Troubleshooting the PIX to Pass Data Traffic on an Established IPSec Tunnel

Document ID: 18957

Contents

Introduction

This document addresses and provides a solution to the problem of why a successfully established IPsec tunnel from a Cisco VPN Client to a PIX is unable to pass data.

The inability to pass data on an established IPsec tunnel between a VPN Client and a PIX is frequently encountered when you cannot ping or Telnet from a VPN Client to any hosts on the LAN behind the PIX. In other words, the VPN Client and PIX cannot pass encrypted data between them. This occurs because the PIX has a LAN−to−LAN IPsec tunnel to a router and also a VPN Client. The inability to pass data is the result of a configuration with the same access control list (ACL) for both the nat 0 and the static crypto map for the LAN−to−LAN IPsec peer.

Prerequisites

Requirements

There are no specific requirements for this document.

Components Used

The information in this document is based on these software and hardware versions:

- Cisco Secure PIX Firewall 6.0.1
- Cisco 1720 Router that runs Cisco IOS® Software Release 12.2(6)

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 the Cisco Technical Tips Conventions for more information on document conventions.

Troubleshoot the PIX

Network Diagram
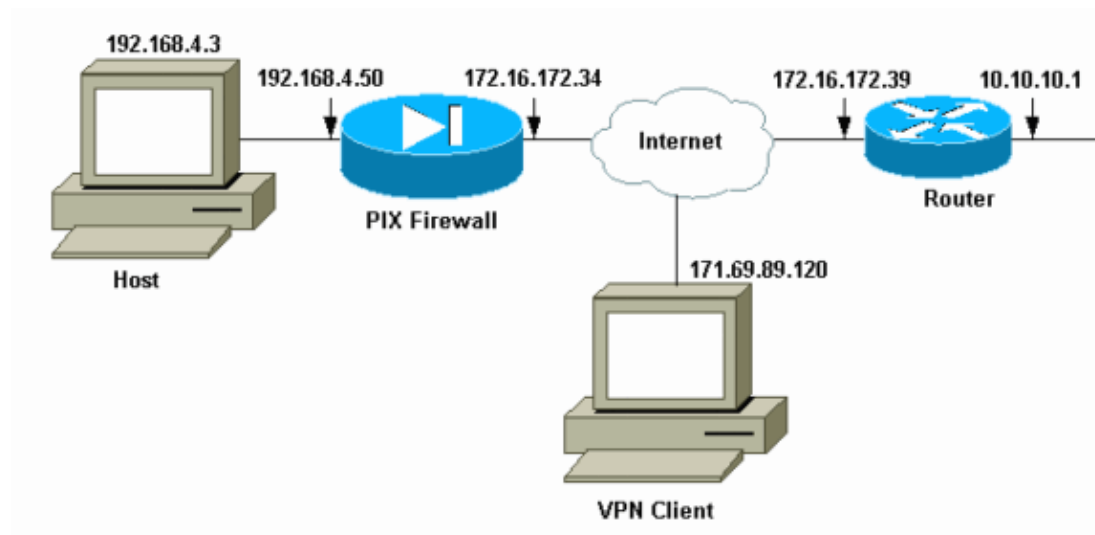
Problematic Sample Configuration

PIX 520

```
pix520-1#write terminal
Building configuration...
: Saved
:
PIX Version 6.0(1)
nameif ethernet0 outside security0
nameif ethernet1 inside security100
enable password 2KFQnbNIdI.2KYOU encrypted
passwd 2KFQnbNIdI.2KYOU encrypted
hostname pix520-1
domain-name vpn.com
fixup protocol ftp 21
fixup protocol http 80
fixup protocol h323 1720
fixup protocol rsh 514
fixup protocol smtp 25
fixup protocol sqlnet 1521
fixup protocol sip 5060
fixup protocol skinny 2000
names

!--- Access-List á40  defines interesting traffic to bypass NAT for VPN
!--- and defines VPN interesting traffic.  This is incorrect.

access-list 140 permit ip 192.168.4.0 255.255.255.0 10.10.10.0 255.255.255.0
access-list 140 permit ip 192.168.4.0 255.255.255.0 10.1.2.0 255.255.255.0
no pager
logging on
logging console debugging
logging monitor debugging
```

```
logging buffered debugging
logging trap debugging
logging history debugging
logging host outside 192.168.2.6
interface ethernet0 auto
interface ethernet1 auto
mtu outside 1500
mtu inside 1500
```

!--- IP addresses on the outside and inside interfaces.

ip address outside 172.16.172.34 255.255.255.240
ip address inside 192.168.4.50 255.255.255.0
```
ip audit info action alarm
ip audit attack action alarm
ip local pool ippool 10.1.2.1-10.1.2.254
no failover
failover timeout 0:00:00
failover poll 15
failover ip address outside 0.0.0.0
failover ip address inside 0.0.0.0
pdm history enable
arp timeout 14400
global (outside) 1 172.16.172.57 netmask 255.255.255.255
```

