

IPsec VPN Router Configuration: The ISAKMP Policy

Michael J. Martin

Previous articles in this series on implementing VPN gateways using Cisco routers discussed the IPsec protocol and basic IPsec VPN connection models. Now we'll learn how to implement ISAKMP policies using IKE to ensure secure VPN configuration.

Setting up an IOS router to utilize IPsec starts with the configuration of the **ISAKMP policy** and the router's **ISAKMP authentication key data**. If the router will be peering with only one other router in a site-to-site topology, the ISAKMP configuration ends there. However, if the router will also be supporting client-to-site peering an additional **IKE Mode Configuration** is needed as well. Before we get to the ISAKMP policy configuration, here are a few safety tips:

- For starters, IOS uses ISAKMP and IKE interchangeably in configuration mode and EXEC mode. Remember that IKE is a protocol that supports ISAKMP -- ISAKMP makes the rules, and IKE plays the game.
- IKE negotiation sends and receives messages using UDP, listening on port 500. This can be a problem if you have a firewall in front of your VPN router or are trying to establish an IPsec client connection through a firewall. Unless you use UDP port 500, traditional IKE will not work.
- IKE does not like Network Address Translation (NAT). IP address-bound pre-shared key authentication will not work when NAT exists between the two IPsec peers. NAT translation modifies source and destination addresses, resulting in mismatches between the key and sending or receiving host. Port Address Translation (PAT), which is used on most stateful-based firewalls, also breaks IPsec connections. However, later versions of IOS address the operational issues between IPsec and NAT/PAT with IPsec NAT transparency and Cisco Tunnel Control Protocol (cTCP). Both of these solutions are invoked during the IKE negotiation phase. NAT transparency adds a NAT discovery phase element to IKE Phase 1 and a NAT traversal option in IKE Phase 2. Operationally, IPsec NAT transparency moves IKE to UDP port 4500 and, if needed, encapsulates IPsec packets into UDP frames.

While NAT transparency addresses some issues, it does not fix them all. IPsec client connections, even with NAT transparency, will not work in environments with strict firewall rules. If UDP port access above 1024 is closed off for the origination of connections, the client cannot establish communication with the gateway. To address this kind of environment, Cisco developed the Tunnel Control Protocol. The cTCP picks up where NAT transparency left off, providing TCP wrapping for IKE and ESP packets. With cTCP, IPsec gateways and clients can be configured to use specific TCP service ports to send IPsec data. This makes it possible to send IPsec traffic through TCP port 80 or 443. That makes it easy to open IPsec client connections in network environments where only limited network services are available.

NAT transparency is enabled by default and is incorporated into the IKE negotiation process of IOS versions that support this enhancement. The Cisco Tunnel Control Protocol needs to be configured and is part of the router's global crypto policy. We will look at configuring cTCP as part of the IKE Mode Configuration.

Configuring an ISAKMP Policy

Well armed with knowledge, let's look at the details of configuring an ISAKMP policy. Start with the most basic step, which is to enable ISAKMP (and IKE) on the router:

Revised August 1, 2008

Page 1 of 7

IPsec VPN Router Configuration: The ISAKMP Policy

Michael J. Martin

```
outlan-rt02(config)#crypto isakmp enable
outlan-rt02(config)#
Oct 13 15:09:27 EST: %CRYPTO-6-ISA_KMP_ON_OFF: ISAKMP is ON
outlan-rt02(config)#
```

Once ISAKMP is enabled, there are five policy parameters that need to be defined to each policy entry. If no policy is defined, a policy using all of the defaults will be used. When creating a policy, if no explicit policy parameter is defined, the default parameter will be used. The policy parameters and default values are:

- **IKE policy encryption** with Data Encryption Standard (DES) as the default,
- **IKE policy hash** with Secure Hash Standard-1 (SHA-1) as the default,
- **IKE key exchange** with Diffie-Hellman Group 1 (768-Bit) as the default,
- **IKE lifetime** with a one-day (86,400 seconds) lifetime as the default, and
- **IKE authentication** with RSA public key as the default.

