Configuring Internet Key Exchange for IPsec VPNs

This module describes how to configure the Internet Key Exchange (IKE) protocol for basic IP Security (IPsec) Virtual Private Networks (VPNs). IKE is a key management protocol standard that is used in conjunction with the IPsec standard. IPsec is an IP security feature that provides robust authentication and encryption of IP packets.

IPsec can be configured without IKE, but IKE enhances IPsec by providing additional features, flexibility, and ease of configuration for the IPsec standard.

IKE is a hybrid protocol, that implements the Oakley key exchange and Skeme key exchange inside the Internet Security Association Key Management Protocol (ISAKMP) framework. (ISAKMP, Oakley, and Skeme are security protocols implemented by IKE.)

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, on page 1
- Prerequisites for IKE Configuration, on page 2
- Restrictions for IKE Configuration, on page 2
- Information About Configuring IKE for IPsec VPNs, on page 3
- How to Configure IKE for IPsec VPNs, on page 8
- Configuration Examples for an IKE Configuration, on page 20
- Where to Go Next, on page 22
- Additional References, on page 22
- Feature Information for Configuring IKE for IPsec VPNs, on page 23

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 https://cfnng.cisco.com/. An account on Cisco.com is not required.

Prerequisites for IKE Configuration

- You should be familiar with the concepts and tasks explained in the module Configuring Security for VPNs with IPsec.
- Ensure that your Access Control Lists (ACLs) are compatible with IKE. Because IKE negotiation uses User Datagram Protocol (UDP) on port 500, your ACLs must be configured so that UDP port 500 traffic is not blocked at interfaces used by IKE and IPsec. In some cases you might need to add a statement to your ACLs to explicitly permit UDP port 500 traffic.

Restrictions for IKE Configuration

- To avoid profiles being locked or leading to DMI degrade state, before using the **config-replace** command to replace a configuration, ensure to shut down the tunnel interface to bring down all crypto sessions, and tunnel configurations.
- The initiating router *must not* have a certificate associated with the remote peer.
- The preshared key *must* be by a fully qualified domain name (FQDN) on both peers. (To configure the preshared key, enter the **crypto isakmp key** command.)
- The communicating routers *must* have a FQDN host entry for each other in their configurations.
- The communicating routers *must* be configured to authenticate by hostname, *not* by IP address; thus, you should use the **crypto isakmp identity hostname** command.
- Use **show crypto eli** command to determine the software encryption limitations for your device. Without any hardware modules, the limitations are as follows:
  - 1000 IPsec security associations (SAs)
  - 100 IKE SAs
  - 50 Diffie-Hellman (DH) session keys
- Disable the crypto batch functionality, by using the **no crypto batch allowed** command to increase the performance of a TCP flow on a Site-to-site VPN. However, disabling the crypto batch functionality might have an impact on CPU utilization.
- Starting with Cisco IOS Release 15.0(1)SY and later, you cannot configure IPSec Network Security features using **crypto ipsec** commands on Cisco Catalyst 6500 Series switches. For IPSec support on these switches, you must use a hardware encryption engine.

Information About Configuring IKE for IPsec VPNs

Supported Standards for Use with IKE

Cisco implements the following standards:

- IPsec—IP Security Protocol. 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; it uses IKE to handle negotiation of protocols and algorithms based on local policy and to 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.
- ISAKMP—Internet Security Association and Key Management Protocol. A protocol framework that defines payload formats, the mechanics of implementing a key exchange protocol, and the negotiation of a security association.
- Oakley—A key exchange protocol that defines how to derive authenticated keying material.
- Skeme—A key exchange protocol that defines how to derive authenticated keying material, with rapid key refreshment.

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-256 and DH Groups 14 or higher. For more information about the latest Cisco cryptographic recommendations, see the Next Generation Encryption (NGE) white paper.

The component technologies implemented for use by IKE include the following:

- AES—Advanced Encryption Standard. A cryptographic algorithm that protects sensitive, unclassified information. AES is privacy transform for IPsec and IKE and has been developed to replace the Data Encryption Standard (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. IKE implements the 56-bit DES-CBC with Explicit IV standard. Cipher Block Chaining (CBC) requires an initialization vector (IV) to start encryption. The IV is explicitly given in the IPsec packet.

