

# Internet Key Exchange Security Protocol

---

## Description

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

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

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

## Benefits

IKE automatically negotiates IPsec security associations (SAs) and enables IPsec secure communications without costly manual preconfiguration.

Specifically, IKE provides these benefits:

- Eliminates the need to manually specify all the IPsec security parameters in the crypto maps at both peers.
- Allows you to specify a lifetime for the IPsec security association.
- Allows encryption keys to change during IPsec sessions.
- Allows IPsec to provide anti-replay services.
- Permits Certification Authority (CA) support for a manageable, scalable IPsec implementation.
- Allows dynamic authentication of peers.

## Supported Standards

Cisco implements the following standards:

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

For more information on IPSec, see the “IPSec Network Security” feature documentation.

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

IKE is implemented per the latest version of the “The Internet Key Exchange,” Internet Draft (draft-ietf-ipsec-isakmp-oakley-xx.txt).

**ISAKMP**—The Internet Security Association and Key Management Protocol. A protocol framework which defines payload formats, the mechanics of implementing a key exchange protocol, and the negotiation of a security association.

ISAKMP is implemented per the latest version of the “Internet Security Association and Key Management Protocol (ISAKMP)” Internet Draft (draft-ietf-ipsec-isakmp-xx.txt).

**Oakley**—A key exchange protocol which defines how to derive authenticated keying material.

**Skeme**—A key exchange protocol which defines how to derive authenticated keying material, with rapid key refreshment.

The component technologies implemented for use by IKE include:

- **DES**—The Data Encryption Standard (DES) 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.

- **Diffie-Hellman**—A public-key cryptography protocol which allows two parties to establish a shared secret over an unsecure communications channel. Diffie-Hellman is used within IKE to establish session keys. 768-bit and 1024-bit Diffie-Hellman groups are supported.
- **MD5 (HMAC variant)**—MD5 (Message Digest 5) is a hash algorithm used to authenticate packet data. HMAC is a variant which provides an additional level of hashing.
- **SHA (HMAC variant)**—SHA (Secure Hash Algorithm) is a hash algorithm used to authenticate packet data. HMAC is a variant which 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 Adleman. RSA signatures provides non-repudiation while RSA encrypted nonces provide repudiation.

IKE interoperates with the following standard:

**X.509v3 certificates**—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 wish 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).

## List of Terms

**anti-replay**—A security service in which the receiver can reject old or duplicate packets in order to protect itself against replay attacks. IPSec provides optional anti-replay services by use of a sequence number combined with the use of authentication.

**data authentication**—Includes two concepts:

- Data integrity (verify that data has not been altered).

- Data origin authentication (verify that the data was actually sent by the claimed sender).

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

**peer**—In the context of this document, a peer refers to a router or other device that participates in IPsec and IKE.

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

**repudiation**—A quality that prevents a third party from being able to prove that a communication between two other parties ever took place. This is a desirable quality if you do not want your communications to be traceable. **Non-repudiation** is the opposite quality—a third party can prove that a communication between two other parties took place. Non-repudiation is desirable if you want to be able to trace your communications and prove that they occurred.

**security association**—A security association (SA) describes how two or more entities will utilize security services to communicate securely. For example, an IPsec SA defines the encryption algorithm (if used), the authentication algorithm, and the shared session key to be used during the IPsec connection.

Both IPsec and IKE require and use SAs to identify the parameters of their connections. IKE can negotiate and establish its own SA. The IPsec SA is established either by IKE or by manual user configuration.

## Platforms

This feature is supported on these platforms:

- Cisco 1600 series
- Cisco 2500 series
- Cisco 2600 series
- Cisco 3600 series
- Cisco 4000 series (Cisco 4000, 4000-M, 4500, 4500-M, 4700, 4700-M)
- Cisco 7200 series
- Cisco 7500 series
- Cisco AS5300

## Configuration Tasks

To configure IKE, perform the following tasks:

- Enable or Disable IKE
- Ensure Access Lists Are Compatible with IKE
- Create IKE Policies
- Manually Configure RSA Keys (Optional, depending on IKE parameters)
- Configure Pre-Shared Keys (Optional, depending on IKE parameters)

- Clear IKE Connections (Optional)
- Troubleshoot IKE (Optional)

## Enable or Disable IKE

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

If you do not want IKE to be used with your IPSec implementation, you can disable it at all IPSec peers. Note that IKE must be enabled or disabled at all IPSec peers; you cannot have a mix of IKE-enabled and IKE-disabled peers within your IPSec network.

If you disable IKE, you will have to make these concessions at the peers:

- You must manually specify all the IPSec security associations in the crypto maps at all peers. (Crypto map configuration is described in the “IPSec Network Security” feature documentation.)
- The peers’ IPSec security associations will never time out for a given IPSec session.
- During IPSec sessions between the peers, the encryption keys will never change.
- Anti-replay services will not be available between the peers.
- Certification Authority (CA) support cannot be used.

To disable or enable IKE, perform one of the following tasks in global configuration mode:

Task	Command
Disable IKE.	<b>no crypto isakmp enable</b>
Enable IKE.	<b>crypto isakmp enable</b>

If you disable IKE, you can skip the rest of the tasks in this chapter and go directly to IPSec configuration as described in the “IPSec Network Security” feature documentation.

## Ensure Access Lists Are Compatible with IKE

IKE negotiation uses UDP on port 500. Ensure that your access lists are configured so that UDP port 500 traffic is not blocked at interfaces used by IKE and IPSec. In some cases you might need to add a statement to your access lists to explicitly permit UDP port 500 traffic.

## Create 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.

To create an IKE policy, follow the guidelines in these sections:

- Why Do You Need to Create These Policies?
- What Parameters Do You Define in a Policy?
- How Do IKE Peers Agree Upon a Matching Policy?
- Which Value Should You Select for Each Parameter?
- Creating Policies
- Additional Configuration Required for IKE Policies

## Why Do You Need to Create These Policies?

IKE negotiations must be protected, so each IKE negotiation begins by each peer agreeing on a common (shared) IKE policy. This policy states which security parameters will be used to protect subsequent IKE negotiations.

After the two peers agree upon 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 will match a remote peer's policy.

## What Parameters Do You Define in a Policy?

There are five parameters to define in each IKE policy:

Parameter	Accepted Values	Keyword	Default Value
encryption algorithm	56-bit DES-CBC	<b>des</b>	56-bit DES-CBC
hash algorithm	SHA-1 (HMAC variant)	<b>sha</b>	SHA-1
	MD5 (HMAC variant)	<b>md5</b>	
authentication method	RSA signatures	<b>rsa-sig</b>	RSA signatures
	RSA encrypted nonces	<b>rsa-encr</b>	
	pre-shared keys	<b>pre-share</b>	
Diffie-Hellman group identifier	768-bit Diffie-Hellman or	<b>1</b>	768-bit Diffie-Hellman
	1024-bit Diffie-Hellman	<b>2</b>	
security association's lifetime <sup>1</sup>	can specify any number of seconds	-	86400 seconds (one day)

1. For information about this lifetime and how it is used, see the command description for the **lifetime (IKE policy)** command.

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

## How Do IKE Peers Agree Upon a Matching Policy?

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

A match is made when both policies from the two peers contain the same encryption, hash, authentication, and Diffie-Hellman parameter values, and when the remote peer's policy specifies a lifetime 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—will be used.)

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

If a match is found, IKE will complete negotiation, and IPSec security associations will be created.

---

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

---

## Which Value Should You Select for Each Parameter?

You can select certain values for each parameter, per the IKE standard. But why chose 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 other device’s supported value. Aside from this, there is often a trade-off between security and performance, and many of these parameter values represent such a trade-off. You should evaluate the level of your network’s security risks and your tolerance for these risks. Then the following tips might help you select which value to specify for each parameter.

- The encryption algorithm currently has only one option: 56-bit DES-CBC.
- 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. There has been a demonstrated successful (but extremely difficult) attack against MD5; however, the HMAC variant used by IKE prevents this attack.

- The authentication method has three options: RSA signatures, RSA encrypted nonces, and pre-shared keys.
  - RSA signatures provides non-repudiation 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 requires use of a Certification Authority (CA). Using a CA can dramatically improve the manageability and scalability of your IPsec network.
  - RSA encrypted nonces provides repudiation for the IKE negotiation (you cannot prove to a third party that you had an IKE negotiation with the remote peer). This is used to prevent a third party from knowing about your activity over the network.