You may recall that peers need to negotiate a common ISAKMP policy in order to establish an IPsec peer relationship. So depending on the devices you expect to peer with, you may need multiple ISAKMP policies. Each ISAKMP policy is assigned a unique priority number between 1 and 10,000. The policy with priority number 1 is considered the highest priority policy. The policy negotiation starts with the policy numbered closest to 1. It is common practice to start policy numbering at 10, this way if you need to insert policy with a higher priority once the router is in production you have some space to work with.

Another ISAKMP policy priority numbering trick has to do with the ISAKMP policies used for IPsec client support. ISAKMP policies that support IPsec client connections have two policy components: the ISAKMP policy and the IKE Mode Configuration policy. The "client" ISAKMP policy should have the lowest priority if the router is going to support peer relationships between IPsec gateways and IPsec clients. This avoids having a gateway-to-gateway IKE negotiation request for username and password information. (In later versions of IOS, this can be overridden by adding *no-xauth* at the end of a pre-shared key definition). Now, let's move on to creating a policy:

```
outlan-rt02(config)#crypto isakmp policy 10
```

The first parameter we need to define is the encryption algorithm. IOS supports two encryption algorithms: Data Encryption Algorithm (DEA) and Rijndael. Data Encryption Standard (DES) and Triple DES (3DES) standards are based on DEA. DES and 3DES are block ciphers that utilize a 64-bit block encrypted with a 56-bit key. The difference between the two is that 3DES runs three encryption rounds for each data block, while DES runs only one.

The Advanced Encryption Standard (AES) is block cipher based on the Rijndael algorithm. AES uses a 128-bit block size with three key-size options of 126 bits, 192 bits, or 256 bits. DES and 3DES are outdated, but are widely supported in hardware on various Cisco router platforms, either on the router's logic board or through the use of an encryption adapter. The use of 3DES on a router using only a software encryption engine is very processor-intensive and is not scalable beyond a few

IPsec VPN Router Configuration: The ISAKMP Policy

Michael J. Martin

tunnels. Later versions of the IOS support AES; this also holds true for the hardware-based encryption options. AES is more secure and also far more efficient than 3DES. Limited implementations using AES in software can be accomplished. Common practice is to use DES or 3DES, but if the option is available, use AES-256.

```
outlan-rt02(config-isakmp)#encryption 3des
```

The next step is to define the ISAKMP hash algorithm. The IOS supports two hash protocols: Message-Digest algorithm 5 and Secure Hash Algorithm. Message-Digest algorithm 5 (MD5) is a single-pass hash algorithm that generates a 128-bit hash. Secure Hash Algorithm (SHA-1), is operationally similar to MD5, but generates a 160-bit hash. SHA-1 is considered the more current of the two algorithms, but both are really past their prime. It is expected that later IOS version will support SHA-2, which is far more secure, with support for four different hash lengths (224, 256, 384, and 512 bits).

```
outlan-rt02(config-isakmp)#hash sha
```

Next we define what Diffie-Hellman (DH) modulus will be used. The original RFC defined two; DH Group 1 uses a 768-bit modulus and DH Group 2 uses a 1024-bit modulus. The larger the value, the more random the key and the more secure the key is. IOS supports Group 1, Group 2 and Group 5. DH Group 5 uses a 1536-bit modulus. Common practice is to use Group 2, because Group 5 is not supported on all IOS versions and is not supported by the Cisco VPN client. Another thing to keep in mind is that the longer the modulus, the longer time it takes for the CPU to generate the key. On lower end routers, it's a good idea to use a smaller DH modulus.

```
outlan-rt02(config-isakmp)#group 2
```

With the SA algorithm parameters out of the way, we need to define the SA lifetime. There are a few ways of looking at SA lifetime. When an SA expires, a new SA and new SPI are generated or deleted. Shorter SA lifetimes are more secure. Additionally, recovery from router crashes and reloads are faster. If one peer goes down and the other stays up, in some instances new SAs will not be established until the previous one expires. This is particularly true on gateway routers that support hundreds of tunnels. Setting an ISAKMP keepalive addresses this to a large degree, but is easy to forget to set. On the other hand, longer SA lifetimes have less ISAKMP processing overhead. While ISAKMP negotiation is not typically a tremendous processing burden, a short SA lifetime can become so on routers with a large number of peer relationships, depending on the router platform. This configuration example that uses a 5-minute SA lifetime:

```
outlan-rt02(config-isakmp)#lifetime 300
```

In a site-to-site router configuration, the last ISAKMP parameter we need to define is the authentication parameter. IOS supports three authentication RSA signatures, RSA nonces and pre-shared keys. Using RSA signatures for authentication configures the router to use X.509 certificate-based authentication. This is the most secure option, but requires deploying and managing a certificate authority server. Because of that requirement, it is the least utilized option. Further information on RSA signatures can be obtained on Cisco's website.

