interface virtual-template 101 type tunnel`	Creates a virtual template interface that can be configured interface and enters interface configuration mode.
Step 21	**ip vrf forwarding** *vrf-name* **Example:** `Device(config-if)# ip vrf forwarding VRF-100-1`	Associates a VRF instance with a virtual-template interface.
Step 22	**ip unnumbered** *type number* **Example:**	Enables IP processing on an interface without assigning an explicit IP address to the interface.

	Command or Action	Purpose
	`Device(config-if)# ip unnumbered GigabitEthernet 0.0`	
Step 23	**tunnel mode ipsec ipv4** **Example:** `Device(config-if)# tunnel mode ipsec ipv4`	Defines the mode for the tunnel.
Step 24	**tunnel protection profile ipsec** *profile-name* **Example:** `Device(config-if)# tunnel protection ipsec profile PROF`	Associates a tunnel interface with an IPsec profile.
Step 25	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode, and returns to privileged EXEC mode.

Configuring Multi-SA Support for Dynamic Virtual Tunnel Interfaces Using IKEv2

Perform the following tasks to configure Multi-SA for DVTIs using IKEv2:

Defining an AAA Attribute List

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **aaa new-model**
4. **aaa authorization network** *list-name* **local**
5. **aaa attribute list** *list-name*
6. **attribute type** *name value*
7. **attribute type** *name value*
8. **aaa session-id common**
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables the AAA access control model.
Step 4	**aaa authorization network** *list-name* **local** **Example:** `Device(config)# aaa authorization network group-list local`	Sets parameters that restrict user access to a network.
Step 5	**aaa attribute list** *list-name* **Example:** `Device(config)# aaa attribute list aaa-cisco-ikev2-profile-100-1`	Specifies an AAA attribute list that is defined in global configuration mode. • The “interface-config” attribute in the AAA attribute list is used to apply interface commands on the virtual access interface associated with the IKEv2 session.
Step 6	**attribute type** *name value* **Example:** `Device(config)# attribute type interface-config "ip vrf forwarding VRF-100-1"`	Defines an attribute type that is to be added to an attribute list locally on a device.
Step 7	**attribute type** *name value* **Example:** `Device(config)# attribute type interface-config "ip unnumbered Ethernet 0/0"`	Defines an attribute type that is to be added to an attribute list locally on a device.
Step 8	**aaa session-id common** **Example:** `Device(config)# aaa session-id common`	Ensures that the same session ID will be used for each AAA accounting service type within a call.
Step 9	**end** **Example:** `Device(config)# end`	Exits global configuration mode, and returns to privileged EXEC mode.

Configuring the VRF

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip vrf** *vrf-name*
4. **rd** *route-distinguisher*

5. **route-target export** *route-target-ext-community*
6. **route-target import** *route-target-ext-community*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip vrf** *vrf-name* **Example:** `Device(config)# ip vrf VRF-100-1`	Defines the VRF instance and enters VRF configuration mode.
Step 4	**rd** *route-distinguisher* **Example:** `Device(config-vrf)# rd 100:21`	Creates routing and forwarding tables for a VRF.
Step 5	**route-target export** *route-target-ext-community* **Example:** `Device(config-vrf)# route-target export 101:1`	(Optional) Creates a route-target export extended community for a VRF.
Step 6	**route-target import** *route-target-ext-community* **Example:** `Device(config-vrf)# route-target import 101:1`	(Optional) Creates a route-target import extended community for a VRF.
Step 7	**end** **Example:** `Device(config)# end`	Exits VRF configuration mode, and returns to privileged EXEC mode.

Configuring Internet Key Exchange Version 2 (IKEv2)

Configuring IKEv2 Proposal

Refer to the “IKEv2 Smart Defaults” section for information on the default IKEv2 proposal.

Perform this task to override the default IKEv2 proposal or to manually configure the proposals if you do not want to use the default proposal.

An IKEv2 proposal is a set of transforms used in the negotiation of IKEv2 SA as part of the IKE_SA_INIT exchange. An IKEv2 proposal is regarded as complete only when it has at least an encryption algorithm, an integrity algorithm, and a Diffie-Hellman (DH) group configured. If no proposal is configured and attached to an IKEv2 policy, the default proposal in the default IKEv2 policy is used in negotiation.

Note Security threats, as well as the cryptographic technologies to help protect against them, are constantly changing. For more information about the latest Cisco cryptographic recommendations, see the Next Generation Encryption (NGE) white paper.

Although the IKEv2 proposal is similar to the **crypto isakmp policy** command, the IKEv2 proposal differs as follows:

- An IKEv2 proposal allows configuring one or more transforms for each transform type.
- An IKEv2 proposal does not have any associated priority.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto ikev2 proposal** *name*
4. **encryption** *encryption-type...*
5. **integrity** *integrity-type...*
6. **group** *group-type...*
7. **prf** *prf-algorithm*
8. **end**
9. **show crypto ikev2 proposal** [*name* | **default**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto ikev2 proposal** *name* **Example:** `Device(config)# crypto ikev2 proposal proposal1`	Overrides the default IKEv2 proposal, defines an IKEv2 proposal name, and enters IKEv2 proposal configuration mode.
Step 4	**encryption** *encryption-type...* **Example:** `Device(config-ikev2-proposal)# encryption aes-cbc-128 aes-cbc-192`	Specifies one or more transforms of the encryption type, which are as follows: • **3des** (No longer recommended) • **aes-cbc-128** • **aes-cbc-192** • **aes-cbc-256**

	Command or Action	Purpose
		• **aes-gcm-128** • **aes-gcm-256**
Step 5	**integrity** *integrity-type...* **Example:** `Device(config-ikev2-proposal)# integrity sha1`	Specifies one or more transforms of the integrity algorithm type, which are as follows: • The **md5** keyword specifies MD5 (HMAC variant) as the hash algorithm. (No longer recommended) • The **sha1** keyword specifies SHA-1 (HMAC variant) as the hash algorithm. • The **sha256** keyword specifies SHA-2 family 256-bit (HMAC variant) as the hash algorithm. • The **sha384** keyword specifies SHA-2 family 384-bit (HMAC variant) as the hash algorithm. • The **sha512** keyword specifies SHA-2 family 512-bit (HMAC variant) as the hash algorithm. **Note** An integrity algorithm type cannot be specified if you specify Advanced Encryption Standard (AES) in Galois/Counter Mode (AES GCM) as the encryption type.

Step 6	**group** *group-type...*
	Example:
	`Device(config-ikev2-proposal)# group 14`
		Specifies the Diffie-Hellman (DH) group identifier.
		• The default DH group identifiers are group 2 and 5 in the IKEv2 proposal.
		  • **1**—768-bit DH (No longer recommended).
		  • **2**—1024-bit DH (No longer recommended).
		  • **5**—1536-bit DH (No longer recommended).
		  • **14**—Specifies the 2048-bit DH group.
		  • **15**—Specifies the 3072-bit DH group.
		  • **16**—Specifies the 4096-bit DH group.
		  • **19**—Specifies the 256-bit elliptic curve DH (ECDH) group.
		  • **20**—Specifies the 384-bit ECDH group.
		  • **24**—Specifies the 2048-bit DH group.
		The group chosen must be strong enough (have enough bits) to protect the IPsec keys during negotiation. A generally accepted guideline recommends the use of a 2048-bit group after 2013 (until 2030). Either group 14 or group 24 can be selected to meet this guideline. Even if a longer-lived security method is needed, the use of Elliptic Curve Cryptography is recommended, but group 15 and group 16 can also be considered.

	Command or Action	Purpose
Step 7	**prf** *prf-algorithm* **Example:** `Device(config-ikev2-proposal)# prf sha256 sha512`	Specifies one or more of the Pseudo-Random Function (PRF) algorithm, which are as follows: • **md5** • **sha1** • **sha256** • **sha384** • **sha512** **Note** This step is mandatory if the encryption type is AES-GCM—**aes-gmc-128** or **aes-gmc-256**. If the encryption algorithm is not AES-GCM, the PRF algorithm is the same as the specified integrity algorithm. However, you can specify a PRF algorithm, if required.
Step 8	**end** **Example:** `Device(config-ikev2-proposal)# end`	Exits IKEv2 proposal configuration mode and returns to privileged EXEC mode.
Step 9	**show crypto ikev2 proposal** [*name* | **default**] **Example:** `Device# show crypto ikev2 proposal default`	(Optional) Displays the IKEv2 proposal.

