Security for VPNs with IPsec Configuration Guide, Cisco IOS Release 15M&T

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

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED “AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

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

CONTENTS

CHAPTER 4

CHAPTER 5

CHAPTER 1

Configuring Security for VPNs with IPsec

This module describes how to configure basic IPsec VPNs. IPsec is a framework of open standards developed by the IETF. It provides security for the transmission of sensitive information over unprotected networks such as the Internet. IPsec acts at the network layer, protecting and authenticating IP packets between participating IPsec devices ("peers"), such as Cisco routers.

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.

Prerequisites for Configuring Security for VPNs with IPsec

IKE Configuration

You must configure Internet Key Exchange (IKE) as described in the module *Configuring Internet Key Exchange for IPsec VPNs*.

Note If you decide not to use IKE, you must still disable it as described in the module *Configuring Internet Key Exchange for IPsec VPNs*.

Ensure Access Lists Are Compatible with IPsec

IKE uses UDP port 500. The IPsec encapsulating security payload (ESP) and authentication header (AH) protocols use protocol numbers 50 and 51, respectively. Ensure that your access lists are configured so that traffic from protocol 50, 51, and UDP port 500 are not blocked at interfaces used by IPsec. In some cases, you might need to add a statement to your access lists to explicitly permit this traffic.

Restrictions for Configuring Security for VPNs with IPsec

Cisco IPsec Policy Map MIB

The MIB OID objects are displayed only when an IPsec session is up.

Discontiguous Access Control Lists

Crypto maps using access control lists (ACLs) that have discontiguous masks are not supported.

IPv4 Packets with IP Options Set

The following platforms do not support encrypting IPv4 packets with IP options set:

Cisco ASR1001 and ASR1000 routers with ESP-5, ESP-10, ESP-20, and ESP-40.

Physical Interface and Crypto Map

A crypto map on a physical interface is not supported, if the physical interface is the source interface of a tunnel protection interface.

NAT Configuration

If you use Network Address Translation (NAT), you should configure static NAT so that IPsec works properly. In general, NAT should occur before the router performs IPsec encapsulation; in other words, IPsec should work with global addresses.

Unicast IP Datagram Application Only

IPsec can be applied to unicast IP datagrams only. Because the IPsec Working Group has not yet addressed the issue of group key distribution, IPsec does not currently work with multicasts or broadcast IP datagrams.

IPv6 BFD with ISM

IPv6 BFD with ISM is not supported on ISRG2 routers.

Unsupported Interface Types

- Crypto VPNs are not supported on the bridge domain interfaces (BDI).
- Crypto maps are not supported on tunnel interface and port-channel interface.

Information About Configuring Security for VPNs with IPsec

Supported Standards

Cisco implements the following standards with this feature:

- IPsec—IPsec is a framework of open standards that provides data confidentiality, data integrity, and data authentication between participating peers. IPsec provides these security services at the IP layer; IPsec uses IKE to handle negotiation of protocols and algorithms based on the local policy, and generate the encryption and authentication keys to be used by IPsec. IPsec can be used to protect one or more data flows between a pair of hosts, between a pair of security gateways, or between a security gateway and a host.

  Note The term IPsec is sometimes used to describe the entire protocol of IPsec data services and IKE security protocols, and is also sometimes used to describe only the data services.

- IKE (IKEv1 and IKEv2)—A hybrid protocol that implements Oakley and SKEME key exchanges inside the Internet Security Association and Key Management Protocol (ISAKMP) framework. While IKE is used with other protocols, its initial implementation is with the IPsec protocol. IKE provides authentication of IPsec peers, negotiates IPsec security associations, and establishes IPsec keys.

The component technologies implemented for IPsec include:

Note Cisco no longer recommends using DES, 3DES, MD5 (including HMAC variant), and Diffie-Hellman (DH) groups 1, 2 and 5; instead, you should use AES, SHA and DH Groups 14 or higher. For more information about the latest Cisco cryptographic recommendations, see the Next Generation Encryption (NGE) white paper.

- AES—Advanced Encryption Standard. A cryptographic algorithm that protects sensitive, unclassified information. AES is a privacy transform for IPsec and IKE and has been developed to replace DES. AES is designed to be more secure than DES. AES offers a larger key size, while ensuring that the only known approach to decrypt a message is for an intruder to try every possible key. AES has a variable key length—the algorithm can specify a 128-bit key (the default), a 192-bit key, or a 256-bit key.

- DES—Data Encryption Standard. An algorithm that is used to encrypt packet data. Cisco software implements the mandatory 56-bit DES-CBC with Explicit IV. Cipher Block Chaining (CBC) requires an initialization vector (IV) to start encryption. The IV is explicitly given in the IPsec packet. For backwards compatibility, Cisco IOS IPsec also implements the RFC 1829 version of ESP DES-CBC.

Cisco IOS also implements Triple DES (168-bit) encryption, depending on the software versions available for a specific platform. Cisco no longer recommends Triple DES (3DES).

Note Cisco IOS images with strong encryption (including, but not limited to 56-bit data encryption feature sets) are subject to United States government export controls, and have a limited distribution. Images to be installed outside the United States require an export license. Customer orders might be denied or subject to delay due to United States government regulations. Contact your sales representative or distributor for more information, or send an e-mail to export@cisco.com.

- SEAL—Software Encryption Algorithm. An alternative algorithm to software-based DES, 3DES, and AES. SEAL encryption uses a 160-bit encryption key and has a lower impact on the CPU when compared to other software-based algorithms.
- SHA-2 and SHA-1 family (HMAC variant)—Secure Hash Algorithm (SHA) 1 and 2. Both SHA-1 and SHA-2 are hash algorithms used to authenticate packet data and verify the integrity verification mechanisms for the IKE protocol. HMAC is a variant that provides an additional level of hashing. SHA-2 family adds the SHA-256 bit hash algorithm and SHA-384 bit hash algorithm. This functionality is part of the Suite-B requirements that comprises four user interface suites of cryptographic algorithms for use with IKE and IPSec that are described in RFC 4869. Each suite consists of an encryption algorithm, a digital signature algorithm, a key agreement algorithm, and a hash or message digest algorithm. See the Configuring Security for VPNs with IPsec feature module for more detailed information about Cisco IOS Suite-B support.
- Diffie-Hellman—A public-key cryptography protocol that allows two parties to establish a shared secret over an unsecure communications channel. Diffie-Hellman is used within IKE to establish session keys. It supports 768-bit (the default), 1024-bit, 1536-bit, 2048-bit, 3072-bit, and 4096-bit DH groups. It also supports a 2048-bit DH group with a 256-bit subgroup, and 256-bit and 384-bit elliptic curve DH (ECDH). Cisco recommends using 2048-bit or larger DH key exchange, or ECDH key exchange.
- MD5 (Hash-based Message Authentication Code (HMAC) variant)—Message digest algorithm 5 (MD5) is a hash algorithm. HMAC is a keyed hash variant used to authenticate data.

IPsec as implemented in Cisco software supports the following additional standards:

- AH—Authentication Header. A security protocol, which provides data authentication and optional anti-replay services. AH is embedded in the data to be protected (a full IP datagram).
- ESP—Encapsulating Security Payload. A security protocol, which provides data privacy services and optional data authentication, and anti-replay services. ESP encapsulates the data to be protected.

Supported Encapsulation

IPsec works with the following serial encapsulations: Frame Relay, High-Level Data-Links Control (HDLC), and PPP.

IPsec also works with Generic Routing Encapsulation (GRE) and IPinIP Layer 3, Layer 2 Forwarding (L2F), Layer 2 Tunneling Protocol (L2TP), Data Link Switching+ (DLSw+), and Source Route Bridging (SRB) tunneling protocols; however, multipoint tunnels are not supported. Other Layer 3 tunneling protocols may not be supported for use with IPsec.

Because the IPsec Working Group has not yet addressed the issue of group key distribution, IPsec currently cannot be used to protect group traffic (such as broadcast or multicast traffic).

IPsec Functionality Overview

IPsec provides the following network security services. (In general, the local security policy dictates the use of one or more of these services.)

- Data confidentiality—The IPsec sender can encrypt packets before transmitting them across a network.
- Data integrity—The IPsec receiver can authenticate packets sent by the IPsec sender to ensure that the data has not been altered during transmission.
- Data origin authentication—The IPsec receiver can authenticate the source of the sent IPsec packets. This service is dependent upon the data integrity service.
- Anti-replay—The IPsec receiver can detect and reject replayed packets.

IPsec provides secure *tunnels* between two peers, such as two routers. You define which packets are considered sensitive and should be sent through these secure tunnels, and you define the parameters that should be used to protect these sensitive packets by specifying the characteristics of these tunnels. When the IPsec peer recognizes a sensitive packet, the peer sets up the appropriate secure tunnel and sends the packet through the tunnel to the remote peer. (The use of the term *tunnel* in this chapter does not refer to using IPsec in tunnel mode.)

More accurately, these *tunnels* are sets of security associations (SAs) that are established between two IPsec peers. The SAs define the protocols and algorithms to be applied to sensitive packets and specify the keying material to be used by the two peers. SAs are unidirectional and are established per security protocol (AH or ESP).

Multiple IPsec tunnels can exist between two peers to secure different data streams, with each tunnel using a separate set of SAs. For example, some data streams only need to be authenticated, while other data streams must both be encrypted and authenticated.

IKEv1 Transform Sets

An Internet Key Exchange version 1 (IKEv1) transform set represents a certain combination of security protocols and algorithms. During the IPsec SA negotiation, the peers agree to use a particular transform set for protecting a particular data flow.

IKEv2 Transform Sets

An Internet Key Exchange version 2 (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, then the default proposal is used in the negotiation. The default proposal is a collection of commonly used algorithms which are as follows:

```
encryption aes-cbc-128 3des
integrity sha1 md5
group 5 2
```

Although the **crypto ikev2 proposal** command is similar to the **crypto isakmp policy priority** command, the IKEv2 proposal differs as follows:

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

Note To use IKEv2 proposals in negotiation, they must be attached to IKEv2 policies. If a proposal is not configured, then the default IKEv2 proposal is used with the default IKEv2 policy.

Transform Sets: A Combination of Security Protocols and Algorithms

About Transform Sets

Note Cisco no longer recommends using ah-md5-hmac, esp-md5-hmac, esp-des or esp-3des. Instead, you should use ah-sha-hmac, esp-sha-hmac or esp-aes. For more information about the latest Cisco cryptographic recommendations, see the Next Generation Encryption (NGE) white paper.

A transform set represents a certain combination of security protocols and algorithms. During the IPsec SA negotiation, the peers agree to use a particular transform set for protecting a particular data flow.

During IPsec security association negotiations with IKE, peers search for an identical transform set for both peers. When such a transform set is found, it is selected and applied to the protected traffic as part of both peers' IPsec SAs. (With manually established SAs, there is no negotiation with the peer, so both sides must specify the same transform set.)

The table below shows allowed transform combinations.

Table 1: Allowed Transform Combinations

Transform Type	Transform	Description
AH Transform (*Pick only one.*)	**ah-md5-hmac**	AH with the MD5 (Message Digest 5) (an HMAC variant) authentication algorithm. (No longer recommended).
	ah-sha-hmac	AH with the SHA (Secure Hash Algorithm) (an HMAC variant) authentication algorithm.

Transform Type	Transform	Description
ESP Encryption Transform (*Pick only one.*)	**esp-aes**	ESP with the 128-bit Advanced Encryption Standard (AES) encryption algorithm.
	esp-gcm **esp-gmac**	The **esp-gcm** and **esp-gmac** transforms are ESPs with either a 128-bit or a 256-bit encryption algorithm. The default for either of these transforms is 128 bits. Both **esp-gcm** and **esp-gmac** transforms cannot be configured together with any other ESP transform within the same crypto IPsec transform set using the **crypto ipsec transform-set** command. The esp-gcm and esp-gmac combinations are not supported on the Cisco ASR 1001 routers with the following ESPs: • ESP-5 • ESP-10 • ESP-20 • ESP-40
	esp-aes 192	ESP with the 192-bit AES encryption algorithm.
	esp-aes 256	ESP with the 256-bit AES encryption algorithm.
	esp-des	ESP with the 56-bit Data Encryption Standard (DES) encryption algorithm. (No longer recommended).
	esp-3des	ESP with the 168-bit DES encryption algorithm (3DES or Triple DES). (No longer recommended).
	esp-null	Null encryption algorithm.
	esp-seal	ESP with the 160-bit SEAL encryption algorithm.
ESP Authentication Transform (*Pick only one.*)	**esp-md5-hmac**	ESP with the MD5 (HMAC variant) authentication algorithm. (No longer recommended).
	esp-sha-hmac	ESP with the SHA (HMAC variant) authentication algorithm.
IP Compression Transform	**comp-lzs**	IP compression with the Lempel-Ziv-Stac (LZS) algorithm

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.

Cisco IOS Suite-B Support for IKE and IPsec Cryptographic Algorithms

Suite-B adds support for four user interface suites of cryptographic algorithms for use with IKE and IPSec that are described in RFC 4869. Each suite consists of an encryption algorithm, a digital signature algorithm, a key agreement algorithm, and a hash or message digest algorithm.

Suite-B is not supported on the following hardware platforms:

- Cisco ASR1001
- ESP-5
- ESP-10
- ESP-20
- ESP-40

Suite-B has the following cryptographic algorithms:

- Suite-B-GCM-128-Provides ESP integrity protection, confidentiality, and IPsec encryption algorithms that use the 128-bit AES using Galois and Counter Mode (AES-GCM) described in RFC 4106. This suite should be used when ESP integrity protection and encryption are both needed.
- Suite-B-GCM-256-Provides ESP integrity protection and confidentiality using 256-bit AES-GCM described in RFC 4106. This suite should be used when ESP integrity protection and encryption are both needed.
- Suite-B-GMAC-128-Provides ESP integrity protection using 128-bit AES- Galois Message Authentication Code (GMAC) described in RFC 4543, but does not provide confidentiality. This suite should be used only when there is no need for ESP encryption.
- Suite-B-GMAC-256-Provides ESP integrity protection using 256-bit AES-GMAC described in RFC 4543, but does not provide confidentiality. This suite should be used only when there is no need for ESP encryption.

IPSec encryption algorithms use AES-GCM when encryption is required and AES-GMAC for message integrity without encryption.

IKE negotiation uses AES Cipher Block Chaining (CBC) mode to provide encryption and Secure Hash Algorithm (SHA)-2 family containing the SHA-256 and SHA-384 hash algorithms, as defined in RFC 4634, to provide the hash functionality. Diffie-Hellman using Elliptic Curves (ECP), as defined in RFC 4753, is used for key exchange and the Elliptic Curve Digital Signature Algorithm (ECDSA), as defined in RFC 4754, to provide authentication.

Suite-B Requirements

Suite-B imposes the following software crypto engine requirements for IKE and IPsec:

- HMAC-SHA256 and HMAC-SHA384 are used as pseudorandom functions; the integrity check within the IKE protocol is used. Optionally, HMAC-SHA512 can be used.
- Elliptic curve groups 19 (256-bit ECP curve) and 20 (384-bit ECP curve) are used as the Diffie-Hellman group in IKE. Optionally, group 21 (521-bit ECP curve) can be used.
- The Elliptic Curve Digital Signature Algorithm (ECDSA) algorithm (256-bit and 384-bit curves) is used for the signature operation within X.509 certificates.

- GCM (16 byte ICV) and GMAC is used for ESP (128-bit and 256-bit keys). Optionally, 192-bit keys can be used.
- Public Key Infrastructure (PKI) support for validation of X.509 certificates using ECDSA signatures must be used.
- PKI support for generating certificate requests using ECDSA signatures and for importing the issued certificates into IOS must be used.
- IKEV2 support for allowing the ECDSA signature (ECDSA-sig) as authentication method must be used.

Where to Find Suite-B Configuration Information

Suite-B configuration support is described in the following documents:

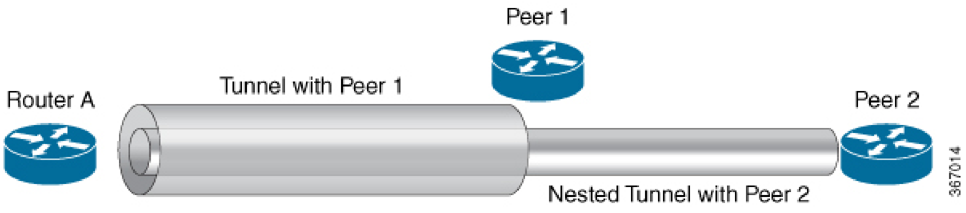
- For more information on SHA-2 family (HMAC variant) and Elliptic Curve (EC) key pair configuration, see the *Configuring Internet Key Exchange for IPsec VPNs* feature module.
- For more information on configuring a transform for an integrity algorithm type, see the "Configuring the IKEv2 Proposal" section in the *Configuring Internet Key Exchange Version 2 (IKEv2) and FlexVPN Site-to-Site* feature module.
- For more information on configuring the ECDSA-sig to be the authentication method for IKEv2, see the "Configuring IKEv2 Profile (Basic)" section in the *Configuring Internet Key Exchange Version 2 (IKEv2) and FlexVPN Site-to-Site* feature module.
- For more information on configuring elliptic curve Diffie-Hellman (ECDH) support for IPsec SA negotiation, see the *Configuring Internet Key Exchange for IPsec VPNs* and *Configuring Internet Key Exchange Version 2 and FlexVPN* feature modules.

For more information on the Suite-B support for certificate enrollment for a PKI, see the *Configuring Certificate Enrollment for a PKI* feature module.

Nested IPsec Tunnels

The following figure illustrates nested IPsec tunnels, where a tunnel is transported inside another tunnel.

Figure 1: Nested IPsec Tunnels



IPsec supports nested tunnels, where a tunnel is transported inside another tunnel. For example, in the above topology Router A has IPsec tunnels with Peer 1 and Peer 2, wherein the tunnel with Peer 2 is transported inside the tunnel with Peer 1. The following are sample configurations to establish nested IPsec tunnels on Router A, Peer 1 and Peer 2:

Configuration snippet on Router A

```
interface Tunnel1
ip address 172.16.0.1 255.255.255.0
```

```
tunnel source FastEthernet0/0
tunnel destination 10.0.0.2
tunnel protection ipsec profile prof1
!
interface Tunnel2
ip address 192.168.1.1 255.255.255.0
tunnel source Tunnel1
tunnel destination 10.0.1.2
tunnel mode ipsec ipv4
tunnel protection ipsec profile prof2
```

Configuration snippet on Peer 1

```
interface Tunnel1
ip address 172.16.0.2 255.255.255.0
tunnel source FastEthernet0/0.110
tunnel destination 10.0.0.1
tunnel protection ipsec profile prof1
```

Configuration snippet on Peer 2

```
interface Tunnel2
ip address 192.168.1.2 255.255.255.0
tunnel source GigabitEthernet0/1
tunnel destination 172.16.0.1
tunnel mode ipsec ipv4
tunnel protection ipsec profile prof2
```

The following nested tunnel combinations are supported with **tunnel protection** command:

- Virtual tunnel interface (VTI) in VTI
- VTI in generic routing encapsulation (GRE) or IPsec
- GRE or IPsec in GRE or IPsec
- GRE or IPsec in VTI

How to Configure IPsec VPNs

Creating Crypto Access Lists

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Do one of the following:
    - **access-list** *access-list-number* {**deny** | **permit**} *protocol source source-wildcard destination destination-wildcard* [**log**]
    - **ip access-list extended** *name*
4. Repeat Step 3 for each crypto access list you want to create.

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	Do one of the following:
	• **access-list** *access-list-number* {**deny** | **permit**} *protocol source source-wildcard destination destination-wildcard* [**log**]
	• **ip access-list extended** *name*
	Example:
	`Device(config)# access-list 100 permit ip 10.0.68.0 0.0.0.255 10.1.1.0 0.0.0.255`
	Example:
	`Device(config)# ip access-list extended vpn-tunnel`
		Specifies conditions to determine which IP packets are protected.
		• You specify conditions using an IP access list designated by either a number or a name. The **access-list** command designates a numbered extended access list; the **ip access-list extended** command designates a named access list.
		• Enable or disable crypto for traffic that matches these conditions.
		Tip Cisco recommends that you configure “mirror image” crypto access lists for use by IPsec and that you avoid using the **any** keyword.

Step 4	Repeat Step 3 for each crypto access list you want to create.	—

What to Do Next

After at least one crypto access list is created, a transform set needs to be defined as described in the “Configuring Transform Sets for IKEv1 and IKEv2 Proposals” section.

Next the crypto access lists need to be associated to particular interfaces when you configure and apply crypto map sets to the interfaces. (Follow the instructions in the “Creating Crypto Map Sets” and “Applying Crypto Map Sets to Interfaces” sections).

Configuring Transform Sets for IKEv1 and IKEv2 Proposals

Perform this task to define a transform set that is to be used by the IPsec peers during IPsec security association negotiations with IKEv1 and IKEv2 proposals.

Restrictions

If you are specifying SEAL encryption, note the following restrictions:

- Your router and the other peer must not have a hardware IPsec encryption.
- Your router and the other peer must support IPsec.
- Your router and the other peer must support the k9 subsystem.

- SEAL encryption is available only on Cisco equipment. Therefore, interoperability is not possible.
- Unlike IKEv1, the authentication method and SA lifetime are not negotiable in IKEv2, and because of this, these parameters cannot be configured under the IKEv2 proposal.

Configuring Transform Sets for IKEv1

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto ipsec transform-set** *transform-set-name transform1* [*transform2* [*transform3*]]
4. **mode** [**tunnel** | **transport**]
5. **end**
6. **clear crypto sa** [**peer** {*ip-address* | *peer-name*} | **sa map** *map-name* | **sa entry** *destination-address protocol spi*]
7. **show crypto ipsec transform-set** [**tag** *transform-set-name*]

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 transform-set** *transform-set-name transform1* [*transform2* [*transform3*]] **Example:** `Device(config)# crypto ipsec transform-set aesset esp-aes 256 esp-sha-hmac`	Defines a transform set and enters crypto transform configuration mode. • There are complex rules defining the entries that you can use for transform arguments. These rules are explained in the command description for the **crypto ipsec transform-set** command, and the table in "About Transform Sets" section provides a list of allowed transform combinations.
Step 4	**mode** [**tunnel** | **transport**] **Example:** `Device(cfg-crypto-tran)# mode transport`	(Optional) Changes the mode associated with the transform set. • The mode setting is applicable only to traffic whose source and destination addresses are the IPsec peer addresses; it is ignored for all other traffic. (All other traffic is in tunnel mode only.)
Step 5	**end** **Example:**	Exits crypto transform configuration mode and enters privileged EXEC mode.

	Command or Action	Purpose
	`Device(cfg-crypto-tran)# end`	
Step 6	**clear crypto sa** [**peer** {*ip-address* | *peer-name*} | **sa map** *map-name* | **sa entry** *destination-address protocol spi*] **Example:** `Device# clear crypto sa`	(Optional) Clears existing IPsec security associations so that any changes to a transform set takes effect on subsequently established security associations. Manually established SAs are reestablished immediately. • Using the **clear crypto sa** command without parameters clears out the full SA database, which clears out active security sessions. • You may also specify the **peer**, **map**, or **entry** keywords to clear out only a subset of the SA database.
Step 7	**show crypto ipsec transform-set** [**tag** *transform-set-name*] **Example:** `Device# show crypto ipsec transform-set`	(Optional) Displays the configured transform sets.

What to Do Next

After you have defined a transform set, you should create a crypto map as specified in the*Creating Crypto Map Sets* section.

Configuring Transform Sets for IKEv2

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto ikev2 proposal** *proposal-name*
4. **encryption** *transform1* [*transform2*] ...
5. **integrity** *transform1* [*transform2*] ...
6. **group** *transform1* [*transform2*] ...
7. **end**
8. **show crypto ikev2 proposal**

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.

	Command or Action	Purpose
Step 3	**crypto ikev2 proposal** *proposal-name* **Example:** `Device(config)# crypto ikev2 proposal proposal-1`	Specifies the name of the proposal and enters crypto IKEv2 proposal configuration mode. • The proposals are referred in IKEv2 policies through the proposal name.
Step 4	**encryption** *transform1* [*transform2*] ... **Example:** `Device(config-ikev2-proposal)# encryption aes-cbc-128`	(Optional) Specifies one or more transforms of the following encryption type: • AES-CBC 128—128-bit AES-CBC • AES-CBC 192—192-bit AES-CBC • AES-CBC 256—256-bit AES-CBC • 3DES—168-bit DES (No longer recommended. AES is the recommended encryption algorithm).
Step 5	**integrity** *transform1* [*transform2*] ... **Example:** `Device(config-ikev2-proposal)# integrity sha1`	(Optional) Specifies one or more transforms of the following integrity type: • 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 • the **sha1** keyword specifies the SHA-1 (HMAC variant) as the hash algorithm. • The **md5** keyword specifies MD5 (HMAC variant) as the hash algorithm. (No longer recommended. SHA-1 is the recommended replacement.)
Step 6	**group** *transform1* [*transform2*] ... **Example:** `Device(config-ikev2-proposal)# group 14`	(Optional) Specifies one or more transforms of the possible DH group type: • **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.

