IPsec Over Cable Sample Configurations and Debugs

Document ID: 12203

Contents

Introduction

Internet Protocol Security (IPsec) is a framework of open standards that ensures secure private communications over IP networks. Based on standards developed by the Internet Engineering Task Force (IETF), IPsec ensures confidentiality, integrity, and authenticity of data communications across a public IP network. IPsec provides a necessary component for a standards−based, flexible solution to deploy a network−wide security policy.

This document provides a configuration example of IPsec between two Cisco cable modems. This configuration creates an encryption tunnel across a cable network between two Cisco uBR9xx Series cable modem routers. All traffic between the two networks is encrypted. But traffic destined for other networks is allowed to pass unencrypted. For small office, home office (SOHO) users, this allows the creation of virtual private networks (VPNs) across a cable network.

Prerequisites

Requirements

There are no specific requirements for this document.

Components Used

The modems must conform to these requirements to configure IPsec on two cable modems:

- Cisco uBR904, uBR905, or uBR924 in Routing Mode
- IPsec 56 Feature Set
- Cisco IOS® Software Release 12.0(5)T or later

In addition, you must have a Cable Modem Termination System (CMTS), which is any Data−over−Cable Service Interface Specifications (DOCSIS)−compliant headend cable router, such as the Cisco uBR7246, Cisco uBR7223, or Cisco uBR7246VXR.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

Conventions

Refer to Cisco Technical Tips Conventions for more information on document conventions.

Background Theory

The example in this document uses a uBR904 cable modem, a uBR924 cable modem, and a uBR7246VXR CMTS. The cable modems run Cisco IOS Software Release 12.1(6), and the CMTS runs Cisco IOS Software Release 12.1(4)EC.

Note: This example is done with manual configuration on the cable modems through the console port. If an automated process is performed through the DOCSIS configuration file (the ios.cfg script is created with the IPsec configuration) then access lists 100 and 101 *cannot* be used. This is because the Cisco implementation of the Simple Network Management Protocol (SNMP) docsDevNmAccess table uses Cisco IOS access lists. It creates one access list per interface. On uBR904, 924, and 905, the first two access lists are generally used (100 and 101). On a cable modem that supports Universal Serial Bus (USB), like the CVA120, three access lists are used (100, 101, and 102).

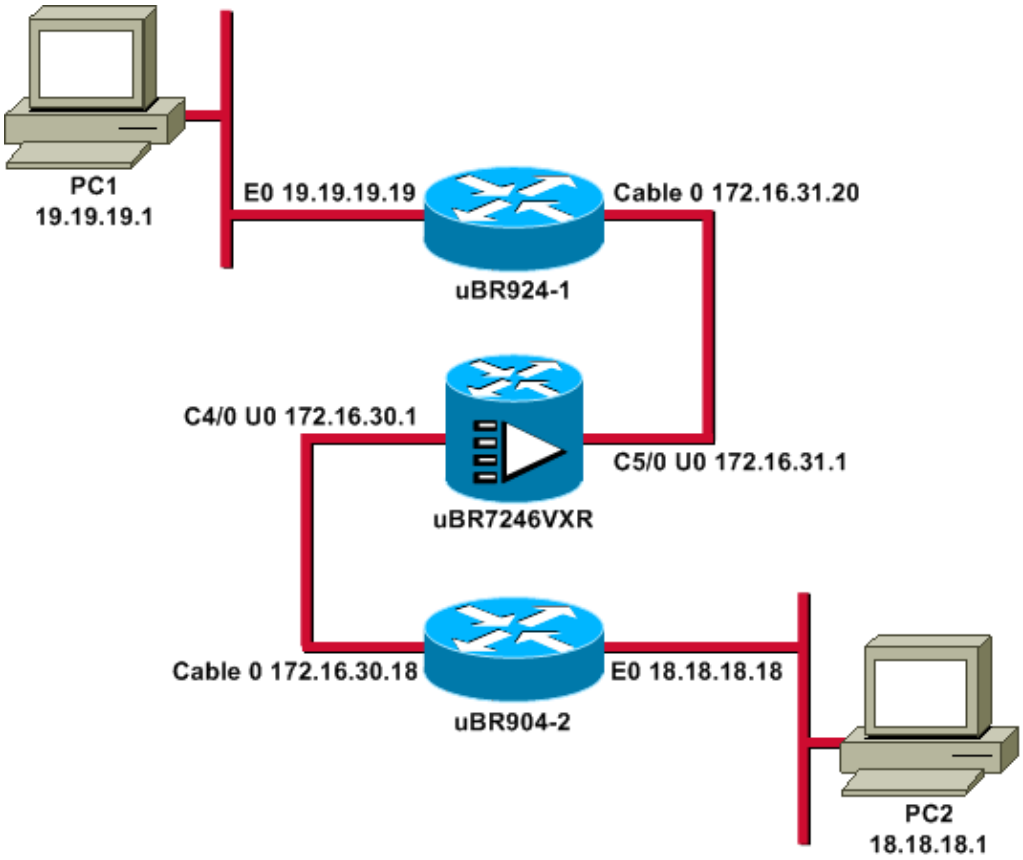
Configure

In this section, you are presented with the information to configure the features described in this document.