Cisco IOS software also implements Triple DES (168-bit) encryption, depending on the software versions available for a specific platform. Triple DES (3DES) is a strong form of encryption that allows sensitive information to be transmitted over untrusted networks. It enables customers, particularly in the finance industry, to utilize network-layer encryption.

Note Cisco IOS images that have 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 that are to be installed outside the United States require an export license. Customer orders might be denied or subject to delay because of United States government regulations. Contact your sales representative or distributor for more information, or send 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 to 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.
- RSA signatures and RSA encrypted nonces—RSA is the public key cryptographic system developed by Ron Rivest, Adi Shamir, and Leonard Adleman. RSA signatures provide nonrepudiation, and RSA encrypted nonces provide repudiation. (Repudation and nonrepudation have to do with traceability.)
- 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—Message Digest 5 (Hash-Based Message Authentication Code (HMAC) variant). A hash algorithm used to authenticate packet data. HMAC is a variant that provides an additional level of hashing.

IKE interoperates with the X.509v3 certificates, which are used with the IKE protocol when authentication requires public keys. This certificate support allows the protected network to scale by providing the equivalent of a digital ID card to each device. When two devices intend to communicate, they exchange digital certificates to prove their identity (thus removing the need to manually exchange public keys with each peer or to manually specify a shared key at each peer).

IKE Benefits

IKE automatically negotiates IPsec security associations (SAs) and enables IPsec secure communications without costly manual preconfiguration. Specifically, IKE provides the following benefits:

- Allows you to specify a lifetime for the IPsec SA.
- Allows encryption keys to change during IPsec sessions.
- Allows IPsec to provide antireplay services.
- Permits certification authority (CA) support for a manageable, scalable IPsec implementation.

- Allows dynamic authentication of peers.

IKE Main Mode and Aggressive Mode

IKE has two phases of key negotiation: phase 1 and phase 2. Phase 1 negotiates a security association (a key) between two IKE peers. The key negotiated in phase 1 enables IKE peers to communicate securely in phase 2. During phase 2 negotiation, IKE establishes keys (security associations) for other applications, such as IPsec.

Phase 1 negotiation can occur using main mode or aggressive mode. Main mode tries to protect all information during the negotiation, meaning that no information is available to a potential attacker. When main mode is used, the identities of the two IKE peers are hidden. Although this mode of operation is very secure, it is relatively costly in terms of the time required to complete the negotiation. Aggressive mode takes less time to negotiate keys between peers; however, it gives up some of the security provided by main mode negotiation. For example, the identities of the two parties trying to establish a security association are exposed to an eavesdropper.

The two modes serve different purposes and have different strengths. Main mode is slower than aggressive mode, but main mode is more secure and more flexible because it can offer an IKE peer more security proposals than aggressive mode. Aggressive mode is less flexible and not as secure, but much faster.

In Cisco IOS software, the two modes are not configurable. The default action for IKE authentication (rsa-sig, rsa-encr, or preshared) is to initiate main mode; however, in cases where there is no corresponding information to initiate authentication, and there is a preshared key associated with the hostname of the peer, Cisco IOS software can initiate aggressive mode. Cisco IOS software will respond in aggressive mode to an IKE peer that initiates aggressive mode.

IKE Policies Security Parameters for IKE Negotiation

An IKE policy defines a combination of security parameters to be used during the IKE negotiation. You must create an IKE policy at each peer participating in the IKE exchange.

If you do not configure any IKE policies, your router will use the default policy, which is always set to the lowest priority and which contains the default value of each parameter.

About IKE Policies

Because IKE negotiations must be protected, each IKE negotiation begins by agreement of both peers on a common (shared) IKE policy. This policy states which security parameters will be used to protect subsequent IKE negotiations and mandates how the peers are authenticated.

After the two peers agree upon a policy, the security parameters of the policy are identified by an SA established at each peer, and these SAs apply to all subsequent IKE traffic during the negotiation.

You can configure multiple, prioritized policies on each peer--e ach with a different combination of parameter values. However, at least one of these policies must contain exactly the same encryption, hash, authentication, and Diffie-Hellman parameter values as one of the policies on the remote peer. For each policy that you create, you assign a unique priority (1 through 10,000, with 1 being the highest priority).