	Command or Action	Purpose
		• **24**—Specifies the 2048-bit DH/DSA group.

Step 7	**end**
	Example:
	`Device(config-ikev2-proposal)# end`
		Exits crypto IKEv2 proposal configuration mode and returns to privileged EXEC mode.

Step 8	**show crypto ikev2 proposal** **Example:** `Device# show crypto ikev2 proposal`	(Optional) Displays the parameters for each IKEv2 proposal.

Transform Sets for IKEv2 Examples

The following examples show how to configure a proposal:

IKEv2 Proposal with One Transform for Each Transform Type

```
Device(config)# crypto ikev2 proposal proposal-1
Device(config-ikev2-proposal)# encryption aes-cbc-128
Device(config-ikev2-proposal)# integrity sha1
Device(config-ikev2-proposal)# group 14
```

IKEv2 Proposal with Multiple Transforms for Each Transform Type

```
crypto ikev2 proposal proposal-2
encryption aes-cbc-128 aes-cbc-192
integrity sha1 sha256
group 14 15
```

For a list of transform combinations, see Configuring Security for VPNs with IPsec.

IKEv2 Proposals on the Initiator and Responder

The proposal of the initiator is as follows:

```
Device(config)# crypto ikev2 proposal proposal-1
Device(config-ikev2-proposal)# encryption aes-cbc-128 aes-cbc-196
Device(config-ikev2-proposal)# integrity sha1 sha256
Device(config-ikev2-proposal)# group 14 16
```

The proposal of the responder is as follows:

```
Device(config)# crypto ikev2 proposal proposal-2
Device(config-ikev2-proposal)# encryption aes-cbc-196 aes-cbc-128
Device(config-ikev2-proposal)# integrity sha256 sha1
Device(config-ikev2-proposal)# group 16 14
```

In the scenario, the initiator’s choice of algorithms is preferred and the selected algorithms are as follows:

```
encryption aes-cbc-128
integrity sha1
group 14
```

What to Do Next

After you have defined a transform set, you should create a crypto map as specified in the *Creating Crypto Map Sets* section.

Creating Crypto Map Sets

Creating Static Crypto Maps

When IKE is used to establish SAs, the IPsec peers can negotiate the settings they use for the new security associations. This means that you can specify lists (such as lists of acceptable transforms) within the crypto map entry.

Perform this task to create crypto map entries that use IKE to establish SAs. To create IPv6 crypto map entries, you must use the **ipv6** keyword with the **crypto map** command. For IPv4 crypto maps, use the **crypto map** command without the **ipv6** keyword.

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. **crypto map** [**ipv6**] *map-name seq-num* [**ipsec-isakmp**]
4. **match address** *access-list-id*
5. **set peer** {*hostname* | *ip-address*}
6. **crypto ipsec security-association dummy** {**pps** *rate* | **seconds** *seconds*}
7. **set transform-set** *transform-set-name1* [*transform-set-name2...transform-set-name6*]
8. **set security-association lifetime** {**seconds** *seconds* | **kilobytes** *kilobytes* | **kilobytes disable**}
9. **set security-association level per-host**
10. **set pfs** [**group1** | **group14** | **group15** | **group16** | **group19** | **group2** | **group20** | **group24** | **group5**]
11. **end**
12. **show crypto map** [**interface** *interface* | **tag** *map-name*]

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.

	Command or Action	Purpose
Step 3	**crypto map** [**ipv6**] *map-name seq-num* [**ipsec-isakmp**] **Example:** `Device(config)# crypto map static-map 1 ipsec-isakmp`	Creates or modifies a crypto map entry, and enters crypto map configuration mode. • For IPv4 crypto maps, use the command without the **ipv6** keyword.
Step 4	**match address** *access-list-id* **Example:** `Device(config-crypto-m)# match address vpn-tunnel`	Names an extended access list. • This access list determines the traffic that should be protected by IPsec and the traffic that should not be protected by IPsec security in the context of this crypto map entry.

Step 5	**set peer** {*hostname* | *ip-address*}
	Example:
	`Device(config-crypto-m)# set-peer 192.168.101.1`
		Specifies a remote IPsec peer—the peer to which IPsec protected traffic can be forwarded.
		• Repeat for multiple remote peers.

Step 6	**crypto ipsec security-association dummy** {**pps** *rate* | **seconds** *seconds*}
	Example:
	`Device(config-crypto-m)# set security-association dummy seconds 5`
		Enables generating dummy packets. These dummy packets are generated for all flows created in the crypto map.

Step 7	**set transform-set** *transform-set-name1* [*transform-set-name2...transform-set-name6*] **Example:** `Device(config-crypto-m)# set transform-set aesset`	Specifies the transform sets that are allowed for this crypto map entry. • List multiple transform sets in the order of priority (highest priority first).

Step 8	**set security-association lifetime** {**seconds** *seconds* | **kilobytes** *kilobytes* | **kilobytes disable**}
	Example:
	`Device (config-crypto-m)# set security-association lifetime seconds 2700`
		(Optional) Specifies a SA lifetime for the crypto map entry.
		• By default, the SAs of the crypto map are negotiated according to the global lifetimes, which can be disabled.

Step 9	**set security-association level per-host** **Example:** `Device(config-crypto-m)# set security-association level per-host`	(Optional) Specifies that separate SAs should be established for each source and destination host pair. • By default, a single IPsec “tunnel” can carry traffic for multiple source hosts and multiple destination hosts. **Caution** Use this command with care because multiple streams between given subnets can rapidly consume resources.
Step 10	**set pfs** [**group1** | **group14** | **group15** | **group16** | **group19** | **group2** | **group20** | **group24** | **group5**]	(Optional) Specifies that IPsec either should ask for password forward secrecy (PFS) when requesting new SAs

	Command or Action	Purpose
	Example: `Device(config-crypto-m)# set pfs group14`	for this crypto map entry or should demand PFS in requests received from the IPsec peer. • Group 1 specifies the 768-bit Diffie-Hellman (DH) identifier (default). (No longer recommended). • Group 2 specifies the 1024-bit DH identifier. (No longer recommended). • Group 5 specifies the 1536-bit DH identifier. (No longer recommended) • Group 14 specifies the 2048-bit DH identifier. • Group 15 specifies the 3072-bit DH identifier. • Group 16 specifies the 4096-bit DH identifier. • Group 19 specifies the 256-bit elliptic curve DH (ECDH) identifier. • Group 20 specifies the 384-bit ECDH identifier. • Group 24 specifies the 2048-bit DH/DSA identifier • By default, PFS is not requested. If no group is specified with this command, group 1 is used as the default.

Step 11	**end**
	Example:
	`Device(config-crypto-m)# end`
		Exits crypto map configuration mode and returns to privileged EXEC mode.

Step 12	**show crypto map** [**interface** *interface* | **tag** *map-name*] **Example:** `Device# show crypto map`	Displays your crypto map configuration.

Troubleshooting Tips

Certain configuration changes take effect only when negotiating subsequent SAs. If you want the new settings to take immediate effect, you must clear the existing SAs so that they are reestablished with the changed configuration. If the router is actively processing IPsec traffic, clear only the portion of the SA database that would be affected by the configuration changes (that is, clear only the SAs established by a given crypto map set). Clearing the full SA database should be reserved for large-scale changes, or when the router is processing very little other IPsec traffic.

To clear IPsec SAs, use the **clear crypto sa** command with appropriate parameters. (Omitting all parameters clears out the full SA database, which clears active security sessions.)

What to Do Next

After you have successfully created a static crypto map, you must apply the crypto map set to each interface through which IPsec traffic flows. To complete this task, see the "Applying Crypto Map Sets to Interfaces" section.

Creating Dynamic Crypto Maps

Dynamic crypto map entries specify crypto access lists that limit traffic for which IPsec SAs can be established. A dynamic crypto map entry that does not specify an access list is ignored during traffic filtering. A dynamic crypto map entry with an empty access list causes traffic to be dropped. If there is only one dynamic crypto map entry in the crypto map set, it must specify the acceptable transform sets.

Perform this task to create dynamic crypto map entries that use IKE to establish the SAs.

Note IPv6 addresses are not supported on dynamic crypto maps.

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. **crypto dynamic-map** *dynamic-map-name dynamic-seq-num*
4. **set transform-set** *transform-set-name1* [*transform-set-name2...transform-set-name6*]
5. **match address** *access-list-id*
6. **set peer** {*hostname* | *ip-address*}
7. **set security-association lifetime** {**seconds** *seconds* | **kilobytes** *kilobytes* | **kilobytes disable**}
8. **set pfs** [**group1** | **group14** | **group15** | **group16** | **group19** | **group2** | **group20** | **group24** | **group5**]
9. **exit**
10. **exit**
11. **show crypto dynamic-map** [**tag** *map-name*]
12. **configure terminal**
13. **crypto map** *map-name seq-num* **ipsec-isakmp dynamic** *dynamic-map-name* [**discover**]
14. **exit**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto dynamic-map** *dynamic-map-name dynamic-seq-num* **Example:** `Device(config)# crypto dynamic-map test-map 1`	Creates a dynamic crypto map entry and enters crypto map configuration mode.
Step 4	**set transform-set** *transform-set-name1* [*transform-set-name2...transform-set-name6*] **Example:** `Device(config-crypto-m)# set transform-set aesset`	Specifies the transform sets allowed for the crypto map entry. • List multiple transform sets in the order of priority (highest priority first). This is the only configuration statement required in dynamic crypto map entries.
Step 5	**match address** *access-list-id* **Example:** `Device(config-crypto-m)# match address 101`	(Optional) Specifies the list number or name of an extended access list. • This access list determines which traffic should be protected by IPsec and which traffic should not be protected by IPsec security in the context of this crypto map entry. **Note** Although access lists are optional for dynamic crypto maps, they are highly recommended. • If an access list is configured, the data flow identity proposed by the IPsec peer must fall within a **permit** statement for this crypto access list. • If an access list is not configured, the device accepts any data flow identity proposed by the IPsec peer. However, if an access list is configured but the specified access list does not exist or is empty, the device drops all packets. This is similar to static crypto maps, which require access lists to be specified. • Care must be taken if the **any** keyword is used in the access list, because the access list is used for packet filtering as well as for negotiation. • You must configure a match address; otherwise, the behavior is not secure, and you cannot enable TED because packets are sent in the clear (unencrypted.)
Step 6	**set peer** {*hostname* | *ip-address*} **Example:**	(Optional) Specifies a remote IPsec peer. Repeat this step for multiple remote peers.

	Command or Action	Purpose
	`Device(config-crypto-m)# set peer 192.168.101.1`	**Note** This is rarely configured in dynamic crypto map entries. Dynamic crypto map entries are often used for unknown remote peers.

Step 7	**set security-association lifetime** {**seconds** *seconds* | **kilobytes** *kilobytes* | **kilobytes disable**}
	Example:
	`Device(config-crypto-m)# set security-association lifetime seconds 7200`
		(Optional) Overrides (for a particular crypto map entry) the global lifetime value, which is used when negotiating IP Security SAs.
		Note To minimize the possibility of packet loss when rekeying in high bandwidth environments, you can disable the rekey request triggered by a volume lifetime expiry.

Step 8	**set pfs** [**group1** | **group14** | **group15** | **group16** | **group19** | **group2** | **group20** | **group24** | **group5**] **Example:** `Device(config-crypto-m)# set pfs group14`	(Optional) Specifies that IPsec should ask for PFS when requesting new security associations for this crypto map entry or should demand PFS in requests received from the IPsec peer. • Group 1 specifies the 768-bit Diffie-Hellman (DH) identifier (default). (No longer recommended). • Group 2 specifies the 1024-bit DH identifier. (No longer recommended). • Group 5 specifies the 1536-bit DH identifier. (No longer recommended) • Group 14 specifies the 2048-bit DH identifier. • Group 15 specifies the 3072-bit DH identifier. • Group 16 specifies the 4096-bit DH identifier. • Group 19 specifies the 256-bit elliptic curve DH (ECDH) identifier. • Group 20 specifies the 384-bit ECDH identifier. • Group 24 specifies the 2048-bit DH/DSA identifier • By default, PFS is not requested. If no group is specified with this command, **group1** is used as the default.
Step 9	**exit** **Example:** `Device(config-crypto-m)# exit`	Exits crypto map configuration mode and returns to global configuration mode.
Step 10	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode.

	Command or Action	Purpose
Step 11	**show crypto dynamic-map** [**tag** *map-name*] **Example:** `Device# show crypto dynamic-map`	(Optional) Displays information about dynamic crypto maps.
Step 12	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 13	**crypto map** *map-name seq-num* **ipsec-isakmp dynamic** *dynamic-map-name* [**discover**] **Example:** `Device(config)# crypto map static-map 1 ipsec-isakmp dynamic test-map discover`	(Optional) Adds a dynamic crypto map to a crypto map set. • You should set the crypto map entries referencing dynamic maps to the lowest priority entries in a crypto map set. **Note** You must enter the **discover** keyword to enable TED.
Step 14	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode.

Troubleshooting Tips

Certain configuration changes take effect only when negotiating subsequent SAs. If you want the new settings to take immediate effect, you must clear the existing SAs so that they are reestablished with the changed configuration. If the router is actively processing IPsec traffic, clear only the portion of the SA database that would be affected by the configuration changes (that is, clear only the SAs established by a given crypto map set). Clearing the entire SA database must be reserved for large-scale changes, or when the router is processing minimal IPsec traffic.

To clear IPsec SAs, use the **clear crypto sa** command with appropriate parameters. (Omitting all parameters clears the full SA database, which clears active security sessions.)

What to Do Next

After you have successfully created a crypto map set, you must apply the crypto map set to each interface through which IPsec traffic flows. To complete this task, see the "Applying Crypto Map Sets to Interfaces" section.

Creating Crypto Map Entries to Establish Manual SAs

Perform this task to create crypto map entries to establish manual SAs (that is, when IKE is not used to establish the SAs). To create IPv6 crypto maps entries, you must use the **ipv6** keyword with the **crypto map** command. For IPv4 crypto maps, use the **crypto map** command without the **ipv6** keyword.

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **crypto map** [**ipv6**] *map-name seq-num* [**ipsec-manual**]
4. **match address** *access-list-id*
5. **set peer** {*hostname* | *ip-address*}
6. **set transform-set** *transform-set-name*
7. Do one of the following:
    - **set session-key inbound ah** *spi hex-key-string*
    - **set session-key outbound ah** *spi hex-key-string*
8. Do one of the following:
    - **set session-key inbound esp** *spi* **cipher** *hex-key-string* [**authenticator** *hex-key-string*]
    - **set session-key outbound esp** *spi* **cipher** *hex-key-string* [**authenticator** *hex-key-string*]
9. **exit**
10. **exit**
11. **show crypto map** [**interface** *interface* | **tag** *map-name*]

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 map** [**ipv6**] *map-name seq-num* [**ipsec-manual**] **Example:** `Device(config)# crypto map mymap 10 ipsec-manual`	Specifies the crypto map entry to be created or modified and enters crypto map configuration mode. • For IPv4 crypto maps, use the **crypto map** command without the **ipv6** keyword.
Step 4	**match address** *access-list-id* **Example:** `Device(config-crypto-m)# match address 102`	Names an IPsec access list that determines which traffic should be protected by IPsec and which traffic should not be protected by IPsec in the context of this crypto map entry. • The access list can specify only one **permit** entry when IKE is not used.

Step 5	**set peer** {*hostname*	*ip-address*}
		Example:
		`Device(config-crypto-m)# set peer 10.0.0.5`
			Specifies the remote IPsec peer. This is the peer to which IPsec protected traffic should be forwarded.
			• Only one peer can be specified when IKE is not used.

Step 6	**set transform-set** *transform-set-name*	Specifies which transform set should be used.

	Command or Action	Purpose
	Example: `Device(config-crypto-m)# set transform-set someset`	• This must be the same transform set that is specified in the remote peer's corresponding crypto map entry. **Note** Only one transform set can be specified when IKE is not used.

Step 7	Do one of the following:
	• **set session-key inbound ah** *spi hex-key-string*
	• **set session-key outbound ah** *spi hex-key-string*
	Example:
	`Device(config-crypto-m)# set session-key inbound ah 256 98765432109876549876543210987654`
	Example:
	`Device(config-crypto-m)# set session-key outbound ah 256 fedcbafedcbafedcfedcbafedcbafedc`
		Sets the AH security parameter indexes (SPIs) and keys to apply to inbound and outbound protected traffic if the specified transform set includes the AH protocol.
		• This manually specifies the AH security association to be used with protected traffic.

Step 8	Do one of the following:
	• **set session-key inbound esp** *spi* **cipher** *hex-key-string* [**authenticator** *hex-key-string*]
	• **set session-key outbound esp** *spi* **cipher** *hex-key-string* [**authenticator** *hex-key-string*]
	Example:
	`Device(config-crypto-m)# set session-key inbound esp 256 cipher 0123456789012345`
	Example:
	`Device(config-crypto-m)# set session-key outbound esp 256 cipher abcdefabcdefabcd`
		Sets the Encapsulating Security Payload (ESP) Security Parameter Indexes (SPI) and keys to apply to inbound and outbound protected traffic if the specified transform set includes the ESP protocol.
		Or
		Specifies the cipher keys if the transform set includes an ESP cipher algorithm. Specifies the authenticator keys if the transform set includes an ESP authenticator algorithm.
		• This manually specifies the ESP security association to be used with protected traffic.

Step 9	**exit** **Example:** `Device(config-crypto-m)# exit`	Exits crypto map configuration mode and returns to global configuration mode.
Step 10	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode.
Step 11	**show crypto map** [**interface** *interface* | **tag** *map-name*] **Example:** `Device# show crypto map`	Displays your crypto map configuration.

Troubleshooting Tips

For manually established SAs, you must clear and reinitialize the SAs for the changes to take effect. To clear IPsec SAs, use the **clear crypto sa** command with appropriate parameters. (Omitting all parameters clears the entire SA database, which clears active security sessions.)

What to Do Next

After you have successfully created a crypto map set, you must apply the crypto map set to each interface through which IPsec traffic flows. To complete this task, see the "Applying Crypto Map Sets to Interfaces" section.

Applying Crypto Map Sets to Interfaces

You must apply a crypto map set to each interface through which IPsec traffic flows. Applying the crypto map set to an interface instructs the device to evaluate the interface's traffic against the crypto map set and to use the specified policy during connection or security association negotiation on behalf of traffic to be protected by the crypto map.

Perform this task to apply a crypto map to an interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type/number*
4. **crypto map** *map-name*
5. **exit**
6. **crypto map** *map-name* **local-address** *interface-id*
7. **exit**
8. **show crypto map** [**interface** *interface*]

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	**interface** *type/number* **Example:** `Device(config)# interface FastEthernet 0/0`	Configures an interface and enters interface configuration mode.
Step 4	**crypto map** *map-name* **Example:** `Device(config-if)# crypto map mymap`	Applies a crypto map set to an interface.
Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.

	Command or Action	Purpose
Step 6	**crypto map** *map-name* **local-address** *interface-id* **Example:** `Device(config)# crypto map mymap local-address loopback0`	(Optional) Permits redundant interfaces to share the same crypto map using the same local identity.
Step 7	**exit** **Example:** `Device(config)# exit`	(Optional) Exits global configuration mode.
Step 8	**show crypto map** [**interface** *interface*] **Example:** `Device# show crypto map`	(Optional) Displays your crypto map configuration

Configuration Examples for IPsec VPN

Example: Configuring AES-Based Static Crypto Map

This example shows how a static crypto map is configured and how an AES is defined as the encryption method:

```
crypto isakmp policy 10
 encryption aes 256
 authentication pre-share
 group 14
 lifetime 180
crypto isakmp key cisco123 address 10.0.110.1
!
!
crypto ipsec transform-set aesset esp-aes 256 esp-sha-hmac
 mode transport
!
crypto map aesmap 10 ipsec-isakmp
 set peer 10.0.110.1
 set transform-set aesset
 match address 120
!
!
!
voice call carrier capacity active
!
!
mta receive maximum-recipients 0
!
!
interface FastEthernet0/0
 ip address 10.0.110.2 255.255.255.0
 ip nat outside
 no ip route-cache
 no ip mroute-cache
 duplex auto
 speed auto
 crypto map aesmap
!
interface Serial0/0
 no ip address
 shutdown
!
interface FastEthernet0/1
 ip address 10.0.110.1 255.255.255.0
 ip nat inside
 no ip route-cache
 no ip mroute-cache
 duplex auto
 speed auto
!
ip nat inside source list 110 interface FastEthernet0/0 overload
ip classless
ip route 0.0.0.0 0.0.0.0 10.5.1.1
ip route 10.0.110.0 255.255.255.0 FastEthernet0/0
ip route 172.18.124.0 255.255.255.0 10.5.1.1
ip route 172.18.125.3 255.255.255.255 10.5.1.1
ip http server
!
!
access-list 110 deny   ip 10.0.110.0 0.0.0.255 10.0.110.0 0.0.0.255
access-list 110 permit ip 10.0.110.0 0.0.0.255 any
access-list 120 permit ip 10.0.110.0 0.0.0.255 10.0.110.0 0.0.0.255
!
```

Additional References for Configuring Security for VPNs with IPsec

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
IKE, IPsec, and PKI configuration 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
IKE configuration	*Configuring Internet Key Exchange for IPsec VPNs*
Suite-B SHA-2 family (HMAC variant) and Elliptic Curve (EC) key pair configuration	*Configuring Internet Key Exchange for IPsec VPNs*

Related Topic	Document Title
Suite-B Integrity algorithm type transform configuration	*Configuring Internet Key Exchange Version 2 (IKEv2)*
Suite-B Elliptic Curve Digital Signature Algorithm (ECDSA) signature (ECDSA-sig) authentication method configuration for IKEv2	*Configuring Internet Key Exchange Version 2 (IKEv2)*
Suite-B Elliptic curve Diffie-Hellman (ECDH) support for IPsec SA negotiation	• *Configuring Internet Key Exchange for IPsec VPNs* • *Configuring Internet Key Exchange Version 2 (IKEv2) and FlexVPN Site-to-Site*
Suite-B support for certificate enrollment for a PKI	*Configuring Certificate Enrollment for a PKI*
Set Up VPN between Cisco ASR 100 Series and Google Cloud Platform	Set Up VPN between Cisco ASR 100 Series and Google Cloud Platform
Recommended cryptographic algorithms	Next Generation Encryption

Standards

Standards	Title
None	—

MIBs

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

RFCs

RFCs	Title
RFC 2401	*Security Architecture for the Internet Protocol*
RFC 2402	*IP Authentication Header*
RFC 2403	*The Use of HMAC-MD5-96 within ESP and AH*
RFC 2404	*The Use of HMAC-SHA-1-96 within ESP and AH*
RFC 2405	*The ESP DES-CBC Cipher Algorithm With Explicit IV*

RFCs	Title
RFC 2406	*IP Encapsulating Security Payload (ESP)*
RFC 2407	*The Internet IP Security Domain of Interpretation for ISAKMP*
RFC 2408	*Internet Security Association and Key Management Protocol (ISAKMP)*

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 Configuring Security for VPNs with IPsec

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

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

Table 2: Feature Information for Configuring Security for IPsec VPNs

Feature Name	Software Releases	Feature Information
Advanced Encryption Standard	12.2(8)T	This feature adds support for the new encryption standard AES, which is a privacy transform for IPsec and IKE and has been developed to replace DES. The following commands were modified by this feature: **crypto ipsec transform-set**, **encryption (IKE policy)**, **show crypto ipsec transform-set**, **show crypto isakmp policy**.
DES/3DES/AES VPN Encryption Module (AIM-VPN/EPII, AIM-VPN/HPII, AIM-VPN/BPII Family)	12.3(7)T	This feature describes in which VPN encryption hardware AIM and NM are supported, in certain Cisco IOS software releases.