Note: Use the Command Lookup Tool (registered customers only) to find additional information about the commands in this document.

Network Diagram

This document uses this network setup:



Note: All of the IP addresses in this diagram have a 24−bit mask.

Configurations

This document uses these configurations:

- uBR924−1
- uBR904−2
- uBR7246VXR

uBR924−1

```
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
!
hostname ubr924-1
!
enable password ww
!
!
!
!
clock timezone - -8
ip subnet-zero
no ip finger
!
ip audit notify log
ip audit po max-events 100
!
!
crypto isakmp policy 10
```

```
!--- Creates an Internet Key Exchange (IKE) policy with the specified priority
!--- number of 10. The range for the priority is 1 to 10000, where 1 is the
!--- highest priority. This command also enters Internet Security Association
!--- and Key Management Protocol (ISAKMP) policy configuration command mode.

 hash md5

!--- Specifies the MD5 (HMAC variant) hash algorithm for packet authentication.

 authentication pre-share

!--- Specifies that the authentication keys are pre-shared, as opposed to
!--- dynamically negotiated using Rivest, Shamir, and Adelman (RSA) public
!--- key signatures.

 group 2

!--- Diffie-Hellman group for key negotiation.

 lifetime 3600

!--- Defines how long, in seconds, each security association should exist before
!--- it expires. Its range is 60 to 86400, and in this case, it is 1 hour.

crypto isakmp key mykey address 18.18.18.18

!--- Specifies the pre-shared key that should be used with the peer at the
!--- specific IP address. The key can be any arbitrary alphanumeric key up to
!--- 128 characters. The key is case-sensitive and must be entered identically
!--- on both routers. In this case, the key is mykey and the peer is the
!--- Ethernet address of uBR904-2
.
!
crypto IPSec transform-set TUNNELSET ah-md5-hmac esp-des

!--- Establishes the transform set to use for IPsec encryption. As many as
!--- three transformations can be specified for a set. Authentication Header
!--- and ESP are in use. Another common transform set used in industry is
!--- esp-des esp-md5-hmac.

!
crypto map MYMAP local-address Ethernet0

!--- Creates the MYMAP crypto map and applies it to the Ethernet0 interface.

crypto map MYMAP 10 ipsec-isakmp

!--- Creates a crypto map numbered 10 and enters crypto map configuration mode.

 set peer 18.18.18.18

!--- Identifies the IP address for the destination peer router. In this case,
!--- the Ethernet interface of the remote cable modem (ubr904-2) is used.

 set transform-set TUNNELSET

!--- Sets the crypto map to use the transform set previously created.

 match address 101

!--- Sets the crypto map to use the access list that specifies the type of
!--- traffic to be encrypted.
!--- Do not use access lists 100, 101, and 102 if the IPsec config is
!--- downloaded through the ios.cfg in the DOCSIS configuration file.
```

```
!
!
!
!
voice-port 0
 input gain -2
 output attenuation 0
!
voice-port 1
 input gain -2
 output attenuation 0
!
!
!
interface Ethernet0
 ip address 19.19.19.19 255.255.255.0
 ip rip send version 2
 ip rip receive version 2
 no ip route-cache
 no ip mroute-cache
!
interface cable-modem0
 ip rip send version 2
 ip rip receive version 2
 no ip route-cache
 no ip mroute-cache
 cable-modem downstream saved channel 525000000 39 1
 cable-modem mac-timer t2 40000
 no cable-modem compliant bridge
 crypto map MYMAP

!--- Applies the previously created crypto map to the cable interface.

!
router rip
 version 2
 network 19.0.0.0
 network 172.16.0.0
!
ip default-gateway 172.16.31.1
ip classless
ip http server
!
access-list 101 permit ip 19.19.19.0 0.0.0.255 18.18.18.0 0.0.0.255

!--- Access list that identifies the traffic to be encrypted. In this case,
!--- it is setting traffic from the local Ethernet network to the remote
!--- Ethernet network.

snmp-server manager
!
line con 0
 transport input none
line vty 0 4
 password ww
 login
!
end
```

The configuration of the other cable modem is very similar, so most of the comments in the previous configuration are omitted.