An IKE security association that is authenticated with RSA encrypted nonces is harder to crack than one authenticated with RSA signatures, though using the nonces vs. signatures can somewhat slow the negotiations.

RSA encrypted nonces require that peers possess each other’s public keys but do not use a Certification Authority. Instead, there are two ways for peers to get each others’ public keys:

1) During configuration you manually configure RSA keys (as described in the section “Manually Configure RSA Keys”) or

2) If your local peer has previously used RSA signatures 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.)

- Pre-shared keys are clumsy to use if your secured network is large, and do not scale well with a growing network. However, they do not require use of a Certification Authority, as do RSA signatures.
- The Diffie-Hellman group identifier has two options: 768-bit or 1024-bit Diffie-Hellman. 1024-bit Diffie-Hellman is harder to crack, but requires more CPU time to execute.
- The security association’s lifetime can be set to any value.

As a general rule, the shorter the lifetime (up to a point), the more secure your IKE negotiations will be. 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 (IKE policy)** command.

## Creating Policies

You can create multiple IKE policies, each with a different combination of parameter values. For each policy that you create, you 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 does not necessarily have to be the same; see details in the section “How Do IKE Peers Agree Upon a Matching Policy?”)

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

To configure a policy, perform the following tasks starting in global configuration mode:

Task	Command
Identify the policy to create. (Each policy is uniquely identified by the priority number you assign.)  (This command puts you into the config-isakmp command mode.)	<b>crypto isakmp policy</b> <i>priority</i>
Specify the encryption algorithm.	<b>encryption</b> <b>des</b>
Specify the hash algorithm.	<b>hash</b> { <b>sha</b>   <b>md5</b> }
Specify the authentication method.	<b>authentication</b> { <b>rsa-sig</b>   <b>rsa-encr</b>   <b>pre-share</b> }
Specify the Diffie-Hellman group identifier.	<b>group</b> { <b>1</b>   <b>2</b> }
Specify the security association’s lifetime.	<b>lifetime</b> <i>seconds</i>
Exit the config-isakmp command mode.	<b>exit</b>
(Optional) View all existing IKE policies.  (Perform this task in EXEC mode.)	<b>show crypto isakmp policy</b>

If you do not specify a value for a parameter, the default value is assigned.

---

**Note** The default policy and the default values for configured policies do not show up in the configuration when you issue a **show running** command. Instead, to see the default policy and any default values within configured policies, use the **show crypto isakmp policy** command.)

---

## Additional Configuration Required for IKE Policies

Depending on which authentication method you specify in your IKE policies, you need to do certain additional configuration 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 must configure the peers to obtain certificates from a Certification Authority (CA). (And, of course, the CA must be properly configured to issue the certificates.) Configure this certificate support as described in the “Certification Authority Interoperability” feature documentation.

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

- **RSA encrypted nonces method:**

If you specify RSA encrypted nonces as the authentication method in a policy, you need to ensure that each peer has the other peers’ public keys.

Unlike RSA signatures, the RSA encrypted nonces method does not use certificates to exchange public keys. Instead, you ensure that each peer has the others’ public keys as follows:

- Either manually configure RSA keys as described in the section “Manually Configure RSA Keys,” or
- Ensure that an IKE exchange using RSA signatures has already occurred between the peers. (The peers’ public keys are exchanged during the RSA-signatures-based IKE negotiations.)

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 will be used the first time because the peers do not yet have each others’ public keys. Then, future IKE negotiations will be able to use RSA encrypted nonces because the public keys will have been exchanged.

Of course, this alternative requires that you have Certification Authority support configured.

- **Pre-shared keys authentication method:**

If you specify pre-shared keys as the authentication method in a policy, you must configure these pre-shared keys as described in the section “Configure Pre-Shared Keys.”

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

## Manually Configure 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 certification authority (CA).

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

- Generate RSA Keys
- Set ISAKMP Identity

- Specify All the Other Peers' RSA Public Keys

## Generate RSA Keys

To generate RSA keys, perform the following tasks starting in global configuration mode:

Task	Command
Generate RSA keys.	<b>crypto key generate rsa [usage-keys]</b>
View the generated RSA public key (in EXEC mode).	<b>show crypto key mypubkey rsa</b>

Remember to repeat these tasks at each peer (without CA support) that uses RSA encrypted nonces in an IKE policy.

## Set ISAKMP Identity

You should set the ISAKMP identity for each peer that uses pre-shared 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 the router's ISAKMP identity set.

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

To set a peer's ISAKMP identity, perform the following tasks in global configuration mode:

Task	Command
<b>At the local peer:</b> Specify the peer's ISAKMP identity by IP address or by hostname. <sup>1</sup>	<b>crypto isakmp identity {address   hostname}</b>
<b>At all remote peers:</b> If the local peer's ISAKMP identity was specified using a hostname, map the peer's hostname to its IP address(es) at all the remote peers. (This step might be unnecessary if the hostname/address is already mapped in a DNS server.)	<b>ip host hostname address1 [address2...address8]</b>

1. See the **crypto isakmp identity** command description for guidelines for when to use the IP address vs. the hostname.

Remember to repeat these tasks at each peer that uses pre-shared keys in an IKE policy.

## Specify All the Other Peers' RSA Public Keys

At each peer, specify all the other peers' RSA public keys by performing the following tasks starting in global configuration mode:

Task	Command
<b>Step 1</b> Enter public key configuration mode.	<b>crypto key pubkey-chain rsa</b>

Task	Command
<p><b>Step 2</b> Indicate which remote peer's RSA public key you are going to specify.</p> <p>If the remote peer uses its hostname as its ISAKMP identity, use the <b>named-key</b> command and specify the remote peer's fully qualified domain name (such as somerouter.companyx.com) as the <i>key-name</i>.</p> <p>If the remote peer uses its IP address as its ISAKMP identity, use the <b>addressed-key</b> command and specify the remote peer's IP address as the <i>key-address</i>.</p>	<p><b>named-key</b> <i>key-name</i> [<b>encryption</b>   <b>signature</b>]</p> <p>or</p> <p><b>addressed-key</b> <i>key-address</i> [<b>encryption</b>   <b>signature</b>]</p>
<p><b>Step 3</b> If you used a fully qualified domain name to name the remote peer in Step 2 (using the <b>named-key</b> command), you can optionally specify the remote peer's IP address.</p>	<p><b>address</b> <i>ip-address</i></p>
<p><b>Step 4</b> Specify the remote peer's RSA public key. This is the key viewed by the remote peer's administrator previously when he generated his router's RSA keys.</p>	<p><b>key-string</b></p> <p><i>key-string</i></p> <p><b>quit</b></p>
<p><b>Step 5</b> Repeat steps 2 through 4 to specify the RSA public keys of all the other IPsec peers that use RSA encrypted nonces in an IKE policy.</p>	
<p><b>Step 6</b> Return to global configuration mode.</p>	<p><b>exit</b></p>

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

To view RSA public keys while or after you configure them, perform the following tasks in EXEC mode:

Task	Command
View a list of all the RSA public keys stored on your router, or view details of a particular RSA public key stored on your router.	<b>show crypto key pubkey-chain rsa</b> { <b>name</b> <i>key-name</i>   <b>address</b> <i>key-address</i> }

## Configure Pre-Shared Keys

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

- First, set each peer's ISAKMP identity. Each peer's identity should be set to either its hostname or by its IP address. By default, a peer's identity is set to its IP address. Setting ISAKMP identities is described previously in the section "Set ISAKMP Identity."
- Next, specify the shared keys at each peer. Note that a given pre-shared 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.

To specify pre-shared keys at a peer, perform the following tasks in global configuration mode;

Task	Command
<p><b>At the local peer:</b> Specify the shared key to be used with a particular remote peer.</p> <p>If the remote peer specified their ISAKMP identity with an address, use the <b>address</b> keyword in this step; otherwise use the <b>hostname</b> keyword in this step.</p>	<p><b>crypto isakmp key</b> <i>keystring</i> <b>address</b> <i>peer-address</i></p> <p>or</p> <p><b>crypto isakmp key</b> <i>keystring</i> <b>hostname</b> <i>peer-hostname</i></p>
<p><b>At the remote peer:</b> Specify the shared key to be used with the local peer. This is the same key you just specified at the local peer.</p> <p>If the local peer specified their ISAKMP identity with an address, use the <b>address</b> keyword in this step; otherwise use the <b>hostname</b> keyword in this step.</p>	<p><b>crypto isakmp key</b> <i>keystring</i> <b>address</b> <i>peer-address</i></p> <p>or</p> <p><b>crypto isakmp key</b> <i>keystring</i> <b>hostname</b> <i>peer-hostname</i></p>
Repeat the previous two steps for each remote peer.	