Feature Name	Software Releases	Feature Information
IKEv2 Proposal Support	15.1(1)T	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, then the default proposal is used in negotiation. The following commands were modified by this feature: **crypto ikev2 proposal, encryption (ikev2 proposal), group (ikev2 proposal), integrity (ikev2 proposal), show crypto ikev2 proposal.**
IPv6 Support for IPsec and IKEv2	15.1(4)M 15.1(1)SY	This feature allows IPv6 addresses to be added to IPsec and IKEv2 protocols. The following commands were introduced or modified: **crypto map (global IPsec)**, **crypto map (isakmp)**, **crypto map (Xauth)**, **ipv6 crypto map**.
Option to Disable Volume-based IPsec Lifetime Rekey	15.0(1)M	This feature allows customers to disable the IPsec security association rekey when processing large amounts of data. The following commands were modified by this feature: **crypto ipsec security association lifetime**, **set security-association lifetime**.
SEAL Encryption	12.3(7)T	This feature adds support for SEAL encryption in IPsec. The following command was modified by this feature: **crypto ipsec transform-set**.
Software IPPCP (LZS) with Hardware Encryption	12.2(13)T	This feature allows customers to use LZS software compression with IPsec when a VPN module is in Cisco 2600 and Cisco 3600 series routers.
Suite-B Support in IOS SW Crypto	15.1(2)T	Suite-B adds support for four user interface suites of cryptographic algorithms for use with IKE and IPSec that are described in RFC 4869. Each suite consists of an encryption algorithm, a digital signature algorithm, a key agreement algorithm, and a hash or message digest algorithm. The following command was modified by this feature: **crypto ipsec transform-set**.

Glossary

anti-replay—Security service where the receiver can reject old or duplicate packets to protect itself against replay attacks. IPsec provides this optional service by use of a sequence number combined with the use of data authentication. Cisco IOS XE IPsec provides this service whenever it provides the data authentication service, except for manually established SAs (that is, SAs established by configuration and not by IKE).

data authentication—Verification of the integrity and origin of the data. Data authentication can refer either to integrity alone or to both of these concepts (although data origin authentication is dependent upon data integrity).

data confidentiality—Security service in which the protected data cannot be observed.

data flow—Grouping of traffic, identified by a combination of source address or mask, destination address or mask, IP next protocol field, and source and destination ports, where the protocol and port fields can have the values of **any**. IPsec protection is applied to data flows.

IKE—Internet Key Exchange. IKE establishes a shared security policy and authenticates keys for services (such as IPSec) that require keys. Before any IPSec traffic can be passed, each router/firewall/host must verify the identity of its peer. This can be done by manually entering preshared keys into both hosts or by a CA service.

IPsec—IP Security. A framework of open standards that provides data confidentiality, data integrity, and data authentication between participating peers. IPSec provides these security services at the IP layer. IPSec uses IKE to handle the negotiation of protocols and algorithms based on local policy and to generate the encryption and authentication keys to be used by IPSec. IPSec can protect one or more data flows between a pair of hosts, between a pair of security gateways, or between a security gateway and a host.

peer—In the context of this module, a “peer” is a router or other device that participates in IPsec.

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

SA—security association. Description of how two or more entities use security services in the context of a particular security protocol (AH or ESP) to communicate securely on behalf of a particular data flow. The transform and the shared secret keys are used for protecting the traffic.

SPI—security parameter index. A number which, together with a destination IP address and security protocol, uniquely identifies a particular security association. Without IKE, the SPI is manually specified for each security association.

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

tunnel—In the context of this module, “tunnel” is a secure communication path between two peers, such as two routers. It does not refer to using IPsec in tunnel mode.

CHAPTER 2

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).

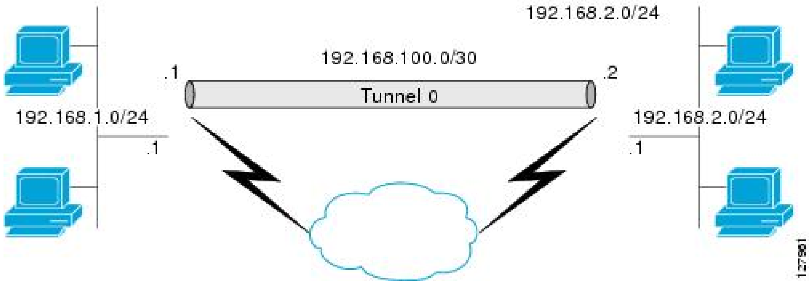
Static Virtual Tunnel Interfaces

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 2: 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.

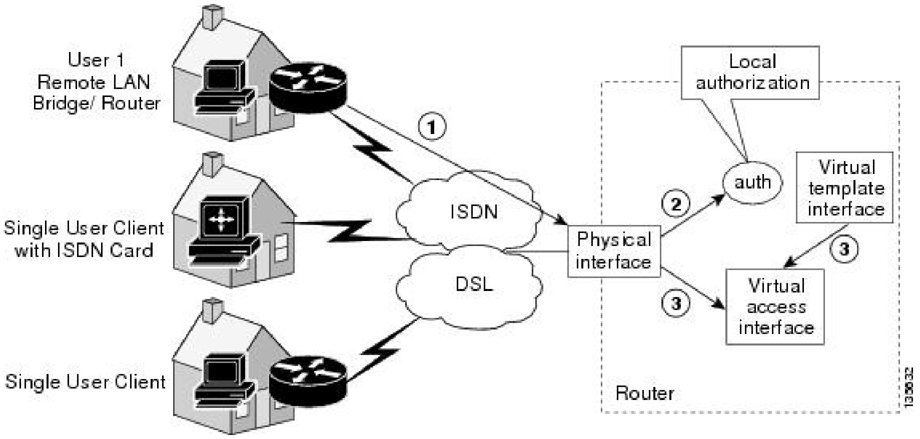
DVTIs provide efficiency in the use of IP addresses and provide secure connectivity. DVTIs allow dynamically downloadable per-group and per-user policies to be configured on a RADIUS server. The per-group or per-user definition can be created using an extended authentication (Xauth) User or Unity group, 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 3: Dynamic IPsec VTI



The authentication shown in the figure above follows this path:

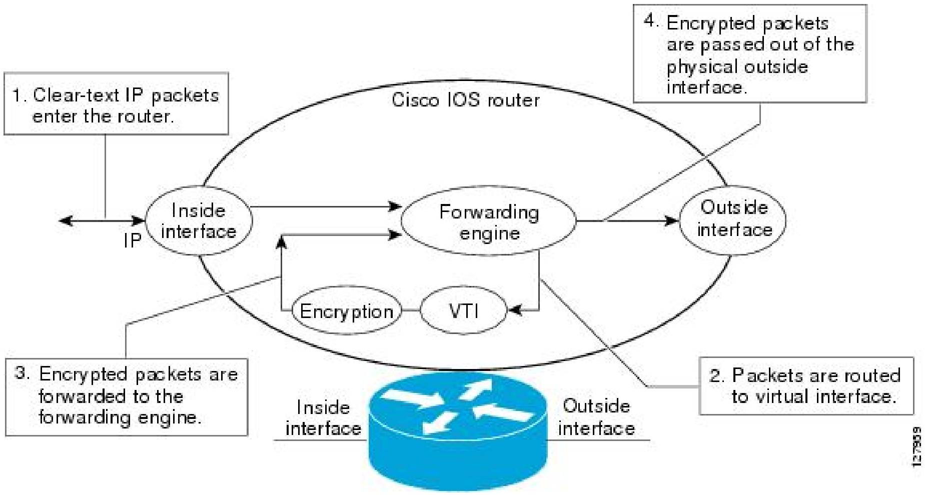
1. User 1 calls the router.
2. Router 1 authenticates User 1.
3. IPsec clones the virtual access interface from the virtual template interface.

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.

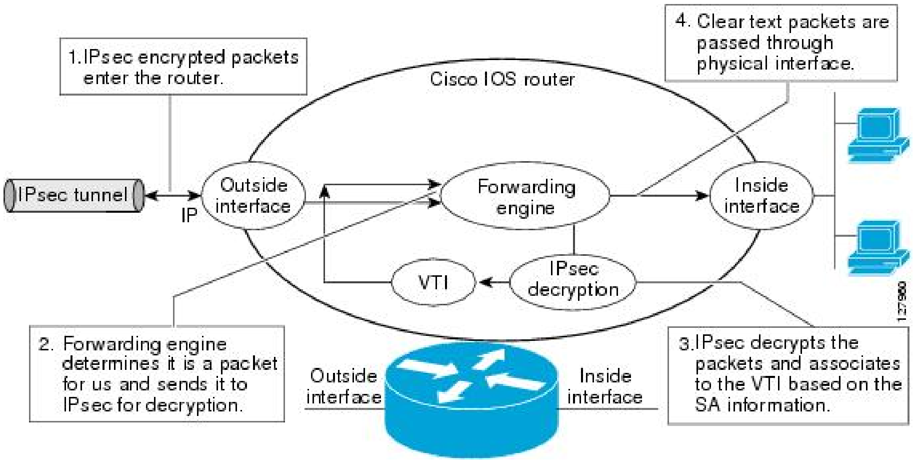
Figure 4: Packet Flow into the IPsec Tunnel


1. Clear-text IP packets enter the router.
Cisco IOS router
4. Encrypted packets are passed out of the physical outside interface.
IP
Inside interface
Forwarding engine
Outside interface
Encryption
VTI
3. Encrypted packets are forwarded to the forwarding engine.
Inside interface
Outside interface
2. Packets are routed to virtual interface.
127959

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 5: Packet Flow out of the IPsec Tunnel


1.IPsec encrypted packets enter the router.
Cisco IOS router
4. Clear text packets are passed through physical interface.
IPsec tunnel
IP
Outside interface
Forwarding engine
Inside interface
VTI
IPsec decryption
2. Forwarding engine determines it is a packet for us and sends it to IPsec for decryption.
Outside interface
Inside interface
3.IPsec decrypts the packets and associates to the VTI based on the SA information.
127960

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 40.

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 isakmp profile** *profile-name* **Example:** `Device(config)# crypto isakmp 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 6: 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

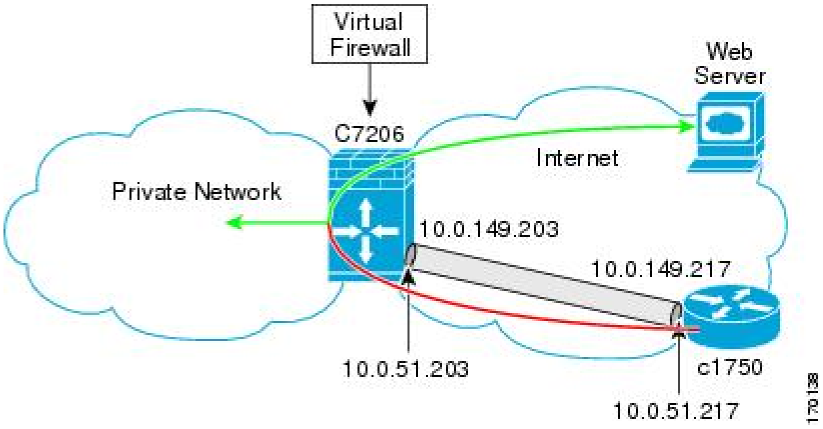
```
hostname cisco 7206
.
.
class-map match-all VTI
 match any
!
policy-map VTI
  class VTI
  police cir 2000000
    conform-action transmit
    exceed-action drop
!
```

```
.
.
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 7: 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 3: 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.

CHAPTER 3

L2TP IPsec Support for NAT and PAT Windows Clients

The L2TP IPsec Support for NAT and PAT Windows Clients feature allows mulitple Windows client to connect to an IPsec-enabled Cisco IOS Layer 2 Tunneling Protocol (L2TP) Network Server (LNS) through a network address translation (NAT) or port address translation (PAT) server.

When a Windows client connects to an IPsec-enabled Cisco IOS LNS router through a NAT or PAT server and another Windows client connects to the same Cisco IOS LNS router, the first client's connection is terminated. The L2TP IPsec Support for NAT and PAT Windows Clients feature ensures that Windows client connections in this environment are established and maintained until the connection is closed.

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.

Prerequisites for L2TP IPsec Support for NAT and PAT Windows Clients

- Windows clients environment, IPsec-enabled Cisco IOS LNS routers and a NAT or PAT server between the Windows clients and the Cisco IOS LNS router.
- You must understand the following concepts and configuration requirements:
    - Cisco IOS LNS routers
    - IPsec
    - L2TP
    - NAT and PAT
    - Windows 2000

Restrictions for L2TP IPsec Support for NAT and PAT Windows Clients

- The L2TP IPsec Support for NAT and PAT Windows Clients feature is tested only with Windows 2000 L2TP/IPsec clients running hotfix 818043.
- PAT is not the default behavior and is incompatible with IPsec because PAT changes the LNS header port information.
- L2TP requires Windows clients to have Microsoft DUN configured. L2TP is only supported by Windows 2000 MS-DUN (Windows 95, Windows 98, or Windows NT do not support L2TP).
- Windows clients cannot connect to an Cisco IOS L2TP over IPsec server if a NAT server is used to translate the messages from the router. To enable the connection, connect the router parallelly to the NAT server so that Network Address Translation Traversal (NAT-T) is not required or use an alternate protocol such as Point-to-Point Tunnelling Protocol (PPTP), IPsec, or SSL.

Information About L2TP IPsec Support for NAT and PAT Windows Clients

How L2TP IPsec Support for NAT and PAT Windows Clients Works

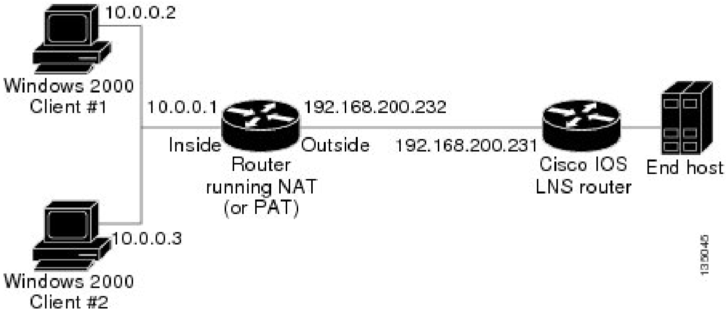
When a Windows client connects to an IPsec-enabled Cisco IOS LNS router through a NAT or PAT server and another Windows client connects to the same Cisco IOS LNS router, the first client's connection is terminated.

Note If IPsec is not enabled or there is no NAT or PAT server, multiple Windows clients can connect to the Cisco IOS LNS router.

L2TP IPsec Support for NAT and PAT Windows Clients Feature not Enabled

The figure below shows two Windows 2000 clients that are trying to connect to an end host through a router running NAT or PAT and IPsec-enabled Cisco IOS LNS router.

Figure 8: Multiple Windows 2000 Clients, NAT Router, and Cisco IOS LNS Router



The Windows 2000 Client #1 establishes an IPsec-protected L2TP tunnel to the Cisco IOS LNS router. The Windows 2000 Client #1 and the Cisco IOS LNS router recognize that there is a NAT router located between them and the NAT router is enabled with IPsec and NAT-Traversal (NAT-T). The Windows 2000 Client #1 attempts to establish an IPsec security association (SA) and requests a transport mode (which it does by default) with proxies from 10.0.0.2, its local address, to 192.168.200.231, the Cisco IOS LNS router's address.

In transport mode, NAT, running on the router, translates all outgoing connections (including 10.0.0.2) to its outside IP address (192.168.200.232), at which the address the traffic arrives. However, NAT cannot modify the L2TP port designation (1701), which is protected by the IPsec encrypted area. So, the local address now is 192.168.200.231, the remote address the 192.168.200.232 and the remote port is 1701. The traffic that matches the tunnel 192.168.200.231, port 1701 is sent to the Windows 2000 Client #1.

Windows 2000 Client #2 establishes an IPsec-protected L2TP tunnel to the Cisco IOS LNS router and NAT translates outgoing connections to its outside IP address (192.168.200.232) again, NAT cannot modify the L2TP port designation (1701) similar to Windows Client #1. The traffic that matches tunnel 192.168.200.231, port 1701 is now sent to Windows 2000 Client #2. which ends Windows Client #1's connection with the Cisco IOS LNS router since it is no longer receiving traffic.

L2TP IPsec Support for NAT and PAT Windows Clients Feature Enabled

When the L2TP IPsec Support for NAT and PAT Windows Clients feature is enabled, IPsec can translate the L2TP ports after decryption. This feature allows IPsec to map traffic from different hosts to different source ports. L2TP can now distinguish between traffic destined for multiple Windows 2000 clients.

When an security association (SA) is created, a translated port is assigned to the SA. This port is client-specific. The same port is used for any new SA created by that client. When an encrypted request is received and decrypted, the source port is translated from the standard value 1701 to a client specific value. The request with the translated port is then forwarded to L2TP.

As shown in the above figure, with port address translation enabled, the Windows 2000 Client #1 is assigned to the translated port number 1024, and Windows 2000 Client #2 is assigned to the translated port number 1025.

When L2TP sends the reply packet, it uses the translated port number and creates a packet to that destination port. IPsec uses the destination port number to select the SA with which to encrypt the packet. Before encrypting the packet, IPsec translates the destination port back to the standard port number 1701, which the Windows 2000 client expects. IPsec encrypts the packet either with the SA to Windows 2000 Client #1 if the destination port is 1024 or with the SA to Windows 2000 Client #2 if the destination port is 1025. The traffic is now sent to the appropriate client, and multiple Windows clients can be connected to a Cisco IOS LNS router through a NAT server at the same time.

The connection is maintained until one of the following actions occurs:

- The IPsec connection is closed.
- The NAT or PAT router ends the session.
- The Cisco IOS LNS router closes the session.
- A Windows client closes the session.

How to Enable L2TP IPsec Support for NAT and PAT Windows Clients

Enabling L2TP IPsec Support

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Do one of the following:
    - **crypto map** *map-name seq-num* [**ipsec-isakmp**]
    - **crypto dynamic-map** *dynamic-map-name dynamic-seq-num*
4. **set nat demux**
5. **end**
6. Do one of the following:
    - **show crypto map** [**interface** *interface* | **tag** *map-name*]
    - **show crypto dynamic-map** [**tag** *map-name*]
7. **show crypto ipsec sa**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

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

Step 3	Do one of the following:
	• **crypto map** *map-name seq-num* [**ipsec-isakmp**]
	• **crypto dynamic-map** *dynamic-map-name dynamic-seq-num*
	Example:
	`Router(config)# crypto map static map 5`
	Example:
	`Router(config)# crypto dynamic-map dynamic-map 10`
		Creates a static crypto map entry and enters crypto map configuration mode.
		or
		Creates a dynamic crypto map entry and enters crypto map configuration mode.

Step 4	**set nat demux** **Example:** `Router(config-crypto-map)# set nat demux`	Enables L2TP—IPsec support.

Step 5	**end**
	Example:
	`Router(config-crypto-map)# end`
		Exits crypto map configuration mode and returns to privileged EXEC mode.

Step 6	Do one of the following:
	• **show crypto map** [**interface** *interface* | **tag** *map-name*]
	• **show crypto dynamic-map** [**tag** *map-name*]
	Example:
	`Router# show crypto map`
	Example:
	`Router# show crypto dynamic-map 10`
		(Optional) Displays the crypto map configuration information.
		or
		(Optional) Displays the dynamic crypto map configuration information.

Step 7	**show crypto ipsec sa** **Example:** `Router# show crypto ipsec sa`	(Optional) Displays the settings used by current SAs.

Configuration Examples for L2TP IPsec Support for NAT and PAT Windows Clients

Example: Dynamic Map Configuration

The following example shows how to enable the L2TP IPsec Support for NAT and PAT Windows Clients feature for a dynamic crypto map:

```
!

version 12.3
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname 72_LNS
!
boot-start-marker
boot-end-marker
!
aaa new-model
!
!
aaa authentication ppp default local
aaa session-id common
ip subnet-zero
!
!
no ip cef
no ip domain lookup
ip domain name cisco.com
ip dhcp excluded-address 198.51.100.1
ip dhcp excluded-address 198.51.100.10
!
!
ip vrf VPN
 rd 1:1
!
! Enable virtual private networking.
vpdn enable
vpdn ip udp ignore checksum
!
! Default L2TP VPDN group
vpdn-group L2TP
!
! Enables the LNS to accept dial in requests; specifies L2TP as the tunneling
! protocol; specifies the number of the virtual templates used to clone
! virtual-access interfaces.
 accept-dialin
  protocol l2tp
  virtual-template 1

! Disables L2TP tunnel authentication.
no l2tp tunnel authentication
!
!
crypto keyring L2TP
```

```
  pre-shared-key address 0.0.0.0 0.0.0.0 key *****
!
!Defines an Internet Key Exchange (IKE) policy and assigns priority 1.
crypto isakmp policy 1
 encr aes
 authentication pre-share
 group 14
 lifetime 3600
!
crypto isakmp key cisco hostname w2k01
crypto isakmp keepalive 3600
!
crypto ipsec security-association lifetime seconds 600
!
!Defines a transform set.
crypto ipsec transform-set TS1 esp-aes esp-sha-hmac
 mode transport
!
! Names the dynamic crypto map entry and enters crypto map configuration mode; Enables
! L2TP--IPSec support; Specifies which transform sets can be used with the crypto map
! entry.
crypto dynamic-map DYN_MAP 10
 set nat demux
 set transform-set TS1!
!
crypto map CRYP_MAP 6000 ipsec-isakmp dynamic DYN_MAP
!
interface Loopback0
 ip address 198.51.100.1 255.255.255.0
!
interface FastEthernet0/0
 ip address 198.51.100.110 255.255.255.0
 no ip route-cache
 duplex full
 speed 100
 crypto map CRYP_MAP
!
interface FastEthernet0/1
 ip address 198.51.100.1 255.255.255.0
 duplex full
 speed 100
!
interface FastEthernet2/0
 ip address 172.19.192.138 255.255.255.0
 duplex full
!
interface Virtual-Template1
 ip unnumbered Loopback0
 peer default ip address pool POOL
 ppp mtu adaptive
 ppp authentication chap ms-chap
!
router ospf 1
 log-adjacency-changes
 redistribute static subnets
 network 198.51.100.10 0.0.0.255 area 0
!
ip local pool POOL 198.51.100.100 198.51.100.110
ip classless
ip route 171.0.0.0 255.0.0.0 172.19.192.1
!
no ip http server
no ip http secure-server
!
```

```
!
control-plane
!
gatekeeper
 shutdown!
!
line con 0
 exec-timeout 0 0
 logging synchronous
 stopbits 1
line aux 0
 stopbits 1
line vty 0 4
!
end
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Security commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	• Cisco IOS Security Command Reference 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 and encryption	"Configuring Security for VPNs with IPsec"
Recommended cryptographic algorithms	Next Generation Encryption

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 L2TP IPsec Support for NAT and PAT Windows Clients

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

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

Table 4: Feature Information for L2TP IPsec Support for NAT and PAT Windows Clients

Feature Name	Releases	Feature Information
L2TP IPsec Support for NAT and PAT Windows Clients	12.3(11)T4 12.4(1)	The L2TP IPsec Support for NAT and PAT Windows Clients feature allows mulitple Windows client to connect to an IPsec-enabled Cisco IOS Layer 2 Tunneling Protocol (L2TP) Network Server (LNS) through a network address translation (NAT) or port address translation (PAT) server. In 12.3(11)T4, this feature was introduced. The following commands were modified by this feature: **crypto dynamic-map, crypto map, set nat demux, show crypto dynamic-map, show crypto map, show crypto ipsec sa.**

CHAPTER 4

Deleting Crypto Sessions of Revoked Peer Certificates

The Delete Crypto Sessions of Revoked Peer Certificates on CRL Download feature deletes an active crypto session with a peer if its certificate is found to be revoked when downloading a new CRL.

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 Deleting Crypto Sessions of Revoked Peer Certificates

- If revocation check is turned off and this feature is enabled, the IKE database is not populated with the number of sessions. The show outputs do not display information about the deleted sessions.
- Frequent enabling and disabling of this feature (with active sessions on the device) is not recommended.
- Frequent CRL downloads (in a span of 30 minutes) for the same issuername (CA server) is not recommended.

- CRL cache must be enabled. CRL caching cannot be disabled for trustpoint-based prefetch. However, it is possible to disable CRL caching for URL-based prefetch.
- In case of autoenrollment on IKE, the sessions are not deleted until the next IKE rekey, whereas in case of IKEv2, the tunnel must be cleared manually or wait until the certificate expires.
- If IKE has database of "issuer-name" and "SN" populated and receives a notification from PKI about certificate revocation, IKE would act on the PKI notification.