uBR904-2

```
version 12.1
no service pad
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
!
hostname ubr904-2
!
enable password ww
!
!
!
!
!
clock timezone - -8
ip subnet-zero
no ip finger
!
!
!
crypto isakmp policy 10
 hash md5
 authentication pre-share
 group 2
 lifetime 3600
crypto isakmp key mykey address 19.19.19.19
!
!
crypto IPSec transform-set TUNNELSET ah-md5-hmac ESP-Des
!
crypto map MYMAP local-address Ethernet0
crypto map MYMAP 10 ipsec-isakmp
 set peer 19.19.19.19

!--- Identifies the IP address for the destination peer router. In this case,
!--- the Ethernet interface of the remote cable modem (uBR924-1) is used.

 set transform-set TUNNELSET
 match address 101
!
!
!
!
interface Ethernet0
 ip address 18.18.18.18 255.255.255.0
 ip rip send version 2
 ip rip receive version 2
!
interface cable-modem0
 ip rip send version 2
 ip rip receive version 2
 no keepalive
 cable-modem downstream saved channel 555000000 42 1
 cable-modem Mac-timer t2 40000
 no cable-modem compliant bridge
 crypto map MYMAP
!
router rip
 version 2
 network 18.0.0.0
 network 172.16.0.0
!
ip default-gateway 172.16.30.1
ip classless
no ip http server
!
```

```
access-list 101 permit ip 18.18.18.0 0.0.0.255 19.19.19.0 0.0.0.255
snmp-server manager
!
line con 0
 transport input none
line vty 0 4
 password ww
 login
!
end
```

The CMTS uBR7246VXR also runs Routing Information Protocol (RIP) version 2, so that the routing works. This is the RIP configuration used on the CMTS:

uBR7246VXR
router rip version 2 network 172.16.0.0 no auto-summary

Verify

Use this section to confirm that your configuration works properly.

In order to verify that IPsec works:

- Verify these things:
    1. The Cisco IOS software supports IPsec.
    2. The running configuration is correct.
    3. Interfaces are up.
    4. Routing works.
    5. The access list defined to encrypt traffic is correct.
- Create traffic and look at the Encrypt and Decrypt, to see the amount that is increasing.
- Turn on debugs for crypto.

The Output Interpreter Tool (registered customers only) (OIT) supports certain **show** commands. Use the OIT to view an analysis of **show** command output.

Issue the **show version** command on both cable modems.

```
ubr924-1#show version
Cisco Internetwork Operating System Software
IOS (tm) 920 Software (UBR920-K1O3SV4Y556I-M), Version 12.1(6),
RELEASE SOFTWARE (fc1)
Copyright (c) 1986-2000 by Cisco Systems, Inc.
Compiled Wed 27-Dec-00 16:36 by kellythw
Image text-base: 0x800100A0, data-base: 0x806C1C20

ROM: System Bootstrap, Version 12.0(6r)T3, RELEASE SOFTWARE (fc1)

ubr924-1 uptime is 1 hour, 47 minutes
System returned to ROM by reload at 10:39:05 - Fri Feb 9 2001
System restarted at 10:40:05 - Fri Feb 9 2001
System image file is "flash:ubr920-k1o3sv4y556i-mz.121-6"

cisco uBR920 CM (MPC850) processor (revision 3.e)
with 15872K/1024K bytes of memory.
```

```
Processor board ID FAA0422Q04F
Bridging software.
1 Ethernet/IEEE 802.3 interface(s)
1 Cable Modem network interface(s)
3968K bytes of processor board System flash (Read/Write)
1536K bytes of processor board Boot flash (Read/Write)

Configuration register is 0x2102
```

The uBR924−1 runs Cisco IOS Software Release 12.1(6) with the VALUE SMALL OFFICE/VOICE/FW IPSec 56 Feature Set.

```
ubr904−2#show version
Cisco Internetwork Operating System Software
IOS (TM) 900 Software (UBR900−K1OY556I−M), Version 12.1(6),
RELEASE SOFTWARE (fc1)
Copyright (c) 1986−2000 by cisco Systems, Inc.
Compiled Wed 27−DEC−00 11:06 by kellythw
Image text−base: 0x08004000, database: 0x085714DC

ROM: System Bootstrap, Version 11.2(19980518:195057), RELEASED SOFTWARE
ROM: 900 Software (UBR900−RBOOT−M), Version 11.3(11)NA,
EARLY DEPLOYMENT RELEASE SOFTWARE (fc1)

ubr904−2 uptime is 1 hour, 48 minutes
System returned to ROM by reload at 10:38:44 − Fri Feb 9 2001
System restarted at 10:40:37 − Fri Feb 9 2001
System image file is "flash:ubr900−k1oy556i−mz.121−6"

cisco uBR900 CM (68360) processor (revision D)
with 8192K bytes of memory.
Processor board ID FAA0235Q0ZS
Bridging software.
1 Ethernet/IEEE 802.3 interface(s)
1 Cable Modem network interface(s)
4096K bytes of processor board System flash (Read/Write)
2048K bytes of processor board Boot flash (Read/Write)

Configuration register is 0x2102
```

