

IPsec VPN Authentication Generating and Exchanging Pre-shared Keys

Michael J. Martin

Previous articles in this series on implementing VPN gateways using Cisco routers discussed the IPsec protocol, VPN connection models, and implementing ISAKMP policies using IKE to ensure secure VPN configuration. The final step of IKE and ISAKMP configuration is authentication key configuration.

Pre-shared Keys

Let's start with an easy authentication option: pre-shared keys. For pre-shared key authentication to work, a common key is defined on each host. The key definition binds the key to the remote peer's ISAKMP identity. From a security perspective, the best practice is to use a unique key for each peer pair. Pre-shared keys are configured using the global configuration command `<crypto isakmp key {0=unencrypted | 6=encrypted} {key string} {ip address or hostname}>`.

The ISAKMP identity is the interface from which the remote router will send ISAKMP messages to the local peer. If a router supports ISAKMP on multiple interfaces, then hostnames should be used for ISAKMP identity. To configure a router to send its hostname in its ISAKMP peer negotiations, use the global command `<crypto isakmp identity {hostname | address}>`. The default is to use the router's ISAKMP interface IP address. If the hostname option is used to identify its ISAKMP negotiations, the remote peer hosts need to have hostname-to-IP-address mapping, achieved with `<ip host {hostname} {IP address1} {IP Address 2...IP Address 8}>`.

A router can also use DNS for hostname resolution, but local hostname definitions are faster and don't break if there is a DNS server problem. Below is a pre-shared key configuration example between two routers. The local router sends its hostname, and the remote machine sends its IP address. First is the configuration on the local peer router (both the IP address and hostname keys have been defined):

```
!  
hostname outlan-rt01  
!  
!  
crypto isakmp policy 10  
  encr 3des  
  hash md5  
  authentication pre-share  
  group 2  
crypto isakmp key secretkey address 192.168.10.3  
crypto isakmp identity hostname  
!  
interface FastEthernet0/0  
  ip address 172.30.80.17 255.255.255.252  
!  
The configuration on the remote peer router looks like this:  
!  
hostname inlan-rt01  
!  
ip host outlan-rt01 172.30.80.17  
!
```

IPsec VPN Authentication Generating and Exchanging Pre-shared Keys

Michael J. Martin

```
!  
crypto isakmp policy 10  
  encr 3des  
  hash md5  
  authentication pre-share  
  group 2  
crypto isakmp key secretkey address outlan-rt01  
!  
interface FastEthernet1/0  
  ip address 192.168.10.3 255.255.255.252  
!
```

There are two points to note about this example. First, notice that only the local router has an ISAKMP identify definition in the configuration. This is because the IP address is sent as default. Second, as a rule you should not mix ISAKMP identity methods. Either use all IP address or hostname to minimize configuration mistakes. We will look more into using hostnames for ISAKMP identity when we cover topology configurations in future articles.

RSA Nonces

Now let's move on to configuring RSA nonces for authentication. This ISAKMP policy example uses manually exchanged RSA key authentication:

```
crypto isakmp policy 15  
  encr 3des  
  hash md5  
  authentication rsa-encr  
  group 5  
  lifetime 300
```

Let's say we want to apply the router authentication key example above between outlan-rt01 and inlan-rt01. The only change we need to make to the ISAKMP policy is the addition of the authentication parameter (in this example, both hosts used their IP addresses as ISAKMP identity):

```
!  
hostname outlan-rt01  
!  
!  
crypto isakmp policy 10  
  encr 3des  
  hash md5  
  authentication rsa-encr  
  group 2  
!  
interface FastEthernet0/0  
  ip address 172.30.80.17 255.255.255.252
```

IPsec VPN Authentication Generating and Exchanging Pre-shared Keys

Michael J. Martin

```
hostname inlan-rt01
!
!
crypto isakmp policy 10
  encr 3des
  hash md5
  authentication rsa-encr
  group 2
!
interface FastEthernet1/0
  ip address 192.168.10.3 255.255.255.252
```

In order for outlan-rt01 and inlan-rt01 to authenticate, two things must happen. First, each router needs to generate an RSA key pair (public and private). Then, they need to share their public keys with each other. The basic process is the same for both routers, so we will look at the process for outlan-rt01. In order to generate an RSA key pair, the router must have a hostname and IP domain name defined:

```
router(config)# hostname outlan-rt01
outlan-rt01(config)# ip domain-name outlan.net
```

Once the hostname and IP domain name have been configured, the next step is to generate the RSA keys. The IOS supports exportable and non-exportable keys, and both key types can support RSA nonce authentication. However, it's a good idea to generate exportable keys. That way if your environment grows to the point where deploying a certification authority is warranted, you have the ability to use the same keys, making the transition much easier.