Information About Deleting Crypto Sessions of Revoked Peer Certificates

How a Crypto Session is Deleted

1. When negotiating via certificate authentication, the peer sends the CERT payload to the device, whcih parses each certificate to store information about serial number and the issuer names. This information forms the list of serial numbers issued by the corresponding CA server and is passed to PKI for revocation check.
2. If the revocation-check crl command is configured for a trustpoint, PKI informs IKE about the revocation check thereby disabling IKE from unnecessarily storing unwanted peer certification information.
3. After a successful CRL download, PKI sends IKE a notification, which contains the "issuer-name." The CRL signature and content is verified. If there is no change in CRL content, PKI does not notify IKE.
4. If PKI notifies IKE containing the issuer name, IKE prepares a list of serial numbers for an issuer name and passes this list to PKI to verify if the serial numbers in the list are revoked.
5. PKI performs revocation check on the serial number list received from the IKE and checks the list against the downloaded CRL. The revoked serial number list is returned to IKE.
6. On a notification from PKI containing the list of revoked serial numbers, IKE identifies and deletes sessions pertaining to those serial numbers those sessions.

How to Enable Deletion of Crypto Sessions for Revoked Peer Certificates

Enabling Deletion of Crypto Sessions

Perform this task to enable the deletion of crypto sessions for revoked certificates.

SUMMARY STEPS

1. **enable**
2. **clear crypto session**
3. **configure terminal**

4. Do one of the following:
    - **crypto isakmp disconnect-revoked-peers**
    - **crypto ikev2 disconnect-revoked-peers**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear crypto session** **Example:** `Device# clear crypto session`	(Optional) Deletes IPsec crypto sessions and IKE and security associations. **Note** Use this command to enable the feature for previously established sessions, else the feature is enabled for new sessions only.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 4	Do one of the following:
	• **crypto isakmp disconnect-revoked-peers**
	• **crypto ikev2 disconnect-revoked-peers**
	Example:
	`Device(config)# crypto isakmp disconnect-revoked-peers`
	Example:
	`Device(config)# crypto ikev2 disconnect-revoked-peers`
		Disconnects IKE or IKEv2 crypto sessions with peers having revoked certificates.
		For this command to take effect, reconnected the existing sessions.

Step 5	**end**
	Example:
	`Device(config)# exit`
		Exits global configuration mode and returns to privileged EXEC mode.

Verifying the Delete Crypto Session Capability for a Revoked Peer Certificate

Perform this task to verify if the delete crypto session capability is displayed in the show output.

SUMMARY STEPS

1. **enable**
2. **show crypto isakmp peers**

3. **show crypto ikev2 session detail**

DETAILED STEPS

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **show crypto isakmp peers**

Example:

```
Device# show crypto isakmp peers
```

Displays Internet Security Association and Key Management Protocol (ISAKMP) peer descriptions.

Step 3 **show crypto ikev2 session detail**

Example:

```
Device# show crypto ikev2 session detail
```

Displays the status of active Internet Key Exchange Version 2 (IKEv2) sessions.

Configuration Examples for Deleting Crypto Sessions of Revoked Peer Certificates

Example: Enabling Deletion of Crypto Sessions for an IKE Session

```
Device> enable
Device# clear crypto session
Device# configure terminal
Device(config)# crypto isakmp disconnect-revoked-peers
Device# show crypto isakmp peers

Peer: 150.1.1.2 Port: 500 Local: 150.1.1.1
 Phase1 id: 150.1.1.2
 Disconnect Revoked Peer: Enabled
```

Example: Enabling Deletion of Crypto Sessions for an IKEv2 Session

```
Device> enable
Device# clear crypto session
Device# configure terminal
Device(config)# crypto ikev2 disconnect-revoked-peers
Device# show crypto ikev2 session detail
```

```
Session-id:1, Status:UP-ACTIVE, IKE count:1, CHILD count:1
Tunnel-id   Local          Remote           fvrf/ivrf          Status
1           10.0.0.1/500    10.0.0.2/500      (none)/(none)      READY
      Encr: 3DES, Hash: SHA96, DH Grp:2, Auth: PSK
      Life/Remaining/Active Time: 86400/86157/248 sec
      CE id: 0, Session-id: 1, MIB-id: 1
      Status Description: Negotiation done
      Local spi: 750CBE827434A245       Remote spi: 4353FEDBABEBF24C
      Local id:          10.0.0.1        Remote id:          10.0.0.2
      Local req mess id:    0           Remote req mess id: 0
      Local next mess id:    0          Remote next mess id: 2
      Local req queued:    0            Remote req queued: 0
      Local window:    5                Remote window: 5
      DPD configured for 0 seconds
      NAT-T is not detected
      Disconnect Revoked Peer: Enabled
Child sa: local selector  10.0.0.1/0 - 10.0.0.1/65535
          remote selector 10.0.0.2/0 - 10.0.0.2/65535
          ESP spi in/out: 0x9360A95/0x6C340600
          CPI in/out: 0x9FE5/0xC776
          AH spi in/out: 0x0/0x0
          Encr: AES CBC, keysize: 128, esp_hmac: SHA96
          ah_hmac: Unknown - 0, comp: IPCOMP_LZS, mode tunnel
```

Additional References for Deleting Crypto Sessions of Revoked Peers

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Commands 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*
Configuring IKE	*Configuring Internet Key Exchange for IPsec VPNs*
Configuring IKEv2	*Configuring Internet Key Exchange Version 2 and FlexVPN Site-to-Site*
Recommended cryptographic algorithms	*Next Generation Encryption*

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 Deleting Crypto Sessions of Revoked Peer Certificates

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 5: Feature Information for Deleting Crypto Sessions of Revoked Peer Certificates

Feature Name	Releases	Feature Information
Delete crypto session(s) of revoked peer cert(s) on CRL download	15.4(3)M	The Delete Crypto Sessions of Revoked Peer Certificates on CRL Download feature deletes an active crypto session with a peer if its certificate is found to be revoked when downloading a new CRL. The following commands were introduced or modified: **crypto ikev2 disconnect-revoked-peers**, **crypto isakmp disconnect-revoked-peers**, **show crypto isakmp peers**, **show crypto ikev2 session detail**.

CHAPTER 5

SafeNet IPsec VPN Client Support

The SafeNet IPsec VPN Client Support feature allows you to limit the scope of an Internet Security Association and Key Management Protocol (ISAKMP) profile or ISAKMP keyring configuration to a local termination address or interface. The benefit of this feature is that different customers can use the same peer identities and ISAKMP keys by using different local termination addresses.

History for the SafeNet IPsec VPN Client Support Feature

Release	Modification
12.3(14)T	This feature was introduced.
12.2(18)SXE	This feature was integrated into Cisco IOS Release 12.2(18)SXE.

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.

Prerequisites for SafeNet IPsec VPN Client Support

- You must understand how to configure ISAKMP profiles and ISAKMP keyrings.

Restrictions for SafeNet IPsec VPN Client Support

- The local address option works only for the primary address of an interface.
- If an IP address is provided, the administrator has to ensure that the connection of the peer terminates to the address that is provided.
- If the IP address does not exist on the device, or if the interface does not have an IP address, the ISAKMP profile or ISAKMP keyring will be effectively disabled.

Information About SafeNet IPsec VPN Client Support

ISAKMP Profile and ISAKMP Keyring Configurations Background

Prior to Cisco IOS Release 12.3(14)T, ISAKMP-profile and ISAKMP-keyring configurations could be only global, meaning that the scope of these configurations could not be limited by any locally defined parameters (VRF instances were an exception). For example, if an ISAKMP keyring contained a preshared key for address 10.11.12.13, the same key would be used if the peer had the address 10.11.12.13, irrespective of the interface or local address to which the peer was connected. There are situations, however, in which users prefer that associate keyrings be bound not only with virtual route forwarding (VRF) instances but also to a particular interface. For example, if instead of VRF instances, there are virtual LANS, and the Internet Key Exchange (IKE) is negotiated with a group of peers using one fixed virtual LAN (VLAN) interface. Such a group of peers uses a single preshared key, so if keyrings could be bound to an interface, it would be easy to define a wildcard key without risking that the keys would also be used for other customers.

Sometimes the identities of the peer are not in the control of the administrator, and even if the same peer negotiates for different customers, the local termination address is the only way to distinguish the peer. After such a distinction is made, if the traffic is sent to different VRF instances, configuring an ISAKMP profile is the only way to distinguish the peer. Unfortunately, when the peer uses an identical identity for all such situations, the ISAKMP profile cannot distinguish among the negotiations. For such scenarios, it would be beneficial to bind ISAKMP profiles to a local termination address. If a local termination address could be assigned, identical identities from the peer would not be a problem.

Local Termination Address or Interface

Effective with Cisco IOS Release 12.3(14)T, the SafeNet IPsec VPN Client Support feature allows you to limit the scope of ISAKMP profiles and ISAKMP keyrings to a local termination address or interface.

Benefit of SafeNet IPsec VPN Client Support

The benefit of this feature is that different customers can use the same peer identities and ISAKMP keys by using different local termination addresses.

How to Configure SafeNet IPsec VPN Client Support

This section contains the following procedures. The first two configurations are independent of each other.

Limiting an ISAKMP Profile to a Local Termination Address or Interface

To configure an ISAKMP profile and limit it to a local termination address or interface, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto isakmp profile** *profile-name*
4. **keyring** *keyring-name*
5. **match identity address** *address*
6. **local-address** {*interface-name* | *ip-address* [*vrf-tag*]}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto isakmp profile** *profile-name* **Example:** `Router (config)# crypto isakmp profile profile1`	Defines an ISAKMP profile and enters ISAKMP profile configuration mode.
Step 4	**keyring** *keyring-name* **Example:** `Router (conf-isa-profile)# keyring keyring1`	(Optional) Configures a keyring with an ISAKMP profile. • A keyring is not needed inside an ISAKMP profile for local termination to work. Local termination works even if Rivest, Shamir, and Adelman (RSA) certificates are used.
Step 5	**match identity address** *address* **Example:**	Matches an identity from a peer in an ISAKMP profile.

	Command or Action	Purpose
	`Router (conf-isa-profile)# match identity address 10.0.0.0 255.0.0.0`	

Step 6	**local-address** {*interface-name* | *ip-address* [*vrf-tag*]}
	Example:
	`Router (conf-isa-profile)# local-address serial2/0`
		Limits the scope of an ISAKMP profile or an ISAKMP keyring configuration to a local termination address or interface.

Limiting a Keyring to a Local Termination Address or Interface

To configure an ISAKMP keyring and limit its scope to a local termination address or interface, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto keyring** *keyring-name*
4. **local-address** {*interface-name* |*ip-address*[*vrf-tag*]}
5. **pre-shared-key address** *address*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto keyring** *keyring-name* **Example:** `Router (config)# crypto keyring keyring1`	Defines a crypto keyring to be used during IKE authentication and enters keyring configuration mode.

Step 4	**local-address** {*interface-name* |*ip-address*[*vrf-tag*]}
	Example:
	`Router (conf-keyring)# local-address serial2/0`
		Limits the scope of an ISAKMP profile or an ISAKMP keyring configuration to a local termination address or interface.

Step 5	**pre-shared-key address** *address* **Example:**	Defines a preshared key to be used for IKE authentication.

	Command or Action	Purpose
	`Router (conf-keyring)# pre-shared-key address 10.0.0.1`	

Monitoring and Maintaining SafeNet IPsec VPN Client Support

The following **debug** and **show** commands may be used to monitor and maintain the configuration in which you limited the scope of an ISAKMP profile or ISAKMP keyring to a local termination address or interface.

SUMMARY STEPS

1. **enable**
2. **debug crypto isakmp**
3. **show crypto isakmp profile**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**debug crypto isakmp** **Example:** `Router# debug crypto isakmp`	Displays messages about IKE events.
Step 3	**show crypto isakmp profile** **Example:** `Router# show crypto isakmp profile`	Lists all the ISAKMP profiles that are defined on a router.

Examples

debug crypto isakmp Command Output for an ISAKMP Keyring That IsBound to Local Termination Addresses Example

You have an ISAKMP configuration as follows (the address of serial2/0 is 10.0.0.1, and the address of serial2/1 is 10.0.0.2),

```
crypto keyring keyring1
! Scope of the keyring is limited to interface serial2/0.
  local-address serial2/0
  ! The following is the key string used by the peer.
  pre-shared-key address 10.0.0.3 key somerandomkeystring
crypto keyring keyring2
  local-address serial2/1
  ! The following is the keystring used by the peer coming into serial2/1.
   pre-shared-key address 10.0.0.3 key someotherkeystring
```

and if the connection is coming into serial2/0, keyring1 is chosen as the source of the preshared key (and keyring2 is ignored because it is bound to serial2/1), you would see the following output:

```
Router# debug crypto isakmp
*Feb 11 15:01:29.595: ISAKMP:(0:0:N/A:0):Keyring keyring2 is bound to
  10.0.0.0, skipping
*Feb 11 15:01:29.595: ISAKMP:(0:0:N/A:0):Looking for a matching key for
  10.0.0.3 in keyring1
*Feb 11 15:01:29.595: ISAKMP:(0:0:N/A:0): : success
*Feb 11 15:01:29.595: ISAKMP:(0:0:N/A:0):found peer pre-shared key
  matching 10.0.0.3
*Feb 11 15:01:29.595: ISAKMP:(0:0:N/A:0): local preshared key found
```

debug crypto isakmp Command Output for an ISAKMP ProfileThat Is Boundto a Local Termination Address Example

If you have the following configuration,

```
crypto isakmp profile profile1
  keyring keyring1
  match identity address 10.0.0.0 255.0.0.0
  local-address serial2/0
crypto isakmp profile profile2
  keyring keyring1
  keyring keyring2
  self-identity fqdn
  match identity address 10.0.0.1 255.255.255.255
  local-address serial2/1
```

and the connection is coming through the local terminal address serial2/0, you will see the following output:

```
Router# debug crypto isakmp
*Feb 11 15:01:29.935: ISAKMP:(0:0:N/A:0):
Profile profile2 bound to 10.0.0.0 skipped
*Feb 11 15:01:29.935: ISAKMP:(0:1:SW:1):: peer matches profile1 profile
```

show crypto isakmp profile Command Output Example

The following is an example of typical **show** command output for an ISAKMP profile that is bound to serial2/0:

```
Router# show crypto isakmp profile
ISAKMP PROFILE profile1
  Identities matched are:
    ip-address 10.0.0.0 255.0.0.0
  Certificate maps matched are:
  keyring(s): keyring1
  trustpoint(s): <all>
  Interface binding: serial2/0 (10.20.0.1:global)
```

Troubleshooting SafeNet IPsec VPN Client Support

If an ISAKMP profile or ISAKMP keyring fails to be selected, you should double-check the local-address binding in the ISAKMP profile or ISAKMP keyring configuration and follow the output of the IKE debugs to determine whether the peer is correctly terminating on the address. You may remove the local-address binding (to make the scope of the profile or keyring global) and check to determine whether the profile or keyring is selected to confirm the situation.

Configuration Examples for SafeNet IPsec VPN Client Support

This section contains the following configuration, **debug** command, and **show** command examples.

ISAKMP Profile Bound to a Local Interface Example

The following example shows that the ISAKMP profile is bound to a local interface:

```
crypto isakmp profile profile1
  keyring keyring1
  match identity address 10.0.0.0 255.0.0.0
```

local-address serial2/0

ISAKMP Keyring Bound to a Local Interface Example

The following example shows that the ISAKMP keyring is bound only to interface serial2/0:

```
crypto keyring
  local-address serial2/0
```

pre-shared-key address 10.0.0.1

ISAKMP Keyring Bound to a Local IP Address Example

The following example shows that the ISAKMP keyring is bound only to IP address 10.0.0.2:

```
crypto keyring keyring1
  local-address 10.0.0.2
```

pre-shared-key address 10.0.0.2 key

ISAKMP Keyring Bound to an IP Address and Limited to a VRF Example

The following example shows that an ISAKMP keyring is bound to IP address 10.34.35.36 and that the scope is limited to VRF examplevrf1:

```
ip vrf examplevrf1
  rd 12:3456
crypto keyring ring1
  local-address 10.34.35.36 examplevrf1
interface ethernet2/0
  ip vrf forwarding examplevrf1
  ip address 10.34.35.36 255.255.0.0
```

Additional References

The following sections provide references related to SafeNet IPsec VPN Client Support.

Related DocumentsStandards

Related Topic	Document Title
Configuring ISAKMP profiles and ISAKMP keyrings	VRF-Aware IPsec
Security commands	*Cisco IOS Security Command Reference*

Standard	Title
No new or modified standards are supported by this feature.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature.	To locate and download MIBs for selected platforms, Cisco IOS software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

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

Technical Assistance

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

CHAPTER 6

Ability to Disable Extended Authentication for Static IPsec Peers

The Ability to Disable Extended Authentication for Static IPsec Peers feature allows users to disable extended authentication (Xauth), preventing the routers from being prompted for Xauth information--username and password.

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.

Feature Overview

Without the ability to disable Xauth, a user cannot select which peer on the same crypto map should use Xauth. That is, if a user has router-to-router IP security (IPsec) on the same crypto map as a virtual private network (VPN)-client-to-Cisco-IOS IPsec, both peers are prompted for a username and password. In addition, a remote static peer (a Cisco IOS router) cannot establish an Internet Key Exchange (IKE) security association (SA) with the local Cisco IOS router. (Xauth is not an optional exchange, so if a peer does not respond to an Xauth request, the IKE SA is deleted.) Thus, the same interface cannot be used to terminate IPsec to VPN clients (that need Xauth) as well as other Cisco IOS routers (that cannot respond to Xauth) unless this feature is implemented.

Benefits

If VPN-client-to-Cisco-IOS IPsec and router-to-router IPsec exist on a single interface, the Ability to Disable Extended Authentication for Static IPsec Peers feature allows a user to disable Xauth while configuring the preshared key for router-to-router IPsec. Thus, the router will not prompt the peer for a username and password, which are transmitted when Xauth occurs for VPN-client-to-Cisco-IOS IPsec.

Restrictions

Xauth can be disabled only if preshared keys are used as the authentication mechanism for the given crypto map.

Related Documents

- "Configuring Internet Key Exchange for IPsec VPNs" chapter in the *Cisco IOS Security Configuration Guide: Secure Connectivity*
- "Configuring Security for VPNs with IPsec" chapter in the *Cisco IOS Security Configuration Guide: Secure Connectivity*
- *Cisco IOS Security Command Reference*

Supported Standards MIBs and RFCs

Standards

None

MIBs

None

To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL:

http://www.cisco.com/go/mibs

RFCs

No new or modified RFCs are supported by this feature.

Prerequisites

Before you can disable Xauth for static IPsec peers, you must complete the following tasks:

- Enable authentication, authorization, and accounting (AAA).

Note Configuring AAA is required only if the VPN-client-to-Cisco-IOS is using AAA authentication.

- Configure an IPsec transform.
- Configure a static crypto map.
- Configure ISAKMP policy.

Configuration Tasks

See the following sections for configuration tasks for the Ability to Disable Extended Authentication for Static IPsec Peers feature. Each task in the list is identified as either required or optional.

- Disabling Xauth for Static IPsec Peers, on page 115

Disabling Xauth for Static IPsec Peers

To disable Xauth for router-to-router IPsec, use the following command in global configuration mode:

Command	Purpose
`Router(config)# crypto isakmp key keystring address peer-address [mask] [no-xauth]`	Configures a preshared authentication key. Use the **no-xauth** keyword if router-to-router IPsec is on the same crypto map as VPN-client-to-Cisco IOS IPsec. This keyword prevents the router from prompting the peer for Xauth information. You must configure the local and remote peer for preshared keys. **Note** According to the design of preshared key authentication in IKE main mode, preshared keys must be based on the IP address of the peers. Although you can send hostname as the identity of preshared key authentication, the key is searched on the IP address of the peer; if the key is not found (based on the IP address) the negotiation will fail.

Configuration Examples

Disabling Xauth for Static IPsec Peers Configuration

The following example shows how the local peer specifies the preshared key, designates the remote peer by its IP address, and disables Xauth:

crypto isakmp key sharedkeystring address 172.21.230.33 no-xauth

Feature Information for Ability to Disable Xauth for Static IPsec Peers

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 6: Feature Information for Ability to Disable Xauth for Static IPsec Peers

Feature Name	Releases	Feature Information
Ability to Disable Extended Authentication for Static IPsec Peers	12.2(4)T	This feature allows users to disable Xauth, preventing the routers from being prompted for Xauth information. The following command was modified: **crypto isakmp key**.

CHAPTER 7

Crypto Conditional Debug Support

The Crypto Conditional Debug Support feature introduces three new command-line interfaces (CLIs) that allow users to debug an IP Security (IPSec) tunnel on the basis of predefined crypto conditions such as the peer IP address, connection-ID of a crypto engine, and security parameter index (SPI). By limiting debug messages to specific IPSec operations and reducing the amount of debug output, users can better troubleshoot a router with a large number of tunnels.

Feature History for Crypto Conditional Debug Support

Feature History	
Release	Modification
12.3(2)T	This feature was introduced.

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.

Prerequisites for Crypto Conditional Debug Support

To use the new crypto CLIs, you must be using a crypto image such as the k8 or k9 subsystem.

Restrictions for Crypto Conditional Debug Support

- This feature does not support debug message filtering for hardware crypto engines.
- Although conditional debugging is useful for troubleshooting peer-specific or functionality related Internet Key Exchange (IKE) and IPSec problems, conditional debugging may not be able to define and check large numbers of debug conditions.
- Because extra space is needed to store the debug condition values, additional processing overhead is added to the CPU and memory usage is increased. Thus, enabling crypto conditional debugging on a router with heavy traffic should be used with caution.

Information About Crypto Conditional Debug Support

Supported Condition Types

The new crypto conditional debug CLIs-- debug crypto condition , debug crypto condition unmatched , and show crypto debug-condition --allow you to specify conditions (filter values) in which to generate and display debug messages related only to the specified conditions. The table below lists the supported condition types.

Table 7: Supported Condition Types for Crypto Debug CLI

Condition Type (Keyword)	Description
connid [1]	An integer between 1-32766. Relevant debug messages will be shown if the current IPSec operation uses this value as the connection ID to interface with the crypto engine.
flowid 1	An integer between 1-32766. Relevant debug messages will be shown if the current IPSec operation uses this value as the flow-ID to interface with the crypto engine.
FVRF	The name string of a virtual private network (VPN) routing and forwarding (VRF) instance. Relevant debug messages will be shown if the current IPSec operation uses this VRF instance as its front-door VRF (FVRF).
IVRF	The name string of a VRF instance. Relevant debug messages will be shown if the current IPSec operation uses this VRF instance as its inside VRF (IVRF).
peer group	A Unity group-name string. Relevant debug messages will be shown if the peer is using this group name as its identity.
peer hostname	A fully qualified domain name (FQDN) string. Relevant debug messages will be shown if the peer is using this string as its identity; for example, if the peer is enabling IKE Xauth with this FQDN string.
peer ipaddress	A single IP address. Relevant debug messages will be shown if the current IPSec operation is related to the IP address of this peer.

Condition Type (Keyword)	Description
peer subnet	A subnet and a subnet mask that specify a range of peer IP addresses. Relevant debug messages will be shown if the IP address of the current IPSec peer falls into the specified subnet range.
peer username	A username string. Relevant debug messages will be shown if the peer is using this username as its identity; for example, if the peer is enabling IKE Extended Authentication (Xauth) with this username.
SPI [1]	A 32-bit unsigned integer. Relevant debug messages will be shown if the current IPSec operation uses this value as the SPI.

[1] If an IPSec connid, flowid, or SPI is used as a debug condition, the debug messages for a related IPSec flow are generated. An IPSec flow has two connids, flowids, and SPIs--one inbound and one outbound. Both two connids, flowids, and SPIs can be used as the debug condition that triggers debug messages for the IPSec flow.

How to Enable Crypto Conditional Debug Support

Enabling Crypto Conditional Debug Messages

Performance Considerations

- Before enabling crypto conditional debugging, you must decide what debug condition types (also known as debug filters) and values will be used. The volume of debug messages is dependent on the number of conditions you define.

Note Specifying numerous debug conditions may consume CPU cycles and negatively affect router performance.

- Your router will perform conditional debugging only after at least one of the global crypto debug commands--**debug crypto isakmp**, **debug crypto ipsec**, and **debug crypto engine**--has been enabled. This requirement helps to ensure that the performance of the router will not be impacted when conditional debugging is not being used.

Disable Crypto Debug Conditions

If you choose to disable crypto conditional debugging, you must first disable any crypto global debug CLIs you have issued ; thereafter, you can disable conditional debugging.

