Implementing Internet Key Exchange Security Protocol on Cisco IOS XR Software

Internet Key Exchange (IKE) is a key management protocol standard that is used in conjunction with the IP Security (IPSec) standard. IPSec is a feature that provides robust authentication and encryption of IP packets.

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

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

This module describes the new and revised tasks you need to implement IKE on your Cisco IOS XR network.

For a complete description of the IKE commands used in this chapter, see the Internet Key Exchange Security Protocol Commands on Cisco IOS XR Software module of the Cisco IOS XR System Security Command Reference. To locate documentation of other commands that appear in this chapter, use the command reference master index, or search online.

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release 2.0</td>
<td>This feature was introduced on the Cisco CRS-1.</td>
</tr>
<tr>
<td>Release 3.0</td>
<td>No modification.</td>
</tr>
<tr>
<td>Release 3.2</td>
<td>Support was added for the Cisco XR 12000 Series Router.</td>
</tr>
<tr>
<td>Release 3.3.0</td>
<td>No modification.</td>
</tr>
</tbody>
</table>
Contents

- Prerequisites, page SC-18
- Information About Implementing Internet Key Exchange Security Protocol, page SC-18
- How to Implement IKE Security Protocol, page SC-26
- Additional References, page SC-40

Prerequisites

The following prerequisites are required to implement Internet Key Exchange:

- You must be in a user group associated with a task group that includes the proper task IDs for security commands. For detailed information about user groups and task IDs, see the Configuring AAA Services on Cisco IOS XR Software module of the Cisco IOS XR System Security Configuration Guide.
- You must install and activate the Package Installation Envelope (PIE) for the security software. For detailed information about optional PIE installation, see the Cisco IOS XR Getting Started Guide.

Information About Implementing Internet Key Exchange Security Protocol

To implement IKE, you should understand the following concepts:

- Supported Standards, page SC-18
- Concessions for Not Enabling IKE, page SC-20
- IKE Policies, page SC-20
- ISAKMP Identity, page SC-24
- Mask Preshared Keys, page SC-24
- Preshared Keys Using a AAA Server, page SC-25
- Internet Key Exchange Mode Configuration, page SC-25
- Internet Key Exchange Extended Authentication, page SC-26

Supported Standards

Cisco implements the following standards:

- IKE—Internet Key Exchange. A hybrid protocol that implements Oakley and Skeme key exchanges inside the ISAKMP framework. IKE can be used with other protocols, but its initial implementation is with the IPSec protocol. IKE provides authentication of the IPSec peers, negotiates IPSec keys, and negotiates IPSec security associations (SAs).

  IKE is implemented following RFC 2409, The Internet Key Exchange.
• 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 is used to protect one or more data flows between a pair of hosts, a pair of security gateways, or a security gateway and a host.

For more information on IPSec, see the Implementing IPSec Network Security on Cisco IOS XR Software module of the Cisco IOS XR System Security Configuration Guide.

• 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.

ISAKMP is implemented following the latest version of the Internet Security Association and Key Management Protocol (ISAKMP) Internet Draft (RFC 2408).

• 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.

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

• 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 XR 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 sent over untrusted networks. It enables customers, particularly in the finance industry, to use network-layer encryption.

• AES—Advanced Encryption Standard. 128-bit, 192-bit, and 256-bit AESs are supported.

Note Cisco IOS XR images that have strong encryption (including, but not limited to, 56-bit data encryption feature sets) are subject to U.S. 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 U.S. government regulations. Contact your sales representative or distributor for more information, or send e-mail to export@cisco.com.

• 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. 768-bit, 1024-bit, and 1536-bit Diffie-Hellman groups are supported.

• MD5 (HMAC variant)—Message Digest 5. A hash algorithm used to authenticate packet data. HMAC is a variant that provides an additional level of hashing.

• SHA (HMAC variant)—Secure Hash Algorithm. A hash algorithm used to authenticate packet data. HMAC is a variant that provides an additional level of hashing.

• RSA signatures and RSA encrypted nonces—RSA is the public key cryptographic system developed by Ron Rivest, Adi Shamir, and Leonard Adelman. RSA signatures provide nonrepudiation, and RSA encrypted nonces provide repudiation. (Repudiation and nonrepudiation are associated with traceability.)
IKE interoperates with the X.509v3 certificates standard. It is 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 want 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).