*!--- The **nat 0** command bypasses NAT for the packets destined over the IPsec tunnel.*

Nat (inside) 0 access-list 140
```
Nat (inside) 1 0.0.0.0 0.0.0.0 0 0
route outside 0.0.0.0 0.0.0.0 172.16.172.33 1
timeout xlate 3:00:00
timeout conn 1:00:00 half-closed 0:10:00 udp 0:02:00 rpc 0:10:00 h323
0:05:00 sip
0:30:00 sip_media 0:02:00
timeout uauth 0:05:00 absolute
aaa-server TACACS+ protocol tacacs+
AAA-server RADIUS protocol radius
AAA-server mytest protocol tacacs+
AAA-server nasir protocol radius
snmp-server host outside 192.168.2.6
no snmp-server location
no snmp-server contact
snmp-server community public
snmp-server enable traps
floodguard enable
```

*!--- The **sysopt** command bypasses conduits or ACLs that check to be applied*
!--- on the inbound VPN packets after decryption.

sysopt connection permit-ipsec
```
no sysopt route dnat
```

*!--- The **crypto ipsec** command defines IPsec encryption and authen algo.*

crypto ipsec transform-set myset esp-des esp-md5-hmac
```
crypto dynamic-map dynmap 10 set transform-set myset
```

*!--- The **crypto map** commands define the IPsec*
!--- Security Assocation (SA) (Phase II SA) parameters.

```
crypto map mymap 5 ipsec-isakmp
```
crypto map mymap 5 match address 140
```
crypto map mymap 5 set peer 172.16.172.39
crypto map mymap 5 set transform-set myset
crypto map mymap 10 ipsec-isakmp dynamic dynmap
crypto map mymap interface outside
```

```
isakmp enable outside

!--- The isakmp key command defines the pre-shared key for the peer address.

isakmp key ******** address 172.16.172.39 netmask 255.255.255.255 no-xauth
no-config-mode
isakmp identity address

!--- The isakmp policy defines the Phase 1 SA parameters.

isakmp policy 10 authentication pre-share
isakmp policy 10 encryption des
isakmp policy 10 hash sha
isakmp policy 10 group 2
isakmp policy 10 lifetime 86400
isakmp policy 20 authentication pre-share
isakmp policy 20 encryption Des
isakmp policy 20 hash sha
isakmp policy 20 group 1
isakmp policy 20 lifetime 86400
vpngroup vpn3000 address-pool ippool
vpngroup vpn3000 idle-time 1800
vpngroup vpn3000 password ********
telnet 192.168.4.0 255.255.255.0 inside
telnet 171.69.89.82 255.255.255.255 inside
telnet timeout 5
ssh 172.0.0.0 255.0.0.0 outside
ssh 171.0.0.0 255.255.255.0 outside
ssh 171.0.0.0 255.0.0.0 outside
ssh timeout 60
terminal width 80
Cryptochecksum:55948dc706cc700e9c10e1d24a8b125c
```

In the problematic configuration the interesting traffic, or the traffic to be encrypted for the LAN−to−LAN tunnel, is defined by ACL 140. The configuration uses the same ACL as the nat 0 ACL.

Understand the General Sequence of Events

When an IP packet arrives at the inside interface of the PIX, Network Address Translation (NAT) is checked. After that, ACLs for the crypto maps are checked.

- **How nat 0 is used.**

  The nat 0 ACL defines what should not be included in NAT. The ACL in the **nat 0** command defines the source and destination address for which the NAT rules on the PIX are disabled. Therefore, an IP packet that has a source and destination address that matches the ACL defined in the **nat 0** command bypasses all the NAT rules on the PIX.

  In order to implement LAN−to−LAN tunnels between a PIX and another VPN device with the help of the private addresses, use the **nat 0** command to bypass NAT. The rules on the PIX firewall prevent the private addresses from being included in NAT while these rules go to the remote LAN over the IPsec tunnel.
- **How the crypto ACL is used.**

  After the NAT inspections, the PIX checks the source and destination of each IP packet that arrives at its inside interface to match the ACLs defined in the static and dynamic crypto maps. If the PIX finds a match with the ACL, the PIX takes any of these steps:

◆ If there is no current IPsec Security Association (SA) already built with the peer IPsec device for the traffic, the PIX initiates the IPsec negotiations. Once the SAs are built, it encrypts the packet and sends it over the IPsec tunnel to the IPsec peer.
◆ If there is already an IPsec SA built with the peer, the PIX encrypts the IP packet and sends the encrypted packet to the peer IPsec device.