Remember to repeat these tasks at each peer that uses pre-shared keys in an IKE policy.

## Clear IKE Connections

If you want, you can clear existing IKE connections.

To clear IKE connections, perform the following tasks in EXEC mode:

Task	Command
View existing IKE connections; note the connection identifiers for connections you wish to clear.	<b>show crypto isakmp sa</b>
Clear IKE connections.	<b>clear crypto isakmp</b> [ <i>connection-id</i> ]

## Troubleshoot IKE

To assist in IKE troubleshooting, perform the following tasks in EXEC mode:

Task	Command
View the parameters for each configured IKE policy.	<b>show crypto isakmp policy</b>
View all current IKE security associations.	<b>show crypto isakmp sa</b>
Display <b>debug</b> messages about IKE events.	<b>debug crypto isakmp</b>

## Configuration Examples

This example creates two IKE policies, with policy 15 as the highest priority, policy 20 as the next priority and the existing default priority as the lowest priority.

This example also creates a pre-shared key to be used with policy 20 with the remote peer whose IP address is 171.69.224.33.

```
crypto isakmp policy 15
  encryption des
  hash md5
  authentication rsa-sig
  group 2
  lifetime 5000
crypto isakmp policy 20
  authentication pre-share
  lifetime 10000
crypto isakmp key 1234567890 address 171.69.224.33
```

In the above example, the **encryption des** of policy 15 would not appear in the written configuration because this is the default value for the encryption algorithm parameter.

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

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

Note that although the output shows “no volume limit” for the lifetimes, you can currently only configure a time lifetime (such as 86400 seconds); volume limit lifetimes are not configurable.

## Command Reference

This section documents new or modified commands. All other commands used with this feature are documented in the Cisco IOS Release 11.3 command references.

The new debug command used with IKE is described in the “Debug Commands” section.

- **address**
- **addressed-key**
- **authentication (IKE policy)**
- **clear crypto isakmp**
- **crypto isakmp enable**
- **crypto isakmp identity**
- **crypto isakmp key**
- **crypto isakmp policy**
- **crypto key generate rsa**
- **crypto key pubkey-chain rsa**
- **encryption (IKE policy)**
- **group (IKE policy)**
- **hash (IKE policy)**
- **key-string**
- **lifetime (IKE policy)**
- **named-key**
- **show crypto isakmp policy**
- **show crypto isakmp sa**
- **show crypto key mypubkey rsa**
- **show crypto key pubkey-chain rsa**

## address

To specify the IP address of the remote peer's RSA public key you will manually configure, use the **address** public key configuration command. This command should only be used when the router has a single interface that processes IPSec.

```
address ip-address
```

### Syntax Description

*ip-address* Specifies the IP address of the remote peer.

### Default

This command has no defaults.

### Command Mode

Public key configuration

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

Use this command in conjunction with the **named-key** command to specify which IPSec peer's RSA public key you will manually configure next.

### Examples

This example manually specifies the RSA public keys of an IPSec peer.

```
myrouter(config)# crypto key pubkey-chain rsa  
myrouter(config-pubkey-chain)# named-key otherpeer.companyx.com  
myrouter(config-pubkey-key)# address 10.5.5.1  
myrouter(config-pubkey-key)# key-string  
myrouter(config-pubkey)# 005C300D 06092A86 4886F70D 01010105  
myrouter(config-pubkey)# 00034B00 30480241 00C5E23B 55D6AB22  
myrouter(config-pubkey)# 04AEF1BA A54028A6 9ACC01C5 129D99E4  
myrouter(config-pubkey)# 64CAB820 847EDAD9 DF0B4E4C 73A05DD2  
myrouter(config-pubkey)# BD62A8A9 FA603DD2 E2A8A6F8 98F76E28  
myrouter(config-pubkey)# D58AD221 B583D7A4 71020301 0001  
myrouter(config-pubkey)# quit  
myrouter(config-pubkey-key)# exit  
myrouter(config-pubkey-chain)# exit  
myrouter(config)#
```

### Related Commands

**addressed-key**  
**crypto key pubkey-chain rsa**  
**key-string**  
**show crypto key pubkey-chain rsa**

---

## addressed-key

To specify which peer's RSA public key you will manually configure, use the **addressed-key** public key chain configuration command.

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

### Syntax Description

<i>key-address</i>	Specifies the IP address of the remote peer's RSA keys.
<b>encryption</b>	(Optional) Indicates that the RSA public key to be specified will be an encryption special usage key.
<b>signature</b>	(Optional) Indicates that the RSA public key to be specified will be a signature special usage key.

### Default

If neither the **encryption** nor **signature** keywords are used, general purpose keys will be specified.

### Command Mode

Public key chain configuration. Using this command puts you into public key configuration mode.

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

Use this command or the **named-key** command to specify which IPSec peer's RSA public key you will manually configure next.

Follow this command with the **key-string** command to specify the key.

If the IPSec remote peer generated general purpose RSA keys, do not use the **encryption** or **signature** keywords.

If the IPSec remote peer generated special usage keys, you must manually specify both keys: perform this command and the **key-string** command two times and use the **encryption** and **signature** keywords respectively.

## Examples

This example manually specifies the RSA public keys of two IPSec peers. The peer at 10.5.5.1 uses general purpose keys, and the other peer uses special usage keys.

```
myrouter(config)# crypto key pubkey-chain rsa
myrouter(config-pubkey-chain)# named-key otherpeer.companyx.com
myrouter(config-pubkey-key)# address 10.5.5.1
myrouter(config-pubkey-key)# key-string
myrouter(config-pubkey)# 005C300D 06092A86 4886F70D 01010105
myrouter(config-pubkey)# 00034B00 30480241 00C5E23B 55D6AB22
myrouter(config-pubkey)# 04AEF1BA A54028A6 9ACC01C5 129D99E4
myrouter(config-pubkey)# 64CAB820 847EDAD9 DFOB4E4C 73A05DD2
myrouter(config-pubkey)# BD62A8A9 FA603DD2 E2A8A6F8 98F76E28
myrouter(config-pubkey)# D58AD221 B583D7A4 71020301 0001
myrouter(config-pubkey)# quit
myrouter(config-pubkey-key)# exit
myrouter(config-pubkey-chain)# addressed-key 10.1.1.2 encryption
myrouter(config-pubkey-key)# key-string
myrouter(config-pubkey)# 00302017 4A7D385B 1234EF29 335FC973
myrouter(config-pubkey)# 2DD50A37 C4F4B0FD 9DADE748 429618D5
myrouter(config-pubkey)# 18242BA3 2EDFBDD3 4296142A DDF7D3D8
myrouter(config-pubkey)# 08407685 2F2190A0 0B43F1BD 9A8A26DB
myrouter(config-pubkey)# 07953829 791FCDE9 A98420F0 6A82045B
myrouter(config-pubkey)# 90288A26 DBC64468 7789F76E EE21
myrouter(config-pubkey)# quit
myrouter(config-pubkey-key)# exit
myrouter(config-pubkey-chain)# addressed-key 10.1.1.2 signature
myrouter(config-pubkey-key)# key-string
myrouter(config-pubkey)# 0738BC7A 2BC3E9F0 679B00FE 53987BCC
myrouter(config-pubkey)# 01030201 42DD06AF E228D24C 458AD228
myrouter(config-pubkey)# 58BB5DDD F4836401 2A2D7163 219F882E
myrouter(config-pubkey)# 64CE69D4 B583748A 241BED0F 6E7F2F16
myrouter(config-pubkey)# 0DE0986E DF02031F 4B0B0912 F68200C4
myrouter(config-pubkey)# C625C389 0BFF3321 A2598935 C1B1
myrouter(config-pubkey)# quit
myrouter(config-pubkey-key)# exit
myrouter(config-pubkey-chain)# exit
myrouter(config)#
```

## Related Commands

**crypto key pubkey-chain rsa**

**key-string**

**named-key**

**show crypto key pubkey-chain rsa**

## authentication (IKE policy)

To specify the authentication method within an IKE policy, use the **authentication (IKE policy)** ISAKMP policy configuration command. IKE policies define a set of parameters to be used during IKE negotiation. To reset the authentication method to the default value, use the **no** form of the command.

```
authentication { rsa-sig | rsa-encr | pre-share }  
no authentication
```

### Syntax Description

<b>rsa-sig</b>	Specifies RSA signatures as the authentication method.
<b>rsa-encr</b>	Specifies RSA encrypted nonces as the authentication method.
<b>pre-share</b>	Specifies pre-shared keys as the authentication method.