Concessions for Not Enabling IKE

IKE is disabled by default in Cisco IOS XR software. If you do not enable IKE, you must make these concessions at the peers:

- You must manually specify all IPSec security associations in the crypto profiles at all peers. (Crypto profile configuration is described in the module Implementing IPSec Network Security on Cisco IOS XR Software.)
- The IPSec security associations of the peer s never timeout for a given IPSec session.
- During IPSec sessions between the peers, the encryption keys never change.
- Antireplay services are not available between the peers.
- Certification authority (CA) support cannot be used.

IKE Policies

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

Before you create and configure IKE policies you should understand the following concepts:

- IKE Policy Creation, page SC-20
- Definition of Policy Parameters, page SC-21
- IKE Peer Agreement for Matching Policies, page SC-21
- Value Selection for Parameters, page SC-22
- Policy Creation, page SC-23
- Additional Configuration Required for IKE Policies, page SC-23

IKE Policy Creation

IKE negotiations must be protected, so 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 on a policy, the security parameters of the policy are identified by a security association established at each peer, and these security associations apply to all subsequent IKE traffic during the negotiation.

You can create multiple, prioritized policies at each peer to ensure that at least one policy matches the policy of a remote peer.
Definition of Policy Parameters

Table 2 lists the five parameters to define in each IKE policy.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Accepted Values</th>
<th>Keyword</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encryption algorithm</td>
<td>56-bit DES-CBC 168-bit DES 128-bit AES 192-bit AES 256-bit AES</td>
<td>des</td>
<td>56-bit DES-CBC</td>
</tr>
<tr>
<td>Hash algorithm</td>
<td>SHA-1 (HMAC variant) MD5 (HMAC variant)</td>
<td>sha</td>
<td>SHA-1</td>
</tr>
<tr>
<td>Authentication method</td>
<td>RSA signatures RSA encrypted nonces Preshared keys</td>
<td>rsa-sig</td>
<td>RSA signatures</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rsa-encr</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pre-share</td>
<td></td>
</tr>
<tr>
<td>Diffie-Hellman group</td>
<td>768-bit Diffie-Hellman or 1024-bit Diffie-Hellman 1536-bit Diffie-Hellman</td>
<td>1</td>
<td>768-bit Diffie-Hellman</td>
</tr>
<tr>
<td>identifier</td>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Lifetime of the security</td>
<td>Any number of seconds — 86400 seconds (1 day)</td>
<td>—</td>
<td>86400 seconds (1 day)</td>
</tr>
<tr>
<td>association</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. For information about this lifetime and how it is used, see the command description for the `lifetime` command.

These parameters apply to the IKE negotiations when the IKE security association is established.

IKE Peer Agreement for Matching Policies

When the IKE negotiation begins, IKE looks 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, and when the remote peer’s policy specifies a lifetime that is less than or equal to the lifetime in the policy being compared. (If the lifetimes are not identical, the shorter lifetime—from the remote peer’s policy—is used.)

If no acceptable match is found, IKE refuses negotiation and IPSec is not established.

If a match is found, IKE completes negotiation, and IPSec security associations are created.

Note: Depending on which authentication method is specified in a policy, additional configuration might be required (as described in the “Additional Configuration Required for IKE Policies” section on page SC-23). If a peer’s policy does not have the required companion configuration, the peer does not submit the policy when attempting to find a matching policy with the remote peer.
Value Selection for Parameters

You can select certain values for each parameter, following the IKE standard. But why choose one value over another?

