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 how to implement IKE on the Cisco IOS XR Software.

Note 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 module, use the command reference master index, or search online.

Feature History for Implementing Internet Key Exchange Security Protocol on Cisco XR 12000 Series Router

Release	Modification
Release 3.2	This feature was introduced on the Cisco XR 12000 Series Router.
Release 3.3.0	No modification.
Release 3.4.0	• Support was added for IKE . • Support was added to implement IKE for locally sourced and destined traffic.
Release 3.5.0	• The IP Security VPN Monitoring feature was added. • Banner, Auto-Update, and Browser-Proxy features were added to aid in managing a Cisco Easy VPN remote device. • Pushing a configuration URL through a mode-configuration exchange feature was supported.

Release 3.6.0	Information was introduced on how to limit an IKE peer to a predefined policy set in the context of IPSec. The existing example on how to configure Cisco Easy VPN for use with a local AAA-method server was updated and an example was introduced for how to configure this for a remote AAA-method server. Conceptual information was introduced (Information About Cisco Easy VPN and the Cisco Easy VPN Server, page 168) to explain what Cisco Easy VPN is and to provide context for the sections Cisco Easy VPN Server, page 169 and Configuring Client Group Attributes for Cisco Easy VPN Server, page 180 Four existing Cisco Easy VPN procedures were incorporated into one new procedure titled Configuring Client Group Attributes for Cisco Easy VPN Server, page 180. A duplicate procedure titled *How to Configure an ISAKMP Profile with Locally Sourced and Destined Traffic* was removed. The same information appears in the previously existing How to Configure the ISAKMP Profile, page 201. Cross references to configuration procedures that explain how to complete some previously referenced tasks were introduced to help readers locate the information in the current chapter or in other chapters. Some information in the chapter was reorganized to improve readability.
Release 3.7.0	No modification.
Release 3.8.0	Information was edited to make clearer which features are supported on the Cisco XR 12000 Series Router exclusively.
Release 3.9.0	No modification.

Contents

Prerequisites for Implementing Internet Key Exchange

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. The command reference guides include the task IDs required for each command.
- If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.
- You must install and activate the package installation envelope (PIE) for the security software.

  For detailed information about optional PIE installation, see *Cisco IOS XR System Management Configuration Guide*.

Information About Implementing IKE Security Protocol Configurations for IPSec Networks

To implement IKE, you should understand the following concepts:

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 Network 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. Standards of 128-bit, 192-bit, and 256-bit 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 insecure 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* in *System Security Configuration Guide*.)
- The IPSec security associations of the peers never time out for a given IPSec session.
- During IPSec sessions between the peers, the encryption keys never change.
- Anti-replay 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 159
- Definition of Policy Parameters, page 159
- IKE Peer Agreement for Matching Policies, page 160
- Limitation of an IKE Peer to a Specific Set of Policies, page 161
- Value Selection for Parameters, page 161
- Policy Creation, page 162
- Additional Configuration Required for IKE Policies, page 162

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 1 lists the five parameters to define in each IKE policy.

Table 1 IKE Policy Parameter Definitions

Parameter	Accepted Values	Keyword	Default Value
Encryption algorithm	56-bit DES-CBC	**des**	56-bit DES-CBC
	168-bit DES	**3des**	
	128-bit AES	**aes**	
	192-bit AES	**aes 192**	
	256-bit AES	**aes 256**	
Hash algorithm	SHA-1 (HMAC variant)	**sha**	SHA-1
	MD5 (HMAC variant)	**md5**	
Authentication method	RSA signatures	**rsa-sig**	RSA signatures
	RSA encrypted nonces	**rsa-encr**	
	Preshared keys	**pre-share**	
Diffie-Hellman group identifier	768-bit Diffie-Hellman or	**1**	768-bit Diffie-Hellman
	1024-bit Diffie-Hellman	**2**	
	1536-bit Diffie-Hellman	**5**	
Lifetime of the security association[1]	Any number of seconds	—	86400 seconds (1 day)

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 sends 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 (designated by the lowest priority number) 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 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. (See related information in the "Limitation of an IKE Peer to a Specific Set of Policies" section.)

If a match is found, IKE completes negotiation, and an ISAKMP security association (SA) is created.To establish an ISAKMP SA pre-shared key or certificate, a match must be configured. Without a match, no ISAKMP SA can be established.

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 162). 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.

Limitation of an IKE Peer to a Specific Set of Policies

Cisco VPN clients are preconfigured with all available policies, and propose all of these policies when connecting to the hub. The hub must then select the "first-match" policy. However, some users may have a need to restrict the use of strong encryption algorithms between the local and remote peer when they connect through the IPSec gateway. Because the Cisco VPN client does not allow users to choose which policy (and therefore which encryption algorithm) to use, these users may instead configure policy sets that in effect create such restrictions. Matches between peer and policy set are then restricted or allowed, based on a match with the local IP address (or tunnel source configured at the SVI) identified in the policy set.

For example, an IPSec hub is configured with six policies, but the policy set is configured with only three of these six. When a remote client tries to initiate a tunnel and refers to this SVI tunnel source address, the policy set is matched. IKE looks for a match among the three policies dictated by the policy set, starting from the highest to the lowest priority number (the lower the number, the higher the priority). If no match exists among these three policies, no tunnel can be established.

If a remote peer tries to connect to an SVI, whose local IP address does not restrict it to certain IKE policies, then the default behavior described under "IKE Peer Agreement for Matching Policies" is operational.

You may configure up to five ISAKMP policies within a single policy set.

For information about how to limit an IKE peer to a specific set of policies, see Limiting an IKE Peer to Use a Specific Policy Set, page 178 of this module.

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.

    You can also exchange the public keys manually, as described in the "Manually Configuring RSA Keys" section on page 184.

  - 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 184).
  - 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 Hellman.

  The 1024-bit Diffie-Hellman and 1536-bit Diffie Hellman 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 160.)

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 184.

- 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 184.
  - 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 ISAKMP Preshared Keys in ISAKMP Keyrings" section on page 192.

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 hostname 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 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 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.

ISAKMP Profile Overview

The ISAKMP profile is an enhancement to Internet Security Association and Key Management Protocol (ISAKMP) configurations. It enables modularity of ISAKMP configuration for Phase-1 negotiations. This modularity allows mapping different ISAKMP parameters to different IP Security (IPSec) tunnels, and mapping different IPSec tunnels to different VPN forwarding and routing (VRF) instances. Currently, many applications and enhancements use the ISAKMP profile, including quality of service (QoS), router certificate management, and Multiprotocol Label Switching (MPLS) VPN configurations.

An ISAKMP profile is a repository for IKE Phase-1 and IKE Phase-1.5 (also known as Xauth) configuration for a set of peers. An ISAKMP profile applies parameters to an incoming IPSec connection identified uniquely through its concept of match identity criteria. These criteria are based on the IKE identity that is presented by incoming IKE connections and includes IP address, fully qualified domain name (FQDN), and group (the Virtual Private Network [VPN] remote client grouping). The granularity of the match identity criteria imposes the granularity of applying the specified parameters. The ISAKMP profile applies parameters specific to each profile, such as trust points, peer identities, and Xauth authentication, authorization, and accounting (AAA) list, and so forth. For information about Xauth, see Internet Key Exchange Mode Configuration, page 166.

Consider the following guidelines on when to use the ISAKMP profile:

- You have a router with two or more IPSec connections that require differing Phase-1 parameters for different peers (for example, when you want to configure site-to-site and remote access on the same router).
- You have an IPSec configuration using VRF-aware IPSec, which allows the use of single IP address to connect to different peers with different IKE Phase-1 parameters. For an example of this configuration, see Configuring VRF-Aware: Example, page 212.
- When different custom Internet Key Exchange (IKE) Phase-1 policies may be needed for different peers. One determining factor might be whether you are applying Xauth to a specific peer, rather than applying it to every connection.

Note Remote-access IPSec, VRF-aware IPSec, and Xauth are supported only on the Cisco XR 12000 Series Router.

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 keyring** 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:

- A security association (SA) cannot be established between the IPSec peers until all IPSec peers are configured for the same preshared key. (An SA is a description of how two or more entities use security services to communicate securely on behalf of a particular data flow.)
- 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 keys individually, as appropriate, to each party. Otherwise, an untrusted party may obtain access to protected data.

Preshared Keys Using an AAA-Method Server

Preshared keys do not scale well in a large Virtual Private Network (VPN) unless you use a certification authority (CA). When dynamic IP addressing such as DHCP or PPP dialup is used, the changing IP address can make key lookup difficult or impossible unless you use a mask preshared key. On the other hand, mask preshared keys are not very secure, because then large number of users receive the same secret, thereby reducing security.

Configuring preshared keys using an authentication, authorization, and accounting (AAA) server allows individual users to have their own key, which is stored on an external AAA server. This makes it possible to centrally manage the user database and to link it to an existing AAA database. It also gives each user a unique, more secure preshared key.

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

- Configure the AAA server. See Configuring ISAKMP Identity, page 184 and Configuring ISAKMP Preshared Keys in ISAKMP Keyrings, page 192.
- Configure a dynamic crypto ISAKMP profile. See Defining Group Policy Information for Mode Configuration.
- Configure extended authentication (optional). See Internet Key Exchange Extended Authentication, page 166.
- Configure ISAKMP policy. See Configuring IKE Policies, page 172.