Key generation is accomplished using the global configuration command `<crypto key generate rsa label {label string}>` for non-exportable keys and `<crypto key generate rsa exportable label {label string}>` for exportable keys. The label is optional; if no label is specified the key name will be the hostname.domain-name. It is good practice to label the keys, because there may be occasions where you require multiple keys. Labeling makes sure you can find them and prevents you from overwriting by mistake. As for the key's modulus size, IOS supports key sizes between 512 and 2048. A key size of 1024 bits is more than adequate for our purposes.

```
outlan-rt01(config)#crypto key generate rsa exportable label outlan-rt01
The name for the keys will be: outlan-rt01
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]: 1024
% Generating 1024 bit RSA keys, keys will be exportable...[OK]
```

IPsec VPN Authentication Generating and Exchanging Pre-shared Keys

Michael J. Martin

```
outlan-rt01(config)#
```

Public Keys

Now that we've generated RSA keys, we need to get our public key onto inlan-rt01. To start this process, we must view outlan-rt01's public key. From the EXEC shell, we use the command <show crypto key mypubkey rsa {key label}>. If no keys on the router exist, the output will look like this:

```
outlan-rt01#show crypto key mypubkey rsa
outlan-rt01#
```

If one or more keys exist, the output will look like this:

```
outlan-rt01#show crypto key mypubkey rsa outlan-rt01
% Key pair was generated at: 01:03:58 UTC Apr 25 2002
Key name: outlan-rt01
Storage Device: not specified
Usage: General Purpose Key
Key is exportable.
Key Data:
 30819F30 0D06092A 864886F7 0D010101 05000381 8D003081 89028181 009F100B
 36665972 E97AD0B1 BC863579 66B67706 F9B009E9 39FF9C24 59D64250 5B45B2EF
 6F6EBA57 2635235A CCC2DEF7 11653C33 956E92BD 45ED2B4E CDEFB15F 40CCAE7C
 F5E06445 17FEAF2B 75BD936C E75465A0 9F7EEB52 1D387EBB E78B553B 1E56957D
 4E607481 E3CF0482 8C672F6D F772170D 6B599060 BB96D7B2 9DEA29E7 CD020301 0001
outlan-rt01#
```

In some cases, we may not like a key or we end up with two keys when we only want one. To delete a key, in global configuration mode we run <crypto key zeroize rsa {key name}>. Once a key is deleted, it is immediate and there is no recovery. So be sure when you decide to "zeroize" a key. The example below deletes a key named outlan-rt01.outlan.net:

```
outlan-rt01#config t
Enter configuration commands, one per line. End with CNTL/Z.
outlan-rt01(config)#crypto key zeroize rsa outlan-rt01.outlan.net
% Keys to be removed are named named 'outlan-rt01.outlan.net'.
% All router certs issued using these keys will also be removed.
Do you really want to remove these keys? [yes/no]: yes
outlan-rt01(config)#
```

The RSA keys are stored in the private-config file on the nvram file system:

```
outlan-rt01#dir nvram:
Directory of nvram:/

   27  -rw-          751      startup-config
   28  ----           24      private-config
    1  -rw-           0      ifIndex-table
```

IPsec VPN Authentication Generating and Exchanging Pre-shared Keys

Michael J. Martin

2 ----

27

persistent-data

```
29688 bytes total (26813 bytes free)
outlan-rt01#
```

At this point, we know how to create and delete RSA keys, and we have created one for outlan-rt01.

We also need to configure inlan-rt01 to use it. This is accomplished by adding outlan-rt01's RSA public key to inlan-rt01's public keychain.

To view the public key on outlan-rt01, we run:

```
outlan-rt01#sh crypto key mypubkey rsa outlan-rt01
% Key pair was generated at: 01:40:40 UTC Apr 25 2002
Key name: outlan-rt01
Storage Device: not specified
Usage: General Purpose Key
Key is exportable.
Key Data:
 30819F30 0D06092A 864886F7 0D010101 05000381 8D003081 89028181 00E45800
 259A0BB9 D0D1E847 2D9E5045 6EB03D8B 1F03F1F3 C2B93EE6 35888B31 DF2E3384
 71C7C331 11D6308D E41511C5 ADC45E2B 340B458B 63DC16E7 AA9FE214 C35941F1
 E3A5B136 752D963C 94B7892B B8A5B1F5 D13042D9 6754DDDB 40DAEFD6 D50A0AF2
 255499F6 448F7F59 E2823792 79696875 48649C7A 22838305 28622634 A3020301 0001
outlan-rt01#
```

Before we add the key to inlan-rt01's public RSA public keychain, let's first see if the key is already defined on inlan-rt01. (It's always a good idea to check, even if you think it's not there.):

```
inlan-rt01#sh crypto key pubkey-chain rsa
Codes: M - Manually configured, C - Extracted from certificate
```

```
inlan-rt01#
```

The key is not there, so we need to define and import it manually. To do this, we must have a copy of outlan-rt01's key or an EXEC session open on the router so we can copy the key. Then we open an EXEC session on inlan-rt01, enter configuration mode and then enter public keychain configuration mode:

```
inlan-rt01#config t
Enter configuration commands, one per line. End with CNTL/Z.
inlan-rt01(config)#
inlan-rt01(config)#crypto key pubkey-chain rsa
inlan-rt01(config-pubkey-chain)#
```

Once in keychain mode, we need to first associate an IP address with the key. The router will be using its IP address as its ISAKMP identity.