Tip If you are interoperating with a device that supports only one of the values for a parameter, your choice is limited to the value supported by the other device. Aside from this limitation, there is often a trade-off between security and performance, and many of these parameter values represent such a trade-off. You should evaluate the level of security risks for your network and your tolerance for these risks.

IKE Peers Agreeing Upon a Matching IKE Policy

When the IKE negotiation begins, IKE searches for an IKE policy that is the same on both peers. The peer that initiates the negotiation will send all its policies to the remote peer, and the remote peer will try to find a match. The remote peer looks for a match by comparing its own highest priority policy against the policies received from the other peer. The remote peer checks each of its policies in order of its priority (highest priority first) until a match is found.

A match is made when both policies from the two peers contain the same encryption, hash, authentication, and Diffie-Hellman parameter values.

If a match is found, IKE will complete negotiation, and IPsec security associations will be created. If no acceptable match is found, IKE refuses negotiation and IPsec will not be established.

Note The parameter values apply to the IKE negotiations after the IKE SA is established. Depending on the authentication method specified in a policy, additional configuration might be required (as described in the section IKE Authentication, on page 6). If a peer's policy does not have the required companion configuration, the peer will not submit the policy when attempting to find a matching policy with the remote peer.

IKE Authentication

IKE authentication consists of the following options and each authentication method requires additional configuration.

RSA Signatures

With RSA signatures, you can configure the peers to obtain certificates from a CA. (The CA must be properly configured to issue the certificates.) Using a CA can dramatically improve the manageability and scalability of your IPsec network. Additionally, RSA signature-based authentication uses only two public key operations, whereas RSA encryption uses four public key operations, making it costlier in terms of overall performance. To properly configure CA support, see the module "Deploying RSA Keys Within a PKI."

The certificates are used by each peer to exchange public keys securely. (RSA signatures requires that each peer has the public signature key of the remote peer.) When both peers have valid certificates, they will automatically exchange public keys with each other as part of any IKE negotiation in which RSA signatures are used.

You can also exchange the public keys manually, as described in the section "Configuring RSA Keys Manually for RSA Encrypted Nonces, on page 13."

RSA signatures provide nonrepudiation for the IKE negotiation. And, you can prove to a third party after the fact that you did indeed have an IKE negotiation with the remote peer.

RSA Encrypted Nonces

With RSA encrypted nonces, you must ensure that each peer has the public keys of the other peers.

Unlike RSA signatures, the RSA encrypted nonces method cannot use certificates to exchange public keys. Instead, you ensure that each peer has the other's public keys by one of the following methods:

- Manually configuring RSA keys as described in the section "Configuring RSA Keys Manually for RSA Encrypted Nonces, on page 13."
- Ensuring that an IKE exchange using RSA signatures with certificates has already occurred between the peers. (The peers' public keys are exchanged during the RSA-signatures-based IKE negotiations if certificates are used.) To make that the IKE exchange happens, specify two policies: a higher-priority policy with RSA encrypted nonces and a lower-priority policy with RSA signatures. When IKE negotiations occur, RSA signatures will be used the first time because the peers do not yet have each other's public keys. Then future IKE negotiations can use RSA encrypted nonces because the public keys will have been exchanged. This alternative requires that you already have CA support configured.

RSA encrypted nonces provide repudiation for the IKE negotiation; however, unlike RSA signatures, you cannot prove to a third party that you had an IKE negotiation with the remote peer.

Preshared Keys

Preshared Keys An Overview

Preshared keys are clumsy to use if your secured network is large, and they do not scale well with a growing network. However, they do not require use of a CA, as do RSA signatures, and might be easier to set up in a small network with fewer than ten nodes. RSA signatures also can be considered more secure when compared with preshared key authentication.

Note If RSA encryption is configured and signature mode is negotiated (and certificates are used for signature mode), the peer will request both signature and encryption keys. Basically, the router will request as many keys as the configuration will support. If RSA encryption is not configured, it will just request a signature key.

ISAKMP Identity Setting for Preshared Keys

You should set the ISAKMP identity for each peer that uses preshared keys in an IKE policy.

When two peers use IKE to establish IPsec SAs, each peer sends its identity to the remote peer. Each peer sends either its hostname or its IP address, depending on how you have set the ISAKMP identity of the router.