• **Dynamic ACL.**

Once a VPN Client connects to the PIX with the help of IPsec, the PIX creates a dynamic ACL that specifies the source and destination address to use in order to define the interesting traffic for this IPsec connection.

Understand the Problematic Series of Events on the PIX

A common configuration mistake is to use the same ACL for nat 0 and the static crypto maps. These sections discuss why this leads to an error and how to rectify the problem.

The PIX configuration shows that the nat 0 ACL 140 bypasses NAT when IP packets go from network 192.168.4.0/24 to networks 10.10.10.0/24 and 10.1.2.0/24 (network address defined in the IP local pool ipool). Additionally, ACL 140 defines the interesting traffic for the static crypto map for peer 172.16.172.39.

When an IP packet comes to the PIX inside interface, the NAT check completes and then the PIX checks the ACLs in the crypto maps. The PIX starts with the crypto map with the lowest instance number. This is because the static crypto map in the previous example has the lowest instance number, the ACL 140 is checked. Next, the dynamic ACL for the dynamic crypto map is checked. In this configuration, the ACL 140 is defined to encrypt traffic that goes from network 192.168.4.0 /24 to networks 10.10.10.0/24 0 and 10.1.2.0 /24. However, for the LAN–to–LAN tunnel, you only want to encrypt traffic between networks 192.168.4.0 /24 and 10.10.10.0 /24. This is how the IPsec peer router defines its crypto ACL.

Understand the Problematic Series of Events on the PIX

When a client establishes an IPsec connection to the PIX, it is assigned an IP address from the IP local pool. In this instance, the client is assigned 10.1.2.1. The PIX also generates a dynamic ACL, as this **show crypto map** command output shows:

```
Crypto Map "mymap" 20 ipsec-isakmp
Peer = 171.69.89.120
access-list dynacl2 permit ip host 172.16.172.34 host 10.1.2.1 (hitcnt=0)
dynamic (created from dynamic map dynmap/10)
Current peer: 171.69.89.120
Security association lifetime: 4608000 kilobytes/28800 seconds
PFS (Y/N): N
Transform sets={ myset, }
Crypto Map "mymap" 30 ipsec-isakmp
Peer = 171.69.89.120
access-list dynacl3 permit ip any host 10.1.2.1 (hitcnt=0)
dynamic (created from dynamic map dynmap/10)
Current peer: 171.69.89.120
Security association lifetime: 4608000 kilobytes/28800 seconds
PFS (Y/N): N
Transform sets={ myset, }
pix520-1(config)#
```

The **show crypto map** command also shows the static crypto map:

```
Crypto Map: "mymap" interfaces: { outside }
Crypto Map "mymap" 5 ipsec-isakmp
Peer = 172.16.172.39
```

```
access-list 140 permit ip 192.168.4.0 255.255.255.0 10.10.10.0255.255.255.0
   (hitcnt=45)
access-list 140 permit ip 192.168.4.0 255.255.255.0 10.1.2.0 255.255.255.0
   (hitcnt=84)
Current peer: 172.16.172.39
Security association lifetime: 4608000 kilobytes/28800 seconds
PFS (Y/N): N
Transform sets={ myset,}
```

Once the IPsec tunnel is established between the client and the PIX, the client initiates a ping to the host 192.168.4.3. When it receives the echo request, the host 192.168.4.3 replies with an echo–reply as this output of the **debug icmp trace** command shows.

```
27: Inbound ICMP echo request (len 32 id 2 seq 7680)
   10.1.2.1 > 192.168.4.3> 192.168.4.3
28: Outbound ICMP echo reply (Len 32 id 2 seq 7680)
   192.168.4.3 >192.168.4.3 > 10.1.2.1
29: Inbound ICMP echo request (Len 32 id 2 seq 7936)
   10.1.2.1 > 192.168.4.3> 192.168.4.3
30: Outbound ICMP echo reply (Len 32 id 2 seq 7936)
   192.168.4.3 >192.168.4.3 > 10.1.2.1
```