Configuring IKEv2 Policies

See the “IKEv2 Smart Defaults” section for information about the default IKEv2 policy.

Perform this task to override the default IKEv2 policy or to manually configure the policies if you do not want to use the default policy.

An IKEv2 policy must contain at least one proposal to be considered as complete and can have match statements, which are used as selection criteria to select a policy for negotiation. During the initial exchange, the local address (IPv4 or IPv6) and the Front Door VRF (FVRF) of the negotiating SA are matched with the policy and the proposal is selected.

The following rules apply to the match statements:

- An IKEv2 policy without any match statements will match all peers in the global FVRF.
- An IKEv2 policy can have only one match FVRF statement.
- An IKEv2 policy can have one or more match address local statements.
- When a policy is selected, multiple match statements of the same type are logically ORed and match statements of different types are logically ANDed.
- There is no precedence between match statements of different types.
- Configuration of overlapping policies is considered a misconfiguration. In the case of multiple, possible policy matches, the first policy is selected.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto ikev2 policy** *name*
4. **proposal** *name*
5. **match fvrf** {*fvrf-name* | **any**}
6. **match address local** {*ipv4-address* | *ipv6-address*}
7. **end**
8. **show crypto ikev2 policy** [*policy-name* | **default**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto ikev2 policy** *name* **Example:** `Device(config)# crypto ikev2 policy policy1`	Overrides the default IKEv2 policy, defines an IKEv2 policy name, and enters IKEv2 policy configuration mode.
Step 4	**proposal** *name* **Example:** `Device(config-ikev2-policy)# proposal proposal1`	Specifies the proposals that must be used with the policy. • The proposals are prioritized in the order of listing. **Note** You must specify at least one proposal. You can specify additional proposals with each proposal in a separate statement.
Step 5	**match fvrf** {*fvrf-name*	**any**} **Example:** `Device(config-ikev2-policy)# match fvrf any`	(Optional) Matches the policy based on a user-configured FVRF or any FVRF. • The default is global FVRF. **Note** The **match fvrf any** command must be explicitly configured in order to match any VRF. The FVRF specifies the VRF in which the IKEv2 packets are negotiated.
Step 6	**match address local** {*ipv4-address*	*ipv6-address*} **Example:** `Device(config-ikev2-policy)# match address local 10.0.0.1`	(Optional) Matches the policy based on the local IPv4 or IPv6 address. • The default matches all the addresses in the configured FVRF.

	Command or Action	Purpose
Step 7	**end** **Example:** `Device(config-ikev2-policy)# end`	Exits IKEv2 policy configuration mode and returns to privileged EXEC mode.
Step 8	**show crypto ikev2 policy** [*policy-name* | **default**] **Example:** `Device# show crypto ikev2 policy policy1`	(Optional) Displays the IKEv2 policy.

Configuring the IKEv2 Keyring

Perform this task to configure the IKEv2 key ring if the local or remote authentication method is a preshared key.

IKEv2 key ring keys must be configured in the peer configuration submode that defines a peer subblock. An IKEv2 key ring can have multiple peer subblocks. A peer subblock contains a single symmetric or asymmetric key pair for a peer or peer group identified by any combination of the hostname, identity, and IP address.

IKEv2 key rings are independent of IKEv1 key rings. The key differences are as follows:

- IKEv2 key rings support symmetric and asymmetric preshared keys.
- IKEv2 key rings do not support Rivest, Shamir, and Adleman (RSA) public keys.
- IKEv2 key rings are specified in the IKEv2 profile and are not looked up, unlike IKEv1, where keys are looked up on receipt of MM1 to negotiate the preshared key authentication method. The authentication method is not negotiated in IKEv2.
- IKEv2 key rings are not associated with VPN routing and forwarding (VRF) during configuration. The VRF of an IKEv2 key ring is the VRF of the IKEv2 profile that refers to the key ring.
- A single key ring can be specified in an IKEv2 profile, unlike an IKEv1 profile, which can specify multiple key rings.
- A single key ring can be specified in more than one IKEv2 profile, if the same keys are shared across peers matching different profiles.
- An IKEv2 key ring is structured as one or more peer subblocks.

On an IKEv2 initiator, the IKEv2 key ring key lookup is performed using the peer's hostname or the address, in that order. On an IKEv2 responder, the key lookup is performed using the peer's IKEv2 identity or the address, in that order.

Note You cannot configure the same identity in more than one peer.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto ikev2 keyring** *keyring-name*
4. **peer** *name*

5. **description** *line-of-description*
6. **hostname** *name*
7. **address** {*ipv4-address* [*mask*] | *ipv6-address prefix*}
8. **identity** {**address** {*ipv4-address* | *ipv6-address*} | **fqdn domain** *domain-name* | **email domain** *domain-name* | **key-id** *key-id*}
9. **pre-shared-key** {**local** | **remote**} [**0** | **6**] *line* **hex** *hexadecimal-string*
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto ikev2 keyring** *keyring-name* **Example:** `Device(config)# crypto ikev2 keyring kyr1`	Defines an IKEv2 key ring and enters IKEv2 key ring configuration mode.
Step 4	**peer** *name* **Example:** `Device(config-ikev2-keyring)# peer peer1`	Defines the peer or peer group and enters IKEv2 key ring peer configuration mode.
Step 5	**description** *line-of-description* **Example:** `Device(config-ikev2-keyring-peer)# description this is the first peer`	(Optional) Describes the peer or peer group.
Step 6	**hostname** *name* **Example:** `Device(config-ikev2-keyring-peer)# hostname host1`	Specifies the peer using a hostname.
Step 7	**address** {*ipv4-address* [*mask*]	*ipv6-address prefix*} **Example:** `Device(config-ikev2-keyring-peer)# address 10.0.0.1 255.255.255.0`	Specifies an IPv4 or IPv6 address or range for the peer. **Note** This IP address is the IKE endpoint address and is independent of the identity address.
Step 8	**identity** {**address** {*ipv4-address*	*ipv6-address*}	**fqdn domain** *domain-name*	**email domain** *domain-name*	**key-id** *key-id*} **Example:** `Device(config-ikev2-keyring-peer)# identity address 10.0.0.5`	Identifies the IKEv2 peer through the following identities: • E-mail • Fully qualified domain name (FQDN) • IPv4 or IPv6 address

	Command or Action	Purpose
		• Key ID **Note** The identity is available for key lookup on the IKEv2 responder only.
Step 9	**pre-shared-key** {**local** | **remote**} [**0** | **6**] *line* **hex** *hexadecimal-string* **Example:** `Device(config-ikev2-keyring-peer)# pre-shared-key local key1`	Specifies the preshared key for the peer.
Step 10	**end** **Example:** `Device(config-ikev2-keyring-peer)# end`	Exits IKEv2 key ring peer configuration mode and returns to privileged EXEC mode.

Configuring an IKEv2 Profile (Basic)

Perform this task to configure the mandatory commands for an IKEv2 profile.

An IKEv2 profile is a repository of nonnegotiable parameters of the IKE security association (SA) (such as local or remote identities and authentication methods) and services available to authenticated peers that match the profile. An IKEv2 profile must be configured and associated with either a crypto map or an IPsec profile on the IKEv2 initiator. Use the **set ikev2-profile** *profile-name* command to associate a profile with a crypto map or an IPsec profile. To disassociate the profile, use the **no** form of the command.

The following rules apply to match statements:

- An IKEv2 profile must contain a match identity or a match certificate statement; otherwise, the profile is considered incomplete and is not used. An IKEv2 profile can have more than one match identity or match certificate statements.
- An IKEv2 profile must have a single match Front Door VPN routing and forwarding (FVRF) statement.
- When a profile is selected, multiple match statements of the same type are logically ORed, and multiple match statements of different types are logically ANDed.
- The match identity and match certificate statements are considered to be the same type of statements and are ORed.
- Configuration of overlapping profiles is considered a misconfiguration. In the case of multiple profile matches, no profile is selected.