The uBR904−2 runs Cisco IOS Software Release 12.1(6) with SMALL OFFICE/FW IPSec 56 Feature Set.

```
ubr924−1#show ip interface brief
Interface            IP−Address      OK? Method Status          Protocol
Ethernet0            19.19.19.19     YES NVRAM  up              up
cable−modem0         172.16.31.20    YES unset  up              up

ubr904−2#show ip interface brief
Interface            IP−Address      OK? Method Status          Protocol
Ethernet0            18.18.18.18     YES NVRAM  up              up
cable−modem0         172.16.30.18    YES unset  up              up
```

From the last command, you can see that the Ethernet interfaces are up. The IP addresses of the Ethernet interfaces were manually entered. The cable interfaces are also up and they learned their IP addresses through DHCP. Because these cable addresses are dynamically assigned, they cannot be used as peers in the IPSec configuration.

```
ubr924−1#show ip route
Codes: C − connected, S − static, I − IGRP, R − RIP, M − mobile, B − BGP
       D − EIGRP, EX − EIGRP external, O − OSPF, IA − OSPF inter area
       N1 − OSPF NSSA external type 1, N2 − OSPF NSSA external type 2
       E1 − OSPF external type 1, E2 − OSPF external type 2, E − EGP
       i − IS−IS, L1 − ISIS level−1, L2 − ISIS level−2, ia − ISIS inter area
```

```
       * − candidate default, U − per−user static route, o − ODR
       P − periodic downloaded static route

Gateway of last resort is 172.16.31.1 to network 0.0.0.0

     19.0.0.0/24 is subnetted, 1 subnets
C       19.19.19.0 is directly connected, Ethernet0
R    18.0.0.0/8 [120/2] via 172.16.31.1, 00:00:23, cable−modem0
     172.16.0.0/16 is variably subnetted, 4 subnets, 3 masks
R       172.16.135.0/25 [120/1] via 172.16.31.1, 00:00:23, cable−modem0
R       172.16.29.0/27 [120/1] via 172.16.31.1, 00:00:23, cable−modem0
R       172.16.30.0/24 [120/1] via 172.16.31.1, 00:00:23, cable−modem0
C       172.16.31.0/24 is directly connected, cable−modem0
R    192.168.99.0/24 [120/3] via 172.16.31.1, 00:00:24, cable−modem0
     10.0.0.0/24 is subnetted, 2 subnets
R       10.10.10.0 [120/2] via 172.16.31.1, 00:00:24, cable−modem0
S*   0.0.0.0/0 [1/0] via 172.16.31.1
```

You can see from this output that uBR924−1 is learning about route 18.18.18.0, which is the Ethernet interface of uBR904−2.

```
ubr904−2#show ip route
Codes: C − connected, S − static, I − IGRP, R − RIP, M − mobile, B − BGP
       D − EIGRP, EX − EIGRP external, O − OSPF, IA − OSPF inter area
       N1 − OSPF NSSA external type 1, N2 − OSPF NSSA external type 2
       E1 − OSPF external type 1, E2 − OSPF external type 2, E − EGP
       i − ISIS, L1 − ISIS level−1, L2 − ISIS level−2, IA − ISIS inter area
       * − candidate default, U − per−user static route, o − ODR
       P − periodic downloaded static route

Gateway of last resort is 172.16.30.1 to network 0.0.0.0

R    19.0.0.0/8 [120/2] via 172.16.30.1, 00:00:17, cable−modem0
     18.0.0.0/24 is subnetted, 1 subnets
C       18.18.18.0 is directly connected, Ethernet0
     172.16.0.0/16 is variably subnetted, 4 subnets, 3 masks
R       172.16.135.0/25 [120/1] via 172.16.30.1, 00:00:17, cable−modem0
R       172.16.29.224/27 [120/1] via 172.16.30.1, 00:00:17, cable−modem0
C       172.16.30.0/24 is directly connected, cable−modem0
R       172.16.31.0/24 [120/1] via 172.16.30.1, 00:00:17, cable−modem0
R    192.168.99.0/24 [120/3] via 172.16.30.1, 00:00:18, cable−modem0
     10.0.0.0/24 is subnetted, 1 subnets
R       10.10.10.0 [120/2] via 172.16.30.1, 00:00:18, cable−modem0
S*   0.0.0.0/0 [1/0] via 172.16.30.1
```

From the routing table of uBR904−2, you can see that the network for the Ethernet of uBR924−1 is in the routing table.

Note: There might be cases where you cannot run a routing protocol between the two cable modems. In such cases, you must add static routes on the CMTS to direct traffic for the Ethernet interfaces of the cable modems.

The next thing to check is the certification of the access list; issue the **show access−lists** command on both routers.

```
ubr924−1#show access−lists
Extended IP access list 101
    permit ip 19.19.19.0 0.0.0.255 18.18.18.0 0.0.0.255 (2045 matches)

ubr904−2#show access−lists
Extended IP access list 101
    permit ip 18.18.18.0 0.0.0.255 19.19.19.0 0.0.0.255 (2059 matches)
```

The access list set the IPsec session when the LAN behind uBR924−1 (19.19.19.0) sends IP traffic to the LAN behind uBR904−2 (18.18.18.0), and vice versa. Do *not* use "any" on the access lists, because it creates problems. Refer to Configuring IPsec Network Security for more details.