However, the echo reply does not reach the VPN Client (host 10.1.2.1), and the ping fails. You can see this with the help of the **show crypto ipsec sa** command on the PIX. This output shows that the PIX decrypts 120 packets that come from the VPN Client, but it does not encrypt any packets or send the encrypted packets to the client. Therefore, the number of packets encapsulated is zero.

```
pix520-1(config)#show crypto ipsec sa
interface: outside
Crypto map tag: mymap, local addr. 172.16.172.34
local ident (addr/mask/prot/port): (0.0.0.0/0.0.0.0/0/0)
remote ident (addr/mask/prot/port): (10.1.2.1/255.255.255.255/0/0)
current_peer: 171.69.89.120
dynamic allocated peer ip: 10.1.2.1
PERMIT, flags={}
#pkts encaps: 0, #pkts encrypt: 0, #pkts digest 0

!--- No packets encrypted and sent to client.

#pkts decaps: 120, #pkts decrypt: 120, #pkts verify 120

!--- 120 packets received from client.

#pkts compressed: 0, #pkts decompressed: 0
#pkts not compressed: 0, #pkts compr. failed: 0, #pkts decompress failed: 0
#send errors 0, #recv errors 0
local crypto endpt.: 172.16.172.34, remote crypto endpt.: 171.69.89.120
path mtu 1500, ipsec overhead 56, media mtu 1500
current outbound spi: 33a45029
inbound esp sas:
spi: 0x279fc5e9(664782313)
transform: ESP-Des esp-md5-hmac ,
in use settings ={Tunnel, }
slot: 0, conn id: 5, crypto map: mymap
sa timing: remaining key lifetime (k/sec): (4607985/27809)
IV size: 8 bytes
replay detection support: Y
inbound ah sas:
inbound pcp sas:
outbound ESP sas:
spi: 0x33a45029(866406441)
transform: ESP-Des esp-md5-hmac ,
in use settings ={Tunnel, }
slot: 0, conn id: 6, crypto map: mymap
```

```
         sa timing: remaining key lifetime (k/sec): (4608000/27809)
         IV size: 8 bytes
         replay detection support: Y
         outbound ah sas:
         outbound PCP sas:
         local ident (addr/mask/prot/port): (192.168.4.0/255.255.255.0/0/0)
         remote ident (addr/mask/prot/port): (10.10.10.0/255.255.255.0/0/0)
         current_peer: 172.16.172.39
         PERMIT, flags={origin_is_acl,}
         #pkts encaps: 10, #pkts encrypt: 10, #pkts digest 10
         #pkts decaps: 23, #pkts decrypt: 23, #pkts verify 23
         #pkts compressed: 0, #pkts decompressed: 0
         #pkts not compressed: 0, #pkts compr. Failed: 0, #pkts decompress failed: 0
         #send errors 0, #recv errors 0
         local crypto endpt.: 172.16.172.34, remote crypto endpt.: 172.16.172.39
         path mtu 1500, ipsec overhead 56, media mtu 1500
         current outbound spi: f264e92c
         inbound ESP sas:
         spi: 0x2772b869(661829737)
         transform: ESP-Des esp-md5-hmac ,
         in use settings ={Tunnel, }
         slot: 0, conn id: 1, crypto map: mymap
         sa timing: remaining key lifetime (k/sec): (4607997/2420)
         IV size: 8 bytes
         replay detection support: Y
         inbound ah sas:
         inbound PCP sas:
         outbound ESP sas:
         spi: 0xf264e92c(4066699564)
         transform: ESP-Des esp-md5-hmac ,
         in use settings ={Tunnel, }
         slot: 0, conn id: 2, crypto map: mymap
         sa timing: remaining key lifetime (k/sec): (4607999/2420)
         IV size: 8 bytes
         replay detection support: Y
         outbound ah sas:
         outbound PCP sas:
```

Note: When the host 192.168.4.3 replies to the echo request, the IP packet comes to the inside interface of the PIX.

```
         38: Outbound ICMP echo reply (Len 32 id 2 seq 8960)
            192.168.4.3 >192.168.4.3 > 10.1.2.1
```

Once the IP packet arrives at the inside interface, the PIX checks the nat 0 ACL 140 and determines that the source and destination addresses of the IP packet matches the ACL. Therefore, this IP packet bypasses all the NAT rules on the PIX. Next, the crypto ACLs are checked. Since the static crypto map has the lowest instance number, its ACL is checked first. Since this example uses ACL 140 for the static crypto map, the PIX checks this ACL. Now, the IP packet has a source address of 192.168.4.3 and a destination of 10.1.2.1. Since this matches the ACL 140, the PIX thinks that this IP packet is intended for the LAN–to–LAN IPsec tunnel with peer 172.16.172.39 (contrary to our objectives). Therefore, it checks the SA database to see if there is already a current SA with peer 172.16.72.39 for this traffic. As the output of the **show crypto ipsec sa** command shows, no SA exists for this traffic. The PIX does not encrypt or send the packet to the VPN Client. Instead, it initiates another IPsec negotiation with peer 172.16.172.39 as this output shows:

```
         crypto_isakmp_process_block: src 172.16.172.39, dest 172.16.172.34
         return status is IKMP_NO_ERR_NO_TRANS02303: sa_request, (key eng. msg.)
         src= 172.16.172.34, dest= 172.16.172.39,
         src_proxy= 192.168.4.0/255.255.255.0/0/0 (type=4),
         dest_proxy= 10.1.2.0/255.255.255.0/0/0 (type=4), protocol= ESP, transform=
         ESP-Des esp-md5-hmac , lifedur= 28800s and 4608000kb,
         spi= 0x0(0), conn_id= 0, keysize= 0, flags= 0x4004
         702303: sa_request, (key Eng. msg.) src= 172.16.172.34, dest=
```

```
      172.16.172.39, src_proxy= 192.168.4.0/255.255.255.0/0/0 (type=4),
      dest_proxy= 10.1.2.0/255.255.255.0/0/0 (type=4), protocol= ESP, transform=
      ESP-Des esp-md5-hmac , lifedur= 28800s and 4608000kb,
      spi= 0x0(0), conn_id= 0, keysize= 0, flags= 0x4004
      ISAKMP (0): sending NOTIFY message 36137 protocol 1
      return status is IKMP_NO_ERR_NO_TRANSIPSEC(key_engine): request timer
      fired: count = 2,
      (identity) local= 172.16.172.34, remote= 172.16.172.39,
      local_proxy= 192.168.4.0/255.255.255.0/0/0 (type=4),
      remote_proxy= 10.1.2.0/255.255.255.0/0/0 (type=4)
```