Use the **show crypto ikev2 profile** *profile-name* command to display the IKEv2 profile.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto ikev2 profile** *profile-name*
4. **description** *line-of-description*
5. **aaa accounting** {**psk** | **cert** | **eap**} *list-name*

6. **authentication** {**local** {**rsa-sig** | **pre-share** [**key** {**0** | **6**} *password*}] | **ecdsa-sig** | **eap** [**gtc** | **md5** | **ms-chapv2**] [**username** *username*] [**password** {**0** | **6**} *password*}]} | **remote** {**eap** [**query-identity** | **timeout** *seconds*] | **rsa-sig** | **pre-share** [**key** {**0** | **6**} *password*}] | **ecdsa-sig**}}
7. **dpd** *interval retry-interval* {**on-demand** | **periodic**}
8. **identity local** {**address** {*ipv4-address* | *ipv6-address*} | **dn** | **email** *email-string* | **fqdn** *fqdn-string* | **key-id** *opaque-string*}
9. **initial-contact force**
10. **ivrf** *name*
11. **keyring** {**local** *keyring-name* | **aaa** *list-name* [**name-mangler** *mangler-name* | **password** *password*] }
12. **lifetime** *seconds*
13. **match** {**address local** {*ipv4-address* | *ipv6-address* | **interface** *name*} | **certificate** *certificate-map* | **fvrf** {*fvrf-name* | **any**} | **identity remote address** {*ipv4-address* [*mask*] | *ipv6-address prefix*} | {**email** [*domain string*] | **fqdn** [*domain string*]} *string* | **key-id** *opaque-string*}
14. **nat keepalive** *seconds*
15. **pki trustpoint** *trustpoint-label* [**sign** | **verify**]
16. **redirect gateway auth**
17. **virtual-template** *number* **mode auto**
18. **shutdown**
19. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto ikev2 profile** *profile-name* **Example:** `Device(config)# crypto ikev2 profile profile1`	Defines an IKEv2 profile and enters IKEv2 profile configuration mode.
Step 4	**description** *line-of-description* **Example:** `Device(config-ikev2-profile)# description This is an IKEv2 profile`	(Optional) Describes the profile.
Step 5	**aaa accounting** {**psk**	**cert**	**eap**} *list-name* **Example:** `Device(config-ikev2-profile)# aaa accounting eap list1`	(Optional) Enables authentication, authorization, and accounting (AAA) accounting method lists for IPsec sessions.

	Command or Action	Purpose
		Note If the **psk**, **cert**, or **eap** keyword is not specified, the AAA accounting method list is used irrespective of the peer authentication method.
Step 6	**authentication** {**local** {**rsa-sig** | **pre-share** [**key** {**0** | **6**} *password*}] | **ecdsa-sig** | **eap** [**gtc** | **md5** | **ms-chapv2**] [**username** *username*] [**password** {**0** | **6**} *password*}]} | **remote** {**eap** [**query-identity** | **timeout** *seconds*] | **rsa-sig** | **pre-share** [**key** {**0** | **6**} *password*}] | **ecdsa-sig**}} **Example:** `Device(config-ikev2-profile)# authentication local ecdsa-sig`	Specifies the local or remote authentication method. • **rsa-sig**—Specifies RSA-sig as the authentication method. • **pre-share**—Specifies the preshared key as the authentication method. • **ecdsa-sig**—Specifies ECDSA-sig as the authentication method. • **eap**—Specifies EAP as the remote authentication method. • **query-identity**—Queries the EAP identity from the peer. • **timeout** *seconds*—Specifies the duration, in seconds, to wait for the next IKE_AUTH request after sending the first IKE_AUTH response. **Note** You can specify only one local authentication method but multiple remote authentication methods.
Step 7	**dpd** *interval retry-interval* {**on-demand** | **periodic**} **Example:** `Device(config-ikev2-profile)# dpd 30 6 on-demand`	(Optional) Configures Dead Peer Detection (DPD) globally for peers matching the profile. • Dead Peer Detection (DPD) is disabled by default. **Note** In the example in this step, the first DPD is sent after 30 seconds when there is no incoming ESP traffic. After waiting for 6 seconds (which is the specified retry interval), DPD retries are sent agressively 5 times in intervals of 6 seconds each. So, a total of 66 seconds (30 + 6 + 6 * 5 = 66) elapses before a crypto session is torn down because of DPD.
Step 8	**identity local** {**address** {*ipv4-address* | *ipv6-address*} | **dn** | **email** *email-string* | **fqdn** *fqdn-string* | **key-id** *opaque-string*} **Example:** `Device(config-ikev2-profile)# identity local email abc@example.com`	(Optional) Specifies the local IKEv2 identity type. **Note** If the local authentication method is a preshared key, the default local identity is the IP address. If the local authentication method is a Rivest, Shamir, and Adleman (RSA) signature, the default local identity is a Distinguished Name.

	Command or Action	Purpose
Step 9	**initial-contact force** **Example:** `Device(config-ikev2-profile)# initial-contact force`	Enforces initial contact processing if the initial contact notification is not received in the IKE_AUTH exchange.
Step 10	**ivrf** *name* **Example:** `Device(config-ikev2-profile)# ivrf vrf1`	(Optional) Specifies a user-defined VPN routing and forwarding (VRF) or global VRF if the IKEv2 profile is attached to a crypto map. • If the IKEv2 profile is used for tunnel protection, the Inside VRF (IVRF) for the tunnel interface should be configured on the tunnel interface. **Note** IVRF specifies the VRF for cleartext packets. The default value for IVRF is FVRF.
Step 11	**keyring** {**local** *keyring-name* | **aaa** *list-name* [**name-mangler** *mangler-name* | **password** *password*] } **Example:** `Device(config-ikev2-profile)# keyring aaa keyring1 name-mangler mangler1`	Specifies the local or AAA-based key ring that must be used with the local and remote preshared key authentication method. **Note** You can specify only one key ring. Local AAA is not supported for AAA-based preshared keys. **Note** Depending on your release, the **local** keyword and the **name-mangler** *mangler-name* keyword-argument pair should be used. **Note** When using AAA, the default password for a Radius access request is "cisco". You can use the **password** keyword within the **keyring** command to change the password.
Step 12	**lifetime** *seconds* **Example:** `Device(config-ikev2-profile)# lifetime 1000`	Specifies the lifetime, in seconds, for the IKEv2 SA.
Step 13	**match** {**address local** {*ipv4-address* | *ipv6-address* | **interface** *name*} | **certificate** *certificate-map* | **fvrf** {*fvrf-name* | **any**} | **identity remote address** {*ipv4-address* [*mask*] | *ipv6-address prefix*} | {**email** [*domain string*] | **fqdn** [*domain string*]} *string* | **key-id** *opaque-string*} **Example:** `Device(config-ikev2-profile)# match address local interface Ethernet 2/0`	Uses match statements to select an IKEv2 profile for a peer.
Step 14	**nat keepalive** *seconds* **Example:** `Device(config-ikev2-profile)# nat keepalive 500`	(Optional) Enables NAT keepalive and specifies the duration in seconds. • NAT is disabled by default.

	Command or Action	Purpose
Step 15	**pki trustpoint** *trustpoint-label* [**sign** | **verify**] **Example:** `Device(config-ikev2-profile)# pki trustpoint tsp1 sign`	Specifies Public Key Infrastructure (PKI) trustpoints for use with the RSA signature authentication method. **Note** If the **sign** or **verify** keyword is not specified, the trustpoint is used for signing and verification. **Note** In contrast to IKEv1, a trustpoint must be configured in an IKEv2 profile for certificate-based authentication to succeed. There is no fallback for globally configured trustpoints if this command is not present in the configuration. The trustpoint configuration applies to the IKEv2 initiator and responder.
Step 16	**redirect gateway auth** **Example:** `Device(config-ikev2-profile)# redirect gateway auth`	Enables the redirect mechanism on the gateway on SA authentication. **Note** The redirect mechanism is specific to the IKEv2 profiles.
Step 17	**virtual-template** *number* **mode auto** **Example:** `Device(config-ikev2-profile)# virtual-template 1 mode auto`	(Optional) Specifies the virtual template for cloning a virtual access interface (VAI). • **mode auto**—Enables the tunnel mode auto selection feature. **Note** For the IPsec Dynamic Virtual Tunnel Interface (DVTI), a virtual template must be specified in an IKEv2 profile, without which an IKEv2 session is not initiated.
Step 18	**shutdown** **Example:** `Device(config-ikev2-profile)# shutdown`	(Optional) Shuts down the IKEv2 profile.
Step 19	**end** **Example:** `Device(config-ikev2-profile)# end`	Exits IKEv2 profile configuration mode and returns to privileged EXEC mode.