There is no IPsec traffic. Issue the **show crypto engine connection active** command.

```
ubr924−1#show crypto engine connection active
ID Interface    IP−Address    State    Algorithm           Encrypt  Decrypt
1                              set     HMAC_MD5+DES_56_CB        0        0

ubr904−2#show crypto engine connection active
ID Interface    IP−Address    State    Algorithm           Encrypt  Decrypt
1                              set     HMAC_MD5+DES_56_CB        0        0
```

There are no IPsec connections because no traffic has matched the access lists.

Note: Refer to Important Information on Debug Commands before you use **debug** commands.

The next step is to turn on some crypto debugs to generate interesting traffic.

In this example, these debugs are turned on:

- **debug crypto engine**
- **debug crypto IPsec**
- **debug crypto key−exchange**
- **debug crypto isakmp**

You must first generate some interesting traffic to see the output of the debugs. Issue an extended ping from the Ethernet port of uBR904−2 to the PC on uBR924−1 (IP address 19.19.19.1).

```
ubr904−2#ping ip
Target IP address: 19.19.19.1

!−−− IP address of PC1 behind the Ethernet of uBR924−1.

Repeat count [5]: 100

!−−− Sends 100 pings.

Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface: 18.18.18.18

!−−− IP address of the Ethernet behind uBR904−2.

Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 100, 100−byte ICMP Echos to 19.19.19.1, timeout is 2 seconds:
```

The uBR924−2 shows this debug output:

```
ubr904−2#
01:50:37: IPSec(sa_request): ,
 (key eng. msg.) src= 18.18.18.18, dest= 19.19.19.19,
    src_proxy= 18.18.18.0/255.255.255.0/0/0 (type=4),
```

```
    dest_proxy= 19.19.19.0/255.255.255.0/0/0 (type=4),
    protocol= AH, transform= ah-md5-hmac ,
    lifedur= 3600s and 4608000kb,
    spi= 0x19911A16(428939798), conn_id= 0, keysize= 0, flags= 0x4004
01:50:37: IPSec(sa_request): ,
  (key Eng. msg.) src= 18.18.18.18, dest= 19.19.19.19,
    src_proxy= 18.18.18.0/255.255.255.0/0/0 (type=4),
    dest_proxy= 19.19.19.0/255.255.255.0/0/0 (type=4),
    protocol= ESP, transform= ESP-Des ,
    lifedur= 3600s and 4608000kb,
    spi= 0x7091981(118036865), conn_id= 0, keysize= 0, flags= 0x4004
01:50:37: ISAKMP: received ke message (1/2)
01:50:37: ISAKMP (0:1): sitting IDLE. Starting QM immediately (QM_IDLE)
01:50:37: ISAKMP (0:1): beginning Quick Mode exchange, M-ID of 1108017901
01:50:37: CryptoEngine0: generate hmac context for conn id 1
01:50:37: ISAKMP (1): sending packet to 19.19.19.19 (I) QM_IDLE
01:50:37: ISAKMP (1): received packet from 19.19.19.19 (I) QM_IDLE
01:50:37: CryptoEngine0: generate hmac context for conn id 1
01:50:37: ISAKMP (0:1): processing SA payload. message ID = 1108017901
01:50:37: ISAKMP (0:1): Checking IPSec proposal 1
01:50:37: ISAKMP: transform 1, AH_MD5
01:50:37: ISAKMP:   attributes in transform:
01:50:3.!!!!!!!!!!!!!!!!!!!!!!7: ISAKMP:      encaps is 1
01:50:37: ISAKMP:      SA life type in seconds
01:50:37: ISAKMP:      SA life duration (basic) of 3600
01:50:37: ISAKMP:      SA life type in kilobytes
01:50:37: ISAKMP:      SA life duration (VPI) of  0x0 0x46 0x50 0x0
01:50:37: ISAKMP:      authenticator is HMAC-MD5
01:50:37: validate proposal 0
01:50:37: ISAKMP (0:1): atts are acceptable.
01:50:37: ISAKMP (0:1): Checking IPSec proposal 1
01:50:37: ISAKMP: transform 1, ESP_DES
01:50:37: ISAKMP:   attributes in transform:
01:50:37: ISAKMP:      encaps is 1
01:50:37: ISAKMP:      SA life type in seconds
01:50:37: ISAKMP:      SA life duration (basic) of 3600
01:50:37: ISAKMP:      SA life type in kilobytes
01:50:37: ISAKMP:      SA life duration (VPI) of  0x0 0x46 0x50 0x0
01:50:37: validate proposal 0
01:50:37: ISAKMP (0:1): atts are acceptable.
01:50:37: IPSec(validate_proposal_request): proposal part #1,
  (key Eng. msg.) dest= 19.19.19.19, src= 18.18.18.18,
  dest_proxy= 19.19.1!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 99 percent (99/100), round-trip min/avg/max = 30/40/70 ms
ubr904-2#
```

Notice that the first ping failed. This is because it needs to establish the connection.