The IPsec negotiation fails for these reasons:

- The peer 172.16.172.39 defines only networks 10.10.10.0/24 and 192.168.4.0/24 as the interesting traffic in its ACL for the crypto map peer 172.16.172.34.
- The proxy identities do not match during the IPsec negotiation between the two peers.
- If the peer initiates the negotiation and the local configuration specifies perfect forward secrecy (PFS), the peer must perform a PFS exchange or the negotiation fails. If the local configuration does not specify a group, a default of group1 is assumed, and an offer of either group1 or group2 is accepted. If the local configuration specifies group2, that group must be part of the peer's offer or the negotiation fails. If the local configuration does not specify PFS, it accepts any offer of PFS from the peer. The 1024–bit Diffie–Hellman prime modulus group, group2, provides more security than group1, but requires more processing time than group1.

    Note: The **crypto map set pfs** command sets IPsec to ask for PFS when it requests new SAs for this crypto map entry. Use the **no crypto map set pfs** command to specify that IPsec not request PFS. This command is only available for IPsec–ISAKMP crypto–map entries and dynamic crypto map entries. By default, PFS is not requested. With PFS, every time a new SA is negotiated, a new Diffie–Hellman exchange occurs. This requires additional processing time. PFS adds another level of security because if one key is ever cracked by an attacker, only the data sent with that key is compromised. During negotiation, this command causes IPsec to request PFS when it requests new SAs for the crypto map entry. The default (group1) is sent if the **set pfs** statement does not specify a group.

    Note: IKE negotiations with a remote peer can hang when a PIX firewall has numerous tunnels that originate from the PIX firewall and terminate on a single remote peer. This problem occurs when PFS is not enabled, and the local peer requests many simultaneous rekey requests. If this problem occurs, the IKE SA does not recover until it times out or until you manually clear it with the **clear [crypto] isakmp sa** command. PIX firewall units configured with many tunnels to many peers or many clients that share the same tunnel are not affected by this problem. If your configuration is affected, enable PFS with the **crypto map mapname seqnum set pfs** command.

The IP packets on the PIX are ultimately dropped.

Understand the Solution

The correct method to rectify this error is to define two separate ACLs for nat 0 and the static crypto maps. In order to do this, the example defines ACL 190 for the **nat 0** command and uses the modified ACL 140 for the static crypto map, as this output shows.

PIX 520–1
| ```
pix520-1(config)#
pix520-1(config)#write terminal
Building configuration...
: Saved
:
``` |

```
PIX Version 6.0(1)
nameif ethernet0 outside security0
nameif ethernet1 inside security100
enable password 2KFQnbNIdI.2KYOU encrypted
passwd 2KFQnbNIdI.2KYOU encrypted
hostname pix520-1
domain-name vpn.com
fixup protocol ftp 21
fixup protocol http 80
fixup protocol h323 1720
fixup protocol rsh 514
fixup protocol smtp 25
fixup protocol sqlnet 1521
fixup protocol sip 5060
fixup protocol skinny 2000
names
```

!--- Access list 140 defines interesting traffic in order to bypass NAT for VPN.

access-list 140 permit ip 192.168.4.0 255.255.255.0 10.10.10.0255.255.255.0

!--- Defines VPN interesting traffic.

access-list 190 permit ip 192.168.4.0 255.255.255.0 10.10.10.0255.255.255.0
access-list 190 permit ip 192.168.4.0 255.255.255.0 10.1.2.0 255.255.255.0
```
no pager
logging on
logging console debugging
logging monitor debugging
logging buffered debugging
logging trap debugging