Configuring IPsec Mixed Mode Support for SVTIs

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto IPsec profile** *profile-name*
4. **set transform-set** *transform-set-name* [*transform-set-name2...transform-set-name6*]
5. **exit**

6. **interface** *type number*
7. **ip address** *address mask*
8. Do one of the following:
   - **tunnel mode ipsec ipv4 v6-overlay**
   - **tunnel mode ipsec ipv6 v4-overlay**
9. **tunnel source** *interface-type interface-type*
10. **tunnel destination** *ip-address*
11. **tunnel protection IPsec profile** *profile-name*
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto IPsec profile** *profile-name* **Example:** `Device(config)# crypto IPsec profile PROF`	Defines the IPsec parameters that are to be used for IPsec encryption between two IPsec devices, and enters IPsec profile configuration mode.
Step 4	**set transform-set** *transform-set-name* [*transform-set-name2...transform-set-name6*] **Example:** `Device(ipsec-profile)# set transform-set tset`	Specifies which transform sets can be used with the crypto map entry.
Step 5	**exit** **Example:** `Device(ipsec-profile)# exit`	Exits IPsec profile configuration mode, and enters global configuration mode.
Step 6	**interface** *type number* **Example:** `Device(config)# interface tunnel 0`	Specifies the interface on which the tunnel will be configured and enters interface configuration mode.
Step 7	**ip address** *address mask* **Example:**	Specifies the IP address and mask.

	Command or Action	Purpose
	`Device(config-if)# ip address 10.1.1.1 255.255.255.0`	
Step 8	Do one of the following: • **tunnel mode ipsec ipv4 v6-overlay** • **tunnel mode ipsec ipv6 v4-overlay** **Example:** `Device(config-if)# tunnel mode ipsec ipv4 v6-overlay`	Defines the mode for the tunnel.
Step 9	**tunnel source** *interface-type interface-type* **Example:** `Device(config-if)# tunnel source loopback 0`	Specifies the tunnel source as a loopback interface.
Step 10	**tunnel destination** *ip-address* **Example:** `Device(config-if)# tunnel destination 172.16.1.1`	Identifies the IP address of the tunnel destination.
Step 11	**tunnel protection IPsec profile** *profile-name* **Example:** `Device(config-if)# tunnel protection IPsec profile PROF`	Associates a tunnel interface with an IPsec profile.
Step 12	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring IPsec Mixed Mode Support for Dynamic VTIs

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto ipsec profile** *profile-name*
4. **set mixed mode**
5. **set transform-set** *transform-set-name* [*transform-set-name2...transform-set-name6*]
6. **exit**
7. **interface virtual-template** *number* **type tunnel**
8. **tunnel mode ipsec ipv4**
9. **tunnel protection IPsec profile** *profile-name*
10. **exit**
11. **crypto isakamp profile** *profile-name*

12. **match identity address** *ip-address mask*
13. **virtual template** *template-number*
14. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto ipsec profile** *profile-name* **Example:** `Device(config)# crypto ipsec profile PROF`	Defines the IPsec parameters that are to be used for IPsec encryption between two IPsec devices and enters IPsec profile configuration mode.
Step 4	**set mixed mode** **Example:** `Device(config)# set mixed mode`	Defines the IPsec parameters that are to be used for IPsec encryption between two IPsec devices and enters IPsec profile configuration mode.
Step 5	**set transform-set** *transform-set-name* [*transform-set-name2...transform-set-name6*] **Example:** `Device(ipsec-profile)# set transform-set tset`	Specifies which transform sets can be used with the crypto map entry.
Step 6	**exit** **Example:** `Device(ipsec-profile)# exit`	Exits ipsec profile configuration mode and enters global configuration mode.
Step 7	**interface virtual-template** *number* **type tunnel** **Example:** `Device(config)# interface virtual-template 2 type tunnel`	Defines a virtual-template tunnel interface and enters interface configuration mode.
Step 8	**tunnel mode ipsec ipv4** **Example:** `Device(config-if)# tunnel mode ipsec ipv4`	Defines the mode for the tunnel.
Step 9	**tunnel protection IPsec profile** *profile-name* **Example:** `Device(config-if)# tunnel protection ipsec profile PROF`	Associates a tunnel interface with an IPsec profile.

	Command or Action	Purpose
Step 10	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 11	**crypto isakamp profile** *profile-name* **Example:** `Device(config)# crypto isakamp profile profile1`	Defines the ISAKMP profile to be used for the virtual template.
Step 12	**match identity address** *ip-address mask* **Example:** `Device(conf-isa-prof)# match identity address 10.1.1.0 255.255.255.0`	Matches an identity from the ISAKMP profile and enters isakmp-profile configuration mode.
Step 13	**virtual template** *template-number* **Example:** `Device(config)# virtual-template 1`	Specifies the virtual template attached to the ISAKMP profile.
Step 14	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.

Configuration Examples for IPsec Virtual Tunnel Interfaces

Example: Static Virtual Tunnel Interface with IPsec

The following example configuration uses a preshared key for authentication between peers. VPN traffic is forwarded to the IPsec VTI for encryption and then sent out the physical interface. The tunnel on subnet 10 checks packets for the IPsec policy and passes them to the Crypto Engine (CE) for IPsec encapsulation. The figure below illustrates the IPsec VTI configuration.

Figure 5: VTI with IPsec

Router Configuration

```
version 12.3
service timestamps debug datetime
service timestamps log datetime
hostname 7200-3
no aaa new-model
ip subnet-zero
ip cef
controller ISA 6/1
!
crypto isakmp policy 1
encr aes
authentication pre-share
group 14
```

```
crypto isakmp key Cisco12345 address 0.0.0.0 0.0.0.0
crypto ipsec transform-set T1 esp-aes esp-sha-hmac
crypto ipsec profile P1
set transform-set T1
!
interface Tunnel0
 ip address 10.0.51.203 255.255.255.0

 load-interval 30
 tunnel source 10.0.149.203
 tunnel destination 10.0.149.217
 tunnel mode IPsec ipv4
 tunnel protection IPsec profile P1
!

 ip address 10.0.149.203 255.255.255.0
 duplex full
!

 ip address 10.0.35.203 255.255.255.0
 duplex full
!
ip classless
ip route 10.0.36.0 255.255.255.0 Tunnel0
line con 0
line aux 0
line vty 0 4
end
```

Router Configuration

```
version 12.3
hostname c1750-17
no aaa new-model
ip subnet-zero
ip cef
crypto isakmp policy 1
encr aes
authentication pre-share
group 14
crypto isakmp key Cisco12345 address 0.0.0.0 0.0.0.0
crypto ipsec transform-set T1 esp-aes esp-sha-hmac
crypto ipsec profile P1
set transform-set T1
!
interface Tunnel0
 ip address 10.0.51.217 255.255.255.0

 tunnel source 10.0.149.217
 tunnel destination 10.0.149.203
 tunnel mode ipsec ipv4
 tunnel protection ipsec profile P1
!
interface
 ip address 10.0.149.217 255.255.255.0
 speed 100
 full-duplex
!
interface
 ip address 10.0.36.217 255.255.255.0
 load-interval 30
 full-duplex
!
```

```
ip classless
ip route 10.0.35.0 255.255.255.0 Tunnel0
line con 0
line aux 0
line vty 0 4
end
```

Example: Verifying the Results for the IPsec Static Virtual Tunnel Interface

This section provides information that you can use to confirm that your configuration is working properly. In this display, Tunnel 0 is "up," and the line protocol is "up." If the line protocol is "down," the session is not active.