The uBR924-1 shows this debug output:

```
ubr924-1#
01:50:24: ISAKMP (1): received packet from 18.18.18.18 (R) QM_IDLE
01:50:24: CryptoEngine0: generate hmac context for conn id 1
01:50:24: ISAKMP (0:1): processing SA payload. Message ID = 1108017901
01:50:24: ISAKMP (0:1): Checking IPSec proposal 1
01:50:24: ISAKMP: transform 1, AH_MD5
01:50:24: ISAKMP:   attributes in transform:
01:50:24: ISAKMP:      encaps is 1
01:50:24: ISAKMP:      SA life type in seconds
01:50:24: ISAKMP:      SA life duration (basic) of 3600
01:50:24: ISAKMP:      SA life type in kilobytes
01:50:24: ISAKMP:      SA life duration (VPI) of  0x0 0x46 0x50 0x0
01:50:24: ISAKMP:      authenticator is HMAC-MD5
01:50:24: validate proposal 0
```

```
01:50:24: ISAKMP (0:1): atts are acceptable.
01:50:24: ISAKMP (0:1): Checking IPSec proposal 1
01:50:24: ISAKMP: transform 1, ESP_DES
01:50:24: ISAKMP:   attributes in transform:
01:50:24: ISAKMP:      encaps is 1
01:50:24: ISAKMP:      SA life type in seconds
01:50:24: ISAKMP:      SA life duration (basic) of 3600
01:50:24: ISAKMP:      SA life type in kilobytes
01:50:24: ISAKMP:      SA life duration (VPI) of  0x0 0x46 0x50 0x0
01:50:24: validate proposal 0
01:50:24: ISAKMP (0:1): atts are acceptable.
01:50:24: IPSec(validate_proposal_request): proposal part #1,
  (key Eng. msg.) dest= 19.19.19.19, src= 18.18.18.18,
    dest_proxy= 19.19.19.0/255.255.255.0/0/0 (type=4),
    src_proxy= 18.18.18.0/255.255.255.0/0/0 (type=4),
    protocol= AH, transform= ah-md5-hmac ,
    lifedur= 0s and 0kb,
    spi= 0x0(0), conn_id= 0, keysize= 0, flags= 0x4
01:50:24: IPSec(validate_proposal_request): proposal part #2,
  (key Eng. msg.) dest= 19.19.19.19, src= 18.18.18.18,
    dest_proxy= 19.19.19.0/255.255.255.0/0/0 (type=4),
    src_proxy= 18.18.18.0/255.255.255.0/0/0 (type=4),
    protocol= ESP, transform= ESP-Des ,
    lifedur= 0s and 0kb,
    spi= 0x0(0), conn_id= 0, keysize= 0, flags= 0x4
01:50:24: validate proposal request 0
01:50:24: ISAKMP (0:1): processing NONCE payload. Message ID = 1108017901
01:50:24: ISAKMP (0:1): processing ID payload. Message ID = 1108017901
01:50:24: ISAKMP (1): ID_IPV4_ADDR_SUBNET src 18.18.18.0/255.255.255.0
   prot 0 Port 0
01:50:24: ISAKMP (0:1): processing ID payload. Message ID = 1108017901
01:50:24: ISAKMP (1): ID_IPV4_ADDR_SUBNET dst 19.19.19.0/255.255.255.0
   prot 0 Port 0
01:50:24: ISAKMP (0:1): asking for 2 spis from IPSec
01:50:24: IPSec(key_engine): got a queue event...
01:50:24: IPSec(spi_response): getting spi 393021796 for SA
        from 18.18.18.18     to 19.19.19.19     for prot 2
01:50:24: IPSec(spi_response): getting spi 45686884 for SA
        from 18.18.18.18     to 19.19.19.19     for prot 3
01:50:24: ISAKMP: received ke message (2/2)
01:50:24: CryptoEngine0: generate hmac context for conn id 1
01:50:24: ISAKMP (1): sending packet to 18.18.18.18 (R) QM_IDLE
01:50:24: ISAKMP (1): received packet from 18.18.18.18 (R) QM_IDLE
01:50:24: CryptoEngine0: generate hmac context for conn id 1
01:50:24: IPSec allocate flow 0
01:50:24: IPSec allocate flow 0
01:50:24: ISAKMP (0:1): Creating IPSec SAs
01:50:24:         inbound SA from 18.18.18.18 to 19.19.19.19
                  (proxy 18.18.18.0 to 19.19.19.0)