Note The **reset** keyword can be used to disable all configured conditions at one time.

SUMMARY STEPS

1. **enable**
2. **debug crypto condition** [**connid***integer***engine-id***integer*] [**flowid***integer* **engine-id***integer*] [**fvrf** *string*] [**ivrf** *string*] [**peer** [**group** *string*] [**hostname** *string*] [**ipv4** *ipaddress*] [**subnet** *subnet mask*] [**username** *string*]] [**spi** *integer*] [**reset**]
3. **show crypto debug-condition** {[**peer**] [**connid**] [**spi**] [**fvrf**] [**ivrf**] [**unmatched**]}
4. **debug crypto isakmp**
5. **debug crypto ipsec**
6. **debug crypto engine**
7. debug crypto condition **unmatched** [**isakmp** | **ipsec** | **engine**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**debug crypto condition** [**connid***integer***engine-id***integer*] [**flowid***integer* **engine-id***integer*] [**fvrf** *string*] [**ivrf** *string*] [**peer** [**group** *string*] [**hostname** *string*] [**ipv4** *ipaddress*] [**subnet** *subnet mask*] [**username** *string*]] [**spi** *integer*] [**reset**] **Example:** `Router# debug crypto condition connid 2000 engine-id 1`	Defines conditional debug filters.

Step 3	**show crypto debug-condition** {[**peer**] [**connid**] [**spi**] [**fvrf**] [**ivrf**] [**unmatched**]}
	Example:
	`Router# show crypto debug-condition spi`
		Displays crypto debug conditions that have already been enabled in the router.

Step 4	**debug crypto isakmp**
	Example:
	`Router#`
	`debug crypto isakmp`
		Enables global IKE debugging.

Step 5	**debug crypto ipsec**
	Example:
	`Router#`
	`debug crypto ipsec`
		Enables global IPSec debugging.

	Command or Action	Purpose

Step 6	**debug crypto engine**
	Example:
	`Router#`
	`debug crypto engine`
		Enables global crypto engine debugging.

Step 7	debug crypto condition **unmatched** [**isakmp** | **ipsec** | **engine**] **Example:** `Router# debug crypto condition unmatched ipsec`	(Optional) Displays debug conditional crypto messages when no context information is available to check against debug conditions. If none of the optional keywords are specified, all crypto-related information will be shown.

Enabling Crypto Error Debug Messages

To enable crypto error debug messages, you must perform the following tasks.

debug crypto error CLI

Enabling the **debug crypto error** command displays only error-related debug messages, thereby, allowing you to easily determine why a crypto operation, such as an IKE negotiation, has failed within your system.

Note When enabling this command, ensure that global crypto debug commands are not enabled; otherwise, the global commands will override any possible error-related debug messages.

SUMMARY STEPS

1. **enable**
2. **debug crypto** {**isakmp** | **ipsec** | **engine**} **error**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**debug crypto** {**isakmp** | **ipsec** | **engine**} **error** **Example:** `Router# debug crypto ipsec error`	Enables only error debugging messages for a crypto area.

Configuration Examples for the Crypto Conditional Debug CLIs

Enabling Crypto Conditional Debugging Example

The following example shows how to display debug messages when the peer IP address is 10.1.1.1, 10.1.1.2, or 10.1.1.3, and when the connection-ID 2000 of crypto engine 0 is used. This example also shows how to enable global debug crypto CLIs and enable the **show crypto debug-condition** command to verify conditional settings.

```
Router#
debug crypto condition connid 2000 engine-id 1
Router#
debug crypto condition peer ipv4 10.1.1.1
Router#
debug crypto condition peer ipv4 10.1.1.2
Router#
debug crypto condition peer ipv4 10.1.1.3
Router#
debug crypto condition unmatched
! Verify crypto conditional settings.
Router#
show crypto debug-condition
Crypto conditional debug currently is turned ON
IKE debug context unmatched flag:ON
IPsec debug context unmatched flag:ON
Crypto Engine debug context unmatched flag:ON
IKE peer IP address filters:
10.1.1.1  10.1.1.2   10.1.1.3
Connection-id filters:[connid:engine_id]2000:1,
! Enable global crypto CLIs to start conditional debugging.
Router#
debug crypto isakmp
Router#
debug crypto ipsec
Router#
debug crypto engine
```

Disabling Crypto Conditional Debugging Example

The following example shows how to disable all crypto conditional settings and verify that those settings have been disabled:

```
Router#
debug crypto condition reset
! Verify that all crypto conditional settings have been disabled.
Router#
show crypto debug-condition
Crypto conditional debug currently is turned OFF
IKE debug context unmatched flag:OFF
IPsec debug context unmatched flag:OFF
Crypto Engine debug context unmatched flag:OFF
```

Additional References

The following sections provide references to the Crypto Conditional Debug Support feature.

Related Documents

Related Topic	Document Title
IPSec and IKE configuration tasks	"Internet Key Exchange for IPsec VPNs" section of *Cisco IOS Security Configuration Guide: Secure Connectivity*
IPSec and IKE commands	*Cisco IOS Security Command Reference*

Standards

Standards	Title
None	--

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco IOS software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
None	--

Technical Assistance

Description	Link
Technical Assistance Center (TAC) home page, containing 30,000 pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

CHAPTER 8

VPN Acceleration Module

VPN Acceleration Module (VAM) supports Data Encryption Standard (DES) or Triple DES (3DES) IPsec encryption at a rate greater than full-duplex DS-3 line rate (up to 145 Mbps) for site-to-site VPNs such as intranets and extranets. VAM also supports up to 5000 encrypted tunnels for mixed VPN environments that have both site-to-site and remote access VPN requirements. VAM integrates hardware-assisted Rivest, Shamir, and Adelman (RSA) and IP Payload Compression Protocol (IPPCP) layer 3 compression to accelerate RSA processing, thereby enhancing tunnel setup and improving overall VPN initialization. In environments where bandwidth is costly, VAM provides hardware-based IPPCP Lempel-Ziv-Stac (LZS) processing to compress network traffic before it is encrypted and sent over pay-per-byte WAN connections.

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.

Prerequisites for VPN Acceleration Module

You must configure IPSec and IKE on the router and a crypto map to all interfaces that require encryption service from the VPN Acceleration Module.

Information about VPN Acceleration Module (VAM)

VPN Acceleration Module (VAM) Overview

The VPN Acceleration Module (VAM) is a single-width acceleration module. It provides high-performance, hardware-assisted tunneling and encryption services suitable for VPN remote access, site-to-site intranet, and extranet applications. It also provides platform scalability and security while working with all services necessary for successful VPN deployments—security, quality of service (QoS), firewall and intrusion detection, service-level validation, and management. The VAM off-loads IPsec processing from the main processor, thus freeing resources on the processor engines for other tasks.

The VAM provides hardware-accelerated support for the following multiple encryption functions:

- 56-bit Data Encryption Standard (DES) standard mode: Cipher Block Chaining (CBC)
- 3-Key Triple DES (168-bit)
- Secure Hash Algorithm (SHA)-1 and Message Digest 5 (MD5)
- Rivest, Shamir, Adelman (RSA) public-key algorithm
- Diffie-Hellman key exchange RC4-40

Benefits

The VAM provides the following benefits:

- 10 tunnels per second
- The following number of tunnels based on the corresponding memory of the NPE:
    - 800 tunnels for 64 MB
    - 1600 tunnels for 128 MB
    - 3200 tunnels for 256 MB
    - 5000 tunnels for 512 MB
- RSA encryption
- Accelerated Crypto performance
- Accelerated Internet Key Exchange (IKE)
- Certificate support for automatic authentication using digital certificates
- Dual VAM support

Note Support for dual VAMs is available on a Cisco 7200 series router with an NPE-G1, on Cisco IOS Release 12.2(15)T, 12.1(14)E, and 12.3 Mainline.

- Encryption services to any port adapter installed in the router. The interface on the port adapter must be configured with a crypto map to support IPSec.
- Full-duplex data transmission of over 100 Mbps with various encryption and compression schemes for 300 byte packages
- Hardware-based IPPCP LZS compression
- Network traffic compression that reduces bandwidth utilization
- Online Insertion and Removal (OIR)
- QoS, multiprotocol, and multicast feature interoperation
- Support for full Layer 3 routing, such as Enhanced Interior Gateway Routing Protocol (EIGRP), Open Shortest Path First (OSPF), and Border Gateway Protocol (BGP) across the IPSec VPN
- Up to 145 Mbps throughput using 3DES
- VPN initialization improvements

Performance Results for Single VAM

The following two tables provide performance results for a single VAM on a Cisco 7206VXR with an NPE-G1 processor, an onboard GE, and FE port adapters in slots 3 and 4.

clear_packet _size	crypto_packet_size	out_packet_size
64	96	114
300	336	354
1400	1432	1450
Mixed packet size - 344	378	396

pkt_size (bytes)	# of tunnels	measured_pps (pps)	meas_clear_ndr (Mbps)	meas_crypto_ndr (Mbps)	meas_out_ndr (Mbps)
64	4	65,224	33.39	50.09	59.48
	500	41,888	21.44	32.17	38.20
	1,000	40,480	20.73	31.09	36.92
	5,000	39,408	20.18	30.27	35.94
300	4	38,032	91.28	102.23	107.71
	500	37,184	89.24	99.95	105.31
	1,000	36,064	86.55	96.94	102.13
	5,000	36,016	86.44	96.81	101.99
1400	4	9,984	111.82	114.38	115.81

pkt_size (bytes)	# of tunnels	measured_pps (pps)	meas_clear_ndr (Mbps)	meas_crypto_ndr (Mbps)	meas_out_ndr (Mbps)
	500	9,848	110.29	112.82	114.24
	1,000	9,648	108.06	110.53	111.92
	5,000	9,616	107.70	110.16	111.55
Mixed packet size	4	31,472	86.61	95.17	99.70
	500	31,056	85.47	93.91	98.39
	1,000	30,128	82.91	91.11	95.45
	5,000	29,264	80.53	88.49	92.71

Performance Results for Dual VAMs

The following two tables provide performance results for dual VAMs on a Cisco 7206VXR with an NPE-G1 processor, an onboard GE, and FE port adapters in slots 3 and 4.

clear_packet _size	crypto_packet_size	out_packet_size
64	96	114
300	336	354
1400	1432	1450
Mixed packet size - 344	378	396

pkt_size (bytes)	# of tunnels	measured_pps (pps)	meas_clear_ndr (Mbps)	meas_crypto_ndr (Mbps)	meas_out_ndr (Mbps)
64	4	135,544	69.40	104.10	123.61
	500	61,520	31.50	47.25	56.11
	1,000	56,928	29.15	43.72	51.92
	5,000	43,744	22.40	33.60	39.89
300	4	71,336	171.21	191.75	202.02
	500	60,416	145.00	162.40	171.10
	1,000	56,016	134.44	150.57	158.64
	5,000	42,496	101.99	114.23	120.35
1400	4	18,736	209.84	214.64	217.34
	500	18,424	206.35	211.07	213.72

pkt_size (bytes)	# of tunnels	measured_pps (pps)	meas_clear_ndr (Mbps)	meas_crypto_ndr (Mbps)	meas_out_ndr (Mbps)
	1000	18,352	205.54	210.24	212.88
	5,000	18,352	205.54	210.24	212.88
Mixed packet size	4	60,416	166.26	182.70	191.40
	500	57,888	159.31	175.05	183.40
	1,000	55,488	152.70	167.80	175.79
	5,000	34,272	94.32	103.64	108.57

How To Configure VPN Acceleration Module (VAM)

On power up if the enabled LED is on, the VAM is fully functional and does not require any configuration commands. However, for the VAM to provide encryption services, you must complete the following tasks:

Creating IKE Policies

Before you begin

The following restrictions apply if you are configuring an AES IKE policy:

- Your device must support IPsec and long keys (the "k9" subsystem).
- AES cannot encrypt IPsec and IKE traffic if an acceleration card is present.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto isakmp policy** *priority*
4. **encryption** {**des** | **3des** | **aes** | **aes 192** | **aes 256**}
5. **hash** {**sha** | **sha256** | **sha384** | **md5**}
6. **authentication** {**rsa-sig** | **rsa-encr** | **pre-share**}
7. **group** {**1** | **2** | **5** | **14** | **15** | **16** | **19** | **20** | **24**}
8. **lifetime** *seconds*
9. **exit**
10. **exit**
11. **show crypto isakmp policy**
12. Repeat these steps for each policy you want to create.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto isakmp policy** *priority* **Example:** `Router(config)# crypto isakmp policy 10`	Defines an IKE policy and enters config-isakmp configuration mode. • *priority* —Uniquely identifies the IKE policy and assigns a priority to the policy. Valid values: 1 to 10,000; 1 is the highest priority.

Step 4	**encryption** {**des** | **3des** | **aes** | **aes 192** | **aes 256**}
	Example:
	`Router(config-isakmp)# encryption aes 256`
		Specifies the encryption algorithm.
		• By default, the **des** keyword is used.
		  • **des**—56-bit DES-CBC (No longer recommended. AES is the recommended encryption algorithm)
		  • **3des**—168-bit DES (No longer recommended. AES is the recommended encryption algorithm)
		  • **aes**—128-bit AES
		  • **aes 192**—192-bit AES
		  • **aes 256**—256-bit AES

Step 5	**hash** {**sha** | **sha256** | **sha384** | **md5**}
	Example:
	`Router(config-isakmp)# hash sha`
		Specifies the hash algorithm.
		• By default, SHA-1 (**sha**) is used.
		• 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 **md5** keyword specifies MD5 (HMAC variant) as the hash algorithm. (No longer recommended. SHA-256 is the recommended replacement.)

Step 6	**authentication** {**rsa-sig** | **rsa-encr** | **pre-share**}
	Example:
	`Router(config-isakmp)# authentication pre-share`
		Specifies the authentication method.
		• By default, RSA signatures are used.
		  • **rsa-sig**—RSA signatures require that you configure your peer routers to obtain certificates from a CA.

	Command or Action	Purpose
		• **rsa-encr**—RSA encrypted nonces require that you ensure each peer has the other peer's RSA public keys. • **pre-share**—Preshared keys require that you separately configure these preshared keys.

Step 7	**group** {**1** | **2** | **5** | **14** | **15** | **16** | **19** | **20** | **24**}
	Example:
	`Router(config-isakmp)# group 14`
		Specifies the Diffie-Hellman (DH) group identifier.
		• By default, DH group 1 is used.
		• **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/DSA 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). Group 14 or higher (where possible) 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.

Step 8	**lifetime** *seconds* **Example:** `Router(config-isakmp)# lifetime 180`	Specifies the lifetime of the IKE SA. • *seconds*—Time, in seconds, before each SA expires. Valid values: 60 to 86,400; default value: 86,400. **Note** The shorter the lifetime (up to a point), the more secure your IKE negotiations will be. However, with longer lifetimes, future IPsec SAs can be set up more quickly.
Step 9	**exit** **Example:** `Router(config-isakmp)# exit`	Exits config-isakmp configuration mode.
Step 10	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.

	Command or Action	Purpose
Step 11	**show crypto isakmp policy** **Example:** `Router# show crypto isakmp policy`	(Optional) Displays all existing IKE policies.
Step 12	Repeat these steps for each policy you want to create.	—

Examples

The following sample output from the **show crypto isakmp policy** command displays a warning message after a user tries to configure an IKE encryption method that the hardware does not support:

```
Router# show crypto isakmp policy
Protection suite of priority 1
        encryption algorithm:  AES - Advanced Encryption Standard (256 bit keys).
WARNING:encryption hardware does not support the configured
encryption method for ISAKMP policy 1
        hash algorithm:        Secure Hash Standard 2 (256-bit)
        authentication method: Pre-Shared Key
        Diffie-Hellman group:  #14 (2048 bit)
        lifetime:              3600 seconds, no volume limit
```

Configuring IPsec

After you have completed IKE configuration, configure IPsec at each participating IPsec peer. This section contains basic steps to configure IPsec.

Creating Crypto Access Lists

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Do one of the following:
    - **access-list** *access-list-number* {**deny** | **permit**} *protocol source source-wildcard destination destination-wildcard* [**log**]
    - **ip access-list extended** *name*
4. Repeat Step 3 for each crypto access list you want to create.

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	Do one of the following:
	• **access-list** *access-list-number* {**deny** | **permit**} *protocol source source-wildcard destination destination-wildcard* [**log**]
	• **ip access-list extended** *name*
	Example:
	`Device(config)# access-list 100 permit ip 10.0.68.0 0.0.0.255 10.1.1.0 0.0.0.255`
	Example:
	`Device(config)# ip access-list extended vpn-tunnel`
		Specifies conditions to determine which IP packets are protected.
		• You specify conditions using an IP access list designated by either a number or a name. The **access-list** command designates a numbered extended access list; the **ip access-list extended** command designates a named access list.
		• Enable or disable crypto for traffic that matches these conditions.
		Tip Cisco recommends that you configure “mirror image” crypto access lists for use by IPsec and that you avoid using the **any** keyword.

Step 4	Repeat Step 3 for each crypto access list you want to create.	—

Configuring Transform Sets for IKEv1

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto ipsec transform-set** *transform-set-name transform1* [*transform2* [*transform3*]]
4. **mode** [**tunnel** | **transport**]
5. **end**
6. **clear crypto sa** [**peer** {*ip-address* | *peer-name*} | **sa map** *map-name* | **sa entry** *destination-address protocol spi*]
7. **show crypto ipsec transform-set** [**tag** *transform-set-name*]

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.

	Command or Action	Purpose
Step 3	**crypto ipsec transform-set** *transform-set-name transform1* [*transform2* [*transform3*]] **Example:** `Device(config)# crypto ipsec transform-set aesset esp-aes 256 esp-sha-hmac`	Defines a transform set and enters crypto transform configuration mode. • There are complex rules defining the entries that you can use for transform arguments. These rules are explained in the command description for the **crypto ipsec transform-set** command, and the table in "About Transform Sets" section provides a list of allowed transform combinations.
Step 4	**mode** [**tunnel** | **transport**] **Example:** `Device(cfg-crypto-tran)# mode transport`	(Optional) Changes the mode associated with the transform set. • The mode setting is applicable only to traffic whose source and destination addresses are the IPsec peer addresses; it is ignored for all other traffic. (All other traffic is in tunnel mode only.)

Step 5	**end**
	Example:
	`Device(cfg-crypto-tran)# end`
		Exits crypto transform configuration mode and enters privileged EXEC mode.

Step 6	**clear crypto sa** [**peer** {*ip-address* | *peer-name*} | **sa map** *map-name* | **sa entry** *destination-address protocol spi*] **Example:** `Device# clear crypto sa`	(Optional) Clears existing IPsec security associations so that any changes to a transform set takes effect on subsequently established security associations. Manually established SAs are reestablished immediately. • Using the **clear crypto sa** command without parameters clears out the full SA database, which clears out active security sessions. • You may also specify the **peer**, **map**, or **entry** keywords to clear out only a subset of the SA database.
Step 7	**show crypto ipsec transform-set** [**tag** *transform-set-name*] **Example:** `Device# show crypto ipsec transform-set`	(Optional) Displays the configured transform sets.

Creating Static Crypto Maps

When IKE is used to establish SAs, the IPsec peers can negotiate the settings they use for the new security associations. This means that you can specify lists (such as lists of acceptable transforms) within the crypto map entry.

Perform this task to create crypto map entries that use IKE to establish SAs. To create IPv6 crypto map entries, you must use the **ipv6** keyword with the **crypto map** command. For IPv4 crypto maps, use the **crypto map** command without the **ipv6** keyword.

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. **crypto map** [**ipv6**] *map-name seq-num* [**ipsec-isakmp**]
4. **match address** *access-list-id*
5. **set peer** {*hostname* | *ip-address*}
6. **crypto ipsec security-association dummy** {**pps** *rate* | **seconds** *seconds*}
7. **set transform-set** *transform-set-name1* [*transform-set-name2...transform-set-name6*]
8. **set security-association lifetime** {**seconds** *seconds* | **kilobytes** *kilobytes* | **kilobytes disable**}
9. **set security-association level per-host**
10. **set pfs** [**group1** | **group14** | **group15** | **group16** | **group19** | **group2** | **group20** | **group24** | **group5**]
11. **end**
12. **show crypto map** [**interface** *interface* | **tag** *map-name*]

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 map** [**ipv6**] *map-name seq-num* [**ipsec-isakmp**] **Example:** `Device(config)# crypto map static-map 1 ipsec-isakmp`	Creates or modifies a crypto map entry, and enters crypto map configuration mode. • For IPv4 crypto maps, use the command without the **ipv6** keyword.
Step 4	**match address** *access-list-id* **Example:** `Device(config-crypto-m)# match address vpn-tunnel`	Names an extended access list. • This access list determines the traffic that should be protected by IPsec and the traffic that should not be protected by IPsec security in the context of this crypto map entry.
Step 5	**set peer** {*hostname* | *ip-address*} **Example:**	Specifies a remote IPsec peer—the peer to which IPsec protected traffic can be forwarded.

	Command or Action	Purpose
	`Device(config-crypto-m)# set-peer 192.168.101.1`	• Repeat for multiple remote peers.

Step 6	**crypto ipsec security-association dummy** {**pps** *rate* | **seconds** *seconds*}
	Example:
	`Device(config-crypto-m)# set security-association dummy seconds 5`
		Enables generating dummy packets. These dummy packets are generated for all flows created in the crypto map.

Step 7	**set transform-set** *transform-set-name1* [*transform-set-name2...transform-set-name6*] **Example:** `Device(config-crypto-m)# set transform-set aesset`	Specifies the transform sets that are allowed for this crypto map entry. • List multiple transform sets in the order of priority (highest priority first).

Step 8	**set security-association lifetime** {**seconds** *seconds* | **kilobytes** *kilobytes* | **kilobytes disable**}
	Example:
	`Device (config-crypto-m)# set security-association lifetime seconds 2700`
		(Optional) Specifies a SA lifetime for the crypto map entry.
		• By default, the SAs of the crypto map are negotiated according to the global lifetimes, which can be disabled.

Step 9	**set security-association level per-host** **Example:** `Device(config-crypto-m)# set security-association level per-host`	(Optional) Specifies that separate SAs should be established for each source and destination host pair. • By default, a single IPsec "tunnel" can carry traffic for multiple source hosts and multiple destination hosts. **Caution** Use this command with care because multiple streams between given subnets can rapidly consume resources.
Step 10	**set pfs** [**group1** | **group14** | **group15** | **group16** | **group19** | **group2** | **group20** | **group24** | **group5**] **Example:** `Device(config-crypto-m)# set pfs group14`	(Optional) Specifies that IPsec either should ask for password forward secrecy (PFS) when requesting new SAs for this crypto map entry or should demand PFS in requests received from the IPsec peer. • Group 1 specifies the 768-bit Diffie-Hellman (DH) identifier (default). (No longer recommended). • Group 2 specifies the 1024-bit DH identifier. (No longer recommended). • Group 5 specifies the 1536-bit DH identifier. (No longer recommended) • Group 14 specifies the 2048-bit DH identifier. • Group 15 specifies the 3072-bit DH identifier. • Group 16 specifies the 4096-bit DH identifier.

	Command or Action	Purpose
		• Group 19 specifies the 256-bit elliptic curve DH (ECDH) identifier. • Group 20 specifies the 384-bit ECDH identifier. • Group 24 specifies the 2048-bit DH/DSA identifier • By default, PFS is not requested. If no group is specified with this command, group 1 is used as the default.

Step 11	**end**
	Example:
	`Device(config-crypto-m)# end`
		Exits crypto map configuration mode and returns to privileged EXEC mode.

Step 12	**show crypto map** [**interface** *interface* | **tag** *map-name*] **Example:** `Device# show crypto map`	Displays your crypto map configuration.

Verifying the Configuration

SUMMARY STEPS

1. **show crypto ipsec transform-set**
2. **show crypto map** [**interface** *interface* | **tag** *map-name*]
3. **show crypto ipsec sa** [**map** *map-name* | **address** | **identity** | **detail** | **interface**]

DETAILED STEPS

Step 1 **show crypto ipsec transform-set**

Example:

```
Device# show crypto ipsec transform-set

Transform set combined-des-md5: {esp-des esp-md5-hmac}
   will negotiate = {Tunnel,},
Transform set t1: {esp-des esp-md5-hmac}
   will negotiate = {Tunnel,},
Transform set t100: {ah-sha-hmac}
   will negotiate = {Transport,},
Transform set t2: {ah-sha-hmac}
   will negotiate = {Tunnel,},
   {esp-des}
   will negotiate = {Tunnel,},
```

Displays the transform set configuration.