Verifying the IPsec Static Virtual Tunnel Interface

```
Router# show interface tunnel 0

Tunnel0 is up, line protocol is up
Hardware is Tunnel
Internet address is 10.0.51.203/24
MTU 1514 bytes, BW 9 Kbit, DLY 500000 usec,
reliability 255/255, txload 103/255, rxload 110/255
Encapsulation TUNNEL, loopback not set
Keepalive not set
Tunnel source 10.0.149.203, destination 10.0.149.217
Tunnel protocol/transport ipsec/ip, key disabled, sequencing disabled
Tunnel TTL 255
Checksumming of packets disabled, fast tunneling enabled
Tunnel transmit bandwidth 8000 (kbps)
Tunnel receive bandwidth 8000 (kbps)
Tunnel protection via IPsec (profile "P1")
Last input never, output never, output hang never
Last clearing of "show interface" counters never
Input queue: 1/75/0/0 (size/max/drops/flushes); Total output drops: 0
Queueing strategy: fifo
Output queue: 0/0 (size/max)
30 second input rate 13000 bits/sec, 34 packets/sec
30 second output rate 36000 bits/sec, 34 packets/sec
191320 packets input, 30129126 bytes, 0 no buffer
Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
59968 packets output, 15369696 bytes, 0 underruns
0 output errors, 0 collisions, 0 interface resets
0 output buffer failures, 0 output buffers swapped out

Router# show crypto session

Crypto session current status
Interface: Tunnel0
Session status: UP-ACTIVE
Peer: 10.0.149.217 port 500
IKE SA: local 10.0.149.203/500 remote 10.0.149.217/500 Active
IPsec FLOW: permit ip 0.0.0.0/0.0.0.0 0.0.0.0/0.0.0.0
Active SAs: 4,
Router# show ip route

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
10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
C 10.0.35.0/24 is directly connected, Ethernet3/3
S 10.0.36.0/24 is directly connected, Tunnel0
C 10.0.51.0/24 is directly connected, Tunnel0
C 10.0.149.0/24 is directly connected, Ethernet3/0
```

Example: VRF-Aware Static Virtual Tunnel Interface

To add the VRF to the static VTI example, include the **ipvrf** and **ip vrf forwarding** commands to the configuration as shown in the following example.

Cisco 7206 Router Configuration

```
hostname cisco 7206
.
.
ip vrf sample-vti1
 rd 1:1
 route-target export 1:1
 route-target import 1:1
!
.
.
interface Tunnel0
 ip vrf forwarding sample-vti1
 ip address 10.0.51.217 255.255.255.0
 tunnel source 10.0.149.217
 tunnel destination 10.0.149.203
 tunnel mode ipsec ipv4
 tunnel protection ipsec profile P1
.
.
!
end
```

Example: Static Virtual Tunnel Interface with QoS

You can apply any QoS policy to the tunnel endpoint by including the **service-policy** statement under the tunnel interface. The following example shows how to police traffic out the tunnel interface.

Cisco 7206 Router Configuration

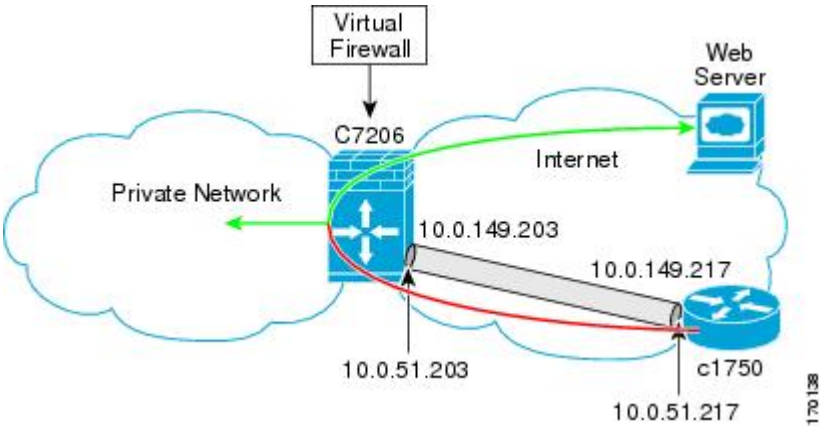
```
hostname cisco 7206
.
.
class-map match-all VTI
 match any
!
policy-map VTI
  class VTI
  police cir 2000000
    conform-action transmit
    exceed-action drop
!
```

```
.
.
interface Tunnel0
 ip address 10.0.51.217 255.255.255.0
 tunnel source 10.0.149.217
 tunnel destination 10.0.149.203
 tunnel mode ipsec ipv4
 tunnel protection ipsec profile P1
 service-policy output VTI
!
.
.
!
end
```

Example: Static Virtual Tunnel Interface with Virtual Firewall

Applying the virtual firewall to the SVTI tunnel allows traffic from the spoke to pass through the hub to reach the Internet. The figure below illustrates an SVTI with the spoke protected inherently by the corporate firewall.

Figure 6: Static VTI with Virtual Firewall



The basic SVTI configuration has been modified to include the virtual firewall definition:

Cisco 7206 Router Configuration

```
hostname cisco 7206
.
.
ip inspect max-incomplete high 1000000
ip inspect max-incomplete low 800000
ip inspect one-minute high 1000000
ip inspect one-minute low 800000
ip inspect tcp synwait-time 60
ip inspect tcp max-incomplete host 100000 block-time 2
ip inspect name IOSFW1 tcp timeout 300
ip inspect name IOSFW1 udp
!
.
.
interface GigabitEthernet0/1
 description Internet Connection
 ip address 172.18.143.246 255.255.255.0
```

```
 ip access-group 100 in
 ip nat outside
!
interface Tunnel0
 ip address 10.0.51.217 255.255.255.0
 ip nat inside
 ip inspect IOSFW1 in
 tunnel source 10.0.149.217
 tunnel destination 10.0.149.203
 tunnel mode ipsec ipv4
 tunnel protection ipsec profile P1
!
ip classless
ip route 0.0.0.0 0.0.0.0 172.18.143.1
!
ip nat translation timeout 120
ip nat translation finrst-timeout 2
ip nat translation max-entries 300000
ip nat pool test1 10.2.100.1 10.2.100.50 netmask 255.255.255.0
ip nat inside source list 110 pool test1 vrf test-vti1 overload
!
access-list 100 permit esp any any
access-list 100 permit udp any eq isakmp any
access-list 100 permit udp any eq non500-isakmp any
access-list 100 permit icmp any any
access-list 110 deny   esp any any
access-list 110 deny   udp any eq isakmp any
access-list 110 permit ip any any
access-list 110 deny   udp any eq non500-isakmp any
!
end
```

Example: Dynamic Virtual Tunnel Interface Easy VPN Server

The following example illustrates the use of the DVTI Easy VPN server, which serves as an IPsec remote access aggregator. The client can be a home user running a Cisco VPN client or a Cisco IOS router configured as an Easy VPN client.

Cisco 7206 Router Configuration

```
hostname cisco 7206
!
aaa new-model
aaa authentication login local_list local
aaa authorization network local_list local
aaa session-id common
!
ip subnet-zero
ip cef
!
username cisco password 0 cisco123
!
controller ISA 1/1
!
crypto isakmp policy 1
 encr aes
 authentication pre-share
 group 14
!
crypto isakmp client configuration group group1
 key cisco123
```

```
 pool group1pool
 save-password
!
crypto isakmp profile vpn1-ra
   match identity group group1
   client authentication list local_list
   isakmp authorization list local_list
   client configuration address respond
   virtual-template 1
!
crypto ipsec transform-set VTI-TS esp-aes esp-sha-hmac
!
crypto ipsec profile test-vti1
 set transform-set VTI-TS
!
interface GigabitEthernet0/1
 description Internet Connection
 ip address 172.18.143.246 255.255.255.0
!
interface GigabitEthernet0/2
 description Internal Network
 ip address 10.2.1.1 255.255.255.0
!
interface Virtual-Template1 type tunnel
 ip unnumbered GigabitEthernet0/1
 ip virtual-reassembly
 tunnel mode ipsec ipv4
 tunnel protection ipsec profile test-vti1
!
ip local pool group1pool 192.168.1.1 192.168.1.4
ip classless
ip route 0.0.0.0 0.0.0.0 172.18.143.1
!
end
```