logging history debugging
logging host outside 192.168.2.6
interface ethernet0 auto
interface ethernet1 auto
mtu outside 1500
mtu inside 1500
ip address outside 172.16.172.34 255.255.255.240
ip address inside 192.168.4.50 255.255.255.0
ip audit info action alarm
ip audit attack action alarm
ip local pool ippool 10.1.2.1-10.1.2.254
no failover
failover timeout 0:00:00
failover poll 15
failover ip address outside 0.0.0.0
failover ip address inside 0.0.0.0
pdm history enable
arp timeout 14400
global (outside) 1 172.16.172.57 netmask 255.255.255.255
```

*!--- The **nat 0** command bypasses NAT for the packets destined over the IPsec tunnel..*

Nat (inside) 0 access-list 190
```
Nat (inside) 1 0.0.0.0 0.0.0.0 0 0
route outside 0.0.0.0 0.0.0.0 172.16.172.33 1
timeout xlate 3:00:00
timeout conn 1:00:00 half-closed 0:10:00 udp 0:02:00 rpc 0:10:00 h323
0:05:00 sip 0:30:00 sip_media 0:02:00
timeout uauth 0:05:00 absolute
AAA-server TACACS+ protocol tacacs+
AAA-server RADIUS protocol radius
AAA-server mytest protocol tacacs+
AAA-server nasir protocol radius
snmp-server host outside 192.168.2.6
```

```
no snmp-server location
no snmp-server contact
snmp-server community public
snmp-server enable traps
floodguard enable
sysopt connection permit-ipsec
no sysopt route dnat
crypto ipsec transform-set myset ESP-Des esp-md5-hmac
crypto dynamic-map dynmap 10 set transform-set myset

!--- The crypto map commands define the IPsec SA (Phase II SA) parameters.

crypto map mymap 5 ipsec-isakmp
crypto map mymap 5 match address 140
crypto map mymap 5 set peer 172.16.172.39
crypto map mymap 5 set transform-set myset
crypto map mymap 10 ipsec-isakmp dynamic dynmap
crypto map mymap interface outside
isakmp enable outside
isakmp key ******** address 172.16.172.39 netmask 255.255.255.255 no-xauth
no-config-mode
isakmp identity address
isakmp policy 10 authentication pre-share
isakmp policy 10 encryption Des
isakmp policy 10 hash sha
isakmp policy 10 group 2
isakmp policy 10 lifetime 86400
isakmp policy 20 authentication pre-share
isakmp policy 20 encryption Des
isakmp policy 20 hash sha
isakmp policy 20 group 1
isakmp policy 20 lifetime 86400
vpngroup vpn3000 address-pool ippool
vpngroup vpn3000 idle-time 1800
vpngroup vpn3000 password ********
telnet 192.168.4.0 255.255.255.0 inside
telnet 171.69.89.82 255.255.255.255 inside
telnet timeout 5
ssh 172.0.0.0 255.0.0.0 outside
ssh 171.0.0.0 255.255.255.0 outside
ssh 171.0.0.0 255.0.0.0 outside
ssh timeout 60
terminal width 80
Cryptochecksum:e2cb98b30d3899597b3af484fae4f9ae
: end
[OK]
pix520-1(config)# pix520-1(config)#show crypto map
```

After the changes are made and the client establishes an IPsec tunnel with the PIX, issue the **show crypto map** command. This command shows that for the static crypto map, the interesting traffic defined by ACL 140 is only 192.168.4.0/24 and 10.10.10.0/24, which was the original objective. In addition, the dynamic access list shows the interesting traffic defined as the client (10.1.2.1) and the PIX (172.16.172.34).

```
pix520-1(config)#show crypto map
Crypto Map: "mymap" interfaces: { outside }
Crypto Map "mymap" 5 ipsec-isakmp
Peer = 172.16.172.39
access-list 140 permit ip 192.168.4.0 255.255.255.0 10.10.10.0255.255.255.0
   (hitcnt=57)
Current peer: 172.16.172.39
Security association lifetime: 4608000 kilobytes/28800 seconds
PFS (Y/N): N
Transform sets={ myset, }
Crypto Map "mymap" 10 ipsec-isakmp
```