### Default

RSA signatures

### Command Mode

ISAKMP policy configuration (config-isakmp)

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

Use this command to specify the authentication method to be used in an IKE policy.

If you specify RSA signatures, you must configure your peer routers to obtain certificates from a Certification Authority (CA).

If you specify RSA encrypted nonces, you must ensure that each peer has the other peer's RSA public keys. (See the **crypto key pubkey-chain rsa**, **addressed-key**, **named-key**, **address**, and **key-string** commands.)

If you specify pre-shared keys, you must also separately configure these pre-shared keys. (See the **crypto isakmp identity** and **crypto isakmp key** commands.)

### Example

This example configures an IKE policy with pre-shared keys as the authentication method (and with all other parameters set to the defaults):

```
MyPeerRouter(config)# crypto isakmp policy 15  
MyPeerRouter(config-isakmp)# authentication pre-share  
MyPeerRouter(config-isakmp)# exit  
MyPeerRouter(config)#
```

Related Commands

**crypto isakmp key**  
**crypto isakmp policy**  
**crypto key generate rsa**  
**encryption (IKE policy)**  
**group (IKE policy)**  
**hash (IKE policy)**  
**lifetime (IKE policy)**  
**show crypto isakmp policy**

## clear crypto isakmp

To clear active IKE connections, use the **clear crypto isakmp** global configuration command.

```
clear crypto isakmp [connection-id]
```

### Syntax Description

*connection-id* (Optional) Specifies which connection to clear. If this argument is not used, all existing connections will be cleared.

### Default

If the *connection-id* argument is not used, all existing IKE connections will be cleared when this command is issued.

### Command Mode

Global configuration

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

Use this command to clear active IKE connections.

### Example

This example clears an IKE connection between two peers connected by interfaces 172.21.114.123 and 172.21.114.67.

```
MyPeerRouter# show crypto isakmp sa
      dst          src          state          conn-id  slot
172.21.114.123  172.21.114.67  QM_IDLE        1         0
155.0.0.2       155.0.0.1      QM_IDLE        8         0

MyPeerRouter# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
MyPeerRouter(config)# clear crypto isakmp 1
MyPeerRouter(config)# exit
MyPeerRouter# show crypto isakmp sa
      dst          src          state          conn-id  slot
155.0.0.2       155.0.0.1      QM_IDLE        8         0

MyPeerRouter#
```

### Related Commands

**show crypto isakmp sa**

## crypto isakmp enable

To globally enable IKE at your peer router, use the **crypto isakmp enable** global configuration command. To disable IKE at the peer, use the **no** form of the command

**crypto isakmp enable**  
**no crypto isakmp enable**

### Syntax Description

This command has no arguments or keywords.

### Default

IKE is enabled.

### Command Mode

Global configuration

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

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

If you do not want IKE to be used in your IPSec implementation, you can disable IKE at all your IPSec peers. If you disable IKE at one peer you must disable it at all your IPSec peers.

If you disable IKE, you will have to make these concessions at the peers:

- You must manually specify all the IPSec security associations (SAs) in the crypto maps at the peers. (Crypto map configuration is described in the “IPSec Network Security” feature documentation.)
- The peers’ IPSec SAs will never time out for a given IPSec session.
- During IPSec sessions between the peers, the encryption keys will never change.
- Anti-replay services will not be available between the peers.
- Certification Authority (CA) support cannot be used.

### Example

This example disables IKE at one peer. (The same command should be issued at all remote peers.)

```
no crypto isakmp enable
```

## crypto isakmp identity

To define the identity the router uses when participating in the IKE protocol, use the **crypto isakmp identity** global configuration command. Set an ISAKMP identity whenever you specify pre-shared keys. To reset the ISAKMP identity to the default value (address), use the **no** form of the command.

```
crypto isakmp identity {address | hostname}  
no crypto isakmp identity
```

### Syntax Description

<b>address</b>	Sets the ISAKMP identity to the IP address of the interface that is used to communicate to the remote peer during IKE negotiations.
<b>hostname</b>	Sets the ISAKMP identity to the hostname concatenated with the domain name (for example, myhost.domain.com).

### Default

The IP address is used for the ISAKMP identity.

### Command Mode

Global configuration

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

Use this command to specify an ISAKMP identity either by IP address or by hostname.

The **address** keyword is typically used when there is only one interface (and therefore only one IP address) that will be used by the peer for IKE negotiations, and the IP address is known.

The **hostname** keyword should be used if there is more than one interface on the peer that might be used for IKE negotiations, or if the interface's IP address is unknown (such as with dynamically-assigned IP addresses).

As a general rule, you should set all peers' identities in the same way, either by IP address or by hostname.

### Examples

The following example uses pre-shared keys at two peers and sets both their ISAKMP identities to IP address.

At the local peer (at 10.0.0.1) the ISAKMP identity is set and the pre-shared key is specified:

```
crypto isakmp identity address
crypto isakmp key sharedkeystring address 198.168.1.33
```

At the remote peer (at 198.168.1.33) the ISAKMP identity is set and the same pre-shared key is specified:

```
crypto isakmp identity address
crypto isakmp key sharedkeystring address 10.0.0.1
```

Note that in the above example if the **crypto isakmp identity** command had never been performed, the ISAKMP identities would have still been set to IP address, the default identity.

The following example uses pre-shared keys at two peers and sets both their ISAKMP identities to hostname.

At the local peer the ISAKMP identity is set and the pre-shared key is specified:

```
crypto isakmp identity hostname
crypto isakmp key sharedkeystring hostname RemoteRouter.companyx.com
ip host RemoteRouter.companyx.com 198.168.0.1
```

At the remote peer the ISAKMP identity is set and the same pre-shared key is specified:

```
crypto isakmp identity hostname
crypto isakmp key sharedkeystring hostname LocalRouter.companyx.com
ip host LocalRouter.companyx.com 10.0.0.1 10.0.0.2
```

In the above example, hostnames are used for the peers' identities. Why? Because the local peer has two interfaces which might be used during an IKE negotiation.

In the above example the IP addresses are also mapped to the hostnames; this mapping would not have been necessary if the routers' hostnames were already mapped in DNS.

### Related Commands

**authentication (IKE policy)**

**crypto isakmp key**

## crypto isakmp key

To configure a pre-shared authentication key, use the **crypto isakmp key** global configuration command. You must configure this key whenever you specify pre-shared keys in an IKE policy. To delete a pre-shared authentication key, use the **no** form of the command.

```
crypto isakmp key keystring address peer-address  
crypto isakmp key keystring hostname peer-hostname  
no crypto isakmp key keystring address peer-address  
no crypto isakmp key keystring hostname peer-hostname
```

### Syntax Description

<i>keystring</i>	Specify the pre-shared key. Use any combination of alpha-numeric characters up to 128 bytes. This pre-shared key must be identical at both peers.
<i>peer-address</i>	Specify the IP address of the remote peer.
<i>hostname</i>	Specify the hostname of the remote peer. This is the peer's hostname concatenated with its domain name (for example, myhost.domain.com).

### Default

There is no default pre-shared authentication key.

### Command Mode

Global configuration

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

Use this command to configure pre-shared authentication keys. You must perform this command at both peers.

If an IKE policy includes pre-shared keys as the authentication method, these pre-shared keys must be configured at both peers—otherwise the policy cannot be used (the policy will not be submitted for matching by the IKE process). The **crypto isakmp key** command is the second task required to configure the pre-shared keys at the peers. (The first task is accomplished with the **crypto isakmp identity** command.)

Use the **address** keyword if the remote peer previously set its ISAKMP identity with its IP address.

Use the **hostname** keyword if the remote peer previously set its ISAKMP identity with its hostname.

With the **hostname** keyword, you might also need to map the remote peer's hostname to all IP addresses of the remote peer interfaces that could be used during the IKE negotiation. (This is done with the **ip host** command which is not documented in this chapter.) You need to map hostname to IP address except when this mapping is already done in a DNS server.

### Example

The remote peer “RemoteRouter” specifies an ISAKMP identity by address:

```
crypto isakmp identity address
```

The local peer “LocalRouter” also specifies an ISAKMP identity, but by hostname:

```
crypto isakmp identity hostname
```

Now, the pre-shared key must be specified at each peer.