Restrictions to Preshared Keys Using an AAA-Method Server

The use of 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 the Cisco IPSec VPN SPAs between remote access clients that have dynamic IP addresses and a corporate gateway, you must dynamically administer scalable IPSec policy on the gateway after authentication of each client. 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 ISAKMP profile to be enforced. For instructions on how to apply mode configuration to a crypto ISAKMP profile, see the "Defining Group Policy Information for Mode Configuration" section on page 174.

For instructions on how to apply mode configuration to a crypto ISAKMP profile, see the "Defining Group Policy Information for Mode Configuration" section on page 174.

Interfaces with crypto ISAKMP profiles, which 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.

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). See the *Configuring AAA Services on Cisco Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide.*
- Configure a **static** crypto ISAKMP profile (required). For configuration details, see the "How to Configure the ISAKMP Profile" section on page 201.
- Configure a **dynamic** crypto ISAKMP profile (optional) . For configuration details, see the "How to Configure the ISAKMP Profile" section on page 201.
- Configure ISAKMP policy (required). For configuration details, see the "Configuring IKE Policies" section on page 172.

Call Admission Control

The Call Admission Control (CAC) for IKE feature describes the application of CAC to the IKE protocol in Cisco IOS XR software. The main function of CAC is to protect the router from severe resource depletion and to prevent crashes. Therefore, the CAC limits the number of simultaneous IKE security associations (SAs, or calls to CAC) that a router can establish. IKE uses SAs to identify the parameters of its connections.

Also, IKE can negotiate and establish its own SA. An IKE SA, which is bidirectional, is used only by IKE.

You can configure a maximum number of active IKE SAs that you want to allow in the system, and thereby limit the CPU resources consumed by the IKE processor global CPU by use of the **crypto isakmp call admission limit** command.

When there is a new SA request from a peer router, IKE determines if the number of active IKE SAs being negotiated meets or exceeds the configured SA limit. If the number is greater than or equal to the limit, the new SA request is rejected and a system log is generated. This log contains the source destination IP address of the SA request.

An IKE SA cannot limit IPSec.

Information About IP Security Monitoring

The IP Security (IPSec) monitoring feature provides session monitoring enhancements that allow you to troubleshoot and monitor the end-user interface. Session monitoring includes the following enhancements:

- Ability to specify an Internet Key Exchange (IKE) peer description in the configuration file.
- Summary listing of crypto session status.
- Ability to clear both IKE and IP Security (IPSec) security associations (SAs) using one command-line interface (CLI).
- Ability to expend the filtering mechanism by using the options from the **show crypto session** command.

To implement IPSec security monitoring, you must understand the following concepts:

- Crypto Sessions Background, page 167
- Per-IKE Peer Description, page 168
- Summary Listing of Crypto Session Status, page 168
- IKE and IPSec Security Exchange Clear Command, page 168

Crypto Sessions Background

A crypto session is a set of IPSec connections (flows) between two crypto endpoints. If the two crypto endpoints use IKE as the keying protocol, they are IKE peers to each other. Typically, a crypto session consists of one IKE security association (for control traffic) and at least two IPSec security associations (for data traffic—one per each direction). There may be duplicated IKE security associations (SAs) and IPSec SAs or duplicated IKE SAs or IPSec SAs for the same session during rekeying or because of simultaneous setup requests from both sides.

Per-IKE Peer Description

The Per-IKE Peer Description function allows you to enter a description of your choice for an IKE peer. The unique peer description, which includes up to 80 characters, is used whenever you are referencing that particular IKE peer. To add the peer description, use the **description (ISAKMP peer)** command.

The primary application of this description field is for monitoring purposes (for example, when using **show** commands or for logging [syslog messages]). The description field is purely informational.

Summary Listing of Crypto Session Status

You can obtain a list of status information for active crypto sessions by using the **show crypto session** command. The listing includes the following summary status of the crypto session:

- Interface
- IKE SAs that are associated with the peer by whom the IPSec SAs are created
- IPSec SAs serving the flows of a session

Up to two IKE SAs and multiple IPSec SAs can be established for the same peer (for the same session), in which case IKE peer descriptions are repeated with different values for the IKE SAs that are associated with the peer and for the IPSec SAs that are serving the flows of the session.

In addition, you can use the **show crypto session** command with the **detail** keyword to obtain more detailed information about the sessions.

IKE and IPSec Security Exchange Clear Command

The **clear crypto session** command allows you to clear both IKE and IPSec. To clear a specific crypto session or a subset of all the sessions (for example, a single tunnel to one remote site), you need to provide session-specific parameters, such as a local or remote IP address, a local or remote port, a front door VPN routing and forwarding (FVRF) name, or an inside VRF (IVRF) name. Typically, the remote IP address is used to specify a single tunnel to be deleted.

If a local IP address is provided as a parameter when you use the **clear crypto session** command, all the sessions (and their IKE SAs and IPSec SAs) that share the IP address as a local crypto endpoint (IKE local address) are cleared. If you do not provide a parameter, all IPSec SAs and IKE SAs that are in the router are deleted.

Information About Cisco Easy VPN and the Cisco Easy VPN Server

Cisco Easy VPN is a software enhancement for existing Cisco routers and security appliances that simplifies VPN deployment for remote offices and teleworkers. Based on Cisco Unified Client Framework, Cisco Easy VPN centralizes VPN management across all Cisco VPN devices and thereby reduces the complexity of VPN deployments. Cisco Easy VPN enables an integration of VPN remote devices, such as Cisco routers, Cisco ASA adaptive security appliances, PIX Firewalls, Cisco VPN concentrators, or software clients, within a single deployment with a consistent policy and key management method, which simplifies remote-side administration.

Cisco Easy VPN consists of two components—Cisco Easy VPN Remote and Cisco Easy VPN Server.

Cisco Easy VPN Remote

The Cisco Easy VPN Remote feature allows routers running Cisco IOS XR software, Cisco ASA adaptive security appliances, Cisco PIX firewalls, and Cisco VPN 3000 concentrators to act as remote VPN clients. As such, these devices can receive security policies from a Cisco Easy VPN Server, minimizing VPN configuration requirements at the remote location. This solutionworks well for remote offices with little IT support, or large CPE deployments where it is impractical to individually configure multiple remote devices.

Cisco Easy VPN Server

The Cisco Easy VPN Server allows routers running Cisco IOS XR software, Cisco ASA adaptive security appliances, Cisco PIX firewalls, and Cisco VPN 3000 concentrators to act as VPN head-end devices in site-to-site or remote-access VPNs, where the remote office devices are using the Cisco Easy VPN Remote feature. Using this feature, security policies defined at the head-end are pushed to the remote VPN device insuring those connections have up-to-date policies in place before the connection is established.

A device enabled with Cisco Easy VPN Server can terminate VPN tunnels initiated by mobile remote workers running Cisco VPN client software on PCs. This allows mobile and remote workers to access their headquarters intranet.

The following subsections provide information about how to manage the Cisco Easy VPN Server by configuring client group attributes.

Client Group Attributes Supporting the Cisco Easy VPN Server

Table 2 describes client group attributes that help you manage a Cisco Easy VPN remote device.

Table 2 Client Group Attributes Supporting Management of a Cisco Easy VPN Remote Device

Attribute	Description
Banner	Configures a Cisco Easy VPN server to push a banner to a Cisco Easy VPN remote device.
Auto-Update[1]	Configures a Cisco Easy VPN server to provide an automated mechanism to make software and firmware upgrades automatically available to a Cisco Easy VPN remote device.
Browser-proxy	Configures a Cisco Easy VPN server so that the Cisco Easy VPN remote device can access resources on the corporate network. With this configuration, users do not need to manually modify the proxy settings of their web browser when connecting nor do they need to manually revert the proxy settings when disconnecting.
Pushing a configuration URL through a mode-configuration exchange	Applies policies and configuration information to a URL that the Cisco Easy VPN server downloads and applies to the running configuration when the IPSec VPN tunnel is active.

1. After a Cisco Easy VPN connection is up, use the **crypto ipsec server send-update** command in EXEC mode to send auto-update notifications at anytime.

Pushing a Configuration URL Through a Mode-Configuration Exchange

When remote devices connect to a corporate gateway for creating an IPsec VPN tunnel, some policy and configuration information must be applied to the remote device when the VPN tunnel is active to allow the remote device to become a part of the corporate VPN. The URL contains the configuration information that the remote device must download and apply to the running configuration.

The configuration that is pushed to the remote device is persistent by default. The configuration is applied when the IPsec tunnel is "up," but it is not withdrawn when the IPsec tunnel goes "down." However, it is possible to write a section of configuration that is transient in nature, in which case, the configuration of the section is reverted when the tunnel is disconnected.

There are no restrictions on where the configuration distribution server is physically located. However, we recommended that a secure protocol such as HTTPS (Secure HTTP) be used to retrieve the configuration. The configuration server is located in the corporate network, so because the transfer happens through the IPsec tunnel, insecure access protocols (such as HTTP) are used.

Regarding backward compatibility: the remote device asks for the CONFIGURATION-URL and CONFIGURATION-VERSION attributes. The server sends the configuration url and version attributes whether they were on the group or user. The standard attribute priority scheme, which was used for all the attributes, are also used. There is no built-in restriction to push the configuration, but bootstrap configurations (such as for the IP address) cannot be sent because those configurations are required to set up the Cisco Easy VPN tunnel, and the CONFIGURATION-URL comes into effect only after the Cisco Easy VPN tunnel comes up.