IPsec VPN Router Configuration: The ISAKMP Policy

Michael J. Martin

RSA nonces can also be used. An RSA nonce is a random number generated by the IKE initiator, encrypted with the recipient's public key. The upside of using RSA nonces is that they are very secure; they also do not require a certificate authority server. The downside is that a peer needs to have the public keys of all of the other peers with which it communicates. That means there is a good degree of labor cost involved in using this method.

The final option is pre-shared keys. Pre-shared keys are used to support both site-to-site and client-to-site VPNs, while the previous two options are used strictly for site-to-site topology configurations. Although pre-shared keys are the least secure method, they are also the most commonly used to authenticate gateway peers. That's because they are quick and easy to set up, and because, with proper security configuration on the gateway, the risk of using a common key between hosts is minimized. In an IPsec client configuration, pre-shared keys are managed using IKE Extended Authentication (Xauth), which is a two-factor authentication method using a user and a group password for authentication. The group password functions essentially as the pre-shared key, and is a common value used by all of the clients and the gateway, while the user password is unique only to the specific client.

We'll look in depth at configuring RSA nonces and pre-shared key configurations for gateways and VPN clients later. Let's choose pre-shared keys for our example; here is the standard configuration:

```
outlan-rt02(config-isakmp)#authentication pre-share
```

We are done with our ISAKMP configuration. Here is what our policy statement looks like:

```
crypto isakmp policy 10
  encr 3des
  hash sha
  lifetime 300
  authentication pre-share
  group 2
!
crypto isakmp keepalive 20 5
crypto isakmp nat keepalive 30
```

Notice that in addition to our ISAKMP policy, there are two keepalive statements. These need to be added as global crypto configuration commands because the default IOS crypto configuration has keepalive services disabled. The ISAKMP keepalive is configured with the global configuration command the `<crypto isakmp keepalive {10-3600 sec}{2-20 sec}>`. With ISAKMP keepalives enabled, the router sends Dead Peer Detection (DPD) messages at intervals between 10 and 3600 seconds. In the event that a response to a DPD is not received, the router then sends the DPD messages at a more aggressive rate -- between 2 and 60 seconds. If the peer router fails to respond after aggressive detection has been activated, the sending router deletes the SA for the failed peer. The `<crypto isakmp nat keepalive {5-3600}>` command is used when the router supports IPsec client connections. In the absence of traffic from the client, a keepalive packet is sent if traffic is not sent before the time interval expires.

IPsec VPN Router Configuration: The ISAKMP Policy

Michael J. Martin

IKE Mode Configuration

Continuing on the topic of IPsec client support, let's move on to the IKE Mode Configuration setup. To support a client-to-site IPsec configuration, the client requires a secure IP identity. The IPsec client's IP address is then used for all IP communication exchanges with the other secured hosts (as defined by the IPsec client policy) protected by the IPsec gateway. This IP address assignment, along with the other entire client configuration parameters (e.g., domain-name, netmask, dns-servers) are defined in the IKE Mode policy. The IKE client configuration is dependent on an ISAKMP policy definition:

```
outlan-rt04#config t
Enter configuration commands, one per line.  End with CNTL/Z.
outlan-rt04(config)#crypto isakmp policy 1000
outlan-rt04(config-isakmp)# encr 3des
outlan-rt04(config-isakmp)# hash md5
outlan-rt04(config-isakmp)# authentication pre-share
outlan-rt04(config-isakmp)# group 2
outlan-rt04(config-isakmp)#exit
outlan-rt04(config)#
```

The IKE Mode Configuration has three parts. The first is the **ISAKMP client group**. This is created using the `<crypto isakmp client configuration group {group name}>` command. This command defines the majority of the client configuration and the group policy information that is used to support the IPsec client connections. The second part is the creation of a client **IP address pool** from which the client configuration group allocates IP address to clients. This is created using the global configuration command `<ip local pool {pool-name} {start-ip} {end-ip}>`. The last part is the split-tunneling **client access policy access control list (ACL)**. This ACL defines the networks that are reachable using the IPsec client IP interface. If this ACL is not defined, the client uses a catch-all access policy that all networks should be reached via the IPsec client IP interface.