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, a trade-off between security and performance often exists, 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. Then the following tips might help you select which value to specify for each parameter:

- The encryption algorithm has five options: 56-bit DES-CBC, 168-bit DES, 128-bit AES, 192-bit AES, and 256-bit AES.
- The hash algorithm has two options: SHA-1 and MD5.
  MD5 has a smaller digest and is considered to be slightly faster than SHA-1. A demonstrated successful (but extremely difficult) attack has been demonstrated against MD5; however, the HMAC variant used by IKE prevents this attack.
- The authentication method has three options: RSA signatures, RSA encrypted nonces, and preshared keys.
  - RSA signatures provide nonrepudiation for the IKE negotiation (you can prove to a third party after the fact that you did indeed have an IKE negotiation with the remote peer).
  - RSA signatures allow the use of a CA. 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 RAS encryption uses four public key operations, making it costlier in terms of overall performance.
  - RSA encrypted nonces provide repudiation for the IKE negotiation (you cannot prove to a third party that you had an IKE negotiation with the remote peer).
  - RSA encrypted nonces require that peers possess each other’s public keys but do not use a certification authority. Instead, two ways exist for peers to get each other’s public keys:
    - During configuration, you manually configure RSA keys (as described in the “Manually Configuring RSA Keys” section on page SC-30).
    - If your local peer has previously used RSA signatures with certificates during a successful IKE negotiation with a remote peer, your local peer already possesses the remote peer’s public key. (The peers’ public keys are exchanged during the RSA-signatures-based IKE negotiations, if certificates are used.)
  - 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 certification authority, 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.
- The Diffie-Hellman group identifier has three options: 768-bit, 1024-bit Diffie-Hellman, and 1536-bit Diffie Helman.
  The 1024-bit Diffie-Hellman and 1536-bit Diffie Helman options are harder to crack but require more CPU time to execute.
- The lifetime of the security association can be set to any value.
As a general rule, the shorter the lifetime (up to a point), the more secure your IKE negotiations. However, with longer lifetimes, future IPSec security associations can be set up more quickly. For more information about this parameter and how it is used, see the command description for the `lifetime` command.

**Policy Creation**

You can create multiple IKE policies, each with a different combination of parameter values. For each policy that you create, assign a unique priority (1 through 10,000, with 1 being the highest priority).

You can configure multiple policies on each peer—but 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. (The lifetime parameter need not necessarily be the same; see details in the “IKE Peer Agreement for Matching Policies” section on page SC-21.)

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

**Additional Configuration Required for IKE Policies**

Depending on the authentication method you specify in your IKE policies, you must perform certain additional configuration tasks before IKE and IPSec can successfully use the IKE policies.

Each authentication method requires additional companion configuration as follows:

- **RSA signatures method.** If you specify RSA signatures as the authentication method in a policy, you may configure the peers to obtain certificates from a CA. (The CA must be properly configured to issue the certificates.) Configure this certificate support as described in the module “Implementing Certification Authority Interoperability.”

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

  You may also want to exchange the public keys manually, as described in the “Manually Configuring RSA Keys” section on page SC-30.

- **RSA encrypted nonces method.** If you specify RSA encrypted nonces as the authentication method in a policy, 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 others’ public keys by one of the following methods:

    - Manually configuring RSA keys, as described in the “Manually Configuring RSA Keys” section on page SC-30.
    
    - 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 this happen, 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 are used the first time because the peers do not yet have each other’s public keys. Then future IKE negotiations are able to use RSA encrypted nonces because the public keys will have been exchanged.

  This alternative requires that you have certification authority support configured.
• Preshared keys authentication method. If you specify preshared keys as the authentication method in a policy, you must configure these preshared keys as described in the “Configuring Preshared Keys” section on page SC-34.

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

**ISAKMP Identity**

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 security associations, each peer sends its identity to the remote peer. Each peer sends either its host name or its IP address, depending on how you have set the ISAKMP identity of the router.

By default, the ISAKMP identity of a peer is the IP address of the peer. If appropriate, you could change the identity to be the peer's host name 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 host names. If some peers use their host names 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 server (DNS) lookup is unable to resolve the identity.

**Mask Preshared Keys**

A mask preshared key lets a group of remote users with the same level of authentication 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 mask 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 a subnet-address value with the `crypto isakmp key` command, it is up to you to use a subnet address, which allows more peers to share the same key. That is, the preshared key is no longer restricted to use between two users.

---

**Note**

We do not recommend using 0.0.0.0 as a subnet address, because it encourages group preshared keys, which allow all peers to have the same group key, thereby reducing the security of your user authentication.