The local peer specifies the pre-shared key and designates the remote peer by its IP address:

```
crypto isakmp key sharedkeystring address 198.168.1.33
```

The remote peer specifies the same pre-shared key and designates the local peer by its hostname:

```
crypto isakmp key sharedkeystring hostname LocalRouter.domain.com
```

The remote peer also maps multiple IP addresses to the same hostname for the local peer. Why? The local peer has two interfaces which both might be used during an IKE negotiation with the local peer. These two interfaces’ IP addresses (10.0.0.1 and 10.0.0.2) are both mapped to the remote peer’s hostname:

```
ip host LocalRouter.domain.com 10.0.0.1 10.0.0.2
```

(This mapping would not have been necessary if LocalRouter.domain.com was already mapped in DNS.)

In this example, a remote peer specifies its ISAKMP identity by address, and the local peer specifies its ISAKMP identity by hostname. Depending on the circumstances in your network, both peers could specify their ISAKMP identity by address, or both by hostname.

### Related Commands

**authentication (IKE policy)**

**crypto isakmp identity**

**ip host**

## crypto isakmp policy

To define an IKE policy, use the **crypto isakmp policy** global configuration command. IKE policies define a set of parameters to be used during the IKE negotiation. To delete an IKE policy, use the **no** form of the command.

```
crypto isakmp policy priority  
no crypto isakmp policy
```

### Syntax Description

*priority* Uniquely identifies the IKE policy and assigns a priority to the policy. Use an integer from 1 to 10,000, with 1 being the highest priority and 10,000 the lowest.

### Default

There is a default policy which is always the lowest priority. This default policy contains default values for the encryption, hash, authentication, Diffie-Hellman group, and lifetime parameters. (The parameter defaults are listed below in the Usage Guidelines section.)

When you create an IKE policy, if you do not specify a value for a particular parameter, the default for that parameter will be used.

### Command Mode

Global configuration

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

Use this command to specify the parameters to be used during an IKE negotiation. (These parameters are used to create the IKE security association [SA].)

When you enter this command, you are put into the ISAKMP policy configuration (config-isakmp) command mode. While in the ISAKMP policy configuration command mode, the following commands are available to specify the parameters in the policy:

- **encryption (IKE policy)**; default = 56-bit DES-CBC
- **hash (IKE policy)**; default = SHA-1
- **authentication (IKE policy)**; default = RSA signatures
- **group (IKE policy)**; default = 768-bit Diffie-Hellman
- **lifetime (IKE policy)**; default = 86400 seconds (one day)

If you do not specify one of these commands for a policy, the default value will be used for that parameter.

To exit the config-isakmp command mode, type **exit**.

You can configure multiple IKE policies on each peer participating in IPSec. When the IKE negotiation begins, it tries to find a common policy configured on both peers, starting with the highest priority policies as specified on the remote peer.

### Example

The following example configures two policies for the peer:

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

The above configuration results in the following policies:

```
MyPeerRouter# show crypto isakmp policy

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

IKE policy 15 is the highest priority, and the default policy is the lowest priority.

### Related Commands

- authentication (IKE policy)**
- encryption (IKE policy)**
- group (IKE policy)**
- hash (IKE policy)**
- lifetime (IKE policy)**
- show crypto isakmp policy**

---

## crypto key generate rsa

To generate RSA key pairs, use the **crypto key generate rsa** global configuration command.

```
crypto key generate rsa [usage-keys]
```

### Syntax Description

**usage-keys** (Optional) Specifies that two RSA special usage key pairs should be generated (i.e. one encryption pair and one signature pair), instead of one general purpose key pair.

### Default

RSA key pairs do not exist. If the **usage-keys** keyword is not used, general purpose keys will be generated.

### Command Mode

Global configuration

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

Use this command to generate RSA key pairs for your Cisco device (such as a router).

RSA keys are generated in pairs—one public RSA key and one private RSA key.

If your router already has RSA keys when you issue this command, you will be warned and prompted to replace the existing keys with new keys.

---

**Note** Before issuing this command, make sure your router has a hostname and IP domain name configured (with the **hostname** and **ip domain-name** commands). You will be unable to complete the **crypto key generate rsa** command without a hostname and IP domain name.

---

This command is never saved in the router configuration; however, the keys generated by this command are saved in the private configuration in NVRAM (which is never displayed to the user or backed up to another device).

There are two mutually-exclusive types of RSA key pairs: special usage keys and general purpose keys. When you generate RSA key pairs, you will be prompted to select whether to generate special usage keys or general purpose keys.

### Special Usage Keys

If you generate special usage keys, two pairs of RSA keys will be generated. One pair will be used with any IKE policy that specifies RSA signatures as the authentication method, and the other pair used with any IKE policy that specifies RSA encrypted nonces as the authentication method.

If you plan to have both types of RSA authentication methods in your IKE policies, you might prefer to generate special usage keys. With special usage keys, each key is not unnecessarily exposed. (Without special usage keys, one key is used for both authentication methods, increasing that key's exposure.)

### General Purpose Keys

If you generate general purpose keys, only one pair of RSA keys will be generated. This pair will be used with IKE policies specifying either RSA signatures or RSA encrypted nonces. Therefore, a general purpose key pair might get used more frequently than a special usage key pair.

### Modulus Length

When you generate RSA keys, you will be prompted to enter a modulus length. A longer modulus could offer stronger security, but takes longer to generate (see Table 1 for sample times) and takes longer to use. Below 512 is normally not recommended. (In certain situations, the shorter modulus may not function properly with IKE, so Cisco recommends using a minimum modulus of 1024.)

**Table 1 Sample Times Required to Generate RSA Keys**

Router	Modulus Length			
	360 bits	512 bits	1024 bits	2048 bits
Cisco 2500	11 seconds	20 seconds	4 minutes, 38 seconds	longer than 1 hour
Cisco 4700	less than 1 second	1 second	4 seconds	50 seconds

### Examples

This example generates special usage RSA keys.

```
myrouter(config)# crypto key generate rsa usage-keys
The name for the keys will be: myrouter.companyx.com
```

```
Choose the size of the key modulus in the range of 360 to 2048 for your Signature Keys.
Choosing a key modulus greater than 512 may take a few minutes.
How many bits in the modulus[512]? <return>
Generating RSA keys.... [OK].
```

```
Choose the size of the key modulus in the range of 360 to 2048 for your Encryption
Keys. Choosing a key modulus greater than 512 may take a few minutes.
How many bits in the modulus[512]? <return>
Generating RSA keys.... [OK].
```

```
myrouter(config)#
```

This example generates general purpose RSA keys. (Note, you cannot generate both special usage and general purpose keys; you can only generate one or the other.)

```
myrouter(config)# crypto key generate rsa
The name for the keys will be: myrouter.companyx.com

Choose the size of the key modulus in the range of 360 to 2048 for your General Purpose
Keys. Choosing a key modulus greater than 512 may take a few minutes.
How many bits in the modulus[512]? <return>
Generating RSA keys.... [OK].

myrouter(config)#
```

### Related Commands

**show crypto key mypubkey rsa**

## crypto key pubkey-chain rsa

To enter public key configuration mode (so you can manually specify other devices' RSA public keys), use the **crypto key pubkey-chain rsa** global configuration command.

**crypto key pubkey-chain rsa**

### Syntax Description

This command has no arguments or keywords.

### Default

This command has no defaults.

### Command Mode

Global configuration. Using this command puts you into public key chain configuration mode.

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

Use this command to enter public key chain configuration mode. Use this command when you need to manually specify other IPsec peers' RSA public keys. You need to specify other peers' keys when you configure RSA encrypted nonces as the authentication method in an IKE policy at your peer router.

### Examples

This example manually specifies the RSA public keys of two other IPsec peers. The remote peers use their IP address as their identity.