01:50:24:         has spi 393021796 and conn_id 2000 and flags 4
01:50:24:         lifetime of 3600 seconds
01:50:24:         lifetime of 4608000 kilobytes
01:50:24:         outbound SA from 19.19.19.19 to 18.18.18.18
                  (proxy 19.19.19.0 to 18.18.18.0)
01:50:24:         has spi 428939798 and conn_id 2001 and flags 4
01:50:24:         lifetime of 3600 seconds
01:50:24:         lifetime of 4608000 kilobytes
01:50:24: ISAKMP (0:1): Creating IPSec SAs
01:50:24:         inbound SA from 18.18.18.18 to 19.19.19.19
                  (proxy 18.18.18.0 to 19.19.19.0)
01:50:24:         has spi 45686884 and conn_id 2002 and flags 4
01:50:24:         lifetime of 3600 seconds
01:50:24:         lifetime of 4608000 kilobytes
01:50:24:         outbound SA from 19.19.19.19 to 18.18.18.18
                  (proxy 19.19.19.0 to 18.18.18.0)
01:50:24:         has spi 118036865 and conn_id 2003 and flags 4
```

```
01:50:25:         lifetime of 3600 seconds
01:50:25:         lifetime of 4608000 kilobytes
01:50:25: ISAKMP (0:1): deleting node 1108017901 error FALSE reason
          "quick mode done (await()"
01:50:25: IPSec(key_engine): got a queue event...
01:50:25: IPSec(initialize_sas): ,
  (key Eng. msg.) dest= 19.19.19.19, src= 18.18.18.18,
    dest_proxy= 19.19.19.0/255.255.255.0/0/0 (type=4),
    src_proxy= 18.18.18.0/255.255.255.0/0/0 (type=4),
    protocol= AH, transform= ah-md5-hmac ,
    lifedur= 3600s and 4608000kb,
    spi= 0x176D0964(393021796), conn_id= 2000, keysize= 0, flags= 0x4
01:50:25: IPSec(initialize_sas): ,
  (key Eng. msg.) src= 19.19.19.19, dest= 18.18.18.18,
    src_proxy= 19.19.19.0/255.255.255.0/0/0 (type=4),
    dest_proxy= 18.18.18.0/255.255.255.0/0/0 (type=4),
    protocol= AH, transform= ah-md5-hmac ,
    lifedur= 3600s and 4608000kb,
    spi= 0x19911A16(428939798), conn_id= 2001, keysize= 0, flags= 0x4
01:50:25: IPSec(initialize_sas): ,
  (key Eng. msg.) dest= 19.19.19.19, src= 18.18.18.18,
    dest_proxy= 19.19.19.0/255.255.255.0/0/0 (type=4),
    src_proxy= 18.18.18.0/255.255.255.0/0/0 (type=4),
    protocol= ESP, transform= ESP-Des ,
    lifedur= 3600s and 4608000kb,
    spi= 0x2B92064(45686884), conn_id= 2002, keysize= 0, flags= 0x4
01:50:25: IPSec(initialize_sas): ,
  (key Eng. msg.) src= 19.19.19.19, dest= 18.18.18.18,
    src_proxy= 19.19.19.0/255.255.255.0/0/0 (type=4),
    dest_proxy= 18.18.18.0/255.255.255.0/0/0 (type=4),
    protocol= ESP, transform= ESP-Des ,
    lifedur= 3600s and 4608000kb,
    spi= 0x7091981(118036865), conn_id= 2003, keysize= 0, flags= 0x4
01:50:25: IPSec(create_sa): sa created,
  (sa) sa_dest= 19.19.19.19, sa_prot= 51,
    sa_spi= 0x176D0964(393021796),
    sa_trans= ah-md5-hmac , sa_conn_id= 2000
01:50:25: IPSec(create_sa): sa created,
  (sa) sa_dest= 18.18.18.18, sa_prot= 51,
    sa_spi= 0x19911A16(428939798),
    sa_trans= ah-md5-hmac , sa_conn_id= 2001
01:50:25: IPSec(create_sa): sa created,
  (sa) sa_dest= 19.19.19.19, sa_prot= 50,
    sa_spi= 0x2B92064(45686884),
    sa_trans= ESP-Des , sa_conn_id= 2002
01:50:25: IPSec(create_sa): sa created,
  (sa) sa_dest= 18.18.18.18, sa_prot= 50,
    sa_spi= 0x7091981(118036865),
    sa_trans= ESP-Des , sa_conn_id= 2003
ubr924-1#
```