IPsec VPN Authentication Generating and Exchanging Pre-shared Keys

Michael J. Martin

```
inlan-rt02(config-pubkey-chain)#addressed-key 172.30.80.18
```

Once the <addressed-key> command is entered, the router enters configure pubkey key mode. To paste the key into the keychain, we start by entering the <key-string> command. Once the command is entered, the router enters public key configuration mode and prompts us to paste in the key data from the other router. We only need to copy the Key Data field (starting with the key data from outlan-rt01):

```
30819F30 0D06092A 864886F7 0D010101 05000381 8D003081 89028181 00E45800
259A0BB9 D0D1E847 2D9E5045 6EB03D8B 1F03F1F3 C2B93EE6 35888B31 DF2E3384
71C7C331 11D6308D E41511C5 ADC45E2B 340B458B 63DC16E7 AA9FE214 C35941F1
E3A5B136 752D963C 94B7892B B8A5B1F5 D13042D9 6754DDDB 40DAEFD6 D50A0AF2
255499F6 448F7F59 E2823792 79696875 48649C7A 22838305 28622634 A3020301
0001
```

```
inlan-rt02(config-pubkey-key)#key-string
Enter a public key as a hexadecimal number ....
```

```
inlan-rt01(config-pubkey)#$0101 05000381 8D003081 89028181 00E45800
inlan-rt01(config-pubkey)#$B03D8B 1F03F1F3 C2B93EE6 35888B31 DF2E3384
inlan-rt01(config-pubkey)#$C45E2B 340B458B 63DC16E7 AA9FE214 C35941F1
inlan-rt01(config-pubkey)#$A5B1F5 D13042D9 6754DDDB 40DAEFD6 D50A0AF2
inlan-rt01(config-pubkey)#quit
inlan-rt01(config-pubkey-key)#exit
inlan-rt01(config-pubkey-chain)#exit
inlan-rt01(config)#exit
inlan-rt01#
```

To ensure the key has been entered correctly, we can compare the keys. On inlan-rt01, run the <show crypto key pubkey-chain rsa address xxx.xxx.xxx.xxx > command:

```
inlan-rt01#sh crypto key pubkey-chain rsa address 172.30.80.17
Key address:      172.30.80.17
Usage: General Purpose Key
Source: Manually entered
Data:
 30819F30 0D06092A 864886F7 0D010101 05000381 8D003081 89028181 00E45800
 259A0BB9 D0D1E847 2D9E5045 6EB03D8B 1F03F1F3 C2B93EE6 35888B31 DF2E3384
 71C7C331 11D6308D E41511C5 ADC45E2B 340B458B 63DC16E7 AA9FE214 C35941F1
 E3A5B136 752D963C 94B7892B B8A5B1F5 D13042D9 6754DDDB 40DAEFD6 D50A0AF2
 255499F6 448F7F59 E2823792 79696875 48649C7A 22838305 28622634 A3020301 0001

inlan-rt01#
```

Then, on outlan-rt01, run <show crypto key mypubkey rsa [key label]>:

```
outlan-rt01#sh crypto key mypubkey rsa outlan-rt01
Revised August 1, 2008
```

IPsec VPN Authentication Generating and Exchanging Pre-shared Keys

Michael J. Martin

% Key pair was generated at: 01:40:40 UTC Apr 25 2002

Key name: outlan-rt01

Storage Device: not specified

Usage: General Purpose Key

Key is exportable.

Key Data:

```
30819F30 0D06092A 864886F7 0D010101 05000381 8D003081 89028181 00E45800
259A0BB9 D0D1E847 2D9E5045 6EB03D8B 1F03F1F3 C2B93EE6 35888B31 DF2E3384
71C7C331 11D6308D E41511C5 ADC45E2B 340B458B 63DC16E7 AA9FE214 C35941F1
E3A5B136 752D963C 94B7892B B8A5B1F5 D13042D9 6754DDDB 40DAEFD6 D50A0AF2
255499F6 448F7F59 E2823792 79696875 48649C7A 22838305 28622634 A3020301 0001
```

outlan-rt01#

The keys on outlan-rt01 and inlan-rt01 match. Now all we need to do is repeat the whole process so that inlan-rt01's public RSA key is in outlan-rt01's RSA public keychain. I am sure at this point you understand the labor involved with using RSA nonces. It never hurts to repeat to yourself, "Security is not cheap -- or easy."