```
myrouter(config)# crypto key pubkey-chain rsa
myrouter(config-pubkey-chain)# addressed-key 10.5.5.1
myrouter(config-pubkey-key)# key-string
myrouter(config-pubkey)# 00302017 4A7D385B 1234EF29 335FC973
myrouter(config-pubkey)# 2DD50A37 C4F4B0FD 9DADE748 429618D5
myrouter(config-pubkey)# 18242BA3 2EDFBDD3 4296142A DDF7D3D8
myrouter(config-pubkey)# 08407685 2F2190A0 0B43F1BD 9A8A26DB
myrouter(config-pubkey)# 07953829 791FCDE9 A98420F0 6A82045B
myrouter(config-pubkey)# 90288A26 DBC64468 7789F76E EE21
myrouter(config-pubkey)# quit
myrouter(config-pubkey-key)# exit
myrouter(config-pubkey-chain)# addressed-key 10.1.1.2
myrouter(config-pubkey-key)# key-string
myrouter(config-pubkey)# 0738BC7A 2BC3E9F0 679B00FE 53987BCC
myrouter(config-pubkey)# 01030201 42DD06AF E228D24C 458AD228
myrouter(config-pubkey)# 58BB5DDD F4836401 2A2D7163 219F882E
myrouter(config-pubkey)# 64CE69D4 B583748A 241BED0F 6E7F2F16
myrouter(config-pubkey)# 0DE0986E DF02031F 4B0B0912 F68200C4
myrouter(config-pubkey)# C625C389 0BFF3321 A2598935 C1B1
myrouter(config-pubkey)# quit
myrouter(config-pubkey-key)# exit
myrouter(config-pubkey-chain)# exit
myrouter(config)#
```

Related Commands

**address**

**addressed-key**

**key-string**

**named-key**

**show crypto key pubkey-chain rsa**

## encryption (IKE policy)

To specify the encryption algorithm within an IKE policy, use the **encryption (IKE policy)** ISAKMP policy configuration command. IKE policies define a set of parameters to be used during IKE negotiation. To reset the encryption algorithm to the default value, use the **no** form of the command.

**encryption des**  
**no encryption**

### Syntax Description

**des** Specifies 56-bit DES-CBC as the encryption algorithm.

### Default

The 56-bit DES-CBC encryption algorithm.

### Command Mode

ISAKMP policy configuration (config-isakmp)

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

Use this command to specify the encryption algorithm to be used in an IKE policy.

### Example

This example configures an IKE policy with the 56-bit DES encryption algorithm (and with all other parameters set to the defaults):

```
MyPeerRouter(config)# crypto isakmp policy 15  
MyPeerRouter(config-isakmp)# encryption des  
MyPeerRouter(config-isakmp)# exit  
MyPeerRouter(config)#
```

### Related Commands

**authentication (IKE policy)**  
**crypto isakmp policy**  
**group (IKE policy)**  
**hash (IKE policy)**  
**lifetime (IKE policy)**  
**show crypto isakmp policy**

## group (IKE policy)

To specify the Diffie-Hellman group identifier within an IKE policy, use the **group (IKE policy)** ISAKMP policy configuration command. IKE policies define a set of parameters to be used during IKE negotiation. To reset the Diffie-Hellman group identifier to the default value, use the **no** form of the command.

```
group { 1 | 2 }  
no group
```

### Syntax Description

- |          |  |
|----------|--|
| <b>1</b> | Specifies the 768-bit Diffie-Hellman group.  |
| <b>2</b> | Specifies the 1024-bit Diffie-Hellman group. |

### Default

768-bit Diffie-Hellman (group 1)

### Command Mode

ISAKMP policy configuration (config-isakmp)

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

Use this command to specify the Diffie-Hellman group to be used in an IKE policy.

### Example

This example configures an IKE policy with the 1024-bit Diffie-Hellman group (and with all other parameters set to the defaults):

```
MyPeerRouter (config) # crypto isakmp policy 15  
MyPeerRouter (config-isakmp) # group 2  
MyPeerRouter (config-isakmp) # exit  
MyPeerRouter (config) #
```

### Related Commands

**authentication (IKE policy)**  
**crypto isakmp policy**  
**encryption (IKE policy)**  
**hash (IKE policy)**  
**lifetime (IKE policy)**  
**show crypto isakmp policy**

## hash (IKE policy)

To specify the hash algorithm within an IKE policy, use the **hash (IKE policy)** ISAKMP policy configuration command. IKE policies define a set of parameters to be used during IKE negotiation. To reset the hash algorithm to the default SHA-1 hash algorithm, use the **no** form of the command.

```
hash {sha | md5}  
no hash
```

### Syntax Description

**sha** Specifies SHA-1 (HMAC variant) as the hash algorithm.

**md5** Specifies MD5 (HMAC variant) as the hash algorithm.

### Default

The SHA-1 hash algorithm.

### Command Mode

ISAKMP policy configuration (config-isakmp)

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

Use this command to specify the hash algorithm to be used in an IKE policy.

### Example

This example configures an IKE policy with the MD5 hash algorithm (and with all other parameters set to the defaults):

```
MyPeerRouter (config) # crypto isakmp policy 15  
MyPeerRouter (config-isakmp) # hash md5  
MyPeerRouter (config-isakmp) # exit  
MyPeerRouter (config) #
```

### Related Commands

**authentication (IKE policy)**  
**crypto isakmp policy**  
**encryption (IKE policy)**  
**group (IKE policy)**  
**lifetime (IKE policy)**  
**show crypto isakmp policy**

## key-string

To manually specify a remote peer's RSA public key, use the **key-string** public key configuration command.

```
key-string  
    key-string  
quit
```

### Syntax Description

*key-string*      Enter the key in hexadecimal format. While entering the key data you can press the return key to continue entering data.

### Default

This command has no defaults.

### Command Mode

Public key configuration

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

Use this command to manually specify the RSA public key of an IPSec peer. Before using this command you must identify the remote peer using either the **addressed-key** or **named-key** command.

If possible, to avoid mistakes, you should cut and paste the key data (instead of attempting to type in the data).

### Examples

This example manually specifies the RSA public keys of an IPSec peer.

```
myrouter(config)# crypto key pubkey-chain rsa  
myrouter(config-pubkey-chain)# named-key otherpeer.companyx.com  
myrouter(config-pubkey-key)# address 10.5.5.1  
myrouter(config-pubkey-key)# key-string  
myrouter(config-pubkey)# 005C300D 06092A86 4886F70D 01010105  
myrouter(config-pubkey)# 00034B00 30480241 00C5E23B 55D6AB22  
myrouter(config-pubkey)# 04AEF1BA A54028A6 9ACC01C5 129D99E4  
myrouter(config-pubkey)# 64CAB820 847EDAD9 DF0B4E4C 73A05DD2  
myrouter(config-pubkey)# BD62A8A9 FA603DD2 E2A8A6F8 98F76E28  
myrouter(config-pubkey)# D58AD221 B583D7A4 71020301 0001  
myrouter(config-pubkey)# quit  
myrouter(config-pubkey-key)# exit  
myrouter(config-pubkey-chain)# exit  
myrouter(config)#
```

Related Commands

**addressed-key**

**crypto key pubkey-chain rsa**

**named-key**

**show crypto key pubkey-chain rsa**

## lifetime (IKE policy)

To specify the lifetime of an IKE security association (SA), use the **lifetime (IKE policy)** ISAKMP policy configuration command. To reset the SA lifetime to the default value, use the **no** form of the command.

**lifetime** *seconds*  
**no lifetime**

### Syntax Description

*seconds* Specifies how many seconds each SA should live before expiring. Use an integer from 60 to 86400 seconds.

### Default

86400 seconds (one day)

### Command Mode

ISAKMP policy configuration (config-isakmp)

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

Use this command to specify how long an IKE SA lives before expiring.

When IKE begins negotiations, the first thing it does is agree upon the security parameters for its own session. The agreed-upon parameters are then referenced by an SA at each peer. The SA is retained by each peer until the SA's lifetime expires. Before an SA expires, it can be reused by subsequent IKE negotiations, which can save time when setting up new IPSec SAs. New SAs are negotiated before current SAs expire.

So, to save setup time for IPSec, configure a longer IKE SA lifetime. However, the shorter the lifetime (up to a point), the more secure the IKE negotiation is likely to be.

Note that when your local peer initiates an IKE negotiation between itself and a remote peer, an IKE policy can be selected only if the lifetime of the remote peer's policy is shorter than or equal to the lifetime of the local peer's policy. Then, if the lifetimes are not equal, the shorter lifetime will be selected. To restate this behavior: If the two peer's policies' lifetimes are not the same, the initiating peer's lifetime must be longer and the responding peer's lifetime must be shorter, and the shorter lifetime will be used.

### Example

This example configures an IKE policy with a security association lifetime of 600 seconds (10 minutes) and with all other parameters set to the defaults:

```
MyPeerRouter (config) # crypto isakmp policy 15
MyPeerRouter (config-isakmp) # lifetime 600
MyPeerRouter (config-isakmp) # exit
MyPeerRouter (config) #
```

Related Commands

**authentication (IKE policy)**

**crypto isakmp policy**

**encryption (IKE policy)**

**group (IKE policy)**

**hash (IKE policy)**

**show crypto isakmp policy**

## named-key

To specify which peer's RSA public key you will manually configure, use the **named-key** public key chain configuration command. This command should only be used when the router has a single interface that processes IPSec.

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

### Syntax Description

<i>key-name</i>	Specifies the name of the remote peer's RSA keys. This is always the fully qualified domain name of the remote peer; for example, router.companyx.com.
<b>encryption</b>	(Optional) Indicates that the RSA public key to be specified will be an encryption special usage key.
<b>signature</b>	(Optional) Indicates that the RSA public key to be specified will be a signature special usage key.

### Default

If neither the **encryption** nor **signature** keywords are used, general purpose keys will be specified.

### Command Mode

Public key chain configuration. Using this command puts you into public key configuration mode.

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

Use this command or the **addressed-key** command to specify which IPSec peer's RSA public key you will manually configure next.

Follow this command with the **key-string** command to specify the key.

If you use the **named-key** command you also need to use the **address** public key configuration command to specify the IP address of the peer.

If the IPSec remote peer generated general purpose RSA keys, do not use the **encryption** or **signature** keywords.

If the IPSec remote peer generated special usage keys, you must manually specify both keys: perform this command and the **key-string** command two times and use the **encryption** and **signature** keywords respectively.

## Examples

This example manually specifies the RSA public keys of two IPSec peers. The peer at 10.5.5.1 uses general purpose keys, and the other peer uses special purpose keys.

```
myrouter(config)# crypto key pubkey-chain rsa
myrouter(config-pubkey-chain)# named-key otherpeer.companyx.com
myrouter(config-pubkey-key)# address 10.5.5.1
myrouter(config-pubkey-key)# key-string
myrouter(config-pubkey)# 005C300D 06092A86 4886F70D 01010105
myrouter(config-pubkey)# 00034B00 30480241 00C5E23B 55D6AB22
myrouter(config-pubkey)# 04AEF1BA A54028A6 9ACC01C5 129D99E4
myrouter(config-pubkey)# 64CAB820 847EDAD9 DFOB4E4C 73A05DD2
myrouter(config-pubkey)# BD62A8A9 FA603DD2 E2A8A6F8 98F76E28
myrouter(config-pubkey)# D58AD221 B583D7A4 71020301 0001
myrouter(config-pubkey)# quit
myrouter(config-pubkey-key)# exit
myrouter(config-pubkey-chain)# addressed-key 10.1.1.2 encryption
myrouter(config-pubkey-key)# key-string
myrouter(config-pubkey)# 00302017 4A7D385B 1234EF29 335FC973
myrouter(config-pubkey)# 2DD50A37 C4F4B0FD 9DADE748 429618D5
myrouter(config-pubkey)# 18242BA3 2EDFBDD3 4296142A DDF7D3D8
myrouter(config-pubkey)# 08407685 2F2190A0 0B43F1BD 9A8A26DB
myrouter(config-pubkey)# 07953829 791FCDE9 A98420F0 6A82045B
myrouter(config-pubkey)# 90288A26 DBC64468 7789F76E EE21
myrouter(config-pubkey)# quit
myrouter(config-pubkey-key)# exit
myrouter(config-pubkey-chain)# addressed-key 10.1.1.2 signature
myrouter(config-pubkey-key)# key-string
myrouter(config-pubkey)# 0738BC7A 2BC3E9F0 679B00FE 098533AB
myrouter(config-pubkey)# 01030201 42DD06AF E228D24C 458AD228
myrouter(config-pubkey)# 58BB5DDD F4836401 2A2D7163 219F882E
myrouter(config-pubkey)# 64CE69D4 B583748A 241BED0F 6E7F2F16
myrouter(config-pubkey)# 0DE0986E DF02031F 4B0B0912 F68200C4
myrouter(config-pubkey)# C625C389 0BFF3321 A2598935 C1B1
myrouter(config-pubkey)# quit
myrouter(config-pubkey-key)# exit
myrouter(config-pubkey-chain)# exit
myrouter(config)#
```

## Related Commands

**address**  
**addressed-key**  
**crypto key pubkey-chain rsa**  
**key-string**  
**show crypto key pubkey-chain rsa**

## show crypto isakmp policy

To view the parameters for each IKE policy, use the **show crypto isakmp policy** EXEC command.

```
show crypto isakmp policy
```

### Syntax Description

This command has no arguments or keywords.

### Command Mode

EXEC

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

### Sample Display

The following is sample output from the **show crypto isakmp policy** command, after two IKE policies have been configured (with priorities 15 and 20 respectively):

```
MyPeerRouter# show crypto isakmp policy

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

Note that although the output shows “no volume limit” for the lifetimes, you can currently only configure a time lifetime (such as 86400 seconds); volume limit lifetimes are not configurable.

### Related Commands

- authentication (IKE policy)**
- crypto isakmp policy**
- encryption (IKE policy)**
- group (IKE policy)**
- hash (IKE policy)**
- lifetime (IKE policy)**

## show crypto isakmp sa

To view all current IKE security associations (SAs) at a peer, use the **show crypto isakmp sa** EXEC command.

**show crypto isakmp sa**

### Syntax Description

This command has no arguments or keywords.

### Command Mode

EXEC

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

### Sample Display

The following is sample output from the **show crypto isakmp sa** command, after IKE negotiations have successfully completed between two peers:

```
MyPeerRouter# show crypto isakmp sa
      dst          src          state          conn-id  slot
172.21.114.123 172.21.114.67  QM_IDLE        1         0
155.0.0.2      155.0.0.1     QM_IDLE        8         0
```

Table 2 through Table 4 show the various states that may be displayed in the output of the **show crypto isakmp sa** command. When an ISAKMP SA exists, it will most likely be in its quiescent state (OAK\_QM\_IDLE). For long exchanges, some of the OAK\_MM\_XXX states may be observed.

**Table 2 States in Main Mode Exchange**

State	Explanation
OAK_MM_NO_STATE	The ISAKMP SA has been created but nothing else has happened yet. It is “larval” at this stage—there is no state.
OAK_MM_SA_SETUP	The peers have agreed on parameters for the ISAKMP SA.
OAK_MM_KEY_EXCH	The peers have exchanged Diffie-Hellman public keys and have generated a shared secret. The ISAKMP SA remains unauthenticated.
OAK_MM_KEY_AUTH	The ISAKMP SA has been authenticated. If the router initiated this exchange, this state transitions immediately to OAK_QM_IDLE and a Quick Mode exchange begins.

**Table 3 States in Aggressive Mode Exchange**

State	Explanation
OAK_AG_NO_STATE	The ISAKMP SA has been created but nothing else has happened yet. It is “larval” at this stage—there is no state.
OAK_AG_INIT_EXCH	The peers have done the first exchange in Aggressive Mode but the SA is not authenticated.
OAK_AG_AUTH	The ISAKMP SA has been authenticated. If the router initiated this exchange, this state transitions immediately to OAK_QM_IDLE and a Quick Mode exchange begins.

**Table 4 States in Quick Mode Exchange**

State	Explanation
OAK_QM_IDLE	The ISAKMP SA is idle. It remains authenticated with its peer and may be used for subsequent Quick Mode exchanges. It is in a quiescent state.

#### Related Commands

**crypto isakmp policy**

**lifetime (IKE policy)**

## show crypto key mypubkey rsa

To view your router's RSA public key(s), use the **show crypto key mypubkey rsa** EXEC command.

```
show crypto key mypubkey rsa
```

### Syntax Description

There are no arguments or keywords with this command.

### Command Mode

EXEC

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

This command displays your router's RSA public key(s).

### Sample Display

The following is sample output from the **show crypto key mypubkey rsa** command. Special usage RSA keys were previously generated for this router using the **crypto key generate rsa** command:

```
% Key pair was generated at: 06:07:49 UTC Jan 13 1996
Key name: myrouter.companyx.com
Usage: Signature Key
Key Data:
005C300D 06092A86 4886F70D 01010105 00034B00 30480241 00C5E23B 55D6AB22
04AEF1BA A54028A6 9ACC01C5 129D99E4 64CAB820 847EDAD9 DF0B4E4C 73A05DD2
BD62A8A9 FA603DD2 E2A8A6F8 98F76E28 D58AD221 B583D7A4 71020301 0001

% Key pair was generated at: 06:07:50 UTC Jan 13 1996
Key name: myrouter.companyx.com
Usage: Encryption Key
Key Data:
00302017 4A7D385B 1234EF29 335FC973 2DD50A37 C4F4B0FD 9DADE748 429618D5
18242BA3 2EDFBDD3 4296142A DDF7D3D8 08407685 2F2190A0 0B43F1BD 9A8A26DB
07953829 791FCDE9 A98420F0 6A82045B 90288A26 DBC64468 7789F76E EE21
```

### Related Commands

**crypto key generate rsa**

## show crypto key pubkey-chain rsa

To view peers' RSA public keys stored on your router, use the **show crypto key pubkey-chain rsa EXEC** command.

```
show crypto key pubkey-chain rsa [name key-name | address key-address]
```

### Syntax Description

**name** *key-name* (Optional) Specify the name of a particular public key to view.

**address** *key-address* (Optional) Specify the address of a particular public key to view.

### Default

If no keywords are used, this command displays a list of all RSA public keys stored on your router.