Step 2 **show crypto map** [**interface** *interface* | **tag** *map-name*]

Example:

```
Device# show crypto map

Crypto Map “abc” 10 ipsec-isakmp
        Peer = 172.21.114.67
        Extended IP access list 141
            access-list 141 permit ip
                source: addr = 172.21.114.123/0.0.0.0
                dest:   addr = 172.21.114.67/0.0.0.0
        Current peer: 172.21.114.67
        Security-association lifetime: 4608000 kilobytes/120 seconds
        PFS (Y/N): N
        Transform sets={t1,}
```

Displays the crypto map configuration.

Step 3 **show crypto ipsec sa** [**map** *map-name* | **address** | **identity** | **detail** | **interface**]

Example:

```
Device# show crypto map ipsec sa interface

    Crypto map tag: abc, local addr. 172.21.114.123
   local ident (addr/mask/prot/port): (172.21.114.123/255.255.255.255/0/0)
   remote ident (addr/mask/prot/port): (172.21.114.67/255.255.255.255/0/0)
    current_peer: 172.21.114.67
     PERMIT, flags={origin_is_acl,}
    #pkts encaps: 10, #pkts encrypt: 10, #pkts digest 10
    #pkts decaps: 10, #pkts decrypt: 10, #pkts verify 10
    #send errors 10, #recv errors 0
     local crypto endpt.: 172.21.114.123, remote crypto endpt.: 172.21.114.67
     path mtu 1500, media mtu 1500
     current outbound spi: 20890A6F
     inbound esp sas:
      spi: 0x257A1039(628756537)
        transform: esp-des esp-md5-hmac,
        in use settings ={Tunnel,}
        slot: 0, conn id: 26, crypto map: router-alice
        sa timing: remaining key lifetime (k/sec): (4607999/90)
        IV size: 8 bytes
        replay detection support: Y
     inbound ah sas:
     outbound esp sas:
      spi: 0x20890A6F(545852015)
        transform: esp-des esp-md5-hmac,
        in use settings ={Tunnel,}
        slot: 0, conn id: 27, crypto map: router-alice
        sa timing: remaining key lifetime (k/sec): (4607999/90)
        IV size: 8 bytes
        replay detection support: Y
     outbound ah sas:
interface: Tunnel0
   Crypto map tag: abc, local addr. 172.21.114.123
   local ident (addr/mask/prot/port): (172.21.114.123/255.255.255.255/0/0)
   remote ident (addr/mask/prot/port): (172.21.114.67/255.255.255.255/0/0)
   current_peer: 172.21.114.67
     PERMIT, flags={origin_is_acl,}
    #pkts encaps: 10, #pkts encrypt: 10, #pkts digest 10
    #pkts decaps: 10, #pkts decrypt: 10, #pkts verify 10
    #send errors 10, #recv errors 0
     local crypto endpt.: 172.21.114.123, remote crypto endpt.: 172.21.114.67
     path mtu 1500, media mtu 1500
     current outbound spi: 20890A6F
     inbound esp sas:
```

```
 spi: 0x257A1039(628756537)
   transform: esp-des esp-md5-hmac,
   in use settings ={Tunnel,}
   slot: 0, conn id: 26, crypto map: router-alice
   sa timing: remaining key lifetime (k/sec): (4607999/90)
   IV size: 8 bytes
   replay detection support: Y
inbound ah sas:
outbound esp sas:
 spi: 0x20890A6F(545852015)
   transform: esp-des esp-md5-hmac,
   in use settings ={Tunnel,}
   slot: 0, conn id: 27, crypto map: router-alice
   sa timing: remaining key lifetime (k/sec): (4607999/90)
   IV size: 8 bytes
   replay detection support: Y
outbound ah sas:
```

Displays information about IPsec security associations.

Troubleshooting Tips

To verify that Cisco IOS software has recognized VAM, enter the **show diag** command and check the output. For example, when the router has the VAM in slot 1, the following output appears:

```
Router# show diag 1
    Slot 1:
         VAM Encryption/Compression engine. Port adapter
         Port adapter is analyzed
         Port adapter insertion time 00:04:45 ago
         EEPROM contents at hardware discovery:
         Hardware Revision        :1.0
         PCB Serial Number        :15485660
         Part Number              :73-5953-04
         Board Revision           :
         RMA Test History         :00
         RMA Number               :0-0-0-0
         RMA History              :00
         Deviation Number         :0-0
         Product Number           :CLEO
         Top Assy. Part Number    :800-10496-04
         CLEI Code                :
         EEPROM format version 4
         EEPROM contents (hex):
           0x00:04 FF 40 02 8A 41 01 00 C1 8B 31 35 34 38 35 36
           0x10:36 30 00 00 00 82 49 17 41 04 42 FF FF 03 00 81
           0x20:00 00 00 00 04 00 80 00 00 00 00 CB 94 43 4C 45
           0x30:4F 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
           0x40:20 C0 46 03 20 00 29 00 04 C6 8A FF FF FF FF FF
           0x50:FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
           0x60:FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
           0x70:FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
```

To see if the VAM is currently processing crypto packets, enter the **show pas vam interface** command. The following is sample output:

```
    Router# show pas vam interface
```

```
Interface VAM 1/1 :
     ds:0x632770C8        idb:0x62813728
     Statistics of packets and bytes that through this interface:
      18 packets in                      18 packets out
    2268 bytes in                      2268 bytes out
       0 paks/sec in                      0 paks/sec out
       0 Kbits/sec in                     0 Kbits/sec out
      83 commands out                    83 commands acknowledged
     ppq_full_err   :0            ppq_rx_err       :0
     cmdq_full_err  :0            cmdq_rx_err      :0
     no_buffer      :0            fallback         :0
     dst_overflow   :0            nr_overflow      :0
     sess_expired   :0            pkt_fragmented   :0
     out_of_mem     :0            access_denied   :0
     invalid_fc     :0            invalid_param    :0
     invalid_handle :0            output_overrun   :0
     input_underrun :0            input_overrun    :0
     key_invalid    :0            packet_invalid   :0
     decrypt_failed :0            verify_failed    :0
     attr_invalid   :0            attr_val_invalid :0
     attr_missing   :0            obj_not_wrap     :0
     bad_imp_hash   :0            cant_fragment    :0
     out_of_handles :0            compr_cancelled  :0
     rng_st_fail    :0            other_errors     :0
     633 seconds since last clear of counters
```

When the VAM processes packets, the “packet in” and “packet out” counters change. Counter “packets out” represents the number of packets directed to the VAM. Counter “packets in” represents the number of packets received from the VAM.

Note In versions prior to Cisco IOS Release 12.2(5)T and Cisco IOS Release 12.1(10)E, upon reboot trap configurations are lost and need to be re-entered.

Monitoring and Maintaining the VPN Acceleration Module

Use the commands below to monitor and maintain the VPN Acceleration Module:

Command	Purpose
Router# **show pas isa interface**	Displays the ISA interface configuration.
Router# **show pas isa controller**	Displays the ISA controller configuration.
Router# **show pas vam interface**	Verifies the VAM is currently processing crypto packets.
Router# **show pas vam controller**	Displays the VAM controller configuration.
Router# **Show version**	Displays integrated service adapter as part of the interfaces.

Configuration Examples for VPN Acceleration

Example: Configuring IKE Policies

In the following example, two IKE policies are created, with policy 15 as the highest priority, policy 20 as the next priority, and the existing default priority as the lowest priority. It also creates a preshared key to be used with policy 20 with the remote peer whose IP address is 192.168.224.33.

```
crypto isakmp policy 15
 encryption 3des
 hash md5
 authentication rsa-sig
 group 2
 lifetime 5000
crypto isakmp policy 20
 authentication pre-share
 lifetime 10000
crypto isakmp key 1234567890 address 192.168.224.33
```

Example: Configuring IPsec Configuration

The following example shows a minimal IPsec configuration where the security associations will be established via IKE:

An IPsec access list defines which traffic to protect:

```
access-list 101 permit ip 10.0.0.0 0.0.0.255 10.2.2.0 0.0.0.255
```

A transform set defines how the traffic will be protected. In this example, transform set "myset1" uses DES encryption and SHA for data packet authentication:

```
crypto ipsec transform-set myset1 esp-des esp-sha
```

Another transform set example is "myset2," which uses Triple DES encryption and MD5 (HMAC variant) for data packet authentication:

```
crypto ipsec transform-set myset2 esp-3des esp-md5-hmac
```

A crypto map joins together the IPsec access list and transform set and specifies where the protected traffic is sent (the remote IPsec peer):

```
crypto map toRemoteSite 10 ipsec-isakmp
 match address 101
 set transform-set myset2
 set peer 10.2.2.5
```

The crypto map is applied to an interface:

```
interface Serial0
 ip address 10.0.0.2
 crypto map toRemoteSite
```

Note In this example, IKE must be enabled.

Additional References for VPN Acceleration Module

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands 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 IPsec*
IKE configuration	*Configuring IKE*
VAM installation and configuration tasks	*VAM Installation and Configuration Guide*

Standards and RFCs

Standard/RFC	Title
RFC 2393	*IP Payload Compression Protocol (IPComp)*
RFC 2395	*IP Payload Compression Using LZS*
RFCs 2401 to 2411	*IPsec—IKE*
RFC 2451	*The ESP CBC-Mode Cipher Algorithms*
IPSec/IKE: RFCs 2401-2411, 2451	

MIBs

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

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 VPN Acceleration Module

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 8: Feature Information for VPN Acceleration Module

Feature Name	Releases	Feature Information
VPN Acceleration Module (VAM)	12.1(9)E 12.1(14)E 12.2(9)YE 12.2(13)T 12.2(15)T 12.3(1)Mainline 12.2(14)SU	VPN Acceleration Module (VAM) supports Data Encryption Standard (DES) or Triple DES (3DES) IPsec encryption at a rate greater than full-duplex DS-3 line rate (up to 145 Mbps) for site-to-site VPNs such as intranets and extranets. VAM also supports up to 5000 encrypted tunnels for mixed VPN environments that have both site-to-site and remote access VPN requirements. VAM integrates hardware-assisted Rivest, Shamir, and Adelman (RSA) and IP Payload Compression Protocol (IPPCP) layer 3 compression to accelerate RSA processing, thereby enhancing tunnel setup and improving overall VPN initialization. In environments where bandwidth is costly, VAM provides hardware-based IPPCP Lempel-Ziv-Stac (LZS) processing to compress network traffic before it is encrypted and sent over pay-per-byte WAN connections. In 12.1(9)E, this feature was introduced on the Cisco 7200 series routers on NPE-225, NPE-400, and NSE-1. In 12.1(14)E, this feature was integrated into Cisco IOS Release 12.1(14)E and support for dual VAMs[2] on the Cisco 7200 series with NPE-G1 was added. In 12.2(9)YE, support for this feature was added to the Cisco 7401ASR router[3]. The following commands were introduced or modified:**crypto engine sw ipsec, show pas vam controller, show pas vam interface.**

[2] Support for dual VAMs is available on a Cisco 7200 series router with NPE-G1 on Cisco IOS Release 12.2(15)T, 12.1(14)E, and 12.3 Mainline only.

[3] The Cisco 7401ASR router is no longer sold.

Glossary

anti-replay—Security service where the receiver can reject old or duplicate packets to protect itself against replay attacks. IPsec provides this optional service by use of a sequence number combined with the use of data authentication. Cisco IOS XE IPsec provides this service whenever it provides the data authentication service, except for manually established SAs (that is, SAs established by configuration and not by IKE).

data authentication—Verification of the integrity and origin of the data. Data authentication can refer either to integrity alone or to both of these concepts (although data origin authentication is dependent upon data integrity).

data confidentiality—Security service in which the protected data cannot be observed.

data flow—Grouping of traffic, identified by a combination of source address or mask, destination address or mask, IP next protocol field, and source and destination ports, where the protocol and port fields can have the values of **any**. IPsec protection is applied to data flows.

IKE—Internet Key Exchange. IKE establishes a shared security policy and authenticates keys for services (such as IPSec) that require keys. Before any IPSec traffic can be passed, each router/firewall/host must verify the identity of its peer. This can be done by manually entering preshared keys into both hosts or by a CA service.

IPsec—IP Security. A framework of open standards that provides data confidentiality, data integrity, and data authentication between participating peers. IPSec provides these security services at the IP layer. IPSec uses IKE to handle the negotiation of protocols and algorithms based on local policy and to generate the encryption and authentication keys to be used by IPSec. IPSec can protect one or more data flows between a pair of hosts, between a pair of security gateways, or between a security gateway and a host.

peer—In the context of this module, a “peer” is a router or other device that participates in IPsec.

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

SA—security association. Description of how two or more entities use security services in the context of a particular security protocol (AH or ESP) to communicate securely on behalf of a particular data flow. The transform and the shared secret keys are used for protecting the traffic.

SPI—security parameter index. A number which, together with a destination IP address and security protocol, uniquely identifies a particular security association. Without IKE, the SPI is manually specified for each security association.

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

tunnel—In the context of this module, “tunnel” is a secure communication path between two peers, such as two routers. It does not refer to using IPsec in tunnel mode.

CHAPTER 9

Option to Disable Hardware Crypto EngineFailover to Software Crypto Engine

The Option to Disable Hardware Crypto Engine Failover to Software Crypto Engine feature gives you the option of configurirng your router so that failover to the software crypto engine does not occur even if the hardware crypto engine fails.

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.

Feature History for Option to Disable Hardware Crypto Engine Failover to Software Crypto Engine

Release	Modification
12.3(14)T	This feature was introduced.

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.

Prerequisites for Option to Disable Hardware Crypto Engine Failover to Software Crypto Engine

- You must have the Cisco IOS IP Security (IPSec) framework configured on your network.

Information About Option to Disable Hardware Crypto Engine Failover to Software Crypto Engine

Hardware Crypto Engine Failover to the Software Crypto Engine Overview

Cisco IOS IPSec traffic can be supported both by a hardware encryption engine and by a software crypto engine (that is, by the main CPU, which is running a software encryption algorithm). If the hardware encryption engine fails, the software on the main CPU attempts to perform the IPSec functions. However, the main CPU software routines have only a small percentage of bandwidth compared with those of the hardware encryption engine. If a sufficient amount of traffic is being handled by the hardware engine, it is possible that on failover, the main CPU may try to handle more traffic than it can, causing the router to fail.

Option to Disable Hardware Crypto Engine Failover

The Option to Disable Hardware Crypto Engine Failover to Software Crypto Engine feature allows you to configure your router so that the hardware crypto engine does not automatically fail over to the software crypto engine.

For situations in which you prefer that the software routines on the main CPU handle the hardware crypto engine failover, the default is that failover does occur.

How to Configure Option to Disable Hardware Crypto Engine Failover to Software Crypto Engine

Disabling Hardware Crypto Engine Failover to the Software Crypto Engine

To disable hardware crypto engine failover to the software crypto engine, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no crypto engine software ipsec**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**no crypto engine software ipsec** **Example:** `Router (config)# no crypto engine software ipsec`	Disables hardware crypto engine failover to the software crypto engine. • To reenable failover, use the **crypto engine software ipsec** form of this command.

Configuration Examples for Option to Disable Hardware Crypto Engine Failover to Software Crypto Engine

Disabled Hardware Crypto Engine Failover Example

The following example shows that hardware crypto engine failover to the software crypto engine has been disabled:

```
version 12.4
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname VPN-Gateway1
!
boot-start-marker
boot-end-marker
!
!
clock timezone EST 0
no aaa new-model
ip subnet-zero
!
!
ip audit po max-events 100
no ftp-server write-enable
!
!
no crypto engine software ipsec
!
crypto isakmp policy 10
 encr aes
 authentication pre-share
 group 14
crypto isakmp key cisco123 address 209.165.201.2!
!
crypto ipsec transform-set basic esp-aes esp-sha-hmac
crypto map mymap 10 ipsec-isakmp
 set peer 209.165.201.2
 set transform-set basic
 match address 101
!
!
interface Ethernet0/0
 ip address 192.168.1.1 255.255.255.0
!
interface Serial1/0
 ip address 209.165.200.2 255.255.255.252
 serial restart-delay 0
 crypto map mymap!
ip classless
ip route 0.0.0.0 0.0.0.0 209.165.200.1
no ip http server
no ip http secure-server
!
!
access-list 101 permit ip 192.168.1.0 0.0.0.255 172.16.2.0 0.0.0.255 access-list 101 remark
 Crypto ACL!
!
!
control-plane
!
!
line con 0
line aux 0
line vty 0 4
!
!
end
```

Additional References

The following sections provide references to the Crypto Conditional Debug Support feature.

Related Documents

Related Topic	Document Title
IPSec and IKE configuration tasks	“ Internet Key Exchange for IPsec VPNs “ module in the *Cisco IOS XE Security Configuration Guide: Secure Connectivity*
IPSec and IKE commands	*Cisco IOS Security Command Reference*

Standards

Standards	Title
None	--

MIBs

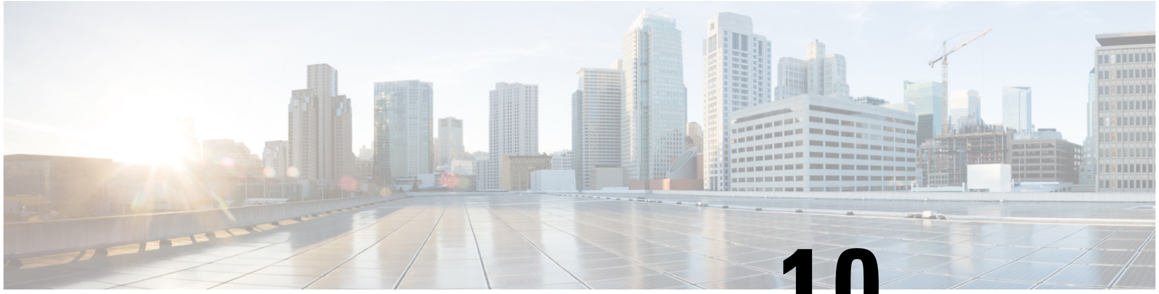
MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco IOS XE releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
None	--

Technical Assistance

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



CHAPTER 10

RFC 430x IPsec Support

The RFC 430x IPsec Support includes features—RFC 430x IPsec Support Phase 1 and RFC430x IPsec Support Phase 2—that implement Internet Key Exchange (IKE) and IPsec behavior as specified in RFC 4301.

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.

Information About RFC 430x IPsec Support

RFC 430x IPsec Support Phase 1

The RFC 430x IPsec Support Phase 1 feature implements Internet Key Exchange (IKE) and IPsec behavior as specified in RFC 4301.

RFC 4301 specifies the base architecture for IPsec-compliant systems. RFC 4301 describes how to provide a set of security services for traffic at the IP layer, in both the IPv4 and IPv6 environments. The RFC 430x IPsec Support Phase 1 feature provides support for the following RFC 4301 implementations on Cisco IOS software.

- **Security association (SA) lifetime**—The lifetime of a security association between IPsec and Internet Key Exchange (IKE) or Internet Key Exchange Version 2 (IKEv2) must not exceed the lifetime of the authentication certificate.

- **OPAQUE selectors**—OPAQUE indicates that the corresponding selector field is not available for verification. When IKEv2 encounters an OPAQUE selector, IKEv2 skips, does not process the OPAQUE selector, and moves to next selector for policy verification.
- **Explicit Congestion Notification (ECN) support**—ECN is propagated when decrypting an IPsec packet thereby ensuring the packet source and destination are aware of congestion that occurs within the network.
- **Fragment processing**—Peers must not send Initial and noninitial fragments in the same tunnel. There must be a separate tunnel mode SA for carrying initial and noninitial fragments and separate tunnel mode SA for noninitial fragments. IPsec peers must support discarding of packets and stateful fragment checking to accommodate bypass traffic.
- **Do not fragment-(DF) bit processing**—DF-bit processing must be set on a per SA basis.
- **Dummy packet generation support**—It should be possible to send dummy packets via IPsec SA to encapsulate the packets when traffic is flowing via IPsec SA tunnel.

RFC 430x IPsec Support Phase 2

The RFC 430x IPsec Support Phase 2 feature provides support for the RFC 4301 implementation of encryption and decryption of Internet Control Message Protocol (ICMP) packets on Cisco IOS software.

ICMP error messages are sent when an ICMP error occurs. For example, when a host is not reachable, the intermediate device sends a message to the originator of the ICMP request that the host is not reachable. When an ICMP error message reaches an IPsec encryption policy, it may not be classified to match an existing SA. So, the packets are classified based on the data inside the ICMP error message. This data contains the source and destination address of the original ICMP message. If an SA is found based on the address in the ICMP error message, the SA is used. If there is no SA, an SA is created if the policy permits. For decryption, the post decrypt check is performed on the data inside the ICMP error message if a valid SA is not found.

The encryption and decryption of ICMP error messages can be verified through the encrypt and decrypt counters displayed in the output of the **show crypto ipsec sa** command.

Use the **debug crypto ipsec** and **debug crypto ipsec error** commands to view ICMP error message classification.

How to Configure RFC 430x IPsec Support

Configuring RFC 430x IPsec Support Globally

Perform this task to configure the RFC 4301 implementations globally.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto ipsec security-association dummy** {**pps** *rate* | **seconds** *seconds*}
4. **crypto ipsec security-association ecn** {**discard** | **propogate**}
5. **exit**

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 security-association dummy** {**pps** *rate* | **seconds** *seconds*}
	Example:
	`Device(config)# crypto ipsec security-association dummy seconds 5`
		Enables the generation and transmission of dummy packets in an IPsec traffic flow.

Step 4	**crypto ipsec security-association ecn** {**discard** | **propogate**}
	Example:
	`Device(config)# crypto ipsec security-association ecn discard`
		Enables the Explicit Congestion Notification (ECN) settings in an IPsec traffic flow.

Step 5	**exit** **Example:** `Device(config-crypto-map)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring RFC 430x IPsec Support Per Crypto Map

Perform this task to configure the RFC 4301 implementations per crypto map.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto map** *map-name seq-num* **ipsec-isakmp**
4. **set ipsec security-association dfbit** {**clear** | **copy** | **set**}
5. **set ipsec security-association dummy** {**pps** *rate* | **seconds** *seconds*}
6. **set ipsec security-association ecn** {**discard** | **propogate**}
7. **end**
8. **show crypto map ipsec sa**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto map** *map-name seq-num* **ipsec-isakmp** **Example:** `Device(config)# crypto map cmap 1 ipsec-isakmp`	Specifies the crypto map entry to be created or modified and enters crypto map configuration mode.

Step 4	**set ipsec security-association dfbit** {**clear** | **copy** | **set**}
	Example:
	`Device(config-crypto-map)# set ipsec security-association dfbit set`
		Enables do not fragment (DF)-bit processing per security association (SA) for an IPsec traffic flow in a crypto map.

Step 5	**set ipsec security-association dummy** {**pps** *rate* | **seconds** *seconds*}
	Example:
	`Device(config-crypto-map)# set ipsec security-association dummy seconds 5`
		Enables the generation and transmission of dummy packets for an IPsec traffic flow in a crypto map.

Step 6	**set ipsec security-association ecn** {**discard** | **propogate**}
	Example:
	`Device(config-crypto-map)# set ipsec security-association ecn propogate`
		Enables the Explicit Congestion Notification (ECN) settings per SA for an IPsec traffic flow in a crypto map.

Step 7	**end**
	Example:
	`Device(config-crypto-map)# end`
		Exits crypto map configuration mode and returns to privileged EXEC mode.

Step 8	**show crypto map ipsec sa** **Example:** `Device# show crypto map ipsec sa`	Displays the settings used by IPsec SAs.