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.
Preshared Keys Using a AAA Server

Preshared keys do not scale well when you are trying to deploy a large scale Virtual Private Network (VPN) without using a CA. When dynamic IP addressing such as DHCP or PPP dialups is used, the changing IP address can make key lookup difficult or impossible unless a mask preshared key is used. However, mask preshared keys are not very secure because a large number of users are given the same secret, thus reducing the security of the secret.

Configuring preshared keys using an authentication, authorization, and accounting (AAA) server allows each user to have his or her own key, which is stored on an external AAA server. This allows for central management of the user database, linking it to an existing AAA database, in addition to allowing every user to have a unique, more secure preshared key.

To configure this feature, you must perform the following tasks at each peer:

- Configure AAA.
- Configure an IPSec transform set.
- Configure a dynamic crypto profile.
- Configure extended authentication (optional)
- Configure ISAKMP policy.

For information on configuring IPSec transform sets and crypto profiles, including enabling an IPSec peer for preshared keys using a AAA server, see the “Configuring Crypto Profiles” section on page SC-67.

Preshared keys using an AAA server have the following restrictions:

- The shared secret can be accessed only in aggressive mode. The ID of the IKE exchange is used as the username to query AAA if no local key can be found on the Cisco IOS XR router to which the user is trying to connect. Aggressive mode provides the ID in the first part of the IKE exchange; main mode does not provide the ID until the latter part of the IKE exchange, which is too late for key lookup.
- Only the following ID types can be used:
  - IPv4 address (can be different from the one assigned by the Internet service provider [ISP])
  - FQDN (fully qualified domain name)
  - E-mail address

Internet Key Exchange 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 IP addresses 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 scalable policy for a very large set of clients irrespective of the IP addresses of those clients.

The client initiation type of IKE mode configuration is supported. The client initiates the configuration mode with the gateway. The gateway responds with an IP address that it has allocated for the client.
Mode configuration must be applied to a crypto profile to be enforced. For instructions on how to apply mode configuration to a crypto profile, see the “Configuring Crypto Profiles” section on page SC-67.

IKE mode configuration has the following restrictions:

- Interfaces with crypto profiles that are configured for IKE mode configuration may experience a slightly longer connection setup time. This longer setup time is true even for IKE peers that refuse to be configured or do not respond to the configuration mode request. In both cases, the gateway initiates the configuration of the client.
- This feature was not designed to enable the configuration mode for every IKE connection by default. Configure this feature at the global crypto profile level.
- The following items in the IETF draft are not currently supported:
  - Configuration attributes other than INTERNAL_IP_ADDRESS
  - Unprotected exchanges

**Internet Key Exchange Extended Authentication**

IKE extended authentication (Xauth) is a draft RFC based on the IKE protocol. Xauth allows all Cisco IOS XR software AAA authentication methods to perform user authentication in a separate phase after the IKE authentication phase 1 exchange. The AAA configuration list name must match the Xauth configuration list name for user authentication to occur.

Xauth does not replace IKE. IKE allows for device authentication and Xauth allows for user authentication, which occurs after IKE device authentication. Xauth occurs after IKE authentication phase 1 but before IKE IPSec SA negotiation phase 2.

To configure Xauth, perform the following tasks:

- Configure AAA (you must set up an authentication list).
- Configure an IPSec transform.
- Configure a static crypto profile.
- Configure ISAKMP policy.
- Configure a dynamic crypto profile (optional).

For information on configuring IPSec transform sets and crypto profiles, see the Implementing IPSec Network Security on Cisco IOS XR Software module.

**How to Implement IKE Security Protocol**

To configure the IKE security protocol, perform the tasks described in the following sections. The tasks in the first three sections are required; the remaining may be optional, depending on which parameters are configured.

- Enabling or Disabling IKE, page SC-27 (required)
- Configuring IKE Policies, page SC-28 (required)
- Manually Configuring RSA Keys, page SC-30 (optional, depending on IKE parameters)
- Configuring Preshared Keys, page SC-34 (optional, depending on IKE parameters)
- Configuring Mask Preshared Keys, page SC-37 (optional, depending on IKE parameters)
Enabling or Disabling IKE

This task enables or disables the Internet Key Exchange protocol.
IKE is disabled by default. IKE need not be enabled for individual interfaces, but it is enabled globally for all interfaces at the router.