Once the IPsec tunnel is created, you can see the connection and the encrypted and decrypted packets.

```
ubr924-1#show crypto engine connection active
ID   Interface      IP-Address     State  Algorithm          Encrypt  Decrypt
   1                                set    HMAC_MD5+DES_56_CB       0        0
2000 cable-modem0   172.16.31.20   set    HMAC_MD5                 0       99
2001 cable-modem0   172.16.31.20   set    HMAC_MD5                99        0
2002 cable-modem0   172.16.31.20   set    DES_56_CBC               0       99
2003 cable-modem0   172.16.31.20   set    DES_56_CBC              99        0
```

The first 200x line shows the 99 packets received. It has to decrypt the packets in order to send them to PC1. The second line shows 99 sent packets. It has to encrypt the packets before it sends them to uBR904-2. The third and fourth lines do the same process, but with ESP-DES transform instead of AH-MD5-HMAC.

Note: If the transform set that is configured on the cable modem is ESP−DES ESP−MD5−HMAC, you only see two autonomous systems (ASs), as opposed to the four shown in the previous **show** command.

```
ubr904−2#show crypto engine connection active
ID   Interface      IP−Address     State  Algorithm          Encrypt  Decrypt
   1                                set    HMAC_MD5+DES_56_CB       0        0
2000 cable−modem0   172.16.30.18   set    HMAC_MD5                 0       99
2001 cable−modem0   172.16.30.18   set    HMAC_MD5                99        0
2002 cable−modem0   172.16.30.18   set    DES_56_CBC               0       99
2003 cable−modem0   172.16.30.18   set    DES_56_CBC              99        0
```

Issue an extended ping to PC2 from uBR924−1 to see if the counters increment for the encrypted and decrypted packets.

```
ubr924−1#ping ip
Target IP address: 18.18.18.1
Repeat count [5]: 50
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface: 19.19.19.19
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 50, 100−byte ICMP Echos to 18.18.18.1, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (50/50), round−trip min/avg/max = 28/30/33 ms

ubr924−1#show crypto engine connection active
ID   Interface      IP−Address     State  Algorithm          Encrypt  Decrypt
   1                                set    HMAC_MD5+DES_56_CB       0        0
2000 cable−modem0   172.16.31.20   set    HMAC_MD5                 0      149
2001 cable−modem0   172.16.31.20   set    HMAC_MD5               149        0
2002 cable−modem0   172.16.31.20   set    DES_56_CBC               0      149
2003 cable−modem0   172.16.31.20   set    DES_56_CBC             149        0

ubr904−2#show crypto engine connection active
ID   Interface      IP−Address     State  Algorithm          Encrypt  Decrypt
   1                                set    HMAC_MD5+DES_56_CB       0        0
2000 cable−modem0   172.16.30.18   set    HMAC_MD5                 0      149
2001 cable−modem0   172.16.30.18   set    HMAC_MD5               149        0
2002 cable−modem0   172.16.30.18   set    DES_56_CBC               0      149
2003 cable−modem0   172.16.30.18   set    DES_56_CBC             149        0
```

Another extended ping can be issued, to see that the counters increment again. This time, send a 500−packet ping from uBR904−2 to the Ethernet interface of uBR924−1 (19.19.19.19).

```
ubr904−2#ping ip
Target IP address: 19.19.19.19
Repeat count [5]: 500
Datagram size [100]: 1000
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface: 18.18.18.18
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
```

```
Type escape sequence to abort.
Sending 500, 1000-byte ICMP Echos to 19.19.19.19, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
01:59:06: IPSec(encapsulate): encaps area too small, moving to new buffer:
idbtype 0, encaps_size 26, header size 60, avail 84!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!
Success rate is 100 percent (500/500), round-trip min/avg/max = 98/135/352 ms

ubr904-2#show crypto engine connection active
ID   Interface       IP-Address      State  Algorithm          Encrypt  Decrypt
   1                                 set    HMAC_MD5+DES_56_CB       0        0
2000 cable-modem0   172.16.30.18     set    HMAC_MD5                 0      649
2001 cable-modem0   172.16.30.18     set    HMAC_MD5               649        0
2002 cable-modem0   172.16.30.18     set    DES_56_CBC               0      649
2003 cable-modem0   172.16.30.18     set    DES_56_CBC             649        0

ubr924-1#show crypto engine connection active
ID   Interface       IP-Address      State  Algorithm          Encrypt  Decrypt
   1                                 set    HMAC_MD5+DES_56_CB       0        0
2000 cable-modem0   172.16.31.20     set    HMAC_MD5                 0      649
2001 cable-modem0   172.16.31.20     set    HMAC_MD5               649        0
2002 cable-modem0   172.16.31.20     set    DES_56_CBC               0      649
2003 cable-modem0   172.16.31.20     set    DES_56_CBC             649        0
```

You can issue the **clear crypto isakmp** and **clear crypto sa** commands to clear the connections. Also, if there is no traffic across the IPsec tunnel during the expiration time, IPsec resets the connection automatically.

Troubleshoot

There is currently no specific information available to troubleshoot this configuration.

Related Information

- **IPsec Network Security Commands**
- **An Introduction to IP Security (IPsec) Encryption − Debug Information**
- **IPsec Configuration Examples**
- **Configuring IPsec Network Security**
- **Configuring the Cisco uBR900 Series Cable Access Routers**
- **Cisco Cable/Broadband Downloads (** **registered customers only)**
- **Broadband Cable Technology Support**
- **Technical Support & Documentation − Cisco Systems**

Updated: May 02, 2006 Document ID: 12203