```
Dynamic map template tag: dynmap
Crypto Map "mymap" 20 ipsec-isakmp
Peer = 171.69.89.120
access-list dynacl4 permit ip host 172.16.172.34 host 10.1.2.1 (hitcnt=0)
dynamic (created from dynamic map dynmap/10)
Current peer: 171.69.89.120
Security association lifetime: 4608000 kilobytes/28800 seconds
PFS (Y/N): N
Transform sets={ myset, }
Crypto Map "mymap" 30 ipsec-isakmp
Peer = 171.69.89.120
access-list dynacl5 permit ip any host 10.1.2.1 (hitcnt=13)
dynamic (created from dynamic map dynmap/10)
Current peer: 171.69.89.120
Security association lifetime: 4608000 kilobytes/28800 seconds
PFS (Y/N): N
Transform sets={ myset, }
```

When VPN Client 10.1.2.1 sends a ping to host 192.168.4.3, the echo reply comes to the inside interface of the PIX. The PIX checks the nat 0 ACL 190 and determines that the IP packet matches the ACL. Therefore, the packet bypasses the NAT rules on the PIX. Next, the PIX checks the static crypto map ACL 140 in order to find a match. This time, the source and destination of the IP packet does not match the ACL 140. Therefore, the PIX checks the dynamic ACL and finds a match. The PIX then checks its SA database to see whether or not an IPsec SA is already established with the client. Since the client has already established an IPsec connection with the PIX, an IPsec SA exists. The PIX then encrypts the packets and sends it to the VPN Client. Use the **show crypto ipsec sa** command output from the PIX to see that packets are both encrypted and decrypted. In this case, the PIX encrypted sixteen packets and sent them to the client. The PIX also received encrypted packets from the VPN Client and decrypted sixteen packets.

```
pix520-1(config)#show crypto ipsec sa
interface: outside
Crypto map tag: mymap, local addr. 172.16.172.34
local ident (addr/mask/prot/port): (0.0.0.0/0.0.0.0/0/0)
remote ident (addr/mask/prot/port): (10.1.2.1/255.255.255.255/0/0)
current_peer: 171.69.89.120
dynamic allocated peer ip: 10.1.2.1
PERMIT, flags={}
#pkts encaps: 16, #pkts encrypt: 16,#pkts digest 16
#pkts decaps: 16, #pkts decrypt: 16, #pkts verify 16
#pkts compressed: 0, #pkts decompressed: 0
#pkts not compressed: 0, #pkts compr. Failed: 0, #pkts decompress failed: 0
#send errors 0, #recv errors 0
local crypto endpt.: 172.16.172.34, remote crypto endpt.: 171.69.89.120
path mtu 1500, ipsec overhead 56, media mtu 1500
current outbound spi: 613d083d
inbound ESP sas:
spi: 0x6adf97df(1793038303)
transform: ESP-Des esp-md5-hmac ,
in use settings ={Tunnel, }
slot: 0, conn id: 4, crypto map: mymap
sa timing: remaining key lifetime (k/sec): (4607998/27420)
IV size: 8 bytes
replay detection support: Y
inbound ah sas:
inbound PCP sas:
outbound ESP sas:
spi: 0x613d083d(1631389757)
transform: ESP-Des esp-md5-hmac ,
in use settings ={Tunnel, }
slot: 0, conn id: 3, crypto map: mymap
sa timing: remaining key lifetime (k/sec): (4607999/27420)
IV size: 8 bytes
replay detection support: Y
outbound ah sas:
```

```
outbound PCP sas:
local ident (addr/mask/prot/port): (192.168.4.0/255.255.255.0/0/0)
remote ident (addr/mask/prot/port): (10.10.10.0/255.255.255.0/0/0)
current_peer: 172.16.172.39
PERMIT, flags={origin_is_acl,}
#pkts encaps: 9, #pkts encrypt: 9, #pkts digest 9
#pkts decaps: 9, #pkts decrypt: 9, #pkts verify 9
#pkts compressed: 0, #pkts decompressed: 0
#pkts not compressed: 0, #pkts compr. Failed: 0, #pkts decompress failed: 0
#send errors 1, #recv errors 0
local crypto endpt.: 172.16.172.34, remote crypto endpt.: 172.16.172.39
path mtu 1500, ipsec overhead 56, media mtu 1500
current outbound spi: 58009c01
inbound ESP sas:
spi: 0x2d408709(759203593)
transform: ESP-Des esp-md5-hmac ,
in use settings ={Tunnel, }
slot: 0, conn id: 2, crypto map: mymap
sa timing: remaining key lifetime (k/sec): (4607998/3319)
IV size: 8 bytes
replay detection support: Y
inbound ah sas:
inbound PCP sas: outbound ESP sas:
spi: 0x58009c01(1476434945)
transform: ESP-Des esp-md5-hmac ,
in use settings ={Tunnel, }
slot: 0, conn id: 1, crypto map: mymap
sa timing: remaining key lifetime (k/sec): (4607999/3319)
IV size: 8 bytes
replay detection support: Y
outbound ah sas:
outbound PCP sas:
pix520-1(config)# sh cr isa sa
Total : 2
Embryonic : 0
dst src state pending created
172.16.172.39 172.16.172.34 QM_IDLE 0 1
172.16.172.34 171.69.89.120 QM_IDLE 0 2
pix520-1(config)# sh cr ipsec sa
```

Router Configuration and show Command Output

Cisco 1720–1