SUMMARY STEPS

1. configure
2. crypto isakmp enable
3. no crypto isakmp enable
4. end
   or
   commit

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td></td>
</tr>
<tr>
<td>configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td></td>
</tr>
<tr>
<td>crypto isakmp enable</td>
<td>Globally enables IKE at the peer router.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RP0/CPU0:router(config)# crypto isakmp enable</td>
<td></td>
</tr>
</tbody>
</table>
### Configuring IKE Policies

This task configures IKE policies.

**SUMMARY STEPS**

1. configure
2. crypto isakmp policy *priority*
3. encryption {des | 3des | aes | aes 192 | aes 256}
4. hash {sha | md5}
5. authentication {pre-share | rsa-sig | rsa-encr}
6. group {1 | 2 | 5}
7. lifetime *seconds*
8. end
   or
   commit
9. show crypto isakmp policy

### Command or Action | Purpose
--- | ---
**Step 3** no crypto isakmp enable | (Optional) Disables IKE at the peer router.  

*Example:*  
RP/0/RP0/CPU0:router(config)# no crypto isakmp enable

**Step 4** end  
or  
commit | Saves configuration changes.  

- When you issue the **end** command, the system prompts you to commit changes:  
  Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:  
  - Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.  
  - Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.  
  - Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.  
  - Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
## Detailed Steps

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong>&lt;br&gt;configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong>&lt;br&gt;RP/0/RP0/CPU0:router# configure</td>
<td></td>
</tr>
</tbody>
</table>
| **Step 2**<br>crypto isakmp policy priority | Identifies the policy to create.  
- Each policy is uniquely identified by the priority number you assign.  
- This command places the router in ISAKMP policy configuration mode. |
| **Example:**<br>RP/0/RP0/CPU0:router(config)# crypto isakmp policy 5 | |
| **Step 3**<br>encryption {des | 3des | aes | aes 192 | aes 256} | Specifies the encryption algorithm. |
| **Example:**<br>RP/0/RP0/CPU0:router(config-isakmp)# encryption aes | |
| **Step 4**<br>hash {sha | md5} | Specifies the hash algorithm. |
| **Example:**<br>RP/0/RP0/CPU0:router(config-isakmp)# hash md5 | |
| **Step 5**<br>authentication {pre-share | rsa-sig | rsa-encr} | Specifies the authentication method. |
| **Example:**<br>RP/0/RP0/CPU0:router(config-isakmp)# authentication rsa-sig | |
| **Step 6**<br>group {1 | 2 | 5} | Specifies the Diffie-Hellman group identifier. |
| **Example:**<br>RP/0/RP0/CPU0:router(config-isakmp)# group 5 | |
| **Step 7**<br>lifetime seconds | Specifies the lifetime of the security association. The range, in seconds, is from 60 to 86400. |
| **Example:**<br>RP/0/RP0/CPU0:router(config-isakmp)# lifetime 50000 | |
How to Implement IKE Security Protocol

Manually Configuring RSA Keys

Manually configure RSA keys when you specify RSA encrypted nonces as the authentication method in an IKE policy and you are not using a CA.

To manually configure RSA keys, perform these tasks at each IPSec peer that uses RSA encrypted nonces in an IKE policy:

- RSA Keys Generation, page SC-30
- Configuring ISAKMP Identity, page SC-30
- Configuring RSA Public Keys of All the Other Peers, page SC-32

### RSA Keys Generation

For instructions on how to generate RSA keys, see the “Generating an RSA Key Pair” section on page SC-7 in the Implementing Certification Authority Interoperability on Cisco IOS XR Software module.

### Configuring ISAKMP Identity

This task configures the ISAKMP identity of a peer.