Example: Verifying the Results for the Dynamic Virtual Tunnel Interface Easy VPN Server

The following examples show that a DVTI has been configured for an Easy VPN server.

```
Router# show running-config interface Virtual-Access2

Building configuration...
Current configuration : 250 bytes
!
interface Virtual-Access2
 ip unnumbered GigabitEthernet0/1
 ip virtual-reassembly
 tunnel source 172.18.143.246
 tunnel destination 172.18.143.208
 tunnel mode ipsec ipv4
 tunnel protection ipsec profile test-vti1
 no tunnel protection ipsec initiate
end
Router# show ip route

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 10.2.1.10 to network 0.0.0.0
```

```
     172.18.0.0/24 is subnetted, 1 subnets
C       172.18.143.0 is directly connected, GigabitEthernet0/1
     192.168.1.0/32 is subnetted, 1 subnets
S       192.168.1.1 [1/0] via 0.0.0.0, Virtual-Access2
     10.0.0.0/24 is subnetted, 1 subnets
C       10.2.1.0 is directly connected, GigabitEthernet0/2
S*   0.0.0.0/0 [1/0] via 172.18.143.1
```

Example: Dynamic Virtual Tunnel Interface Easy VPN Client

The following example shows how you can set up a router as the Easy VPN client. This example uses the same idea as the Easy VPN client that you can run from a PC to connect to a network. The configuration of the Easy VPN server will work for the software client or the Cisco IOS client.

```
hostname cisco 1841
!
no aaa new-model
!
ip cef
!
username cisco password 0 cisco123
!
crypto ipsec client ezvpn CLIENT
 connect manual
 group group1 key cisco123
 mode client
 peer 172.18.143.246
 virtual-interface 1
 username cisco password cisco123
 xauth userid mode local
!
interface Loopback0
 ip address 10.1.1.1 255.255.255.255
!
interface FastEthernet0/0
 description Internet Connection
 ip address 172.18.143.208 255.255.255.0
 crypto ipsec client ezvpn CLIENT
!
interface FastEthernet0/1
 ip address 10.1.1.252 255.255.255.0
 crypto ipsec client ezvpn CLIENT inside
!
interface Virtual-Template1 type tunnel
 ip unnumbered Loopback0
!
ip route 0.0.0.0 0.0.0.0 172.18.143.1 254
!
end
```

The client definition can be set up in many different ways. The mode specified with the **connect** command can be automatic or manual. If the connect mode is set to manual, the IPsec tunnel has to be initiated manually by a user.

Note the use of the **mode** command. The mode can be a client, network-extension, or network-extension-plus. This example indicates the client mode, which means that the client is given a private address from the server. The network-extension mode is different from the client mode in that the client specifies for the server its attached private subnet. Depending on the mode, the routing table on either end will be slightly different. The basic operation of the IPsec tunnel remains the same, regardless of the specified mode.

Example: Verifying the Results for the Dynamic Virtual Tunnel Interface Easy VPN Client

The following examples illustrate different ways to display the status of the DVTI.

```
Router# show running-config interface Virtual-Access2

Building configuration...
Current configuration : 148 bytes
!
interface Virtual-Access2
 ip unnumbered Loopback1
 tunnel source FastEthernet0/0
 tunnel destination 172.18.143.246
 tunnel mode ipsec ipv4
end

Router# show running-config interface Loopback1

Building configuration...
Current configuration : 65 bytes
!
interface Loopback1
 ip address 192.168.1.1 255.255.255.255
end
Router# show ip route
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 172.18.143.1 to network 0.0.0.0
     10.0.0.0/32 is subnetted, 1 subnets
C       10.1.1.1 is directly connected, Loopback0
     172.18.0.0/24 is subnetted, 1 subnets
C       172.18.143.0 is directly connected, FastEthernet0/0
     192.168.1.0/32 is subnetted, 1 subnets
C       192.168.1.1 is directly connected, Loopback1
S*   0.0.0.0/0 [1/0] via 0.0.0.0, Virtual-Access2

Router# show crypto ipsec client ezvpn

Easy VPN Remote Phase: 6
Tunnel name : CLIENT
Inside interface list: FastEthernet0/1
Outside interface: Virtual-Access2 (bound to FastEthernet0/0)
Current State: IPSEC_ACTIVE
Last Event: SOCKET_UP
Address: 192.168.1.1
Mask: 255.255.255.255
Save Password: Allowed
Current EzVPN Peer: 172.18.143.246
```

Example: VRF-Aware IPsec with a Dynamic VTI When VRF Is Configured Under a Virtual Template

The following example shows how to configure VRF-aware IPsec under a virtual template to take advantage of the DVTI:

```
hostname cisco 7206
!
ip vrf VRF-100-1
  rd 1:1
!
ip vrf VRF-100-2
  rd 1:1
!
!
!
crypto keyring cisco-100-1
  pre-shared-key address 10.1.1.1 key cisco-100-1
crypto keyring cisco-100-2
  pre-shared-key address 10.1.2.1 key cisco-100-2
crypto isakmp profile cisco-isakmp-profile-100-1
  keyring cisco-100-1
  match identity address 10.1.1.0 255.255.255.0
  virtual-template 101
crypto isakmp profile cisco-isakmp-profile-100-2
  keyring cisco-100-2
  match identity address 10.1.2.0 255.255.255.0
  virtual-template 102
!
!
crypto ipsec transform-set cisco esp-aes esp-sha-hmac
!
crypto ipsec profile cisco-ipsec-profile-101
  set security-policy limit 3
  set transform-set cisco
!
crypto ipsec profile cisco-ipsec-profile-102
  set security-policy limit 5
  set transform-set Cisco
!
interface Virtual-Template101 type tunnel
  ip vrf forwarding VRF-100-1
  ip unnumbered Ethernet 0/0
  tunnel mode ipsec ipv4
  tunnel protection ipsec profile cisco-ipsec-profile-101
!
interface Virtual-Template102 type tunnel
  ip vrf forwarding VRF-100-2
  ip unnumbered Ethernet 0/0
  tunnel mode ipsec ipv4
  tunnel protection ipsec profile cisco-ipsec-profile-102
!
```

Example: VRF-Aware IPsec with Dynamic VTI When VRF Is Configured Under a Virtual Template with the Gateway Option in an IPsec Profile

The following example shows how to configure VRF-aware IPsec to take advantage of the DVTI, when the VRF is configured under a virtual template with the gateway option in an IPsec profile.

```
hostname ASR 1000
!
ip vrf VRF-100-1
 rd 1:1
!
ip vrf VRF-100-2
```

```
 rd 1:1
!
!
!
crypto keyring cisco-100-1
 pre-shared-key address 10.1.1.1 key cisco-100-1
crypto keyring cisco-100-2
 pre-shared-key address 10.1.2.1 key cisco-100-2
crypto isakmp profile cisco-isakmp-profile-100-1
 keyring cisco-100-1
 match identity address 10.1.1.0 255.255.255.0
 virtual-template 101
crypto isakmp profile cisco-isakmp-profile-100-2
 keyring cisco-100-2
 match identity address 10.1.2.0 255.255.255.0
 virtual-template 102
!
!
crypto ipsec transform-set cisco esp-3des esp-sha-hmac
!
crypto ipsec profile cisco-ipsec-profile-101
 set security-policy limit 3
 set transform-set cisco
 set reverse-route gateway 172.16.0.1
!
crypto ipsec profile cisco-ipsec-profile-102
 set security-policy limit 5
 set transform-set cisco
 set reverse-route gateway 172.16.0.1
!
interface Virtual-Template101 type tunnel
 ip vrf forwarding VRF-100-1
 ip unnumbered Ethernet 0/0
 tunnel mode ipsec ipv4
 tunnel protection ipsec profile cisco-ipsec-profile-101
!
interface Virtual-Template102 type tunnel
 ip vrf forwarding VRF-100-2
 ip unnumbered Ethernet 0/0
 tunnel mode ipsec ipv4
 tunnel protection ipsec profile cisco-ipsec-profile-102
!
```