For configuration details, see Configuring Client Group Attributes for Cisco Easy VPN Server, page 180. For examples on how to configure Cisco Easy VPN for use with either a local or a remote AAA-method server, see Configuring Cisco Easy VPN with a Local AAA-Method Server: Example, page 210 and Configuring Cisco Easy VPN with a Remote AAA-Method Server: Example, page 211.

Information About Elimination of Multiple Proxies in Hub-and-Spoke Networks

Up to now, multiple IPSec security associations (SA) were required in a hub-and-spoke network, with each spoke negotiating multiple proxies, one for each subnet/IP address. This required multiple access list entries and establishment of an IPSec SA for each access-list entry, or line.

Cisco XR 12000 Series Router now acts like an IPSec hub, interoperating in network extension/plus mode, with EzVPN spokes that support the following networking differences:

- A single SA per spoke in a hub-and-spoke network for multiple networks.
- Aggregation of all access-list lines by the single SA to a single-SA **any any** proxy command.
- Network reachability from hub to spoke.
- Termination of multiple spokes on a single service-ipsec interface—a static interface that enables multiple IPSec tunnels to terminate on it alone, as long as the proxies associated with the tunnels do not intersect. This allows the IPSec SPA to uniquely identify the SA within the interface.

Configuration of an **ip any any** proxy command in the interface translates the routes arriving through the **MODECFG_IPV4_ROUTE** attribute during IKE Phase 1.5. into Cisco line card access control lists (ACLs). Each route corresponds to a network, which is taken from behind the spoke and sent from the Cisco Access Control Entry (ACE) to the hub. At the hub, reverse-routing is injected into the routing table with a destination that is the source proxy on the spoke.

IPSec Dead Peer Detection Periodic Message Option

A peer is an IPSec-compliant node capable of establishing IKE channels and negotiating SAs between itself and other peers. Peers can lose their IP connection to other peers due to routing problems, peer reloading, or other situations, resulting in a loss of packet traffic (sometimes called a “black hole”).

The IPSec Dead Peer Detection (DPD) Periodic Message option (defined in RFC 3706) allows you to query the liveliness of the Internet Key Exchange (IKE) peer on your router at regular intervals. The benefit of this configuration approach over that of the default configuration (on-demand dead peer detection), which is the default, is the earlier detection of dead peers.

How to Implement IKE Security Protocol Configurations for IPSec Networks

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

- Enabling or Disabling IKE, page 171 (required)
- Configuring IKE Policies, page 172 (required)
- Defining Group Policy Information for Mode Configuration, page 174 (optional)
- Limiting an IKE Peer to Use a Specific Policy Set, page 178 (optional)
- Configuring Client Group Attributes for Cisco Easy VPN Server, page 180 (optional)
- Configuring Cisco Easy VPN with a Local AAA-Method Server, page 184 (optional)
- Configuring Cisco Easy VPN with a Remote AAA-Method Server, page 184
- Manually Configuring RSA Keys, page 184 (optional, depending on IKE parameters)
- Configuring ISAKMP Preshared Keys in ISAKMP Keyrings, page 192 (optional, depending on IKE parameters)
- Configuring Call Admission Control, page 193 (optional)
- Configuring Crypto Keyrings, page 197 (required)
- Configuring IP Security VPN Monitoring, page 200 (optional)

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**
3. **no crypto isakmp**

4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**crypto isakmp** **Example:** `RP/0/0/CPU0:router(config)# crypto isakmp`	Globally enables IKE at the peer router.
Step 3	**no crypto isakmp** **Example:** `RP/0/0/CPU0:router(config)# no crypto isakmp`	(Optional) Disables IKE at the peer router.

Step 4	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config)# end`
	or
	`RP/0/0/CPU0:router(config)# 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.

Configuring IKE Policies

This task configures IKE policies.

SUMMARY STEPS

1. **configure**

2. **crypto isakmp policy** *priority*
3. **encryption {192-aes** *AES - Advanced Encryption Standard (192-bit keys)* **| 256-aes** *AES - Advanced Encryption Standard (256-bit keys)* **| 3des** *3DES - Three-key triple DES* **| aes** *AES - Advanced Encryption Standard (128 bit keys)* **| des** *DES - Data Encryption Standard (56 bit keys)}*
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**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**crypto isakmp policy** *priority* **Example:** `RP/0/0/CPU0:router(config)# crypto isakmp policy 5`	Identifies the policy to create. Each policy is uniquely identified by the priority number you assign, which can be from 1-10000. This command places the router in ISAKMP policy configuration mode.
Step 3	**encryption {192-aes** *AES - Advanced Encryption Standard (192-bit keys)* **| 256-aes** *AES - Advanced Encryption Standard (256-bit keys)* **| 3des** *3DES - Three-key triple DES* **| aes** *AES - Advanced Encryption Standard (128 bit keys)* **| des** *DES - Data Encryption Standard (56 bit keys)}* **Example:** `RP/0/0/CPU0:router(config-isakmp)# encryption aes`	Specifies the encryption algorithm.
Step 4	**hash** {**sha** | **md5**} **Example:** `RP/0/0/CPU0:router(config-isakmp)# hash md5`	Specifies the hash algorithm. • SHA—Secure-hash-algorithm • MD5—Message-digest-5 **Note** SHA and MD5 can be used to calculate hashed message authentication coding (HMAC).
Step 5	**authentication** {**pre-share** | **rsa-sig** | **rsa-encr**} **Example:** `RP/0/0/CPU0:router(config-isakmp)# authentication rsa-sig`	Specifies the authentication method for this policy as either a pre-shared key, an RSA-encryption, or an RSA signature.

	Command or Action	Purpose
Step 6	**group** {**1** | **2** | **5**} **Example:** `RP/0/0/CPU0:router(config-isakmp)# group 5`	Specifies the Diffie-Hellman group identifier.
Step 7	**lifetime** *seconds* **Example:** `RP/0/0/CPU0:router(config-isakmp)# lifetime 50000`	Specifies the lifetime of the security association. The range, in seconds, is from 60 to 86400.

Step 8	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config-isakmp)# end`
	or
	`RP/0/0/CPU0:router(config-isakmp)# 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.

Step 9	**show crypto isakmp policy** **Example:** `RP/0/0/CPU0:router# show crypto isakmp policy`	(Optional) Displays all existing IKE policies.

Defining Group Policy Information for Mode Configuration

Although users can belong to only one group for each connection, they may belong to specific groups with different policy requirements. Therefore, users may decide to connect to the client using a different group ID by changing their client profile on the VPN device.

This task defines the group policy attributes that are pushed to the client through mode configuration.

SUMMARY STEPS

1. **configure**
2. **crypto isakmp client configuration group** *group-name*

3. **key** *preshared-key*
4. **acl** *acl-name*
5. **backup-server** {*ip-address* | *hostname*}
6. **dns** *primary-server* [*secondary-server*]
7. **domain** *name*
8. **firewall are-u-there**
9. **group-lock**
10. **include-local-lan**
11. **max-logins** *number-of-logins*
12. **max-users** *number-of-users*
13. **netmask** *mask*
14. **pfs**
15. **pool** *name*
16. **save-password**
17. **split-dns** *domain-name*
18. **wins** *primary-server* [*secondary-server*]
19. **end**
    or
    commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**crypto isakmp client configuration group** *group-name* **Example:** `RP/0/0/CPU0:router(config)# crypto isakmp client configuration group cisco`	Specifies which group's policy profile is defined and enters ISAKMP group configuration mode. If no specific group matches and a default group is defined, users are automatically given the default group's policy.
Step 3	**key** *preshared-key* **Example:** `RP/0/0/CPU0:router(config-group)# key samplekey`	Specifies the IKE preshared key for group policy attribute definition. **Note** This command *must* be enabled if the client identifies itself with a preshared key.
Step 4	**acl** *acl-name* **Example:** `RP/0/0/CPU0:router(config-group)# acl group1`	(Optional) Configures split tunneling. • Use the *acl-name* argument to specify a group of ACL rules that represent protected subnets for split tunneling purposes.

	Command or Action	Purpose
Step 5	**backup-server** {ip-address | hostname} **Example:** `RP/0/0/CPU0:router(config-group)# backup-server 10.1.1.1`	Specifies the backup server. • Use the *ip-address* argument to specify the IP address of the server. • Use the *hostname* argument to specify the hostname of the server.
Step 6	**dns** *primary-server* [*secondary-server*] **Example:** `RP/0/0/CPU0:router(config-group)# dns 2.2.2.2 2.3.2.3`	Specifies the primary and secondary Domain Name Service (DNS) addresses. • Use the *primary-server* argument to specify the IP address of the primary DNS. • (Optional) Use the *secondary-server* argument to specify the IP address of the secondary DNS.
Step 7	**domain** *name* **Example:** `RP/0/0/CPU0:router(config-group)# domain cisco.com`	Specifies the DNS domain to which a group belongs. • Use the *name* argument to specify the default name of the DNS domain.
Step 8	**firewall are-u-there** **Example:** `RP/0/0/CPU0:router(config-group)# firewall are-u-there`	Adds the Firewall-Are-U-There attribute to the server group if your PC is running the Black Ice or Zone Alarm personal firewalls.
Step 9	**group-lock** **Example:** `RP/0/0/CPU0:router(config-group)# group-lock`	Allows you to enter your extended authentication (Xauth) username, including the group name, when preshared key authentication is used with IKE.
Step 10	**include-local-lan** **Example:** `RP/0/0/CPU0:router(config-group)# include-local-lan`	Configures the Include-Local-LAN attribute to allow a nonsplit-tunneling connection to access the local subnetwork at the same time as the client.
Step 11	**max-logins** *number-of-logins* **Example:** `RP/0/0/CPU0:router(config-group)# max-logins 8`	Specifies the maximum number of concurrent logins that are allowed for a certain user. • Use the *number-of-logins* argument to specify the number of logins. The value ranges from 0 to 16 and 384.
Step 12	**max-users** *number-of-users* **Example:** `RP/0/0/CPU0:router(config-group)# max-users 1200`	Limits the number of connections to a specific server group. • Use the *number-of-users* argument to specify the number of connected users. The value ranges from 0 to 16 and 384.
Step 13	**netmask** *mask* **Example:** `RP/0/0/CPU0:router(config-group)# netmask 255.255.255.0`	Sets the IP network mask. • Use the *mask* argument to specify the IP network mask.