Remember to repeat these tasks at each peer that uses preshared keys in an IKE policy.
**SUMMARY STEPS**

1. configure
2. crypto isakmp identity {address | hostname}
3. host hostname address1 [address2...address8]
4. end
   or
   commit

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Example: RP/0/RP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> crypto isakmp identity {address</td>
<td>hostname}</td>
</tr>
<tr>
<td>Example: RP/0/RP0/CPU0:router(config)# crypto isakmp identity address</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> host hostname address1 [address2...address8]</td>
<td>(At all remote peers) Maps the peer's host name to its IP addresses at all the remote peers.</td>
</tr>
</tbody>
</table>
| Example: RP/0/RP0/CPU0:router(config)# host host1 10.0.0.5 | • This command is used if the local peer’s ISAKMP identity was specified using a host name.  
• This step might be unnecessary if the host name or address is already mapped in a DNS server. |
| **Step 4** end  
or  
commit | Saves configuration changes. |
| Example: RP/0/RP0/CPU0:router(config)# end  
or  
RP/0/RP0/CPU0:router(config)# commit | • When you issue the end command, the system prompts you to commit changes: |
| | Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]: |
| | • Entering yes saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. |
| | • Entering no exits the configuration session and returns the router to EXEC mode without committing the configuration changes. |
| | • Entering cancel leaves the router in the current configuration session without exiting or committing the configuration changes. |
| | • Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session. |
Configuring RSA Public Keys of All the Other Peers

This task configures the RSA public keys of all the other peers.

Remember to repeat these tasks at each peer that uses RSA encrypted nonces in an IKE policy.

**SUMMARY STEPS**

1. `configure`
2. `crypto key pubkey-chain rsa`
3. `named-key key-name [encryption | signature]`
   or
   `addressed-key key-address [encryption | signature]`
4. `address ip-address`
5. `key-string key-string`
6. `exit`
7. `end`
   or
   `commit`
8. `show crypto key pubkey-chain rsa [name key-name | address key-address]`

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td></td>
</tr>
<tr>
<td><code>configure</code></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RP0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td></td>
</tr>
<tr>
<td><code>crypto key pubkey-chain rsa</code></td>
<td>Specifies RSA public keys for other IPSec peers.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>RP/0/RP0/CPU0:router(config)# crypto key pubkey-chain rsa</td>
<td>Enters public key chain configuration mode.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Command or Action

**Step 3**

```plaintext
named-key key-name [encryption | signature]
```

or

```plaintext
addressed-key key-address [encryption | signature]
```

**Example:**

```plaintext
RP/0/RP0/CPU0:router(config-pubkey-chain)#
named-key otherpeer.example.com
```

or

```plaintext
Example:
RP/0/RP0/CPU0:router(config-pubkey-chain)#
addressed-key 10.5.5.1
```

### Purpose

Indicates which remote peer’s RSA public key you will specify.

- If the remote peer uses its host name as its ISAKMP identity, use the `named-key` command and specify the remote peer’s fully qualified domain name (such as somerouter.example.com) as the `key-name` value.
- Follow this command with the `key-string` command to specify the key.
- If you use the `named-key` command, you also need to use the `address` command to specify the IP address of the peer.

**Step 4**

```plaintext
address ip-address
```

**Example:**

```plaintext
RP/0/RP0/CPU0:router(config-pubkey-key)#
address 10.5.5.1
```

### Purpose

Specifies the remote peer’s IP address.

- You can use this command if you used a fully qualified domain name to name the remote peer in Step 3 (using the `named-key` command).

**Step 5**

```plaintext
key-string key-string
```

**Example:**

```plaintext
RP/0/RP0/CPU0:router(config-pubkey-key)#
key-string 005C300D 06092A86 4886F70D 01010105 ...
```

### Purpose

Specifies the remote peer’s RSA public key.

- This is the key previously displayed by the remote peer’s administrator when the remote router’s RSA keys were generated.
- To avoid mistakes, you should cut and paste the key data (instead of attempting to enter in the data).
- Enter a key on each line. You must enter the `key-string` command before the key.
- When you have finished specifying the remote peer’s RSA key, return to global configuration mode by entering `exit` at the config-pubkey-key prompt.
How to Implement IKE Security Protocol