Example: VRF-Aware IPsec with a Dynamic VTI When VRF Is Configured Under an ISAKMP Profile

```
hostname cisco 7206
!
ip vrf VRF-100-1
  rd 1:1
!
ip vrf VRF-100-2
  rd 1:1
!
crypto keyring cisco-100-1
  pre-shared-key address 10.1.1.1 key cisco-100-1
crypto keyring cisco-100-2
  pre-shared-key address 10.1.2.1 key cisco-100-2
crypto isakmp profile cisco-isakmp-profile-100-1
```

```
  vrf VRF-100-1
  keyring cisco-100-1
  match identity address 10.1.1.0 255.255.255.0
  virtual-template 1
crypto isakmp profile cisco-isakmp-profile-100-2
  vrf VRF-100-2
  keyring cisco-100-2
  match identity address 10.1.2.0 255.255.255.0
  virtual-template 1
!
!
crypto ipsec transform-set cisco esp-aes esp-sha-hmac
crypto ipsec profile cisco-ipsec-profile
  set security-policy limit 3
  set transform-set cisco
!
!
!
interface Virtual-Template 1 type tunnel
  ip unnumbered ethernet 0/0
  tunnel mode ipsec ipv4
  tunnel protection ipsec profile cisco-ipsec-profile
!
!
```

Example: VRF-Aware IPsec with a Dynamic VTI When VRF Is Configured Under an ISAKMP Profile and a Gateway Option in an IPsec Profile

The following example shows how to configure VRF-aware IPsec to take advantage of the DVTI, when the VRF is configured under an ISAKMP profile and a gateway option in an IPsec profile:

```
hostname ASR 1000
!
ip vrf VRF-100-1
 rd 1:1
!
ip vrf VRF-100-2
 rd 1:1
!
crypto keyring cisco-100-1
 pre-shared-key address 10.1.1.1 key cisco-100-1
crypto keyring cisco-100-2
 pre-shared-key address 10.1.2.1 key cisco-100-2
crypto isakmp profile cisco-isakmp-profile-100-1
 vrf VRF-100-1
 keyring cisco-100-1
 match identity address 10.1.1.0 255.255.255.0
 virtual-template 1
crypto isakmp profile cisco-isakmp-profile-100-2
 vrf VRF-100-2
 keyring cisco-100-2
 match identity address 10.1.2.0 255.255.255.0
 virtual-template 1
!
!
crypto ipsec transform-set cisco esp-3des esp-sha-hmac
crypto ipsec profile cisco-ipsec-profile
 set security-policy limit 3
```

```
 set transform-set cisco
 set reverse-route gateway 172.16.0.1
!
!
!
interface Virtual-Template1 type tunnel
 ip unnumbered Ethernet 0/0
 tunnel mode ipsec ipv4
 tunnel protection ipsec profile cisco-ipsec-profile
!
!
```

Example: VRF-Aware IPsec with a Dynamic VTI When a VRF Is Configured Under Both a Virtual Template and an ISAKMP Profile

Note When separate VRFs are configured under an ISAKMP profile and a virtual template, the VRF configured under the virtual template takes precedence. This configuration is not recommended.

The following example shows how to configure VRF-aware IPsec to take advantage of the DVTI when the VRF is configured under both a virtual template and an ISAKMP profile:

```
hostname ASR 1000
.
.
.
ip vrf test-vti2
 rd 1:2
 route-target export 1:1
 route-target import 1:1
!
.
.
.
ip vrf test-vti1
 rd 1:1
 route-target export 1:1
 route-target import 1:1
!
.
.
.
crypto isakmp profile cisco-isakmp-profile
 vrf test-vti2
 keyring key
 match identity address 10.1.1.0 255.255.255.0
!
.
.
.
interface Virtual-Template1 type tunnel
 ip vrf forwarding test-vti1
 ip unnumbered Loopback 0
 ip virtual-reassembly
 tunnel mode ipsec ipv4
 tunnel protection ipsec profile test-vti1
```

```
!
.
.
.
end
```

Example: Configuring Multi-SA Support for Dynamic VTI Using IKEv2

The following examples show how to configure Multi-SA Support for Dynamic VTI using IKEv2:

```
!
!
aaa new-model
!
!
aaa authorization network grp-list local
!
aaa attribute list aaa-cisco-ikev2-profile-100-1
attribute type interface-config "ip vrf forwarding VRF-100-1"
attribute type interface-config "ip unnumbered Ethernet0/0"
!
aaa attribute list aaa-cisco-ikev2-profile-100-2
attribute type interface-config "ip vrf forwarding VRF-100-2"
attribute type interface-config "ip unnumbered Ethernet0/0"
!
aaa attribute list aaa-cisco-ikev2-profile-100-3
attribute type interface-config "ip vrf forwarding VRF-100-3"
attribute type interface-config "ip unnumbered Ethernet0/0"
!
!
!
!
!
aaa session-id common
!
ip vrf VRF-100-1
rd 101:1
 route-target export 101:1
 route-target import 101:1
!
ip vrf VRF-100-2
rd 102:2
route-target export 102:2
route-target import 102:2
!
ip vrf VRF-100-3
rd 103:3
route-target export 103:3
route-target import 103:3
!
!
!
crypto ikev2 authorization policy auth-policy-cisco-ikev2-profile-100-1
aaa attribute list aaa-cisco-ikev2-profile-100-1
ipsec flow-limit 3
!
crypto ikev2 authorization policy auth-policy-cisco-ikev2-profile-100-2
aaa attribute list aaa-cisco-ikev2-profile-100-2
ipsec flow-limit 3
!
crypto ikev2 authorization policy auth-policy-cisco-ikev2-profile-100-3
```

```
aaa attribute list aaa-cisco-ikev2-profile-100-3
ipsec flow-limit 3
!
crypto ikev2 proposal ikev2-proposal
 encryption aes
 integrity sha
 group 14
!
crypto ikev2 policy ikev2-policy
match fvrf any
proposal ikev2-proposal
!
crypto ikev2 keyring cisco-ikev2
peer cisco-100-1
 address 100.1.1.1
 pre-shared-key cisco-100-1
!
peer cisco-100-2
address 100.1.2.1
pre-shared-key cisco-100-2
!
peer cisco-100-3
address 100.1.3.1
pre-shared-key cisco-100-3
!
!
!
crypto ikev2 profile cisco-ikev2-profile-100-1
match fvrf any
match identity remote address 10.1.1.1 255.255.255.255
authentication local pre-share
authentication remote pre-share
keyring cisco-ikev2
aaa authorization group grp-list auth-policy-cisco-ikev2-profile-100-1
virtual-template 1
!
crypto ikev2 profile cisco-ikev2-profile-100-2
match fvrf any
match identity remote address 10.1.2.1 255.255.255.255
authentication local pre-share
authentication remote pre-share
keyring cisco-ikev2
aaa authorization group group-list auth-policy-cisco-ikev2-profile-100-2
virtual-template 1
!
crypto ikev2 profile cisco-ikev2-profile-100-3
match fvrf any
match identity remote address 10.1.3.1 255.255.255.255
authentication local pre-share
authentication remote pre-share
keyring cisco-ikev2
aaa authorization group group-list auth-policy-cisco-ikev2-profile-100-3
virtual-template 1
!
!
crypto ipsec transform-set cisco esp-aes esp-sha-hmac
!
crypto ipsec profile cisco-ipsec-profile
set transform-set cisco
set reverse-route distance 10
set reverse-route tag 321
interface Virtual-Template1 type tunnel
ip unnumbered Ethernet0/0
tunnel mode ipsec ipv4
```

```
tunnel protection ipsec profile cisco-ipsec-profile
!
```