	Command or Action	Purpose
Step 14	**pfs** **Example:** `RP/0/0/CPU0:router(config-group)# pfs`	Configures a server to notify the client of the central-site policy regarding whether PFS is required for any IP Security (IPSec) Security Association (SA).
Step 15	**pool** *name* **Example:** `RP/0/0/CPU0:router(config-group)# pool pool2`	Defines the name of an address-pool in which an address is allocated, if required. Use the *pool-name* argument to specify the name of the local pool address.
Step 16	**save-password** **Example:** `RP/0/0/CPU0:router(config-group)# save-password`	Saves your extended authentication (Xauth) password locally on your PC or Easy VPN client.

Step 17	**split-dns** *domain-name*
	
	Example:
	`RP/0/0/CPU0:router(config-group)# split-dns green.com`
	`RP/0/RP0/CPU0:router(config-group)# split-dns acme.org`
		Specifies a domain name that must be tunneled or resolved to the private network.
		
		• Use the *domain-name* argument to specify the name of the DNS domain that must be tunneled or resolved to the private network.
		
		Note If you have to configure more than one domain name, you have to add a **split-dns** command line for each.

Step 18	**wins** *primary-server* [*secondary-server*] **Example:** `RP/0/0/CPU0:router(config-group)# wins 10.1.1.2 10.1.1.3`	Specifies the primary and secondary Windows Internet Naming Service (WINS) servers. • Use the *primary-server* argument to specify the name of the primary WINS server. • (Optional) Use the *secondary-server* argument to specify the name of the secondary WINS server.

Step 19	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config-group)# end`
	or
	`RP/0/0/CPU0:router(config-group)# 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.

Limiting an IKE Peer to Use a Specific Policy Set

This task describes how to configure IKE to limit the policies it matches with the remote VPN peer to those restricted by the IPSec hub. To restrict the IKE peer to a policy set on a the IPSec hub, the client must negotiate the IP address of the SVI tunnel source (**match identity local-address** command).

Note A policy set may contain up to five IKE policies.

You may create as many policies as needed to add additional encryption methods to be prioritized for matching.

To limit an IKE peer to use a specific policy set, you must also configure the policy set or sets. See Configuring IKE Policies, page 172.

SUMMARY STEPS

1. **configure**
2. **crypto isakmp policy-set** *policy-name*
3. **policy** *policy number*

Note In this step, you can identify up to five previously configured ISAKMP policies to match.

4. **match identity local-address** *A.B.C.D IP address*
5. (Optional) Repeat Step 2 through Step 4, as needed, to configure additional policy sets for specific IP addresses.
6. **end**
   or
   commit
7. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	Enters global configuration mode.
Step 2	**crypto isakmp policy-set** *policy-name* **Example:** `RP/0/RP0/CPU0:router(config)# crypto isakmp policy-set policy_1`	Enters the global crypto configuration mode to create a new policy set by naming it.
Step 3	**policy** *policy number* **Example:** `RP/0/RP0/CPU0:router(config-isakmp-pol-set)# policy 10`	Identifies up to five ISAKMP policies by the match priority number you gave them in Configuring IKE Policies, page 172. (A policy set can contain up to five policies.) Only these policies can be part of the IKE negotiation between the local and the remote peer.

	Command or Action	Purpose
Step 4	**match identity local-address** *A.B.C.D IP address* **Example:** `RP/0/RP0/CPU0:router(config-isakmp-pol-set)# match identity local-address 10.56.8.10`	Identifies the SVI tunnel source to be restricted to a particular policy set. **Note** When users connect to the IP address identified in this step, the ISAKMP policies configured in the policy set defined in Step 2 becomes part of the IKE negotiation.
Step 5	(Optional) Repeat Step 2 through Step 4, as needed, to configure additional policy sets for specific IP addresses.	You may use either multiple ISAKMP policies or multiple IP addresses to create the match.

Step 6	**end**
	or
	commit
	
	Example:
	`RP/0/RP0/CPU0:router(config-isakmp-pol-set)# end`
	or
	`RP/0/RP0/CPU0:router(config-isakmp-pol-set)# 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.

Step 7	**exit**
	
	Example:
	`RP/0/0/CPU0:router(config-isakmp-pol-set)# exit`
	`RP/0/0/CPU0:router(config)#`
		Exits the crypto ISAKMP policy- set configuration mode.

For an example, see Limiting an IKE Peer to a Particular Policy Set Based on Local IP Address: Example, page 209.

Configuring Client Group Attributes for Cisco Easy VPN Server

This task describes how to configure client group attributes for a Cisco Easy VPN server.

SUMMARY STEPS

1. **configure**
2. **crypto isakmp client configuration group** *group-name*
3. **banner** *banner-text*
4. **auto-update client** {*type-of-system*} {**url** *url*} {**rev** *review-version*}
5. **browser-proxy** {*browser-proxy-map-name*}
6. **configuration url** *url*
7. **configuration version** *version*
8. **end**
   or
   commit
9. **exit**
10. **crypto isakmp client configuration browser-proxy** *browser-proxy-name*
11. **proxy** {**auto-detect | bypass-local | exception-list** *semicolon-delimited exception list* **| none | server**}
12. **end**
    or
    commit
13. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**crypto isakmp client configuration group** *group-name*
	
	Example:
	`RP/0/0/CPU0:router(config)# crypto isakmp client configuration group cisco`
	`RP/0/0/CPU0:router(config-group)#`
		Specifies the policy profile for a group and enters ISAKMP group configuration mode.
		• If no specific group matches, you are automatically given the policy of the default group.
		• The default group is also used for other attributes, so these must be checked and updated.

	Command or Action	Purpose
Step 3	**banner** *banner-text* **Example:** `RP/0/0/CPU0:router(config-group)# banner thequickbrowndog`	Specifies the text of the banner.
Step 4	**auto-update client** {*type-of-system*} {**url** *url*} {**rev** *review-version*} **Example:** `RP/0/0/CPU0:router(config-group)# auto-update client Win2000 url http:www.ourcompanysite.com/newclient rev 3.0.1`	Configures automatic update parameters for a Cisco Easy VPN remote device. The example shows that the update parameters are set for a Windows 2000 operating system, a URL of http:www.ourcompanysite.com/newclient, and version 3.0.1. For a list of the reserved names that the Cisco Easy VPN client expects for each type of operating system, which are used for the *type-of-system* argument, see *Cisco IOS XR System Security Command Reference*.
Step 5	**browser-proxy** {*browser-proxy-map-name*} **Example:** `RP/0/0/CPU0:router(config-group)# browser-proxy EZVPN`	Specifies browser-proxy parameter settings to a group.
Step 6	**configuration url** {*url*} **Example:** `RP/0/0/CPU0:router(config-group)# configuration url http://10.10.8.8/easy.cfg`	Specifies the URL the remote device must use to get the configuration from the server.
Step 7	**configuration version** {*version-number*} **Example:** `RP/0/0/CPU0:router(config-group)# configuration version 10`	Specifies the version of the configuration. • The *version-number* argument is an unsigned integer in the range of 1 to 10.

	Command or Action	Purpose

Step 8	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config-group)# end`
	or
	`RP/0/0/CPU0:router(config-group)# 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.

Step 9	**exit** **Example:** `RP/0/0/CPU0:router(config-group)# exit`	Exits the group configuration mode and returns you to global configuration mode.

Step 10	**crypto isakmp client configuration browser-proxy** *browser-proxy-name*
	
	Example:
	`RP/0/0/CPU0:router(config)# crypto isakmp client configuration browser-proxy bproxy`
	`RP/0/0/CPU0:router(config-crypto-isakmp-browser-proxy)#`
		Configures browser-proxy parameters for a Cisco Easy VPN remote device and enters ISAKMP browser proxy configuration mode.

	Command or Action	Purpose
Step 11	**proxy** {**auto-detect** | **bypass-local** | **exception-list** *semicolon-delimited exception list* | **none** | **server**} **Example:** `RP/0/0/CPU0:router(config-crypto-isakmp-browser-proxy)# proxy auto-detect`	Configures proxy parameters for a Cisco Easy VPN remote device: • **auto-detect—**Auto-detects proxy settings. • **bypass-local—**Bypasses proxy settings for local. • **exception-list—**Configures exception list. • **none—**Configures no proxy server. • **server—**Configures proxy server.

Step 12	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config-crypto-isakmp-browser-proxy)# end`
	or
	`RP/0/0/CPU0:router(config-crypto-isakmp-browser-proxy)# 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.