By default, a peer's ISAKMP identity is the IP address of the peer. If appropriate, you could change the identity to be the peer's hostname instead. As a general rule, set the identities of all peers the same way--either all peers should use their IP addresses or all peers should use their hostnames. If some peers use their hostnames and some peers use their IP addresses to identify themselves to each other, IKE negotiations could fail if the identity of a remote peer is not recognized and a Domain Name System (DNS) lookup is unable to resolve the identity.

Mask Preshared Keys

A mask preshared key allows a group of remote users with the same level of authentication to share an IKE preshared key. The preshared key of the remote peer must match the preshared key of the local peer for IKE authentication to occur.

A m ask preshared key is usually distributed through a secure out-of-band channel. In a remote peer-to-local peer scenario, any remote peer with the IKE preshared key configured can establish IKE SAs with the local peer.

If you specify the **mask** keyword with the **crypto isakmp key** command, it is up to you to use a subnet address, which will allow more peers to share the same key. That is, the preshared key is no longer restricted to use between two users.

Note Using 0.0.0.0 as a subnet address is not recommended because it encourages group preshared keys, which allow all peers to have the same group key, thereby reducing the security of your user authentication.

Disable Xauth on a Specific IPsec Peer

Disabling Extended Authentication (Xauth) for static IPsec peers prevents the routers from being prompted for Xauth information--username and password.

IKE Mode Configuration

IKE mode configuration, as defined by the Internet Engineering Task Force (IETF), allows a gateway to download an IP address (and other network-level configuration) to the client as part of an IKE negotiation. Using this exchange, the gateway gives an IP address to the IKE client to be used as an “inner” IP address encapsulated under IPsec. This method provides a known IP address for the client that can be matched against IPsec policy.

To implement IPsec VPNs between remote access clients that have dynamic IP addresses and a corporate gateway, you have to dynamically administer scalable IPsec policy on the gateway once each client is authenticated. With IKE mode configuration, the gateway can set up a scalable policy for a very large set of clients regardless of the IP addresses of those clients.

There are two types of IKE mode configuration:

- Gateway initiation--Gateway initiates the configuration mode with the client. Once the client responds, the IKE modifies the identity of the sender, the message is processed, and the client receives a response.
- Client initiation--Client initiates the configuration mode with the gateway. The gateway responds with an IP address that it has allocated for the client.

How to Configure IKE for IPsec VPNs

If you do not want IKE to be used with your IPsec implementation, you can disable it at all IPsec peers via the **no crypto isakmp** command, skip the rest of this chapter, and begin your IPsec VPN.

IKE is enabled by default. IKE does not have to be enabled for individual interfaces, but it is enabled globally for all interfaces at the router.

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.

Perform the following tasks to provide authentication of IPsec peers, negotiate IPsec SAs, and establish IPsec keys:

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:**	Defines an IKE policy and enters config-isakmp configuration mode.

	Command or Action	Purpose
	`Router(config)# crypto isakmp policy 10`	• *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. • **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.