### Command or Action

<table>
<thead>
<tr>
<th>Step 6</th>
<th>exit</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/RP0/CPU0:router(config-pubkey-key)# exit</td>
</tr>
<tr>
<td>Purpose</td>
<td>Returns to global configuration mode.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 7</th>
<th>end or commit</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/RP0/CPU0:router(config)# end or RP/0/RP0/CPU0:router(config)# commit</td>
</tr>
<tr>
<td>Purpose</td>
<td>Saves configuration changes.</td>
</tr>
<tr>
<td></td>
<td>• When you issue the <strong>end</strong> command, the system prompts you to commit changes:</td>
</tr>
<tr>
<td></td>
<td>Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:</td>
</tr>
<tr>
<td></td>
<td>– Entering <strong>yes</strong> saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.</td>
</tr>
<tr>
<td></td>
<td>– Entering <strong>no</strong> exits the configuration session and returns the router to EXEC mode without committing the configuration changes.</td>
</tr>
<tr>
<td></td>
<td>– Entering <strong>cancel</strong> leaves the router in the current configuration session without exiting or committing the configuration changes.</td>
</tr>
<tr>
<td></td>
<td>• Use the <strong>commit</strong> command to save the configuration changes to the running configuration file and remain within the configuration session.</td>
</tr>
</tbody>
</table>

| Step 8 | show crypto pubkey-chain rsa [name key-name | address key-address] |
|--------|-----------------------|
| **Example:** | RP/0/RP0/CPU0:router# show crypto pubkey-chain rsa (Optional) Displays a list of all the RSA public keys stored on your router. |
| | • Use the optional name or address keyword to display details about a particular RSA public key stored on your router. |

### Configuring Preshared Keys

This task configures preshared keys.

### Prerequisites

To configure preshared keys, perform these tasks at each peer that uses preshared keys in an IKE policy:

- Set the ISAKMP identity of each peer. Each peer’s identity should be set either to its host name or by its IP address. By default, a peer’s identity is set to its IP address. Setting ISAKMP identities is described in the “Configuring ISAKMP Identity” section on page SC-30.

- Specify the shared keys at each peer. Note that a given preshared key is shared between two peers. At a given peer you could specify the same key to share with multiple remote peers; however, a more secure approach is to specify different keys to share between different pairs of peers.

Remember to repeat these tasks at each peer that uses preshared keys in an IKE policy.
### SUMMARY STEPS

1. `configure`
2. `crypto isakmp key keystring address peer-address [subnet-address]`
   or
   `crypto isakmp key keystring hostname peer-hostname`
3. `crypto isakmp key keystring address peer-address [subnet-address]`
   or
   `crypto isakmp key keystring hostname peer-hostname`
4. `end`
   or
   `commit`

### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| **Step 1**
  `configure`  | Enters global configuration mode. |

Example:
```
RP/0/RP0/CPU0:router# configure
```

| **Step 2**
  `crypto isakmp key keystring address peer-address [subnet-address]`
  or
  `crypto isakmp key keystring hostname peer-hostname`  | (At the local peer) Specifies the shared key to be used with a particular remote peer. |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>• You must configure this key whenever you specify preshared keys in an IKE policy. You must enter this command at both peers.</td>
<td></td>
</tr>
<tr>
<td>• If the remote peer specified its ISAKMP identity with an address, use the <code>address</code> keyword in this step; otherwise use the <code>hostname</code> keyword in this step.</td>
<td></td>
</tr>
</tbody>
</table>

Example:
```
RP/0/RP0/CPU0:router(config)# crypto isakmp key sharedkeystring address 172.16.0.2
```

or
```
RP/0/RP0/CPU0:router(config)# crypto isakmp key sharedkeystring hostname remoterouter.domain.com
```
### Step 3

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>crypto isakmp key keystring address peer-address [subnet-address]</code></td>
<td>(At the remote peer) Specifies the shared key to be used with the local peer. This is the same key you just specified at the local peer.</td>
</tr>
<tr>
<td>or <code>crypto isakmp key keystring hostname peer-hostname</code></td>
<td>- If the local peer specified its ISAKMP identity with an address, use the <code>address</code> keyword in this step; otherwise use the <code>hostname</code> keyword in this step.</td>
</tr>
</tbody>
</table>

**Example:**

```
RP/0/RP0/CPU0:router(config)# crypto isakmp key sharedkeystring address 10.0.0.1
```

or

```
RP/0/RP0/CPU0:router(config)# crypto isakmp key sharedkeystring hostname localrouter.example.com
```

### Step 4

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>end</code> or <code>commit</code></td>
<td>Saves configuration changes.</td>
</tr>
</tbody>
</table>

**Example:**

```
RP/0/RP0/CPU0:router(config)# end
```

or

```
RP/0/RP0/CPU0:router(config)# commit
```

- When you issue the `end` command, the system prompts you to commit changes:
  
  Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:
  
  - Entering `yes` saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
  
  - Entering `no` exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
  
  - Entering `cancel` leaves the router in the current configuration session without exiting or committing the configuration changes.