Configuring Cisco Easy VPN with a Local AAA-Method Server

The AAA database stores the users, groups, and task information that controls access to the system. This database can be either local or remote. Whether you use a local or remote database depends on your AAA configuration.

AAA data, such as users, user groups, and task groups, can be stored locally within a secure domain router. The data is stored in the in-memory database and persists in the configuration file. The stored passwords are encrypted.

For detailed information about user groups and task IDs, see the *Configuring AAA Services on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide*.

For a sample configuration of Cisco Easy VPN with a local AAA-method server, see Configuring Cisco Easy VPN with a Local AAA-Method Server: Example, page 210.

Configuring Cisco Easy VPN with a Remote AAA-Method Server

The procedure for configuring a *remote* AAA-method server is identical to that for a local AAA-method server except that you must first define its address in global configuration mode.

Before entering global configuration mode, you must first access the administration plane by executing the **admin** command. The **remote** keyword required for this configuration is only accessible with the **aaa authentication** command and login keyword.

For an example, see Configuring Cisco Easy VPN with a Local AAA-Method Server: Example, page 210.

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:

- Generating RSA Keys, page 184
- Configuring ISAKMP Identity, page 184
- Configuring RSA Public Keys of All the Other Peers, page 186
- Importing a Public Key for RSA based User Authentication, page 189
- Deleting an RSA Public Key from the Router, page 190

Generating RSA Keys

For instructions on how to generate RSA keys, see the "Generating an RSA Key Pair, page 67" inthe *Implementing Certification Authority Interoperability* 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

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**crypto isakmp identity** {**address** | **hostname**} **Example:** `RP/0/0/CPU0:router(config)# crypto isakmp identity address`	(At the local peer) Specifies the peer's ISAKMP identity by IP address or by hostname. See the **crypto isakmp identity** command description for guidelines for when to use the IP address and when to use the hostname.
Step 3	**host** *hostname address1 [address2...address8]* **Example:** `RP/0/0/CPU0:router(config)# host host1 10.0.0.5`	(At all remote peers) Maps the peer's hostname to its IP addresses at all the remote peers. • This command is used if the local peer's ISAKMP identity was specified using a hostname. • This step might be unnecessary if the hostname or address is already mapped in a DNS server.

Step 4	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config)# end`
	or
	`RP/0/0/CPU0:router(config)# 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.

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 keyring** *keyring-name* [**vrf** *fvrf-name*]
3. **rsa-pubkey** {**address** *address* | **name** *fqdn*} [**encryption** | **signature**]
4. **address** *ip-address*
5. **key-string** *key-string*
6. **quit**
7. **end**
   or
   commit
8. **show crypto key pubkey-chain rsa** [**name** *key-name* | **address** *key-address*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**crypto keyring** *keyring-name* [**vrf** *fvrf-name*]
	
	Example:
	`RP/0/0/CPU0:router(config)# crypto keyring vpnkeyring`
	`RP/0/0/CPU0:router(config-keyring)#`
		Defines a crypto keyring during IKE authentication
		• Enters keyring configuration mode.
		• Use the *keyring-name* argument to specify the name of the crypto keyring.
		• (Optional) Use the **vrf** keyword to specify that the front door virtual routing and forwarding (FVRF) name is the keyring that is referenced.

	Command or Action	Purpose

Step 3	**rsa-pubkey** {**address** *address* | **name** *fqdn*} [**encryption** | **signature**]
	
	Example:
	`RP/0/0/CPU0:router(config-keyring)# rsa-pubkey name host.vpn.com`
	`RP/0/0/CPU0:router(config-pubkey)`
		Defines the Rivest, Shamir, and Adelman (RSA) manual key to be used for encryption or signature during Internet Key Exchange (IKE) authentication.
		• Use the **address** keyword to specify the IP address of the RSA public key of the remote peer. The address argument is the IP address of the remote RSA public key of the remote peer that you manually configure.
		• Use the **name** keyword to specify the fully qualified domain name (FQDN) of the peer.
		• Use the **encryption** keyword to specify that the key is used for encryption.
		• Use the **signature** keyword to specify that the key is used for a signature. The **signature** keyword is the default.

Step 4	**address** *ip-address* **Example:** `RP/0/0/CPU0:router(config-pubkey)# address 10.5.5.1`	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.
Step 5	**key-string** ***key-string*** **Example:** `RP/0/0/CPU0:router(config-pubkey)# key-string 005C300D 06092A86 4886F70D 01010105 ...`	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 **quit** at the public key configuration prompt.
Step 6	**quit** **Example:** `RP/0/0/CPU0:router(config-pubkey)# quit`	Returns to global configuration mode.

	Command or Action	Purpose

Step 7	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config)# end`
	or
	`RP/0/0/CPU0:router(config)# 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.

Step 8	**show crypto pubkey-chain rsa** [**name** ***key-name*** | **address** ***key-address***] **Example:** `RP/0/0/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.

Importing a Public Key for RSA based User Authentication

This task imports the RSA public key to the router.

SUMMARY STEPS

1. **configure**
2. **crypto key import authentication rsa** {**address** *address* | **name** *fqdn*}
3. **end**
   or
   commit
4. **show crypto key import authentication rsa** {**address** *address* | **name** *fqdn*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**crypto key import authentication rsa** {**address** *address* | **name** *fqdn*}
	
	Example:
	`RP/0/0/CPU0:router(config)# crypto key import authentication rsa tftp://223.255.254.254/ssh/public.pub (in base64)`
	`RP/0/0/CPU0:router(config-keyring)#`
		Imports the public key to the router.
		• Use the **address** keyword to specify the IP address of the RSA public key of the remote peer. The address argument is the IP address of the remote RSA public key of the remote peer that you manually configure.
		• Use the **name** keyword to specify the fully qualified domain name (FQDN) of the peer.

Step 3	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config)# end`
	or
	`RP/0/0/CPU0:router(config)# 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.

Step 4	**show crypto key import authentication rsa** {**address** *address* | **name** *fqdn*} **Example:** `RP/0/0/CPU0:router# show crypto key import authentication rsa`	(Optional) Displays a list of all the RSA public keys imported to the router. • Use the optional name or address keyword to display details about a particular RSA public key stored on your router.

Deleting an RSA Public Key from the Router

This task deletes the RSA public key from the router.

SUMMARY STEPS

1. **configure**

2. **zeroize crypto key import authentication rsa** {**address** *address* | **name** *fqdn*}
3. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**zeroize crypto key import authentication rsa** {**address** *address* | **name** *fqdn*}
	
	Example:
	`RP/0/0/CPU0:router(config)# zeroize crypto key import authentication rsa tftp://223.255.254.254/ssh/public.pub (in base64)`
	`RP/0/0/CPU0:router(config-keyring)#`
		Deletes the public key from the router.
		• Use the **address** keyword to specify the IP address of the RSA public key of the remote peer. The address argument is the IP address of the remote RSA public key of the remote peer that you manually configure.
		• Use the **name** keyword to specify the fully qualified domain name (FQDN) of the peer.

Step 3	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config)# end`
	or
	`RP/0/0/CPU0:router(config)# 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.

Step 4	**show crypto pubkey-chain rsa** [**name** ***key-name*** | **address** ***key-address***] **Example:** `RP/0/0/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 ISAKMP Preshared Keys in ISAKMP Keyrings

This task configures ISAKMP preshared keys in ISAKMP keyrings.

Prerequisites

To configure ISAKMP preshared keys in ISAKMP keyrings, 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 hostname 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 184.
- 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.
- You must specify the support for masked preshared keys. Remember to repeat these tasks at each peer that uses preshared keys in an IKE policy.

SUMMARY STEPS

1. **configure**
2. **crypto keyring** *keyring-name* [**vrf** *fvrf-name*]
3. **pre-shared-key** {**address** *address* [*mask*] | **hostname** *hostname*} **key** *key*
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**crypto keyring** *keyring-name* [**vrf** *fvrf-name*]
	
	Example:
	`RP/0/0/CPU0:router(config)# crypto keyring vpnkeyring`
	`RP/0/0/CPU0:router(config-keyring)#`
		Defines a crypto keyring during IKE authentication.
		• Use the *keyring-name* argument to specify the name of the crypto keyring.
		• (Optional) Use the **vrf** keyword to specify that the front door virtual routing and forwarding (FVRF) name is the keyring that is referenced.

	Command or Action	Purpose

Step 3	**pre-shared-key** {**address** *address* [*mask*] | **hostname** *hostname*} **key** *key*
	
	Example:
	`RP/0/0/CPU0:router(config-keyring)# pre-shared-key address 10.72.23.11 key vpnkey`
	`RP/0/0/CPU0:router(config-keyring)# pre-shared-key hostname mycisco.com key vpnkey`
		Defines a preshared key for IKE authentication.
		• Use the **address** keyword to specify the IP address of the remote peer or a subnet and mask.
		• (Optional) Use the *mask* argument to match the range of the address.
		• Use the **hostname** keyword to specify the fully qualified domain name (FQDN) of the peer.
		• (Optional) Use the **key** keyword to specify the key.

Step 4	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config-keyring)# end`
	or
	`RP/0/0/CPU0:router(config-keyring)# 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.

Configuring Call Admission Control

These tasks are used to configure Call Admission Control (CAC):

- Configuring the IKE Security Association Limit, page 193
- Configuring the System Resource Limit, page 195

Configuring the IKE Security Association Limit

This task configures the IKE security admission limit.