	Command or Action	Purpose
		• **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.
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
```

Troubleshooting Tips

- Clear (and reinitialize) IPsec SAs by using the **clear crypto sa** EXEC command.

Using the **clear crypto sa** command without parameters will clear out the full SA database, which will clear out active security sessions. You may also specify the **peer**, **map**, or **entry** keywords to clear out only a subset of the SA database. For more information, see the **clear crypto sa** command in the Cisco IOS Security Command Reference.

- The default policy and default values for configured policies do not show up in the configuration when you issue the **show running-config** command. To display the default policy and any default values within configured policies, use the **show crypto isakmp policy** command.
- Any IPsec transforms or IKE encryption methods that the current hardware does not support should be disabled; they are ignored whenever an attempt to negotiate with the peer is made.

If a user enters an IPsec transform or an IKE encryption method that the hardware does not support, a warning message will be generated. These warning messages are also generated at boot time. When an encrypted card is inserted, the current configuration is scanned. If any IPsec transforms or IKE encryption methods are found that are not supported by the hardware, a warning message will be generated.

What to Do Next

Depending on which authentication method you specified in your IKE policies (RSA signatures, RSA encrypted nonces, or preshared keys), you must do certain additional configuration tasks before IKE and IPsec can successfully use the IKE policies. For information on completing these additional tasks, refer to the Configuring IKE Authentication, on page 12."

To configure an AES-based transform set, see the module "Configuring Security for VPNs with IPsec."

Configuring IKE Authentication

After you have created at least one IKE policy in which you specified an authentication method (or accepted the default method), you need to configure an authentication method. IKE policies cannot be used by IPsec until the authentication method is successfully configured.

Note Before configuring IKE authentication, you must have configured at least one IKE policy, which is where the authentication method was specified (or RSA signatures was accepted by default).

To configure IKE authentication, you should perform one of the following tasks, as appropriate:

Configuring RSA Keys Manually for RSA Encrypted Nonces

Note This task can be performed only if a CA is not in use.

To manually configure RSA keys, perform this task for each IPsec peer that uses RSA encrypted nonces in an IKE policy.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto key generate rsa**{**general-keys**} | **usage-keys**} [**label** *key-label*] [**exportable**] [**modulus** *modulus-size*]
4. **crypto key generate ec keysize** [**256** | **384**] [**label** *label-string*]
5. **exit**
6. **show crypto key mypubkey rsa**
7. **configure terminal**
8. **crypto key pubkey-chain rsa**
9. Do one of the following:
   - **named-key** *key-name* [**encryption** | **signature**]
   - **addressed-key** *key-address* [**encryption** | **signature**]
10. **address** *ip-address*
11. **key-string** key-string
12. **quit**
13. Repeat these steps at each peer that uses RSA encrypted nonces in an IKE policy.
14. **exit**
15. **exit**
16. **show crypto key pubkey-chain rsa** [**name** *key-name* | **address** *key-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 key generate rsa**{**general-keys**}	**usage-keys**} [**label** *key-label*] [**exportable**] [**modulus** *modulus-size*] **Example:**	Generates RSA keys. • If a *key-label* argument is not specified, the default value, which is the fully qualified domain name (FQDN) of the router, is used.

	Command or Action	Purpose
	`Router(config)# crypto key generate rsa general-keys modulus 360`	
Step 4	**crypto key generate ec keysize** [**256** | **384**] [**label** *label-string*] **Example:** `Router(config)# crypto key generate ec keysize 256 label Router_1_Key`	Generates EC keys. • The 256 keyword specifies a 256-bit keysize. • The 384 keyword specifies a 384-bit keysize. • A label can be specified for the EC key by using the **label** keyword and *label-string* argument. **Note** If a label is not specified, then FQDN value is used.
Step 5	**exit** **Example:** `Router(config)# exit`	(Optional) Exits global configuration mode.
Step 6	**show crypto key mypubkey rsa** **Example:** `Router# show crypto key mypubkey rsa`	(Optional) Displays the generated RSA public keys.