- Use the `commit` command to save the configuration changes to the running configuration file and remain within the configuration session.
Configuring Mask Preshared Keys

This task configures mask preshared keys at each peer.

Prerequisites

To configure mask preshared keys, perform these tasks at each peer that uses mask preshared keys in an IKE policy:

- Set the ISAKMP identity of each peer. Each peer's identity should be set either to its host name or by its IP address. By default, a peer's identity is set to its IP address. Setting ISAKMP identities is described in the “Configuring ISAKMP Identity” section on page SC-30.

- Specify the shared keys at each peer. Note that a given mask preshared key is shared between two peers. At a given peer you could specify the same key to share with multiple remote peers; however, a more secure approach is to specify different keys to share between different pairs of peers.

Remember to repeat these tasks at each peer that uses mask preshared keys in an IKE policy.

SUMMARY STEPS

1. configure
2. crypto isakmp key keystring address peer-address [subnet-address]
3. end
   or commit

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1 configure</td>
<td>Enters global configuration mode.</td>
</tr>
</tbody>
</table>

Example:
RP/0/RP0/CPU0:router# configure
Creating IKE Policies: Example

This example shows how to create 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
```

Configuration Examples for Implementing IKE Security Protocol

This section provides the following configuration examples:

- Creating IKE Policies: Example, page SC-38
- Configuring Xauth with Static Crypto Profile: Example, page SC-39
In the example, the encryption des of policy 15 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:DES - Data Encryption Standard (56 bit keys)
- hash algorithm:Message Digest 5
- authentication method:Rivest-Shamir-Adelman 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-Adelman Signature
- Diffie-Hellman group:1 (768 bit)
- lifetime:86400 seconds, no volume limit

**Note**
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.

### Configuring Xauth with Static Crypto Profile: Example

In the following example, Xauth is configured with preshared key using AAA local policy:

```plaintext
aaa new-model
aaa authentication login xauthlist local
username user1 password sample1234
crypto ipsec transform-set xauthtransform esp-des esp-md5-hmac
! crypto isakmp policy 1
hash md5
authentication pre-share
crypto isakmp key sample1234 address 192.168.202.145
! crypto profile xauthprofile
  match 192 transform-set xauthtransform
! interface Ethernet1/0
  ip address 192.168.202.147 255.255.255.224
  crypto profile xauthprofile
```
Additional References

The following sections provide references related to implementing the IKE security protocol.

Related Documents

<table>
<thead>
<tr>
<th>Related Topic</th>
<th>Document Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>IKE security protocol commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples</td>
<td>Internet Key Exchange Security Protocol Commands on Cisco IOS XR Software</td>
</tr>
<tr>
<td>IPSec conceptual information and configuration tasks</td>
<td>Implementing IPSec Network Security on Cisco IOS XR Software</td>
</tr>
</tbody>
</table>

Standards

<table>
<thead>
<tr>
<th>Standards</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.</td>
<td>—</td>
</tr>
</tbody>
</table>

MIBs

<table>
<thead>
<tr>
<th>MIBs</th>
<th>MIBs Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>There are no applicable MIBs for this module.</td>
<td>To locate and download MIBs for selected platforms using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL: <a href="http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml">http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml</a></td>
</tr>
</tbody>
</table>

RFCs

<table>
<thead>
<tr>
<th>RFCs</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFC 2409</td>
<td>The Internet Key Exchange</td>
</tr>
<tr>
<td>RFC 2408</td>
<td>Internet Security Association and Key Management Protocol (ISAKMP) Internet Draft</td>
</tr>
</tbody>
</table>
## Technical Assistance

<table>
<thead>
<tr>
<th>Description</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Cisco Technical Support website contains thousands of 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.</td>
<td><a href="http://www.cisco.com/techsupport">http://www.cisco.com/techsupport</a></td>
</tr>
</tbody>
</table>