SUMMARY STEPS

1. **configure**
2. **crypto isakmp call admission limit** {**in-negotiation-sa** *number* | **sa** *number*}

3. **end**
   or
   commit
4. **show cyrpto isakmp call admission statistics**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**crypto isakmp call admission limit** {**in-negotiation-sa** *number* | **sa** *number*} **Example:** `RP/0/0/CPU0:router(config)# crypto isakmp call admission limit sa 25`	Specifies the maximum number of IKE SAs that the router can establish before IKE begins rejecting new SA requests. • Use the **in-negotiation-sa** keyword to specify the maximum number of in-negotiation (embryonic) IKE security associations (SAs) that the router can establish before IKE begins rejecting new SA requests. The range for the number argument is from 1 to 100000. • Use the **sa** keyword to specify the maximum number of active IKE SAs that the router can establish. The range for the *number* argument is from 1 to 100000.

	Command or Action	Purpose

Step 3	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config)# end`
	or
	`RP/0/0/CPU0:router(config)# 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.

Step 4	**show cyrpto isakmp call admission statistics** **Example:** `RP/0/0/CPU0:router# show crypto isakmp call admission statistics`	Monitors crypto CAC statistics.

Configuring the System Resource Limit

This task configures the system resource limit.

SUMMARY STEPS

1. **configure**
2. **crypto isakmp call admission limit** {**cpu** {**total** *percent* | **ike** *percent*}}
3. **end**
   or
   commit
4. **show cyrpto isakmp call admission statistics**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**crypto isakmp call admission limit** {**cpu** {**total** *percent* | **ike** *percent*}} **Example:** `RP/0/0/CPU0:router(config)# crypto isakmp call admission limit cpu total 90`	Specifies the maximum number of IKE SAs that the router can establish before IKE begins rejecting new SA requests. • Use the **cpu** keyword to specify the total resource limit for the CPU usage. • Use the **total** keyword to specify the maximum total CPU usage to accept new calls. The range for the *percent* argument is from 1 to 100. • Use the **ike** keyword to specify the maximum IKE CPU usage to accept new calls. The range for the percent argument is from 1 to 100.

Step 3	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config)# end`
	or
	`RP/0/0/CPU0:router(config)# 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.

Step 4	**show cyrpto isakmp call admission statistics** **Example:** `RP/0/0/CPU0:router# show cyrpto isakmp call admission statistics`	Monitors crypto CAC statistics.

Configuring Crypto Keyrings

A crypto keyring is a repository of preshared and Rivest, Shamir, and Adelman (RSA) public keys. The router can have zero or more keyrings. Each keyring optionally allows the specification of a VRF in which the keys defined in the keyring belong.

This task configures crypto keyrings.

Crypto Keyrings Configuration Guidelines and Restrictions

Follow these guidelines and restrictions when configuring crypto keyrings:

- The VRF associated with a crypto keyring cannot be changed. A different keyring must be configured with the new VRF value.
- Address overlapping in a keyring is not allowed and must be enforced during configuration.
- A crypto keyring is attached to one or more ISAKMP profiles and cannot be deleted while in use.

SUMMARY STEPS

1. **configure**
2. **crypto keyring** *keyring-name* [**vrf** *fvrf-name*]
3. **description** *string*
4. **local-address** *ip-address*
5. **pre-shared-key** {**address** *address* [*mask*] | **hostname** *hostname*} **key** *key*
6. **rsa-pubkey** {**address** *address* | **name** *fqdn*} [**encryption** | **signature**]
7. **key-string** *key-string*
8. **quit**
9. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**crypto keyring** *keyring-name* [**vrf** *fvrf-name*] **Example:** `RP/0/RP0/CPU0:router(config)# crypto keyring vpnkey`	Defines a crypto keyring to be used during IKE authentication. • Use the *keyring-name* argument as the name of the crypto keyring. • Use the **vrf** keyword to specify that the front door virtual routing and forwarding (FVRF) name is the keyring that is referenced. The *fvrf-name* argument must match the FVRF name that was defined during a (VRF) configuration.
Step 3	**description** *string* **Example:** `RP/0/RP0/CPU0:router(config-keyring# description this is a sample keyring`	Creates a one-line description for a keyring. • Use the *string* argument to specify the character string that describes the keyring.
Step 4	**local-address** *ip-address* **Example:** `RP/0/RP0/CPU0:router(config-keyring)# local-address 130.40.1.1`	Limits the scope of an ISAKMP keyring configuration to a local termination address or interface. • Use the *ip-address* argument to specify the IP address to which to bind.
Step 5	**pre-shared-key** {**address** *address* [*mask*] | **hostname** *hostname*} **key** *key* **Example:** `RP/0/RP0/CPU0:router(config-keyring)# pre-shared-key address 10.72.23.11 key vpnkey`	Defines a preshared key to be used for IKE authentication. • Use the **address** keyword to specify the IP address of the remote peer or a subnet and mask. The *mask* argument is optional. • Use the **hostname** keyword to specify the fully qualified domain name (FQDN) of the peer. • Use the **key** keyword to specify the secret.

	Command or Action	Purpose
Step 6	**rsa-pubkey** {**address** *address* | **name** *fqdn*} [**encryption** | **signature**] **Example:** `RP/0/RP0/CPU0:router(config-keyring)# rsa-pubkey name host.vpn.com` `RP/0/RP0/CPU0:router(config-keyring)# rsa-pubkey name host.vpn.com`	Defines a Rivest, Shamir, and Adelman (RSA) public key by address or hostname. • Use the **address** keyword to specify the IP address of the RSA public key of the remote peer. The *address* argument is the IP address of the remote RSA public key of the remote peer that you manually configure. • Use the **name** keyword to specify the FQDN of the peer. • (Optional) Use the **encryption** keyword to specify that the key is used for encryption. • (Optional) Use the **signature** keyword to specify that the key is used for a signature. The **signature** keyword is the default.
Step 7	**key-string** *key-string* **Example:** `RP/0/RP0/CPU0:router(config-pubkey)# key-string 005C300D 06092A86 4886F70D 01010105`	Manually specifies the RSA public key of a remote peer.
Step 8	**quit** **Example:** `RP/0/RP0/CPU0:router(config-pubkey)# quit`	Returns to global configuration mode.

Step 9	**end**
	or
	commit
	
	Example:
	`RP/0/RP0/CPU0:router(config)# end`
	or
	`RP/0/RP0/CPU0:router(config)# 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.

Configuring IP Security VPN Monitoring

The following sections describe how to configure IP Security (IPSec) VPN monitoring:

- Adding the Description of an IKE Peer, page 200 (optional)
- Clearing a Crypto Session, page 201 (optional)

Adding the Description of an IKE Peer

This task allows you to add the description of an IKE peer to an IPSec VPN session.

SUMMARY STEPS

1. **configure**
2. **crypto isakmp peer** {**address** *ip-address* | **hostname** *hostname*} [**description** *line* | **vrf** *fvrf-name*]
3. **description** *line-of-description*
4. **end**
   or
   commit
5. **show crypto isakmp peers** [*ip-address* | **vrf** *vrf-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**crypto isakmp peer** {**address** *ip-address* | **hostname** *hostname*} [**description** *string* | **vrf** *fvrf-name*]
	
	Example:
	`RP/0/0/CPU0:router(config)# crypto isakmp peer address 40.40.40.2`
	`RP/0/0/CPU0:router(config-isakmp-peer)`
		Specifies an IPSec peer for the session.

Step 3	**description** *string* **Example:** `RP/0/0/CPU0:router(config-isakmp-peer)# description citeA`	Adds a line of description for an IKE peer. • Description of peer may be up to 80 characters. •

	Command or Action	Purpose

Step 4	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config-isakmp-peer)# end`
	or
	`RP/0/0/CPU0:router(config-isakmp-peer)# 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.

Step 5	**show crypto isakmp peers** [*ip-address* | **vrf** *vrf-name*] **Example:** `RP/0/0/CPU0:router# show crypto isakmp peers`	Displays peer descriptions.

Clearing a Crypto Session

Use the **clear crypto session** command in EXEC mode to delete the crypto sessions (IP Security [IPSec] and Internet Key Exchange [IKE] security associations [SAs]) for users and groups.

How to Configure the ISAKMP Profile

This task configures the ISAKMP profile (which is a repository of commands for a set of peers) for either a service interface or a tunnel interface.

Note The Cisco XR 12000 Series Router supports both services and tunnel interfaces.

SUMMARY STEPS

1. **configure**
2. **crypto isakmp profile** [**local**] *profile-name*
3. **description** *string*

4. **keepalive disable**
5. **self-identity** {**address** | **fqdn** | **user-fqdn** *user-fqdn*}
6. **keyring** *keyring-name*
7. **match identity** {**group** *group-name* | **address** *address* [*mask*] **vrf** [*fvrf*] | **host** *hostname* | **host domain** *domain-name* | **user** *username* | **user domain** *domain-name*}
8. **set interface { | service-gre** *intf-index* | **service-ipsec}** *intf-index*
9. **set ipsec-profile** *profile-name*
10. **end**
    or
    commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**crypto isakmp profile** [**local**] *profile-name*
	
	Example:
	`RP/0/0/CPU0:router(config)# crypto isakmp profile local vpnprofile`
	`RP/0/0/CPU0:router(config-isa-prof)#`
		Defines an ISAKMP profile and audits IPSec user sessions.
		• (Optional) Use the **local** keyword to specify that the profile is used for locally sourced or terminated traffic.
		Note The **local** keyword is required for use of the **set ipsec-profile** and the **set interface tunnel-ipsec** commands later in this procedure.
		• (Required) Use the *profile-name* argument to specify the name of the user profile.