Step 7	**configure terminal** **Example:** `Router# configure terminal`	Returns to global configuration mode.
Step 8	**crypto key pubkey-chain rsa** **Example:** `Router(config)# crypto key pubkey-chain rsa`	Enters public key chain configuration mode (so you can manually specify the RSA public keys of other devices).

Step 9	Do one of the following:
	• **named-key** *key-name* [**encryption** | **signature**]
	• **addressed-key** *key-address* [**encryption** | **signature**]
	Example:
	`Router(config-pubkey-chain)# named-key otherpeer.example.com`
	Example:
	`Router(config-pubkey-chain)# addressed-key 10.1.1.2 encryption`
		Indicates which remote peer's RSA public key you will specify and enters public key configuration mode.
		• If the remote peer uses its hostname as its ISAKMP identity, use the **named-key** command and specify the remote peer's FQDN, such as somerouter.example.com, as the *key-name*.
		• If the remote peer uses its IP address as its ISAKMP identity, use the **addressed-key** command and specify the remote peer's IP address as the *key-address*.

Step 10	**address** *ip-address* **Example:** `Router(config-pubkey-key)# address 10.5.5.1`	Specifies the IP address of the remote peer. • If you use the **named-key** command, you need to use this command to specify the IP address of the peer.
Step 11	**key-string** key-string **Example:**	Specifies the RSA public key of the remote peer.

	Command or Action	Purpose

	`Router(config-pubkey-key)# key-string`
	Example:
	`Router(config-pubkey)# 00302017 4A7D385B 1234EF29 335FC973`
	Example:
	`Router(config-pubkey)# 2DD50A37 C4F4B0FD 9DADE748 429618D5`
	Example:
	`Router(config-pubkey)# 18242BA3 2EDFBDD3 4296142A DDF7D3D8`
	Example:
	`Router(config-pubkey)# 08407685 2F2190A0 0B43F1BD 9A8A26DB`
	Example:
	`Router(config-pubkey)# 07953829 791FCDE9 A98420F0 6A82045B`
	Example:
	`Router(config-pubkey)# 90288A26 DBC64468 7789F76E EE21`
		• (This key was previously viewed by the administrator of the remote peer when the RSA keys of the remote router were generated.)

Step 12	**quit** **Example:** `Router(config-pubkey-key)# quit`	Returns to public key chain configuration mode.
Step 13	Repeat these steps at each peer that uses RSA encrypted nonces in an IKE policy.	—
Step 14	**exit** **Example:** `Router(config-pubkey-key)# exit`	Returns to global configuration mode.
Step 15	**exit** **Example:** `Router(config)# exit`	Returns to privileged EXEC mode.
Step 16	**show crypto key pubkey-chain rsa** [**name** *key-name* | **address** *key-address*] **Example:** `Router# show crypto key pubkey-chain rsa`	(Optional) Displays either a list of all RSA public keys that are stored on your router or details of a particular RSA key that is stored on your router.

Configuring Preshared Keys

To configure preshared keys, perform these steps for each peer that uses preshared keys in an IKE policy.

Note Preshared keys do not scale well with a growing network. Mask preshared keys have the following restrictions:

- The SA cannot be established between the IPsec peers until all IPsec peers are configured for the same preshared key.
- The mask preshared key must be distinctly different for remote users requiring varying levels of authorization. You must configure a new preshared key for each level of trust and assign the correct keys to the correct parties. Otherwise, an untrusted party may obtain access to protected data.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto isakmp identity** {**address** | **dn** | **hostname**}
4. **ip host** *hostname address1* [*address2...address8*]
5. Do one of the following:
   - **crypto isakmp key** *keystring* **address** *peer-address* [**mask**] [**no-xauth**]
   - **crypto isakmp key** *keystring* **hostname** *hostname* [**no-xauth**]
6. Do one of the following:
   - **crypto isakmp key** *keystring* **address** *peer-address* [**mask**] [**no-xauth**]
   - **crypto isakmp key** *keystring* **hostname** *hostname* [**no-xauth**]
7. Repeat these steps at each peer that uses preshared keys in an IKE policy.

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 identity** {**address**	**dn**	**hostname**} **Example:** `Router(config)# crypto isakmp identity address`	Specifies the peer's ISAKMP identity by IP address, by distinguished name (DN) hostname at the local peer. • **address**--Typically used when only one interface (and therefore only one IP address) will be used by the peer for IKE negotiations, and the IP address is known. • **dn**--Typically used if the DN of a router certificate is to be specified and chosen as the ISAKMP identity during IKE processing. The dn keyword is used only for certificate-based authentication.

	Command or Action	Purpose