The idea behind split tunneling is that an IPsec client host may want to reach some IP nodes via an "unsecured" environment and others via a "secured" environment. The upside of this approach is that with split tunneling enabled, a user can access local LAN devices and the Internet, for example, using the client's LAN interface, without going through the IPsec VPN gateway. The downside is that while the VPN client is active, the host is simultaneously connected to both the unsecured and secured networks. This provides a security risk that can expose secured resources.

The ISAKMP client group needs five required parameters to function properly. Along with base configuration parameters, there are a number of client provisioning parameters that can be defined in the group policy, but these vary to some degree depending on your IOS version. We will look at these additional attributes later, in the client-to-site topology configuration. Here is the basic client group definition using the five parameters:

```
outlan-rt04(config)#crypto isakmp client configuration
group outlan-ras
outlan-rt04(config-isakmp-group)# key outlan-ras
outlan-rt04(config-isakmp-group)# dns 172.30.40.2
outlan-rt04(config-isakmp-group)# domain outlan-ras.net
```

IPsec VPN Router Configuration: The ISAKMP Policy

Michael J. Martin

```
outlan-rt04(config-isakmp-group)# pool outlan-ras
outlan-rt04(config-isakmp-group)# acl outlan-ras-networks
outlan-rt04(config-isakmp-group)#exit
outlan-rt04(config)#
```

```
crypto isakmp client configuration group outlan-ras
  key outlan-ras
  dns 172.30.40.2
  domain outlan-ras.net
  pool outlan-ras
  acl outlan-ras-networks
```

Once the client group definition is completed, we need to create the IP address pool:

```
outlan-rt04(config)#ip local pool outlan-ras 172.30.99.10 172.30.99.100
```

The final step is the client access policy ACL:

```
outlan-rt04(config)#ip access-list extended outlan-ras-networks
outlan-rt04(config-ext-nacl)# permit ip 172.30.40.0 0.0.0.255 172.30.99.0
0.0.0.255
```

Our ISAKMP VPN client support configuration is technically complete. However, if we want to extend VPN client support to hosts connected to other secured networks, we need to configure the Cisco Tunnel Control Protocol. NAT transparency, you should recall, is enabled by default, so enabling cTCP requires the additional global crypto configuration command `<crypto ctcp port {(listener port 1-65535), port, port}>`. The command can be set with or without a port or list of listening ports. If no port is defined, port cTCP listens on port 10000. Here is the cTCP configuration that listens on port HTTP, HTTPS, and the default cTCP service port:

```
outlan-rt04(config)#crypto ctcp port 443 80 10000
```

One thing to keep in mind when configuring cTCP is that if the router is running an HTTP or HTTPS daemon, the IKE service and the HTTP/HTTPS service cannot be running on the same router interface. Below is what the completed ISAKMP client configuration looks like:

```
!
crypto isakmp policy 1000
  encr 3des
  hash md5
  authentication pre-share
  group 2
crypto isakmp keepalive 20 5
crypto isakmp nat keepalive 30
!
crypto isakmp client configuration group outlan-ras
  key outlan-ras
```

IPsec VPN Router Configuration: The ISAKMP Policy

Michael J. Martin

```
dns 172.30.40.2
domain outlan-ras.net
pool outlan-ras
acl outlan-ras-networks
!
crypto ctcp port 443 80 10000
!
ip local pool outlan-ras 172.30.99.10 172.30.99.100
!
ip access-list extended outlan-ras-networks
permit ip 172.30.40.0 0.0.0.255 172.30.99.0 0.0.0.255
```