Step 3	**description** *string* **Example:** `RP/0/0/CPU0:router(config-isa-prof)# description this is a sample profile`	Creates a description for a keyring. • Use the *string* argument to specify the character string that describes the keyring.
Step 4	**keepalive disable** **Example:** `RP/0/0/CPU0:router(config-isa-prof)# keepalive disable`	Lets the gateway send DPD messages to the Cisco IOS XR peer. • Use the **disable** keyword to disable the keepalive global declarations.

	Command or Action	Purpose
Step 5	**self-identity** {**address** | **fqdn** | **user-fqdn** *user-fqdn*} **Example:** `RP/0/0/CPU0:router(config-isa-prof)# self-identity user-fqdn user@vpn.com`	Defines the identity that the local IKE uses to identify itself to the remote peer. • Use the **address** keyword to specify the IP address of the local endpoint. • Use the **fqdn** keyword to specify the fully qualified domain name (FQDN) of the host. • Use the **user-fqdn** keyword to specify that the user FQDN was sent to the remote endpoint.
Step 6	**keyring** *keyring-name* **Example:** `RP/0/0/CPU0:router(config-isa-prof)# keyring vpnkeyring`	Configures a keyring with an ISAKMP profile. • Use the *keyring-name* argument to specify the keyring name, which must match the keyring name that was defined in the global configuration.

Command or Action	Purpose

Step 7 **match identity** {**group** *group-name* | **address** *address* [*mask*] **vrf** [*fvrf*] | **host** *hostname* | **host domain** *domain-name* | **user** *username* | **user domain** *domain-name*}
 	
Example:
`RP/0/0/CPU0:router(config-isa-prof)# match identity group vpngroup`
`RP/0/0/CPU0:router(config-isa-prof-match)#`
	Matches the identity from a peer in an ISAKMP profile.
	• Use the **group** keyword to specify a Unity group that matches identification (ID) type ID_KEY_ID. If RSA signatures are used, the *group-name* argument matches the organizational unit (OU) field of the distinguished name (DN).
	• Use the **address** keyword to match the *address* argument with the ID type ID_IPV4_ADDR.
	• Use the *mask* argument to specify a range of addresses.
	• Use the **vrf** keyword to specify the front door VPN routing and forwarding (VRF) of the peer.
	• Use the *fvrf* argument to match the address in the front door virtual router forwarding (FVRF) Virtual Private Network (VPN) space.
	• Use the **host** keyword to specify an identity that matches the type ID_FQDN, whose fully qualified domain name (FQDN) ends with the domain name.
	• Use the **host domain** keyword to specify an identity that matches type ID_FQDN. The domain name is the same as the *domain-name* argument.
	• Use the **user** keyword to specify an identity that matches the FQDN.
	• Use the **user domain** keyword to specify an identity that matches the type ID_USER_FQDN. When the **user domain** keyword is present, all users having identities of the type ID_USER_FQDN and ending with *domain-name* are matched.

	Command or Action	Purpose

Step 8	**set interface** {**tunnel-ipsec** *intf-index* | **service-gre** *intf-index* | **service-ipsec** *intf-index*}
	
	Example:
	`RP/0/RP0/CPU0:router(config-isa-prof-match)# set interface tunnel-ipsec 50`
	
	or
	
	`RP/0/0/CPU0:router(config-isa-prof-match)# set service-gre 34`
		• **set interface tunnel-ipsec** command is available only if you previously selected the **local** keyword—Predefines the interface instance when IKE negotiates for IPSec service associations (SAs) for the traffic that is locally sourced or terminated and the local endpoint is the IKE responder.
		• **set interface service-gre** and **set interface service-ipsec** commands are available only on the Cisco XR 12000 Series Router—Predefines the interface instances when IKE negotiates for IPSec SAs for the traffic that is remotely sourced and terminated.
		• *intf-index* argument range differs based on whether you configure a tunnel or a service:
		– tunnel = 0 - 429496729
		– service = 1-65535

	Command or Action	Purpose
Step 9	**set ipsec-profile** *profile-name* **Example:** `RP/0/0/CPU0:router(config-isa-prof-match)# set ipsec-profile myprofile`	(Optional) Predefines the IPSec profile instance when IKE negotiates for IPSec service associations (SAs) for the traffic that is locally sourced or terminated and the local endpoint is the IKE responder. • Use the *profile-name* argument to set the name of the IPSec profile. • **Note** Only available if you selected the **local** keyword earlier in this procedure.

Step 10	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config-isa-prof-match)# end`
	or
	`RP/0/0/CPU0:router(config-isa-prof-match)# 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.

How to Configure a Periodic Dead Peer Detection Message

This task configures a periodic or on-demand dead peer detection (DPD) message.

SUMMARY STEPS

1. **configure**
2. **crypto isakmp keepalive** *seconds retry-seconds* [**periodic** | **on-demand**]
3. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**crypto isakmp keepalive** *seconds retry-seconds* [**periodic** | **on-demand**] **Example:** `RP/0/0/CPU0:router(config)# crypto isakmp keepalive 20 20 on-demand`	Uses the IKE security association (SA) feature to provide a mechanism to detect loss of connectivity between two IP Security (IPSec) peers. • Use the *seconds* argument to specify the number of seconds between keepalive messages. The range is from 10 to 3600. • Use the *retry-seconds* argument to specify the number of seconds between retries if keepalive fails. The range is from 2 to 60. • (Optional) Use the **periodic** keyword to specify the keepalive messages that are sent at regular intervals for DPD messages. • (Optional) Use the **on-demand** keyword to specify the DPD retries that are sent on demand.

Step 3	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config)# end`
	or
	`RP/0/0/CPU0:router(config)# 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.

Configuration Examples for Implementing IKE Security Protocol

This section provides the following configuration examples:

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.

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

In the example, the **encryption des** of policy 20 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 - Data Encryption Standard (168 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 a service-ipsec Interface with a Dynamic Profile: Example

The following shows how to configure a service-ipsec interface with a dynamic profile:

```
ipv4 access-list acl1
 10 permit ipv4 any any
!
interface service-ipsec1
 ipv4 address 44.44.44.44 255.255.255.0
 profile ipsec-profile1
 tunnel source 100.0.0.1
 service-location preferred-active 0/4/0
!

crypto isakmp
crypto isakmp policy 10
 authentication pre-share
 group 5
 encryption 3des
 lifetime 86400
!
crypto keyring ring1 vrf default
 pre-shared-key address 40.0.0.1 255.255.255.255 key key1
!
crypto isakmp profile ike-profile1
 keyring ring1
 match identity address 40.0.0.0/16 vrf default
  set interface service-ipsec1
 !
!
crypto isakmp keepalive 60 5
crypto ipsec transform-set tsfm1 esp-3des esp-md5-hmac
!
crypto ipsec profile ipsec-profile1
 set type dynamic
 match acl1 transform-set tsfm1
!
```

Note The service-ipsec interface is supported only on the Cisco XR 12000 Series Router.

Limiting an IKE Peer to a Particular Policy Set Based on Local IP Address: Example

The first part consists of selecting an ISAKMP policy related to the encryption method and identifying the SVI tunnel source. Users connecting to IP address 1.1.1.1 in the following example experience DES as the ISAKMP policy. However, users connecting to IP address 2.2.2.2 experience only AES as the ISAKMP policy.

More than one ISAKMP policy, or more than one IP address, can be used for matches. The rest of configuration remains the same; in other words, the configuration of the ISAKMP profile that matches a group name set to an SVI.

In this particular example, two policies have been configured in the policy set (policy 10 and 20).

Note that the SVI1 and SVI2 tunnel sources are respectively identified in bold as **local-address 1.1.1.1** and **local-address 2.2.2.2** in the example below.

```
RP/0/0/CPU0:router: configure
RP/0/0/CPU0:router(config)# crypto isakmp policy 10
```

```
RP/0/0/CPU0:router(config-isakmp)# encryption des << restricts use to DES only
RP/0/0/CPU0:router(config-isakmp)# group 2
RP/0/0/CPU0:router(config-isakmp)# authentication pre-share

RP/0/0/CPU0:router(config)# crypto isakmp policy 20
RP/0/0/CPU0:router(config-isakmp)# encryption aes << restricts use to AES only
RP/0/0/CPU0:router(config-isakmp)# group 2
RP/0/0/CPU0:router(config-isakmp)# authentication pre-share


RP/0/0/CPU0:router(config)# crypto isakmp policy-set policy_1 << match ID
RP/0/0/CPU0:router(config-isakmp-pol-set)# policy 10 << routing priority
RP/0/0/CPU0:router(config-isakmp-pol-set)# match identity local-address 1.1.1.1

RP/0/0/CPU0:router(config)# crypto isakmp policy-set policy_2 << match ID
RP/0/0/CPU0:router(config-isakmp-pol-set)# policy 20
RP/0/0/CPU0:router(config-isakmp-pol-set)# match identity local-address 2.2.2.2
RP/0/0/CPU0:router(config-isakmp-pol-set)# commit
RP/0/0/CPU0:router(config-isakmp-pol-set)# exit
RP/0/0/CPU0:router(config-isakmp)#
```