		• **hostname**--Should be used if more than one interface on the peer might be used for IKE negotiations, or if the interface's IP address is unknown (such as with dynamically assigned IP addresses).
Step 4	**ip host** *hostname address1* [*address2...address8*] **Example:** `Router(config)# ip host RemoteRouter.example.com 192.168.0.1`	If the local peer's ISAKMP identity was specified using a hostname, maps the peer's host name to its IP address(es) at all the remote peers. (This step might be unnecessary if the hostname or address is already mapped in a DNS server.)

Step 5	Do one of the following:
	• **crypto isakmp key** *keystring* **address** *peer-address* [**mask**] [**no-xauth**]
	• **crypto isakmp key** *keystring* **hostname** *hostname* [**no-xauth**]
	Example:
	`Router(config)# crypto isakmp key sharedkeystring address 192.168.1.33 no-xauth`
	Example:
	`Router(config) crypto isakmp key sharedkeystring hostname RemoteRouter.example.com`
		Specifies at the local peer the shared key to be used with a particular remote peer.
		• If the remote peer specified its ISAKMP identity with an address, use the **address** keyword in this step; otherwise use the **hostname** keyword in this step.
		• **no-xauth**--Prevents the router from prompting the peer for Xauth information.
		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 a hostname as the identity of a 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.

Step 6	Do one of the following:
	• **crypto isakmp key** *keystring* **address** *peer-address* [**mask**] [**no-xauth**]
	• **crypto isakmp key** *keystring* **hostname** *hostname* [**no-xauth**]
	Example:
	`Router(config) crypto isakmp key sharedkeystring address 10.0.0.1`
	Example:
	`Router(config) crypto isakmp key sharedkeystring hostname LocalRouter.example.com`
		Specifies at the remote peer the shared key to be used with the local peer.
		• This is the same key you just specified at the local peer.
		• If the local peer specified its ISAKMP identity with an address, use the **address** keyword in this step; otherwise use the **hostname** keyword in this step.

Step 7	Repeat these steps at each peer that uses preshared keys in an IKE policy.	--

Configuring IKE Mode Configuration

Note IKE mode configuration has the following restrictions:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip local pool** *pool-name start-addr end-addr*
4. **crypto isakmp client configuration address-pool local** *pool-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip local pool** *pool-name start-addr end-addr* **Example:** `Router(config)# ip local pool pool1 172.16.23.0 172.16.23.255`	Defines an existing local address pool that defines a set of addresses.
Step 4	**crypto isakmp client configuration address-pool local** *pool-name* **Example:** `Router(config)# crypto isakmp client configuration address-pool local pool1`	References the local address pool in the IKE configuration.

Configuring an IKE Crypto Map for IPsec SA 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.

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **crypto map** *tag sequence* **ipsec-isakmp**
4. **set pfs** {**group1** | **group2** | **group5** | **group14** | **group15** | **group16**}

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 map** *tag sequence* **ipsec-isakmp** **Example:** `Router(config)# crypto map example 1 ipsec-ipsec-isakmp`	Specifies the crypto map and enters crypto map configuration mode. • The *tag* argument specifies the crypto map. • The *sequence* argument specifies the sequence to insert into the crypto map entry. • The **ipsec-isakmp** keyword specifies IPsec with IKEv1 (ISAKMP).
Step 4	**set pfs** {**group1** | **group2** | **group5** | **group14** | **group15** | **group16**} **Example:** `Router(config-isakmp)# set pfs 14`	Specifies the DH group identifier for IPSec SA negotiation. • By default, DH group 1 is used. • **group1**—768-bit DH (No longer recommended) • **group2**—1024-bit DH (No longer recommended) • **group5**—1536-bit DH (No longer recommended) • **group14**—Specifies the 2048-bit DH group. • **group15**—Specifies the 3072-bit DH group. • **group16**—Specifies the 4096-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 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.

Configuration Examples for an IKE Configuration

Example: Creating IKE Policies

This section contains the following examples, which show how to configure an AES IKE policy and a 3DES IKE policy.

Note Cisco no longer recommends using 3DES; instead, you should use AES. For more information about the latest Cisco cryptographic recommendations, see the Next Generation Encryption (NGE) white paper.

Example: Creating an AES IKE Policy

The following example is sample output from the **show running-config**command. In this example, the AES 256-bit key is enabled.