```
1720-1#show run
Building configuration...
Current configuration : 1592 bytes
!
! Last configuration change at 21:08:49 PST Mon Jan 7 2002
! NVRAM config last updated at 18:18:17 PST Mon Jan 7 2002
!
version 12.2
no parser cache
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
!
hostname 1720-1
!
no logging buffered
enable secret 5 $1$6jAs$tNxI1a/2DYFAtPLyCDXjo/
enable password ww
!
username cisco password 0 cisco
memory-size iomem 15
```

```
clock timezone PST -8
ip subnet-zero
no ip domain-lookup
ip domain-name cisco.com
!
ip ssh time-out 120
ip ssh authentication-retries 3
!
!
```

*!--- The **crypto isakmp policy** command defines the Phase 1 SA parameters.*

```
crypto isakmp policy 15
authentication pre-share
crypto isakmp key cisco123 address 172.16.172.34
!
!
```

*!--- The **crypto ipsec transform-set** command defines IPsec encryption*
!--- and authentication algorithims.

```
crypto ipsec transform-set myset ESP-Des esp-md5-hmac
!
!
```

*!--- The **crypto map** command defines the IPsec SA (Phase II SA) parameters..*

```
crypto map vpn 10 ipsec-isakmp
set peer 172.16.172.34
set transform-set myset
```
match address 150
```
!
!
!
!
!
interface FastEthernet0
ip address 172.16.172.39 255.255.255.240
speed auto
```

!--- The crypto map applied to the outbound interface.

crypto map vpn
```
interface Ethernet0
ip address 10.10.10.1 255.255.255.240
speed auto
no ip route-cache
no ip mroute-cache
!
!
ip classless
ip route 0.0.0.0 0.0.0.0 172.16.172.33
no ip http server
ip pim bidir-enable
!
```

!--- Access-list defines interesting VPN traffic.

access-list 150 permit ip 10.10.10.0 0.0.0.255 192.168.4.0 0.0.0.255
```
!
line con 0
line aux 0
line vty 0 4
exec-timeout 0 0
password cisco
no login
```

```
line vty 5 15
login
!
no scheduler allocate
end
1720-1#
```

```
1720-1#show crypto isa sa
DST src state conn-id slot
172.16.172.39 172.16.172.34 QM_IDLE 132 0
1720-1#show crypto ipsec sa
interface: FastEthernet0
Crypto map tag: vpn, local addr. 172.16.172.39
local ident (addr/mask/prot/port): (10.10.10.0/255.255.255.0/0/0)
remote ident (addr/mask/prot/port): (192.168.4.0/255.255.255.0/0/0)
current_peer: 172.16.172.34
PERMIT, flags={origin_is_acl,}
#pkts encaps: 9 #pkts encrypt: 9 #pkts digest 9
#pkts decaps: 9, #pkts decrypt: 9, #pkts verify 9
#pkts compressed: 0, #pkts decompressed: 0
#pkts not compressed: 0, #pkts compr. Failed: 0, #pkts decompress failed: 0
#send errors 7, #recv errors 0
local crypto endpt.: 172.16.172.39, remote crypto endpt.: 172.16.172.34
path mtu 1500, media mtu 1500
current outbound spi: 2D408709
inbound ESP sas:
spi: 0x58009C01(1476434945)
transform: ESP-Des esp-md5-hmac ,
in use settings ={Tunnel, }

!--- IPsec SA 200 as seen in the show crypto engine connection active command.

slot: 0, conn id: 200, flow_id: 1, crypto map: vpn
sa timing: remaining key lifetime (k/sec): (4607998/3144)
IV size: 8 bytes
replay detection support: Y
inbound ah sas:
inbound PCP sas:
outbound ESP sas:
spi: 0x2D408709(759203593)
transform: ESP-Des esp-md5-hmac ,
in use settings ={Tunnel, }

!--- IPsec SA 201 as seen in the show crypto engine connection active command.

slot: 0, conn id: 201, flow_id: 2, crypto map: vpn
sa timing: remaining key lifetime (k/sec): (4607998/3144)
IV size: 8 bytes
replay detection support: Y
outbound ah sas:
outbound PCP sas:
1720-1#

1720-1#show crypto map
Interfaces using crypto map mymap:
Crypto Map "vpn" 10 ipsec-isakmp
Peer = 172.16.172.34
Extended IP access list 150
access-list 150 permit ip 10.10.10.0 0.0.0.255 192.168.4.0 0.0.0.255
Current peer: 172.16.172.34
Security association lifetime: 4608000 kilobytes/3600 seconds
PFS (Y/N): N
Transform sets={ myset, }
Interfaces using crypto map vpn: FastEthernet0
```

Related Information

- **Cisco PIX Firewall Software**
- **Cisco Secure PIX Firewall Command References**
- **Security Product Field Notices (including PIX)**
- **Requests for Comments (RFCs)**
- **IPsec Negotiation/IKE Protocols**
- **Technical Support & Documentation – Cisco Systems**