Example

The following is sample output from the **show crypto map ipsec sa** command:

```
Device# show crypto map ipsec sa

interface: Tunnel0
 Crypto map tag: Tunnel0-head-0, local addr 3FFE:2002::32F7:DFF:FE54:7FD1
protected vrf: (none)
local ident (addr/mask/prot/port): (3FFE:2002::32F7:DFF:FE54:7FD1/128/47/0)
remote ident (addr/mask/prot/port): (3FFE:2002::C671:FEFF:FE88:EB82/128/47/0)
current_peer 3FFE:2002::C671:FEFF:FE88:EB82 port 500
 PERMIT, flags={origin_is_acl,}
```

```
#pkts encaps: 36, #pkts encrypt: 36, #pkts digest: 36
#pkts decaps: 28, #pkts decrypt: 28, #pkts verify: 28
#pkts compressed: 0, #pkts decompressed: 0
#pkts not compressed: 0, #pkts compr. failed: 0
#pkts not decompressed: 0, #pkts decompress failed: 0
#send errors 0, #recv errors 0
#send dummy packets 852600, #recv dummy packets 424905

local crypto endpt.: 3FFE:2002::32F7:DFF:FE54:7FD1,
remote crypto endpt.: 3FFE:2002::C671:FEFF:FE88:EB82
plaintext mtu 1430, path mtu 1500, ipv6 mtu 1500, ipv6 mtu idb GigabitEthernet0/0/1
current outbound spi: 0xE963D1EC(3915633132)
PFS (Y/N): N, DH group: none
Dummy packet: Initializing

inbound esp sas:
spi: 0xF4E01B9A(4108327834)
 transform: esp-3des esp-md5-hmac,
 in use settings ={Tunnel, }
 conn id: 2053, flow_id: ESG:53, sibling_flags FFFFFFFF80000049, crypto map: Tunnel0-head-0

 sa timing: remaining key lifetime (k/sec): (4608000/2343)
 IV size: 8 bytes
 replay detection support: Y
 Status: ACTIVE(ACTIVE)

inbound ah sas:

inbound pcp sas:

outbound esp sas:
spi: 0xE963D1EC(3915633132)
 transform: esp-3des esp-md5-hmac,
 in use settings ={Tunnel, }
 conn id: 2054, flow_id: ESG:54, sibling_flags FFFFFFFF80000049, crypto map: Tunnel0-head-0

 sa timing: remaining key lifetime (k/sec): (4608000/2343)
 IV size: 8 bytes
 replay detection support: Y
 Status: ACTIVE(ACTIVE)

outbound ah sas:

outbound pcp sas:
```

Configuration Examples for RFC 430x IPsec Support

Example: Configuring RFC 430x IPsec Support Globally

The following examples shows how to configure RFC 430x IPsec Support globally:

```
Device> enable
Device# configure terminal
Device(config)# crypto ipsec security-association dummy seconds 15
Device(config)# crypto ipsec security-association ecn propogate
Device(config-crypto-map)# exit
```

Example: Configuring RFC 430x IPsec Support Per Crypto Map

The following examples shows how to configure RFC 430x IPsec Support per crypto map:

```
Device> enable
Device# configure terminal
Device(config)# crypto map cmap 1 ipsec-isakmp
Device(config-crypto-map)# set security-association copy
Device(config-crypto-map)# set security-association dummy seconds 15
Device(config-crypto-map)# set security-association ecn propogate
Device(config-crypto-map)# end
Device# show crypto map ipsec sa

interface: Tunnel0
 Crypto map tag: Tunnel0-head-0, local addr 3FFE:2002::32F7:DFF:FE54:7FD1
protected vrf: (none)
local ident (addr/mask/prot/port): (3FFE:2002::32F7:DFF:FE54:7FD1/128/47/0)
remote ident (addr/mask/prot/port): (3FFE:2002::C671:FEFF:FE88:EB82/128/47/0)
current_peer 3FFE:2002::C671:FEFF:FE88:EB82 port 500
 PERMIT, flags={origin_is_acl,}
#pkts encaps: 36, #pkts encrypt: 36, #pkts digest: 36
#pkts decaps: 28, #pkts decrypt: 28, #pkts verify: 28
#pkts compressed: 0, #pkts decompressed: 0
#pkts not compressed: 0, #pkts compr. failed: 0
#pkts not decompressed: 0, #pkts decompress failed: 0
#send errors 0, #recv errors 0
#send dummy packets 852600, #recv dummy packets 424905

local crypto endpt.: 3FFE:2002::32F7:DFF:FE54:7FD1,
remote crypto endpt.: 3FFE:2002::C671:FEFF:FE88:EB82
plaintext mtu 1430, path mtu 1500, ipv6 mtu 1500, ipv6 mtu idb GigabitEthernet0/0/1
current outbound spi: 0xE963D1EC(3915633132)
PFS (Y/N): N, DH group: none
Dummy packet: Initializing

inbound esp sas:
spi: 0xF4E01B9A(4108327834)
 transform: esp-3des esp-md5-hmac,
 in use settings ={Tunnel, }
 conn id: 2053, flow_id: ESG:53, sibling_flags FFFFFFFF80000049, crypto map: Tunnel0-head-0

 sa timing: remaining key lifetime (k/sec): (4608000/2343)
 IV size: 8 bytes
 replay detection support: Y
 Status: ACTIVE(ACTIVE)

inbound ah sas:

inbound pcp sas:

outbound esp sas:
spi: 0xE963D1EC(3915633132)
 transform: esp-3des esp-md5-hmac,
 in use settings ={Tunnel, }
 conn id: 2054, flow_id: ESG:54, sibling_flags FFFFFFFF80000049, crypto map: Tunnel0-head-0

 sa timing: remaining key lifetime (k/sec): (4608000/2343)
 IV size: 8 bytes
 replay detection support: Y
 Status: ACTIVE(ACTIVE)

outbound ah sas:
```

```
outbound pcp sas:
```

Additional References for RFC 430x IPsec Support

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
IKEv2 configuration	Configuring Internet Key Exchange Version 2 (IKEv2) and FlexVPN Site-to-Site
Recommended cryptographic algorithms	Next Generation Encryption

Standards and RFCs

Standard/RFC	Title
RFC 4301	*Security Architecture for the Internet Protocol*

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

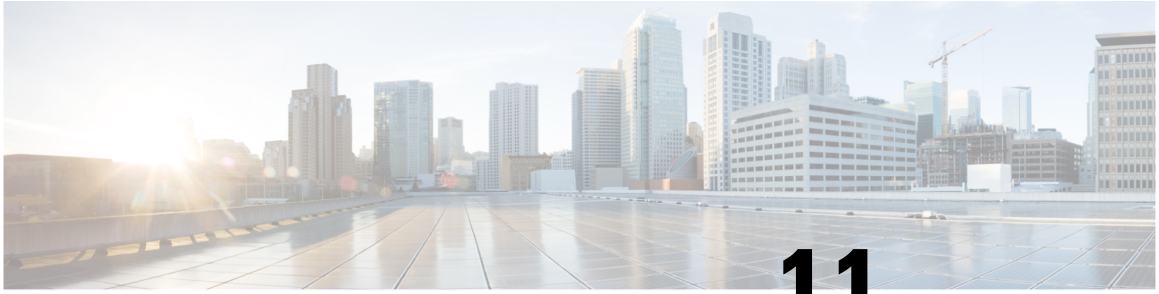
Feature Information for RFC 430x IPsec Support

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 9: Feature Information for RFC430x IPsec Support

Feature Name	Releases	Feature Information
RFC430x IPsec Support Phase 1	15.4(2)T	The RFC 430x IPsec Support Phase 1 feature implements Internet Key Exchange (IKE) and IPsec behavior as specified in RFC 4301. The following commands were introduced or modified: **crypto ipsec security-association dummy, crypto ipsec security-association ecn, set ipsec security-association dfbit, set ipsec security-association dummy, set ipsec security-association ecn, show crypto map ipsec sa.**
RFC430x IPsec Support Phase 2	15.5(2)T	The RFC 430x IPsec Support Phase 2 feature provides support for the RFC 4301 implementation of encryption and decryption of Internet Control Message Protocol (ICMP) packets on Cisco IOS software. No commands were modified or updated for this feature.



CHAPTER 11

Session Initiation Protocol Triggered VPN

Session Initiation Protocol Triggered VPN (SIP-Triggered VPN or VPN-SIP) is a service offered by service providers where a VPN is set up using Session Initiation Protocol (SIP) for on-demand media or application sharing between peers. The VPN-SIP feature defines the process in which two SIP user agents resolve each other's IP addresses, exchange the fingerprints of their self-signed certificates, third-party certificates, or pre-shared key securely, and agree to establish an IPsec-based VPN.

Service providers offer the VPN-SIP service to their customers that have SIP-based services such as bank ATMs or branches. This VPN-SIP service replaces an ISDN connection for backup network functionality. If the primary broadband service link goes down, these bank ATMs or branches connect to their central headend or data centres through the VPN-SIP service.

The SIP server of the service provider, which coordinates the VPN-SIP service, is also used for billing of the service based on the time the service is used.

Information about VPN-SIP

Components for VPN-SIP Solution

VPN-SIP uses IPSec Static Virtual Tunnel Interface (SVTI). IPSec SVTI stays in active (UP) state even when there is no IPSec security association (SA) established between the tunnel interface and the SVTI peer.

The following are three components for the VPN-SIP Solution:

- SIP
- VPN-SIP

- Crypto (IP Security (IPsec), Internet Key Exchange (IKE), Tunnel Protection (TP), Public Key Infrastructure (PKI) modules within crypto)

Sesssion Initiation Protocol

SIP is used as a name resolution mechanism to initiate an IKE session. VPN-SIP uses SIP service to establish a VPN connection to a home or a small business router that does not have a fixed IP address. This connection is achieved using self-signed certificates or pre-shared keys. SIP negotiates the use of IKE for media sessions in the Session Description Protocol (SDP) offer-and-answer model.

SIP is statically configured. One tunnel interface must be configured for each remote SIP number.

SIP also provides billing capabilities for service providers to charge customers based on the SIP number, for using the VPN-SIP service. Billing based on SIP numbers happens in the service provider network and is independent of the end devices like Cisco VPN-SIP routers.

VPN-SIP Solution

VPN-SIP is the central block that coordinates between SIP and Crypto modules, and provides an abstraction between them.

When traffic destined to a remote network behind a SIP number is routed to the tunnel interface, the IPSec control plane gets a trigger from packet switching path as there is no IPSEC SA configured to that peer. IPsec control plane passes the trigger to VPN-SIP as the tunnel is configured for VPN-SIP.

Note Static routes for remote networks for that SIP number must be configured to point to that tunnel interface.

When the VPN-SIP service is triggered, SIP sets up the call with a SIP phone number pair. SIP also passes incoming call details to the VPN-SIP and negotiates IKE media sessions using local address and fingerprint information of the local self-signed certificate or pre-shared key. SIP also passes remote address and fingerprint information to VPN-SIP.

The VPN-SIP service listens to tunnel status updates and invokes SIP to tear down the SIP session. The VPN-SIP service also provides a means to display current and active sessions.

Feature at a glance

The following steps summarize how the VPN-SIP feature works:

- IP SLA monitors the primary link using route tracking. When the primary link fails IP SLA detects this failure.

- Once the primary path fails, IP SLA switches the default route to the higher metric route that is configured on the router.

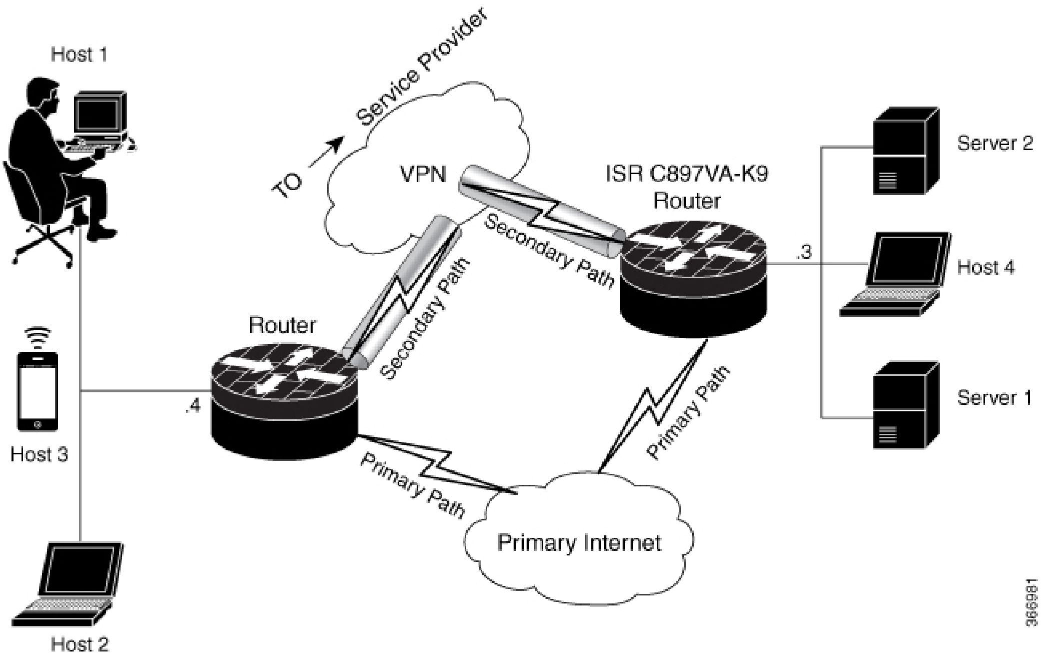
- When relevant traffic tries to flow using the secondary link, SIP sends an invite message to the SIP server to obtain the VPN peer information.

- The router receives the VPN peer information (IP address, local and remote SIP numbers, IKE port, and finger print) and it establishes VPN-SIP tunnel.

- When the primary path comes back up, IP SLA detects the primary path and the route falls back to the original path. When the idle timer expires, IPSec is torn down and a SIP call is disconnected.

Following is the topology for the VPN-SIP solution:

Figure 9: VPN-SIP Topology



SIP Call Flow

The SIP call flow is divided into initiation at the local peer and call receipt at the remote peer.

At SIP Call Intitiation

When packets are routed to an SVTI interface in data plane, the SIP call must be placed to the peer SIP number to resolve its address, so that VPN tunnel can be brought up.

- When local auth-type is PSK, IKEv2 finds the matching key for a peer SIP number. The IKEv2 keyring must be configured with id_key_id type (string) as SIP number for each SIP peer. IKEv2 computes the fingerprint of the looked-up key and passes it to VPN-SIP.
- When local auth-type is a self-signed certificate or an third-party certificate, IKEv2 computes the fingerprint of the local certificate configured under the IKEv2 profile and passes it to the VPN-SIP

The VPN-SIP module interacts with SIP to setup SIP call to the peer. When the call is successful, VPN-SIP sets the tunnel destination of SVTI to the resolved IP address, requesting SVTI to initiate the VPN tunnel.

Note When a wildcard key is required, use the authentication local pre-share key command and the authentication remote pre-share key command in IKEv2 profile.

When SIP call is received at the remote peer

When a SIP call is received from a peer, following interactions occur between various crypto modules:

- The Tunnel Protection helps VPN-SIP module to set tunnel destination address.
- IKEv2 returns local auth-type (PSK or PKI) and local fingerprint to the VPN-SIP module. When local auth-type is PSK, IKEv2 finds a matching key for a corresponding SIP number.

Note IKEv2 only knows peer by its SIP number.

During the SIP call negotiation between peers, each peer must select a unique local IKEv2 port number to be exchanged over the SDP. To support different port numbers for each session, the VPN-SIP module programmatically configures IP Port Address Translation (PAT) to translate between IKEv2 port (4500) and the port number exchanged over SDP. For the translation to work IP NAT must be configured on secondary link and the loopback interface configured as the VPN-SIP tunnel source. The lifetime of the translation is limited to the lifetime of the VPN-SIP session.

SDP Offer and Answer

Following is the sample for SDP offer and answer that is negotiated in the SIP call as defined in RFC 6193:

```
offer SDP
      ...
      m=application 50001 udp ike-esp-udpencap
      c=IN IP4 10.6.6.49
      a=ike-setup:active
      a=fingerprint:SHA-1 \
      b=AS:512
      4A:AD:B9:B1:3F:82:18:3B:54:02:12:DF:3E:5D:49:6B:19:E5:7C:AB
      ...

   answer SDP
      ...
      m=application 50002 udp ike-esp-udpencap
      c=IN IP4 10.6.6.50
      a=ike-setup:passive
      a=fingerprint:SHA-1 \
      b=AS:512
      D2:9F:6F:1E:CD:D3:09:E8:70:65:1A:51:7C:9D:30:4F:21:E4:4A:8E
```

As part of the SDP negotiation, both peers negotiate the maximum bandwidth rate for the VPN-SIP session using the b=AS :number SDP attribute. if the peers mention different bandwidth numbers in their SDP, both of them should honor the minimum value as the maximum bandwidth. If b=AS :number SDP attribute is missing in the offer or answer, the SIP call is not successfully set up.

The negotiated maximum bandwidth is applied on the SVTI tunnel interface through the programmatically configured QoS policy in the output direction. The programmatically configured QoS policy is not applied and session fails, if there is a pre-existing statically configured policy.

Once SIP call is complete and address of the peer is resolved, VPN-SIP sets tunnel destination of SVTI and sends a request to initiate tunnel.

IKEv2 Negotiation

Following is the process for IKEv2 Security Session (SA) negotiation:

- Before starting the session, IKEv2 checks with VPN-SIP if the session is a VPN-SIP session.
- If it's a VPN-SIP session and local auth-type is PSK, IKEv2 looks up the PSK key pair using SIP number of the peer instead of IP address of the peer.
- For validating self-signed certificate, IKEv2 checks if the certificate is self-signed and validates the certificate.
    - In addition to existing AUTH payload validation as part of IKEv2 protocol, IKEv2 calculates hash of the received certificate or looked-up PSK and compares with the fingerprint from SIP negotiation that IKEv2 queries from VPN-SIP module. Only if the fingerprint matches, IKEv2 considers authentication of peer is valid. If not, IKEv2 declares that peer has failed to authenticate and fails the VPN session.

VPN-SIP solution depends on IPSEC idle timer to detect that traffic is no longer routed over the backup VPN. The idle-time configuration under the IPSec Profile is mandatory for session to be disconnected when there is no traffic. 120 seconds is the recommended time.

VPN-SIP and SIP coordinate to tear down SIP call.

When IPsec idle time expires the VPN-SIP module informs the IKEv2 to bring down the IPsec tunnel. VPN-SIP requests the SIP module to disconnect the SIP call, without waiting for confirmation from the IKEv2.

When SIP call disconnect is received from the peer, VPN-SIP module informs the IKEv2 to bring down the IPsec tunnel, and acknowledges to SIP to tear down the SIP call.

Supported Platforms

The VPN-SIP feature is supported on the following platforms:

Cisco Integrated Services Router Generation 2 (ISR G2) 800 Series platforms

- C819H-K9
- C841M-8X-JAIS/K9
- C881-K9
- C891FJ-K9
- C891-24X/K9
- C892FSP-K9
- C897VA-K9
- C899G-LTE-JP-K9
- C899G-LTE-LA-K9
- C819G-LTE-LA-K9
- C819GW-LTE-LA-QK9

Prerequisites for VPN-SIP

- Cisco 4000 Series Integrated Services Routers and Cisco 1000 Integrated Services Routers must have Cisco IOS XE 16.8.1 software installed.
- Security K9 license must be enabled on the router.
- The routers must have a minimum memory of 1 GB.
- For the SIP register request of the SIP User Agent to succeed, the SIP registrar must be available to the VPN-SIP routers.
- The DHCP server must support option 120 and 125 to obtain the SIP server address, which is needed for registration and establishing the SIP session.
- Proper routing configurations must be completed to ensure backup WAN path is used when primary path is down.
- Maximum Transmission Unit (MTU) of the tunnel interface must be less than the MTU of the secondary WAN interface.
- When self-signed or third-party certificates are used for IKEv2 authentication, configure IKEv2 fragmentation on the VPN-SIP router to avoid fragmentation at the IP layer.
- NAT SIP ALG must be disabled.
- Caller ID notification service must be configured in the network.

Restrictions for VPN-SIP

- VPN-SIP and CUBE/SIP gateway cannot be configured on the same device. When CUBE license is active on the device, only CUBE will be functional.
- Only IPv4 is supported for transport and media (IPv4 transport for SIP registration, SIP signaling, and IPv4 packets encrypted over IPv4 transport).
- SIP signalling with peer devices behind NAT is not supported (ICE and STUN are not supported.
- SIP negotiation is supported only in global VRF.
- Remote-access VPN features like private address assignment, configuration mode exchange (CP payloads), routes exchange, are not supported.
- Routing protocols over the VPN-SIP session are not supported.
- Only Rivest-Shamir-Addleman (RSA) server self-signed certificates are supported.
- Pre-shared key lookup functionality using authentication, authorization, and accounting (AAA) is not supported.
- The IPSec idle timer is configured per IPSec profile using the ipsec-profile command. The idle time is the same for all VPN-SIP sessions that use a specific IPSec profile.

- Track objects that are used for IPSLA monitoring, have a maximum limit of 1000 objects in Cisco IOS software. When one track object is used to track one peer router, maximum number of VPN-SIP sessions that one IOS device can have is limited by the maximum number of track objects.
- Only one local SIP number is supported on Cisco IOS software.
- If there is a pre-existing statically configured policy, the programmatically configured QoS policy is not applied and session fails, . Remove any statically configured QoS policy on the SVTI interface.
- Cisco does not support the interoperability with VPN-SIP implementation of other vendors.

How to Configure VPN-SIP

Configuring VPN-SIP

The following steps describe the process of configuring VPN-SIP:

1. Configure the tunnel authentication using third party certificates, self-signed certificates, or pre-shared keys.

    1. Tunnel Authentication using Certificates

        Configure a trustpoint to obtain a certificate from a certification authority (CA) server that is located in the customer's network. This is required for tunnel authentication. Use the following configuration:

```
peer1(config)# crypto pki trustpoint CA
 enrollment url http://10.45.18.132/
 serial-number none
 subject-name CN=peer2
 revocation-check crl
 rsakeypair peer2

peer2(config)# crypto pki authenticate CA
Certificate has the following attributes:
       Fingerprint MD5: F38A9B4C 2D80490C F8E7581B BABE7CBD
      Fingerprint SHA1: 4907CC36 B1957258 5DFE23B2 649E7DDA 99BDB7C3
% Do you accept this certificate? [yes/no]: yes
Trustpoint CA certificate accepted.

peer2(config)#crypto pki enroll CA
%
% Start certificate enrollment ..
% Create a challenge password. You will need to verbally provide this
   password to the CA Administrator in order to revoke your certificate.
   For security reasons your password will not be saved in the configuration.
   Please make a note of it.
Password:
Re-enter password:
% The subject name in the certificate will include: CN=peer2
% The subject name in the certificate will include: peer2
% Include an IP address in the subject name? [no]:
Request certificate from CA? [yes/no]: yes
% Certificate request sent to Certificate Authority
% The 'show crypto pki certificate verbose CA' command will show the fingerprint.
Certificate map for Trustpoint
crypto pki certificate map data 1
issuer-name co cn = orange
```

2. Tunnel authentication using self-signed certificate

   Configure a PKI trust point to generate a self-signed certificate on the device, when authenticating using a self-signed certificate. Use the following configuration:

```
peer4(config)#crypto pki trustpoint Self
    enrollment selfsigned
    revocation-check none
    rsakeypair myRSA
    exit
crypto pki enroll Self

Do you want to continue generating a new Self Signed Certificate? [yes/no]: yes
% Include the router serial number in the subject name? [yes/no]: yes
% Include an IP address in the subject name? [no]: no
Generate Self Signed Router Certificate? [yes/no]: yes

Router Self Signed Certificate successfully created
```

3. Configure tunnel authentication using a pre-shared key

```
crypto ikev2 keyring keys
peer peer1
identity key-id 1234
pre-shared-key key123
```

2. 1. Configure IKEv2 Profile for Certificate

```
crypto ikev2 profile IPROF
match certificate data
identity local key-id 5678
authentication remote rsa-sig
authentication local rsa-sig
keyring local keys
pki trustpoint self
nat force-encap
```

   2. Configure an IKEv2 Profile for pre-shared keys

```
crypto ikev2 profile IPROF
match identity remote any
identity local key-id 5678
authentication remote pre-share
authentication local pre-share
keyring local keys
nat force-encap
```

Note To complete the IKEv2 SA configuration, the **nat force-encap** command must be configured on both peers. Since, UDP encapsulation is negotiated in SDP, IKEv2 must start and continue on port 4500.

3. Configure an IPsec profile

```
crypto ipsec profile IPROF
set security-association idle-time 2000
```

4. Configure a LAN side interface

Note The LAN side interface is connected to the ISRG2 interface on the router.

```
interface Vlan101
         ip address 10.3.3.3 255.255.255.0
         no shutdown
!
     interface GigabitEthernet2
         switchport access vlan 101
         no ip address
```

5. Configure a loopback interface

   The loopback interface is used as the source interface for the secondary VPN tunnel.

```
interface loopback 1
    ip address 10.11.1.1 255.0.0.0
    ip nat inside
```

6. Configure a secondary interface.

Note Make sure the secondary interface is configured to receive the IP address, SIP server address, and vendor specific information via DHCP.

```
interface GigabitEthernet8
     ip dhcp client request sip-server-address
     ip dhcp client request vendor-identifying-specific
     ip address dhcp
    ip nat outside
```

7. Configure the tunnel interface

```
interface Tunnel1
     ip address 10.3.2.1 255.255.255.255
     load-interval 30
     tunnel source Loopback1
     tunnel mode ipsec ipv4
     tunnel destination dynamic
     tunnel protection ipsec profile IPROF ikev2-profile IPROF
     vpn-sip local-number 5678 remote-number 1234 bandwidth 1000
```

   Use the **vpn-sip local-number** *local-number* **remote-number** *remote-number* **bandwidth** *bw-number* command to configure the sVTI interface for VPN-SIP. Bandwidth is the maximum data transmission rate that must be negotiated with this peer and the negotiated value is set on the tunnel interface. Allowed values are 64, 512, and 1000 kbps.