```
Current configuration : 1665 bytes
!
version 12.2
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname "Router1"
!
!
ip subnet-zero
!
!
no ip domain lookup
!
ip audit notify log
ip audit po max-events 100
!
crypto isakmp policy 10
 encryption aes 256
 authentication pre-share
 lifetime 180
crypto isakmp key cisco123 address 10.0.110.1
!
!
crypto ipsec transform-set aesset esp-aes 256 esp-sha-hmac
 mode transport

.
.
.
```

Example: Creating 3DES IKE Policies

This example creates two IKE policies, 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
```

In the example, the encryption DES of policy default would not appear in the written configuration because this is the default value for the encryption algorithm parameter.

If the **show crypto isakmp policy**command is issued with this configuration, the output is as follows:

```
Protection suite priority 15
encryption algorithm:3DES - Triple Data Encryption Standard (168 bit keys)
hash algorithm:Message Digest 5
authentication method:Rivest-Shamir-Adleman Signature
Diffie-Hellman group:#2 (1024 bit)
lifetime:5000 seconds, no volume limit
Protection suite priority 20
encryption algorithm:DES - Data Encryption Standard (56 bit keys)
hash algorithm:Secure Hash Standard
authentication method:preshared Key
Diffie-Hellman group:#1 (768 bit)
lifetime:10000 seconds, no volume limit
Default protection suite
encryption algorithm:DES - Data Encryption Standard (56 bit keys)
hash algorithm:Secure Hash Standard
authentication method:Rivest-Shamir-Adleman Signature
Diffie-Hellman group:#1 (768 bit)
lifetime:86400 seconds, no volume limit
```

Note that although the output shows “no volume limit” for the lifetimes, you can configure only a time lifetime (such as 86,400 seconds); volume-limit lifetimes are not configurable.

Example: Configuring IKE Authentication

The following example shows how to manually specify the RSA public keys of two IPsec peer-- the peer at 10.5.5.1 uses general-purpose keys, and the other peer uses special-usage keys:

```
crypto key pubkey-chain rsa
 named-key otherpeer.example.com
 address 10.5.5.1
 key-string
 005C300D 06092A86 4886F70D 01010105
 00034B00 30480241 00C5E23B 55D6AB22
 04AEF1BA A54028A6 9ACC01C5 129D99E4
 64CAB820 847EDAD9 DF0B4E4C 73A05DD2
 BD62A8A9 FA603DD2 E2A8A6F8 98F76E28
 D58AD221 B583D7A4 71020301 0001
 quit
 exit
 addressed-key 10.1.1.2 encryption
 key-string
 00302017 4A7D385B 1234EF29 335FC973
```

```
2DD50A37 C4F4B0FD 9DADE748 429618D5
18242BA3 2EDFBDD3 4296142A DDF7D3D8
08407685 2F2190A0 0B43F1BD 9A8A26DB
07953829 791FCDE9 A98420F0 6A82045B
90288A26 DBC64468 7789F76E EE21
quit
exit
addressed-key 10.1.1.2 signature
key-string
0738BC7A 2BC3E9F0 679B00FE 53987BCC
01030201 42DD06AF E228D24C 458AD228
58BB5DDD F4836401 2A2D7163 219F882E
64CE69D4 B583748A 241BED0F 6E7F2F16
0DE0986E DF02031F 4B0B0912 F68200C4
C625C389 0BFF3321 A2598935 C1B1
quit
exit
exit
```

Where to Go Next

After you have successfully configured IKE negotiation, you can begin configuring IPsec. For information on completing these tasks, see the module "Configuring Security for VPNs With IPsec."

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 configuration	Configuring Security for VPNs with IPsec
IKE Version 2	Configuring Internet Key Exchange Version 2
Configuring RSA keys to obtain certificates from a CA	Deploying RSA Keys Within a PKI
Suite-B ESP transforms	Configuring Security for VPNs with IPsec

Related Topic	Document Title
Suite-B support for certificate enrollment for a PKI	Configuring Certificate Enrollment for a PKI

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
RFC 2408	Internet Security Association and Key Management Protocol (ISAKMP)
RFC 2409	The Internet Key Exchange (IKE)
RFC 2412	The OAKLEY Key Determination 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 Configuring IKE for IPsec VPNs

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

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

Table 1: Feature Information for Configuring IKE for IPsec VPNs

Feature Name	Releases	Feature Information
Ability to Disable Extended Authentication for Static IPsec Peers	12.2(4)T	This 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. The following command was modified by this feature: **crypto isakmp key.**
Advanced Encryption Standard (AES)	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, crypto ipsec transform-set, show crypto isakmp policy.**
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.**
Suite-B support in IOS SW crypto	15.1(2)T	Suite-B adds support in the Cisco IOS for the SHA-2 family (HMAC variant) hash algorithm 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. This feature also adds elliptic curve Diffie-Hellman (ECDH) support for IPsec SA negotiation. See the Configuring Security for VPNs with IPsec feature module for more detailed information about Cisco IOS Suite-B support. The following command was modified by this feature: **authentication, crypto key generate ec keysize, crypto map, group, hash, set pfs.**