### Command Mode

EXEC

### Usage Guidelines

This command first appeared in Cisco IOS Release 11.3 T.

This command shows RSA public keys stored on your router. This includes peers' RSA public keys manually configured at your router and keys received by your router via other means (such as by a certificate, if CA support is configured).

If a router reboots, any public key derived by certificates will be lost. This is because the router will ask for certificates again, at which time the public key will be derived again.

Use the **name** or **address** keywords to display details about a particular RSA public key stored on your router.

### Sample Display

The following is sample output from the **show crypto key pubkey-chain rsa** command:

```
Codes: M - Manually Configured, C - Extracted from certificate

Code  Usage      IP-address      Name
M     Signature   10.0.0.1       myrouter.companyx.com
M     Encryption  10.0.0.1       myrouter.companyx.com
C     Signature   172.16.0.1     routerA.companyx.com
C     Encryption  172.16.0.1     routerA.companyx.com
C     General     192.168.10.3   routerB.comanyx.com
```

This sample shows manually configured special usage RSA public keys for the peer "somerouter." This sample also shows three keys obtained from peers' certificates: special usage keys for peer "routerA" and a general purpose key for peer "routerB".

Certificate support is used in the above example; if certificate support was not in use, none of the peers' keys would show "C" in the code column, but would all have to be manually configured.

The following is sample output when you issue the command **show crypto key pubkey rsa name somerouter.companyx.com**:

```
Key name: somerouter.companyx.com
Key address: 10.0.0.1
Usage: Signature Key
Source: Manual
Data:
 305C300D 06092A86 4886F70D 01010105 00034B00 30480241 00C5E23B 55D6AB22
 04AEF1BA A54028A6 9ACC01C5 129D99E4 64CAB820 847EDAD9 DF0B4E4C 73A05DD2
 BD62A8A9 FA603DD2 E2A8A6F8 98F76E28 D58AD221 B583D7A4 71020301 0001

Key name: somerouter.companyx.com
Key address: 10.0.0.1
Usage: Encryption Key
Source: Manual
Data:
 00302017 4A7D385B 1234EF29 335FC973 2DD50A37 C4F4B0FD 9DADE748 429618D5
 18242BA3 2EDFBDD3 4296142A DDF7D3D8 08407685 2F2190A0 0B43F1BD 9A8A26DB
 07953829 791FCDE9 A98420F0 6A82045B 90288A26 DBC64468 7789F76E EE21
```

Note that the Source field in the above example indicates “Manual,” meaning that the keys were manually configured on your router, not received in the peer’s certificate.

The following is sample output when you issue the command **show crypto key pubkey rsa address 192.168.10.3**:

```
Key name: routerB.companyx.com
Key address: 192.168.10.3
Usage: General Purpose Key
Source: Certificate
Data:
 0738BC7A 2BC3E9F0 679B00FE 53987BCC 01030201 42DD06AF E228D24C 458AD228
 58BB5DDD F4836401 2A2D7163 219F882E 64CE69D4 B583748A 241BED0F 6E7F2F16
 0DE0986E DF02031F 4B0B0912 F68200C4 C625C389 0BFF3321 A2598935 C1B1
```

Note that the Source field in the above example indicates “Certificate,” meaning that the keys were received by your router by way of the other router’s certificate.

## Debug Commands

The following **debug** command can assist with troubleshooting IKE configuration:

### debug crypto isakmp

Use the **debug crypto isakmp EXEC** command to display messages about IKE events. The **no** form of this command disables debugging output.

```
[no] debug crypto isakmp
```

#### Sample Display

The following shows sample **debug crypto isakmp** output for an IKE peer that initiates an IKE negotiation.

First, IKE negotiates its own security association (SA), checking for a matching IKE policy:

```
MyRouter# debug crypto isakmp
20:26:58: ISAKMP (8): beginning Main Mode exchange
20:26:58: ISAKMP (8): processing SA payload. message ID = 0
20:26:58: ISAKMP (8): Checking ISAKMP transform 1 against priority 10 policy
20:26:58: ISAKMP:      encryption DES-CBC
20:26:58: ISAKMP:      hash SHA
20:26:58: ISAKMP:      default group 1
20:26:58: ISAKMP:      auth pre-share
20:26:58: ISAKMP (8): atts are acceptable. Next payload is 0
```

IKE has found a matching policy. Next, the IKE SA is used by each peer to authenticate the other peer:

```
20:26:58: ISAKMP (8): SA is doing pre-shared key authentication
20:26:59: ISAKMP (8): processing KE payload. message ID = 0
20:26:59: ISAKMP (8): processing NONCE payload. message ID = 0
20:26:59: ISAKMP (8): SKEYID state generated
20:26:59: ISAKMP (8): processing ID payload. message ID = 0
20:26:59: ISAKMP (8): processing HASH payload. message ID = 0
20:26:59: ISAKMP (8): SA has been authenticated
```

Next, IKE negotiates to set up the IPsec SA by searching for a matching transform set:

```
20:26:59: ISAKMP (8): beginning Quick Mode exchange, M-ID of 767162845
20:26:59: ISAKMP (8): processing SA payload. message ID = 767162845
20:26:59: ISAKMP (8): Checking IPsec proposal 1
20:26:59: ISAKMP: transform 1, ESP_DES
20:26:59: ISAKMP:      attributes in transform:
20:26:59: ISAKMP:      encaps is 1
20:26:59: ISAKMP:      SA life type in seconds
20:26:59: ISAKMP:      SA life duration (basic) of 600
20:26:59: ISAKMP:      SA life type in kilobytes
20:26:59: ISAKMP:      SA life duration (VPI) of
0x0 0x46 0x50 0x0
20:26:59: ISAKMP:      authenticator is HMAC-MD5
20:26:59: ISAKMP (8): atts are acceptable.
```

A matching IPSec transform set has been found at the two peers. Now the IPSec SA can be created (one SA is created for each direction):

```
20:26:59: ISAKMP (8): processing NONCE payload. message ID = 767162845
20:26:59: ISAKMP (8): processing ID payload. message ID = 767162845
20:26:59: ISAKMP (8): processing ID payload. message ID = 767162845
20:26:59: ISAKMP (8): Creating IPSec SAs
20:26:59:      inbound SA from 155.0.0.2      to 155.0.0.1      (proxy 155.0.0.2
to 155.0.0.1      )
20:26:59:      has spi 454886490 and conn_id 9 and flags 4
20:26:59:      lifetime of 600 seconds
20:26:59:      lifetime of 4608000 kilobytes
20:26:59:      outbound SA from 155.0.0.1      to 155.0.0.2      (proxy 155.0.0.1
to 155.0.0.2      )
20:26:59:      has spi 75506225 and conn_id 10 and flags 4
20:26:59:      lifetime of 600 seconds
20:26:59:      lifetime of 4608000 kilobytes
```

## Supported MIBs and RFCs

No new or changed MIBs are supported by IKE.

IKE implements/supports the following RFCs and Internet drafts:

- “Internet Security Association and Key Management Protocol (ISAKMP),” draft-ietf-ipsec-isakmp-09.txt, Maughan, D., Schertler, M., Schneider, M., and Turner, J.
- “The Internet Key Exchange (IKE),” draft-ietf-ipsec-isakmp-oakley-08.txt, Harkins, D., and Carrel, D.
- “The IP Security Domain of Interpretation,” draft-ietf-ipsec-ipsec-doi-09.txt, Piper, D.
- RFC 2104: “HMAC: Keyed-Hashing for Message Authentication,” H. Krawczyk, M. Bellare, R. Canetti.

## What to Do Next

After IKE configuration is complete, you can configure IPSec. IPSec configuration is described in the “IPSec Network Security” feature documentation.