Configuring Cisco Easy VPN with a Local AAA-Method Server: Example

The following example shows how to configure Cisco Easy VPN with a local method-AAA server:

```
aaa authorization network author-net-local local
aaa authentication login authen-net-local local

local pool
   ipv4 pool-1 20.20.20.4 20.20.20.255
!
ipv4 access-list acl-3
    10 permit ipv4 any any
!
interface MgmtEth0/0/CPU0/0
    ipv4 address 3.1.73.1 255.255.0.0
!
interface GigabitEthernet0/1/0/1
    ipv4 address 2.0.0.1 255.0.0.0
    negotiation auto
!
interface service-ipsec3
    ipv4 address 30.3.3.3 255.255.0.0
    profile ipsec-prof-ezvpn
    tunnel source 10.20.100.3
    service-location preferred-active 0/2/0
!
crypto isakmp client configuration group group-a
    key group-a-key
    pool pool-1
!
crypto isakmp
crypto isakmp policy 30
    authentication pre-share
    group 2
    encryption aes
    lifetime 180
!
crypto isakmp profile isakmp-prof3
    client authentication list authen-net-local
    match identity group group-a
        set interface service-ipsec3
```

```
!
isakmp authorization list author-net-local
!
crypto ipsec transform-set tsfm3
    transform esp-3des esp-sha-hmac
!
crypto ipsec profile ipsec-prof-ezvpn
    set type dynamic
    match acl-3 transform-set tsfm3
    reverse-route
```

Note Cisco Easy VPN is supported only on the Cisco XR 12000 Series Router.

Configuring Cisco Easy VPN with a Remote AAA-Method Server: Example

On the remote AAA server, system administrators configures two lists, one for authentication and another for authorization.

Also required are the location of the remote AAA server and the administrator login password needed for access.

List names, as defined in the remote AAA-method server, must be added to the crypto ISAKMP profile.

In all other respects, configuration for a remote AAA-method server is the same as for a local AAA-method server. (See also Configuring Cisco Easy VPN with a Local AAA-Method Server: Example, page 210.)

```
aaa group server radius free_radius
 server-private 8.0.0.5 auth-port 1812 acct-port 1813
  key 7 094F471A1A0A
 !
!
aaa authorization network banana group free_radius
aaa authentication login banana group free_radius
local pool
 ipv4 localpool1000 17.1.1.1 17.1.1.250
!
ipv4 access-list remote_list
 10 permit ipv4 any any
!
interface GigabitEthernet0/0/0/CPU0:router(config-isakmp)#1
ipv4 address 2.0.0.2 255.255.255.0
!
interface GigabitEthernet0/0/0/2
 ipv4 address 8.0.0.2 255.255.255.0
!
interface service-ipsec1000
 ipv4 address 50.0.0.1 255.255.255.0
 profile vrf1000-prof-ipsec
 tunnel source 20.0.1.1
 service-location preferred-active 0/0/1
!
crypto isakmp
crypto isakmp policy 10
 authentication pre-share
 group 2
 encryption 3des
 lifetime 100
!
crypto isakmp profile vrf1000-ra
 aaa attribute-priority authorization
```

```
 self-identity address
 client authentication list banana
 match identity group grp1
  set interface service-ipsec1000
 !
 isakmp authorization list banana
!
crypto ipsec transform-set ATT
 transform esp-3des esp-sha-hmac
!
crypto ipsec profile vrf1000-prof-ipsec
 set type dynamic
 match remote_list transform-set ATT
 reverse-route
!
end
```

Note Cisco Easy VPN is supported only on the Cisco XR 12000 Series Router.

Configuring a Local ISAKMP Profile for Preshared Keys in ISAKMP Keyrings: Example

The following example shows how to configure a local ISAKMP profile:

```
interface tunnel-ipsec3001
 ipv4 unnumbered GigabitEthernet0/0/1/1.3001
 profile TUNNEL_IPSEC
 tunnel source GigabitEthernet0/0/1/1.3001
 tunnel destination 1.1.1.6
!

crypto ipsec profile TUNNEL_IPSEC
 set type static
 match TUNNEL_IPSEC transform-set TRANSFORM_SET
 reverse-route
```

Note The **reverse-route** command is not supported on the Cisco CRS-1 Router, and it can be omitted.

```
!
crypto keyring TUNNEL_IPSEC vrf default
 local-address 1.1.1.5
 pre-shared-key address 1.1.1.6 255.255.255.255 key cisco123
 pre-shared-key address 20.0.7.210 255.255.255.255 key cisco123
crypto isakmp profile local TUNNEL_IPSEC
 keyring TUNNEL_IPSEC
 match identity address 1.1.1.6/32 vrf default
  set interface tunnel-ipsec3001

!
```

Configuring VRF-Aware: Example

The following example shows how to configure VRF-aware:

```
ipv4 access-list acl-2_5-1
    10 permit ipv4 any any
```

```
ipv4 access-list acl-2_5-4
    10 permit ipv4 host 2.6.1.3 host 1.7.1.3
vrf IVRF1
!
vrf IVRF2
!
vrf IVRF3
!
vrf FVRF
!
interface GigabitEthernet0/1/0/0.1
    vrf FVRF
    ipv4 address 10.0.83.2 255.255.255.0
!
interface GigabitEthernet0/1/0/1.1
    vrf IVRF1
    ipv4 address 2.6.0.1 255.255.0.0
    dot1q vlan 61
!
interface GigabitEthernet0/1/0/1.2
    vrf IVRF2
    ipv4 address 2.6.1.1 255.255.0.0
    dot1q vlan 62
!
interface GigabitEthernet0/1/0/1.3
    vrf IVRF3
    ipv4 address 2.6.0.1 255.255.0.0
    dot1q vlan 63
!
interface GigabitEthernet0/1/0/0.11
    vrf FVRF

    ipv4 address 10.0.91.1 255.255.255.0
    dot1q vlan 91
!
interface GigabitEthernet0/1/0/0.12
    vrf FVRF
    ipv4 address 10.0.92.1 255.255.255.0
    dot1q vlan 92
!
interface GigabitEthernet0/1/0/0.13
    vrf FVRF
    ipv4 address 10.0.93.1 255.255.255.0
    dot1q vlan 93
interface service-ipsec15
    vrf IVRF1
    ipv4 address 115.115.115.115 255.255.0.0
    service-location preferred-active 0/2/0
    profile ipsec-prof15 tunnel vrf FVRF
    tunnel source 10.20.100.15
interface service-ipsec16
    vrf IVRF2
    ipv4 address 116.116.116.116 255.255.0.0
    service-location preferred-active 0/2/0
    profile ipsec-prof16 tunnel vrf FVRF
    tunnel source 10.20.100.16
    tunnel destination 10.0.85.2

router static
    address-family ipv4 unicast
        1.7.0.3/32 service-ipsec15
        1.7.0.4/32 service-ipsec15
    vrf FVRF
    address-family ipv4 unicast
```

```
    10.0.0.0/16 10.0.83.1

crypto isakmp
crypto isakmp policy 60
    authentication pre-share
    hash sha
    group 5
    encryption aes
    lifetime 86400
!
crypto keyring kr11 vrf FVRF
    pre-shared-key address 10.0.91.2 255.255.255.255 key key-vrf
    pre-shared-key address 10.0.92.2 255.255.255.255 key key-vrf
    pre-shared-key address 10.0.93.2 255.255.255.255 key key-vrf
!
crypto keyring kr12 vrf FVRF
    local-address 10.20.100.16
    pre-shared-key address 0.0.0.0 0.0.0.0 key key16
!
crypto isakmp profile isakmp-prof6
    keyring kr11
    match identity address 10.0.91.2/32 vrf FVRF
        set interface service-ipsec15
    match identity address 10.0.92.2/32 vrf FVRF
        set interface service-ipsec15
    match identity address 10.0.93.2/32 vrf FVRF
        set interface service-ipsec15
!
!
crypto isakmp profile isakmp-prof7
    keyring kr12
    match identity address 10.0.85.2/32 vrf FVRF
        set interface service-ipsec16
```

Note VRF-aware is supported only on the Cisco XR 12000 Series Router.

Additional References

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

Related Documents

Related Topic	Document Title
IKE security protocol commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS XR System Security Command Reference*
IPSec-related object tracking commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS XR System Management Command Reference*

Related Topic	Document Title
Object tracking configuration procedures, including examples	*Cisco IOS XR System Management Configuration Guide*
IPSec network security commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*IPSec Network Security Commands on Cisco IOS XR Software* module in *Cisco IOS XR System Security Command Reference*

Standards

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

MIBs

MIBs	MIBs Link
—	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
RFC 2401	*Security Architecture for the Internet Protocol*
RFC 2402	*IP Authentication Header*
RFC 2403	*The Use of HMAC-MD5-96 within ESP and AH*
RFC 2404	*The Use of HMAC-SHA-1-96 within ESP and AH*
RFC 2405	*The ESP DES-CBC Cipher Algorithm With Explicit IV*
RFC 2406	*IP Encapsulating Security Payload (ESP)*
RFC 2407	*The Internet IP Security Domain of Interpretation for ISAKMP*
RFC 2408	*Internet Security Association and Key Management Protocol (ISAKMP)*
RFC 2409	*The Internet Key Exchange (IKE)*

Technical Assistance

Description	Link
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.	http://www.cisco.com/techsupport