   Once an SVTI is configured for VPN-SIP, changes cannot be made to tunnel mode, tunnel destination, tunnel source, and tunnel protection. To change the mode, source, destination, or tunnel protection you must remove the VPN-SIP configuration from the SVTI interface.

8. Add static default routes

   Add a secondary route with a higher metric.

```
ip route 0.0.0.0 0.0.0.0 Tunnel0 track 1
ip route 0.0.0.0 0.0.0.0 Tunnel1 254
```

9. Configure IP SLA

```
ip sla 1
         icmp-echo 10.11.11.1
         threshold 500
         timeout 500
         frequency 2
        ip sla schedule 1 life forever start-time now
```

10. Configure route tracking

```
track 1 ip sla 1 reachability
```

11. Enable VPN-SIP

```
vpn-sip enable
vpn-sip local-number 5678 address ipv4 GigabitEthernet8
vpn-sip tunnel source Loopback1
vpn-sip logging
```

To configure VPN-SIP, you must configure local SIP number and local address. The **vpn-sip local-number** *SIP-number* **address ipv4** *WAN-interface-name* command configures the local SIP number that is used for SIP call and the associated IPv4 address.

Note Only IPv4 addresses can be configured. Crypto module does not support dual stack.

- Backup WAN interface address may change based on DHCP assignment.

When the primary WAN interface is functional, the destination of the VPN-SIP tunnel is set to the backup WAN interface, so that the tunnel interface is active. Destination is set to IP address of the peer that is learnt from SDP of SIP negotiation when traffic is routed to the tunnel interface. When primary WAN interface fails and the back routes are activated, packets are routed to the sVTI through backup.

Note We recommend that you use an unused non-routable address as the address of the loopback interface and do not configure this loopback interface for any other purpose. Once a loopback interface is configured, VPN-SIP listens to any updates to the interface and blocks them. The **vpn-sip logging** command enables the system logging of VPN-SIP module for events, such as session up, down, or failure.

Verifying VPN-SIP on a Local Router

Verifying Registration Status

```
Peer1# show vpn-sip registration-status
 SIP registration of local number 0388881001 : registered 10.6.6.50
```

Verifying SIP Registrar

```
Peer1#show vpn-sip sip registrar

Line          destination      expires(sec)  contact   transport     call-id
==========================================================================================
0388881001    example.com         2359        10.6.6.50  UDP
3176F988-9EAA11E7-8002AFA0-8EF41435
```

Verifying VPN-SIP Status

```
Peer1#show vpn-sip session detail
VPN-SIP session current status

Interface: Tunnel1
   Session status: SESSION_UP (I)
   Uptime        : 00:00:42
   Remote number : 0388881001 =====> This is the Remote Router's SIP number
   Local number  : 0388882001 =====> Local router's SIP number
   Remote address:port: 10.6.6.49:50002
   Local address:port : 10.6.6.50:50001
   Crypto conn handle: 0x8000017D
   SIP Handle        : 0x800000C7
   SIP callID        : 1554
   Configured/Negotiated bandwidth: 64/64 kbps
```

Verifying Crypto Session

```
Peer1# show crypto session detail
Crypto session current status
Code: C - IKE Configuration mode, D - Dead Peer Detection
K - Keepalives, N - NAT-traversal, T - cTCP encapsulation
X - IKE Extended Authentication, F - IKE Fragmentation
R - IKE Auto Reconnect, U - IKE Dynamic Route Update
S - SIP Vpn-sip

Interface: Tunnel1
Profile: IPROF
Uptime: 00:03:53
Session status: UP-ACTIVE
Peer: 10.6.6.49 port 4500 fvrf: (none) ivrf: (none)
      Phase1_id: 10.6.6.49
      Desc: (none)
  Session ID: 43
  IKEv2 SA: local 10.11.1.1/4500 remote 10.6.6.49/50002 Active
          Capabilities:S connid:1 lifetime:23:56:07 ====> Capabilities:S indicates this is
 a  SIP VPN_SIP Session
  IPSEC FLOW: permit ip 0.0.0.0/0.0.0.0 0.0.0.0/0.0.0.0
        Active SAs: 2, origin: crypto map
        Inbound:  #pkts dec'ed 6 drop 0 life (KB/Sec) 4222536/3366
        Outbound: #pkts enc'ed 4 drop 0 life (KB/Sec) 4222537/3366
```

Verifying IP NAT Translations

```
Peer1#sh ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
udp 2.2.2.2:4500       10.6.6.50:50001    10.6.6.49:50002    10.6.6.49:50002
```

Verifying DHCP SIP Configuration

```
Peer9#show vpn-sip sip dhcp
SIP DHCP Info

SIP-DHCP interface:  GigabitEthernet8

SIP server address:
Domain name:         dns:example.com
```

Verifying VPN-SIP on a Remote Router

Verifying VPN-SIP Registration Status on a Remote Router

```
Peer2# show vpn-sip registration-status
 SIP registration of local number 0388882001 : registered 10.6.6.49
```

Verifying VPN-SIP Registrar on a Remote Router

```
Peer2# show vpn-sip sip registrar
Line          destination      expires(sec)  contact    transport    call-id
==========================================================================================
0388882001    example.com    2478         10.6.6.49   UDP
E6F23809-9EAB11E7-80029279-40B97F59
```

Verifying VPN-SIP Session Details on a Remote Router

```
Peer2# show vpn-sip session detail
VPN-SIP session current status
Interface: Tunnel1
   Session status: SESSION_UP (R)
   Uptime        : 00:00:21
   Remote number : 0388882001 =====> This is the Peer1 Router's SIP number
   Local number  : 0388881001 =====> Local router's SIP number
   Remote address:port: 10.6.6.50:50001
   Local address:port : 10.6.6.49:50002
   Crypto conn handle: 0x8000017E
   SIP Handle        : 0x800000BE
   SIP callID        : 1556
   Configured/Negotiated bandwidth: 1000/64 kbps
```

Verifying Crypto Session Details on a Remote Router

```
Peer2 #show crypto session detail
Crypto session current status
Code: C - IKE Configuration mode, D - Dead Peer Detection
K - Keepalives, N - NAT-traversal, T - cTCP encapsulation
X - IKE Extended Authentication, F - IKE Fragmentation
R - IKE Auto Reconnect, U - IKE Dynamic Route Update
S - SIP VPN-SIP

Interface: Tunnel1
Profile: IPROF
Uptime: 00:02:32
Session status: UP-ACTIVE
Peer: 10.6.6.50 port 50001 fvrf: (none) ivrf: (none)
      Phase1_id: 10.6.6.50
      Desc: (none)
  Session ID: 147
  IKEv2 SA: local 10.17.1.1/4500 remote 10.6.6.50/50001 Active
          Capabilities:S connid:1 lifetime:23:57:28 ====> Capabilities:S indicates this is
 a  SIP VPN-SIP Session
  IPSEC FLOW: permit ip 0.0.0.0/0.0.0.0 0.0.0.0/0.0.0.0
        Active SAs: 2, origin: crypto map
        Inbound:  #pkts dec'ed 4 drop 0 life (KB/Sec) 4293728/3448
        Outbound: #pkts enc'ed 6 drop 0 life (KB/Sec) 4293728/3448
```

Verifying IP NAT Translations on a Remote Router

```
Peer2#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
udp 3.3.3.3:4500       10.6.6.49:50002    10.6.6.50:50001    10.6.6.50:50001
```

Configuration Examples for VPN-SIP

Configuration Examples for VPN-SIP

Using self-signed certificates for authentication

The following is sample configuration to configure VPN-SIP using self-signed certificates for authentication. There is no distinction between initiator and responder role in VPN-SIP. The configuration on a peer node will be identical with local SIP numbers changed.

```
// Self-signed certificate
crypto pki trustpoint selfCert
  rsakeypair myRSA
  enrollment selfsigned
  revocation-check none
!
crypto ikev2 profile vpn-sip-profile
 match identity remote any
 authentication local rsa-sig
 authentication remote rsa-sig
 pki trustpoint selfCert // Use same self-signed trustpoint for sign and verify
 nat force-encap
!
crypto ipsec profile vpn-sip-ipsec
 set security-association idle-time 120
!
vpn-sip enable
vpn-sip local-number 0388883001 address ipv4 GigabitEthernet1
vpn-sip tunnel source Loopback11
vpn-sip logging
!
// one tunnel per peer - configuration is for peer with a SIP-number of 0388884001
int tunnel0
 ip unnumbered loopback 0
 tunnel source loopback11
 tunnel mode ipsec ipv4
 tunnel destination dynamic
 tunnel protection ipsec profile vpn-sip-ipsec ikev2-profile vpn-sip-profile
 vpn-sip local-number 0388883001 remote-number 0388884001 bandwidth 1000
!
// ip unnumbered of tunnel interfaces
int loopback 0
  ip address 10.21.1.1 255.255.255.255
!
int loopback11
ip address 10.9.9.9 255.255.255.255
ip nat inside
!
// one tunnel per peer - this is for peer with SIP-number 0388885001
int tunnel1
 ip unnumbered loopback 0
 tunnel source loopback11
 tunnel mode ipsec ipv4
 tunnel destination dynamic
 tunnel protection ipsec profile vpn-sip-ipsec ikev2-profile iprof
 vpn-sip sip-local 0388883001 sip-remote 0388885001 bandwidth 1000
!
interface GigabitEthernet8
 ip dhcp client request sip-server-address
 ip dhcp client request vendor-identifying-specific
 ip address dhcp
ip nat outside


// backup routes configured with higher AD so that these routes will be activated only when
 primary path goes down. AD need to be chosen to be greater than that of primary route.
ip route 10.0.0.0 255.0.0.0 tunnel 0 250
ip route 10.1.0.0 255.0.0.0 tunnel 0 250
ip route 10.2.0.0 255.0.0.0 tunnel 0 250
ip route 10.3.0.0 255.0.0.0 tunnel 0 250
```

Troubleshooting for VPN-SIP

Viewing Tunnel Interface in Show Output

Symptom

Show VPN-SIP session doesn’t show any information about the tunnel interface. In the following example, information about the tunnel interface, tunnel1 is not shown:

```
Peer5-F#show vpn-sip session
VPN-SIP session current status

Interface: Tunnel2
   Session status: READY_TO_CONNECT
   Remote number : 0334563333
   Local number  : 0623458888
   Remote address:port: 0.0.0.0:0
   Local address:port : 192.30.18.22:0

Interface: Tunnel3
   Session status: READY_TO_CONNECT
   Remote number : 0323452222
   Local number  : 0623458888
   Remote address:port: 0.0.0.0:0
   Local address:port : 192.30.18.22:0

Interface: Tunnel4
   Session status: READY_TO_CONNECT
   Remote number : 0612349999
   Local number  : 0623458888
   Remote address:port: 0.0.0.0:0
   Local address:port : 192.30.18.22:0

Interface: Tunnel6
   Session status: READY_TO_CONNECT
   Remote number : 0634567777
   Local number  : 0623458888
   Remote address:port: 0.0.0.0:0
   Local address:port : 172.30.18.22:0
```

Possible Cause

VPN-SIP is not configured on the tunnel interface

```
Peer5-F#sh run int tun1
Building configuration...

Current configuration : 201 bytes
!
interface Tunnel1
 ip address 10.5.5.5 255.0.0.0
 tunnel source Loopback11
 tunnel mode ipsec ipv4
 tunnel destination dynamic
 tunnel protection ipsec profile test-prof ikev2-profile test
end
```

Recommended Action

Configure VPN-SIP on the tunnel interface.

```
:

Peer5-F#show running interface tunnel 1
Building configuration...

Current configuration : 278 bytes
!
interface Tunnel1
 ip address 10.5.5.5 255.255.255.255
 tunnel source Loopback11
 tunnel mode ipsec ipv4
 tunnel destination dynamic
 tunnel protection ipsec profile test-prof ikev2-profile test
 vpn-sip local-number 0623458888 remote-number 0312341111 bandwidth 1000
end
```

Following is the running output for the above scenario:

```
Peer5-F#show vpn-sip session detail
VPN-SIP session current status

Interface: Tunnel1
   Session status: READY_TO_CONNECT
   Remote number : 0312341111
   Local number  : 0623458888
   Remote address:port: 0.0.0.0:0
   Local address:port : 172.30.18.22:0

   Crypto conn handle: 0x8000002C
   SIP Handle        : 0x0
   SIP callID        : --
   Configured/Negotiated bandwidth: 1000/0 kbps

Interface: Tunnel2
   Session status: READY_TO_CONNECT
   Remote number : 0334563333
   Local number  : 0623458888
   Remote address:port: 0.0.0.0:0
   Local address:port : 172.30.18.22:0
   Crypto conn handle: 0x80000012
   SIP Handle        : 0x0
   SIP callID        : --
   Configured/Negotiated bandwidth: 512/0 kbps

Interface: Tunnel3
   Session status: READY_TO_CONNECT
   Remote number : 0323452222
   Local number  : 0623458888
   Remote address:port: 0.0.0.0:0
   Local address:port : 172.30.18.22:0
   Crypto conn handle: 0x80000031
   SIP Handle        : 0x0
   SIP callID        : --
   Configured/Negotiated bandwidth: 512/0 kbps

Interface: Tunnel4
   Session status: READY_TO_CONNECT
   Remote number : 0612349999
   Local number  : 0623458888
   Remote address:port: 0.0.0.0:0
   Local address:port : 172.30.18.22:0
   Crypto conn handle: 0x8000002F
   SIP Handle        : 0x0
   SIP callID        : --
   Configured/Negotiated bandwidth: 1000/0 kbps

Interface: Tunnel6
   Session status: READY_TO_CONNECT
   Remote number : 0634567777
   Local number  : 0623458888
   Remote address:port: 0.0.0.0:0
   Local address:port : 172.30.18.22:0
   Crypto conn handle: 0x80000026
   SIP Handle        : 0x0
   SIP callID        : --
   Configured/Negotiated bandwidth: 1000/0 kbps
```

Troubleshooting SIP Registration Status

Symptom

SIP registration status is Not Registered

```
Peer5#show vpn-sip sip registrar
Line          destination      expires(sec)  contact
transport     call-id
============================================================

Peer5-F#show vpn-sip registration-status

 SIP registration of local number 0623458888 : not registered
```

Possible Cause

IP address is not configured on the WAN interface.

```
Peer5#show ip interface brief
Interface                  IP-Address      OK? Method Status                Protocol
GigabitEthernet0/0         unassigned      YES unset  down                  down
GigabitEthernet0/1         unassigned      YES unset  up                    up
GigabitEthernet0/2         unassigned      YES unset  down                  down
GigabitEthernet0/3         unassigned      YES unset  down                  down
GigabitEthernet0/4         unassigned      YES unset  up                    up
GigabitEthernet0/5         10.5.5.5        YES manual up                    up
Vlan1                      10.45.1.5        YES NVRAM  up                    up
NVI0                       10.1.1.1         YES unset  up                    up
Loopback1                  10.1.1.1         YES NVRAM  up                    up
Loopback5                  10.5.5.5        YES NVRAM  administratively down down
Loopback11                 10.11.11.11     YES NVRAM  up                    up
Tunnel1                    10.5.5.5         YES NVRAM  up                    down
```

```
Tunnel2                    10.2.2.2         YES NVRAM  up                    down
Tunnel3                    10.3.3.3         YES NVRAM  up                    down
Tunnel4                    10.4.4.4         YES NVRAM  up                    down
Tunnel6                    10.8.8.8         YES NVRAM  up                    down

Peer5-F#show run interface gigabitEthernet 0/4
Building configuration...

Current configuration : 213 bytes
!
interface GigabitEthernet0/4
 ip dhcp client request sip-server-address
 ip dhcp client request vendor-identifying-specific
 no ip address           ====> no IP address
 ip nat outside
 ip virtual-reassembly in
 duplex auto
 speed auto
end
```

Recommended Action

Use the **ip address dhcp** command to configure the interface IP address.

```
Peer5-F#show running-config interface gigabitEthernet 0/4
Building configuration...

Current configuration : 215 bytes
!
interface GigabitEthernet0/4
 ip dhcp client request sip-server-address
 ip dhcp client request vendor-identifying-specific
 ip address dhcp         ====> configure IP address DHCP
 ip nat outside
 ip virtual-reassembly in
 duplex auto
 speed auto
end


Peer5-F#show ip interface brief
Interface                  IP-Address      OK? Method Status                Protocol
GigabitEthernet0/0         unassigned      YES unset  down                  down
GigabitEthernet0/1         unassigned      YES unset  up                    up
GigabitEthernet0/2         unassigned      YES unset  down                  down
GigabitEthernet0/3         unassigned      YES unset  down                  down
GigabitEthernet0/4         172.30.18.22    YES DHCP   up                    up
GigabitEthernet0/5         10.5.5.5        YES manual up                    up
Vlan1                      10.45.1.5        YES NVRAM  up                    up
NVI0                       10.1.1.1         YES unset  up                    up
Loopback1                  10.1.1.1         YES NVRAM  up                    up
Loopback5                  10.5.5.5        YES NVRAM  administratively down down
Loopback11                 10.11.11.11     YES NVRAM  up                    up
Tunnel1                    10.6.5.5         YES NVRAM  up                    down
Tunnel2                    10.2.2.2         YES NVRAM  up                    down
Tunnel3                    10.3.3.3         YES NVRAM  up                    down
Tunnel4                    10.4.4.4         YES NVRAM  up                    down
Tunnel6                    10.8.8.8         YES NVRAM  up                    down

Peer5-F#show vpn-sip sip registrar
Line          destination      expires(sec)  contact
transport     call-id
============================================================
0623458888    example.com   2863          172.30.18.22
UDP           1E83ECF0-AF0611E7-802B8FCF-594EB9E7@122.50.18.22
```

```
Peer5-F#show vpn-sip registration-status

 SIP registration of local number 0623458888 : registered 172.30.18.22
```

Session stuck in Negotiating IKE state

Symptom

VPN-SIP session stuck in Negotiating IKE state.

```
Peer5#show vpn-sip session remote-number 0612349999 detail
VPN-SIP session current status

Interface: Tunnel4
   Session status: NEGOTIATING_IKE (R)
   Uptime        : 00:00:58
   Remote number : 0612349999
   Local number  : 0623458888
   Remote address:port: 172.30.168.3:24825
   Local address:port : 172.30.18.22:50012
   Crypto conn handle: 0x8000002E
   SIP Handle        : 0x8000000C
   SIP callID        : 16
   Configured/Negotiated bandwidth: 1000/1000 kbps
```

Possible Cause

Bad configuration related to IKEv2.

In the following example the Key ID that is configured in the keyring does not match the SIP numberof the remote peer.

```
Peer5-F#show running-config interface tunnel 4
Building configuration...

Current configuration : 276 bytes
!
interface Tunnel4
 ip address 10.4.4.4 255.0.0.0
 tunnel source Loopback11
 tunnel mode ipsec ipv4
 tunnel destination dynamic
 tunnel protection ipsec profile test-prof ikev2-profile test
 VPN-SIP local-number 0623458888 remote-number 0612349999 bandwidth 1000  ====> Remote
number mentioned here doesn’t match the remote number in the keyring
end

IKEv2 Keyring configs:
!
crypto ikev2 keyring keys
 peer peer1
  identity key-id 0312341111
  pre-shared-key psk1
 !
 peer abc
  identity key-id 0345674444
  pre-shared-key psk1
 !
 peer peer2
  identity key-id 0334563333
  pre-shared-key psk10337101690
 !
 peer peer6
```

```
  identity key-id 0634567777
  pre-shared-key cisco123
 !
 peer peer3
  identity key-id 0323452222
  pre-shared-key cisco123
 !
 peer peer4
  identity key-id 0645676666
  pre-shared-key psk1
 !
 peer NONID
  identity fqdn example.com
  pre-shared-key psk1
 !
!
!
crypto ikev2 profile test
 match identity remote any
 identity local key-id 0623458888
 authentication remote pre-share
 authentication local pre-share
 keyring local keys
 dpd 10 6 periodic
 nat force-encap
```

Recommended Action

Correct the keyring configurations.

```
rypto ikev2 keyring keys
 peer peer1
  identity key-id 0312341111
  pre-shared-key psk1
 !
 peer abc
  identity key-id 0345674444
  pre-shared-key psk1
 !
 peer peer2
  identity key-id 0334563333
  pre-shared-key psk1
 !
 peer peer6
  identity key-id 0634567777
  pre-shared-key psk1
 !
 peer peer3
  identity key-id 0323452222
  pre-shared-key psk1
 !
 peer peer4
  identity key-id 0612349999
  pre-shared-key psk1
 !
 peer NONID
  identity fqdn example.com
  pre-shared-key psk1
 !
!
!
crypto ikev2 profile test
 match identity remote any
 identity local key-id 0623458888
 authentication remote pre-share
 authentication local pre-share
 keyring local keys
 dpd 10 6 periodic
 nat force-encap
!

Peer5-F#show vpn-sip session remote-number  0612349999 detail
VPN-SIP session current status

Interface: Tunnel4
   Session status: SESSION_UP (R)
   Uptime        : 00:02:04
   Remote number : 0612349999
   Local number  : 0623458888
   Remote address:port: 172.30.168.3:24845
   Local address:port : 172.30.18.22:50020
   Crypto conn handle: 0x8000004E
   SIP Handle        : 0x80000014
   SIP callID        : 24
   Configured/Negotiated bandwidth: 1000/1000 kbps
```

Troubleshooting Session Initiation

Symptom

Session does not initiate and gets stuck in Negotiating IKE state

Possible Cause

Fagmentation of IKE packets when a large PKI certificate is included in the IKE authentication message.

Recommended Action

Configure IKEv2 fragmentation on the routers.

Debug Commands

The folllwing debug commands are available to debug VPN-SIP configuration:

Table 10: debug commands

Command Name	Description
debug vpn-sip event	Prints debug messages for SVTI registration with VPN-SIP, SIP registration, call setup, and so on.
debug vpn-sip errors	Prints error messages only when an error occurs during initialization, registration, call setup, and so on.
debug vpn-sip sip all	Enables all SIP debugging traces.
debug vpn-sip sip calls	Enables SIP SPI calls debugging trace.
debug vpn-sip sip dhcp	Enables SIP-DHCP debugging trace
debug vpn-sip sip error	Enables SIP error debugging trace
debug vpn-sip sip events	Enables SIP events debugging trace.

Command Name	Description
debug vpn-sip sip feature	Enables feature level debugging.
debug vpn-sip sip function	Enables SIP function debugging trace.
debug vpn-sip sip info	Enables SIP information debugging trace.
debug vpn-sip sip level	Enables information level debugging.
debug vpn-sip sip media	Enables SIP media debugging trace.
debug vpn-sip sip messages	Enables SIP SPI messages debugging trace
debug vpn-sip sip non-call	Enables Non-Call-Context trace (OPTIONS, SUBSCRIBE, and so on)
debug vpn-sip sip preauth	Enable SIP preauth debugging trace.
debug vpn-sip sip states	Enable SIP SPI states debugging trace.
debug vpn-sip sip translate	Enables SIP translation debugging trace.
debug vpn-sip sip transport	Enables SIP transport debugging traces.
debug vpn-sip sip verbose	Enables verbose mode.

Additional References for VPN-SIP

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases

Standards and RFCs

Standard/RFC	Title
RFC 6193 (with Restrictions)	Media Description for the Internet Key Exchange Protocol (IKE) in the Session Description Protocol (SDP)

Feature Information for VPN-SIP

Table 11: Feature Information for VPN-SIP

Feature Name	Releases	Feature Information
Session Initiation Protocol Triggered VPN	Cisco IOS Release 15.7(3)M1	VPN-SIP is a service offered by service providers where a VPN is setup for on-demand media or application sharing between peers, using Session Initiation Protocol (SIP). The following commands were introduced: **nat force-encap, show vpn-sip session, show vpn-sip sip, show vpn-sip registration-status, vpn-sip local-number, vpn-sip logging, vpn-sip tunnel source**.