Example: Dynamic Virtual Tunnel Interface with Virtual Firewall

The DVTI Easy VPN server can be configured behind a virtual firewall. Behind-the-firewall configuration allows users to enter the network, while the network firewall is protected from unauthorized access. The virtual firewall uses Context-Based Access Control (CBAC) and NAT applied to the Internet interface as well as to the virtual template.

```
hostname cisco 7206
.
.
ip inspect max-incomplete high 1000000
ip inspect max-incomplete low 800000
ip inspect one-minute high 1000000
ip inspect one-minute low 800000
ip inspect tcp synwait-time 60
ip inspect tcp max-incomplete host 100000 block-time 2
ip inspect name IOSFW1 tcp timeout 300
ip inspect name IOSFW1 udp
!
.
.
interface GigabitEthernet0/1
 description Internet Connection
 ip address 172.18.143.246 255.255.255.0
 ip access-group 100 in
 ip nat outside
!
interface GigabitEthernet0/2
 description Internal Network
 ip address 10.2.1.1 255.255.255.0
!
interface Virtual-Template1 type tunnel
 ip unnumbered Loopback0
 ip nat inside
 ip inspect IOSFW1 in
 tunnel mode ipsec ipv4
 tunnel protection ipsec profile test-vti1
!
ip classless
ip route 0.0.0.0 0.0.0.0 172.18.143.1
!
ip nat translation timeout 120
ip nat translation finrst-timeout 2
ip nat translation max-entries 300000
ip nat pool test1 10.2.100.1 10.2.100.50 netmask 255.255.255.0
ip nat inside source list 110 pool test1 vrf test-vti1 overload
!
access-list 100 permit esp any any
access-list 100 permit udp any eq isakmp any
access-list 100 permit udp any eq non500-isakmp any
access-list 100 permit icmp any any
access-list 110 deny   esp any any
access-list 110 deny   udp any eq isakmp any
access-list 110 permit ip any any
access-list 110 deny   udp any eq non500-isakmp any
!
end
```

Example: Dynamic Virtual Tunnel Interface with QoS

You can add QoS to the DVTI tunnel by applying the service policy to the virtual template. When the template is cloned to make the virtual access interface, the service policy will also be applied to the virtual access interface. The following example shows the basic DVTI configuration with QoS added.

```
hostname cisco 7206
.
.
class-map match-all VTI
 match any
!
policy-map VTI
  class VTI
  police cir 2000000
    conform-action transmit
    exceed-action drop
!
.
.
interface Virtual-Template1 type tunnel
 ip vrf forwarding test-vti1
 ip unnumbered Loopback0
 ip virtual-reassembly
 tunnel mode ipsec ipv4
 tunnel protection ipsec profile test-vti1
 service-policy output VTI
!
.
.
!
end
```

Example: Dynamic Virtual Tunnel Interface Using GRE with IPsec Protection

```
Router1(config)# crypto ipsec transform-set MyTransformSet esp-aes esp-sha-hmac
Router1(cfg-crypto-trans)# mode transport
Router1(cfg-crypto-trans)# exit
Router1# config terminal
Router1(config)# crypto ipsec profile MyProfile set transform-set MyTransformSet
Router1(config)# interface Tunnel1
Router1(config-if)# description to-3800
Router1(config-if)# ip address 172.29.0.137 255.255.255.252
Router1(config-if)# tunnel source Ethernet0/0
Router1(config-if)# tunnel destination 10.38.38.1
Router1(config-if)# tunnel protection ipsec profile MyProfile
```

The **show interface tunnel** command verifies the tunnel interface configuration.

Note The tunnel transport MTU accounts for IPsec encryption overhead with GRE when used with the above configuration.

```
router1# show interface tunnel 1
```

```
          Tunnel1 is up, line protocol is up
          Hardware is Tunnel
          Description: to-3800
          Internet address is 172.29.0.137/30
          MTU 17880 bytes, BW 100 Kbit/sec, DLY 50000 usec,
              reliability 255/255, txload 1/255, rxload 1/255
          Encapsulation TUNNEL, loopback not set
          Keepalive not set
          Tunnel source 10.39.39.1 (Ethernet0/0), destination 10.38.38.1
           Tunnel Subblocks:
              src-track:
                  Tunnel1 source tracking subblock associated with Ethernet0/0
                   Set of tunnels with source Ethernet0/0, 1 member (includes iterators),
on interface <OK>
          Tunnel protocol/transport GRE/IP
              Key disabled, sequencing disabled
              Checksumming of packets disabled
          Tunnel TTL 255, Fast tunneling enabled
          Path MTU Discovery, ager 10 mins, min MTU 92
          Tunnel transport MTU 1440 bytes
```

Additional References for IPsec Virtual Tunnel Interface

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
Security commands	• Cisco IOS Security Command Reference Commands A to C • Cisco IOS Security Command Reference Commands D to L • Cisco IOS Security Command Reference Commands M to R • Cisco IOS Security Command Reference Commands S to Z
IPsec configuration	*Configuring Security for VPNs with IPsec*
QoS configuration	*Cisco IOS Quality of Service Solutions Configuration Guide*
EasyVPN configuration	• *Cisco Easy VPN Remote* • *Easy VPN Server*
Recommended cryptographic algorithms	Next Generation Encryption

Standards and RFCs

Standard/RFC	Title
RFC 2401	*Security Architecture for the Internet Protocol*
RFC 2408	*Internet Security Association and Key Management Protocol*
RFC 2409	*The Internet Key Exchange (IKE)*

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 IPsec Virtual Tunnel Interfaces

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 IPsec Virtual Tunnel Interfaces

Feature Name	Releases	Feature Configuration Information
Dynamic IPsec VTIs	12.3(7)T 12.3(14)T	Dynamic VTIs enable efficient use of IP addresses and provide secure connectivity. Dynamic VTIs allow dynamically downloadable per-group and per-user policies to be configured on a RADIUS server. IPsec dynamic VTIs allow you to create highly secure connectivity for remote access VPNs. The dynamic VTI simplifies VRF-aware IPsec deployment. The following commands were introduced or modified: **crypto isakmp profile, interface virtual-template, show vtemplate, tunnel mode, virtual-template.**
FlexVPN Mixed Mode Support	15.4(2)T	The FlexVPN Mixed Mode feature provides support for carrying IPv4 traffic over IPsec IPv6 transport. This is the first phase towards providing dual stack support on the IPsec stack. This implementation does not support using a single IPsec security association (SA) pair for both IPv4 and IPv6 traffic. This feature is only supported for Remote Access VPN with IKEv2 and Dynamic VTI.
IKE Profile Based Tunnel Selection	15.3(3)M	The Profile Based Tunnel Selection feature uses the Internet Key Exchange (IKE) or Internet Key Exchange version 2 (IKEv2) profile to select a tunnel interface for an IPsec session thereby allowing tunnel interfaces to share the tunnel source IP address and IPsec transform set without sharing the IPsec security association databases (SADBs) among tunnel interfaces. The following commands were introduced or modified: **tunnel protection ipsec profile.**

Feature Name	Releases	Feature Configuration Information
Multi-SA for Dynamic VTIs	15.2(1)T	The DVTI can accept multiple IPsec selectors that are proposed by the initiator. The following commands were introduced or modified: **set security-policy limit, set reverse-route.**
Static IPsec VTIs	12.2(33)SRA 12.2(33)SXH 12.3(7)T 12.3(14)T	IPsec VTIs provide a routable interface type for terminating IPsec tunnels and an easy way to define protection between sites to form an overlay network. IPsec VTIs simplify configuration of IPsec for protection of remote links, support multicast, and simplify network management and load balancing.
Tunnel Mode Auto Selection	15.4(2)T	The Tunnel Mode Auto Selection feature eases the configuration and spares you about knowing the responder's details. This feature automatically applies the tunneling protocol (GRE or IPsec) and transport protocol (IPv4 or IPv6) on the virtual template as soon as the IKE profile creates the virtual access interface. The following command was introduced or modified: **virtual-template**
Mixed Mode for IPsec VTI	15.6(1)T	The Mixed Mode feature provides support where traffic tunneled is either IPv4 or IPv6 but not both. This implementation supports only Mixed Mode for VTI for both IKEv1 and IKEv2. The following command was introduced or modified: **tunnel mode, crypto ipsec profile** Mixed Mode for IPsec VTI

Feature Name	Releases	Feature Configuration Information
FlexVPN Mixed Mode v6 over v4 Transport		The FlexVPN Mixed Mode v6 over v4 Transport feature provides support for carrying IPv6 traffic over IPsec IPv4 transport. This implementation does not support using a single IPsec security association (SA) pair for both IPv4 and IPv6 traffic.