IOS IKEv1 and IKEv2 Packet Exchange Processes for Profiles with Multiple Certificates

TAC **Document ID: 117633**

Contributed by Michal Garcarz, Cisco TAC Engineer.
Apr 23, 2014

Contents

Introduction

This document describes the Internet Key Exchange Version 1 (IKEv1) and Internet Key Exchange Version 2 (IKEv2) packet exchange processes when certificate authentication is used and the possible problems that might occur.

Here is a list of subjects that are described in this document:

- The certificate selection criteria for the Internet Key Exchange (IKE) initiator and IKE responder
- The IKE profile match criteria when multiple IKE profiles are matched (for overlap and non−overlap scenarios)
- The default settings and behavior when no trust−points are used under the IKE profiles
- The differences between the IKEv1 and the IKEv2 in regards to profile and certificate selection criteria

Note: For details about how to troubleshoot a specific problem, refer to the correct section. Also, a short summary is provided at the end of this document.

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- Cisco IOS® VPN configuration
- IKEv1 and IKEv2 protocols (packet exchange)

Components Used

The information in this document is based on Cisco IOS Version15.3T.

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.

Background Information

The problems that are described in this document arise when multiple trust−points and multiple IKE profiles are used.

The initial examples that are used in this document have an IKEv1 LAN−to−LAN tunnel with two trust−points on each router. At first, it might seem that the configuration is correct. However, the VPN tunnel can be initiated only from one side of the connection because of the way that the ***ca trust−point*** command is used for the Internet Security Association and Key Management Protocol (ISAKMP) profile behavior and for the order of the enrolled certificates in the local store.

A different behavior is configured with the ***ca trust−point*** command for the ISAKMP profile when the router is the ISAKMP initiator. A problem might occur because the ISAKMP initiator is aware of the ISAKMP profile from the start, so the ***ca trust−point*** command that is configured for the profile can influence the payload for the certificate request in Main Mode Packet 3 (MM3). However, when the router is the ISAKMP responder, it binds the inbound traffic to a specific ISAKMP profile after it receives the Main Mode Packet 5 (MM5), which includes the IKE ID that is necessary in order to create the bind. This is why it is not possible to apply any ***ca trust−point*** command for the Main Mode Packet 4 (MM4) packet because the profile is not determined before the MM5.

The order of the certificate requestpayload in the MM3 and MM4 and the impact on the whole negotiation process is explained in this document, as well as the reason that it only allows the connection to be established from one side of the VPN tunnel.

Here is a summary of the IKEv1 initiator and responder behaviors:

	IKEv1 Initiator	***IKEv1 Responder***

Send Request	Sends specific requests only for the trust−points that are configured under the profile	Sends requests for all of the available trust−points
Validate Request	Validates against specific trust−points that are configured under the profile	Validates against specific trust−points that are configured under the profile

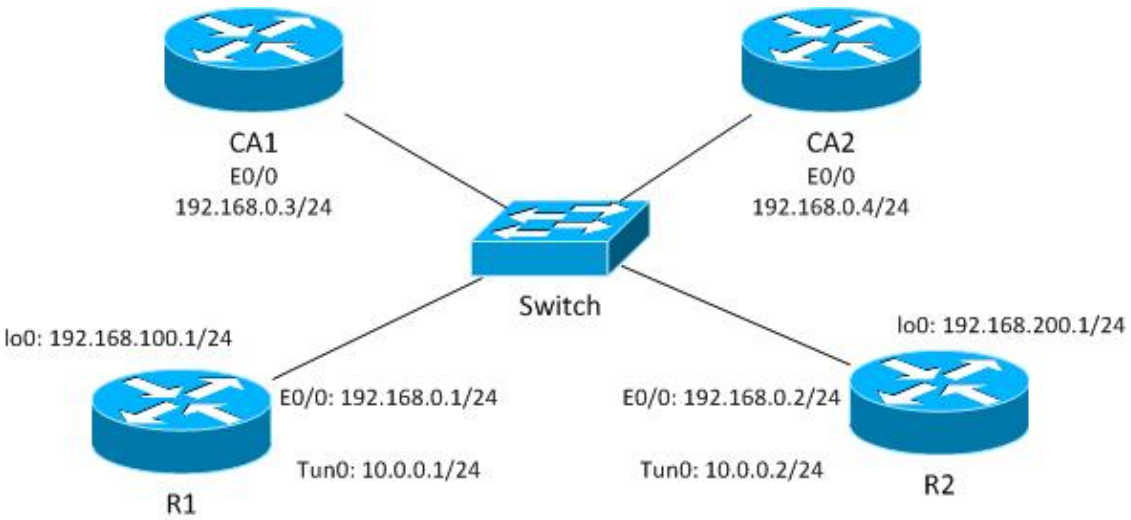
Cisco recommends that you do not use the ***ca trust−point*** command for the ISAKMP responders that have multiple ISAKMP profiles and use globally−configured trust−points. For ISAKMP initiators with multiple ISAKMP profiles, Cisco recommends that you narrow the certificate selection process with the ***ca trust−point*** command in each profile.

The IKEv2 protocol has the same issues as the IKEv1 protocol, but the different behavior of the ***pki trustpoint*** command helps prevent the occurrence of the problems. This is because the ***pki trustpoint*** command is mandatory for the IKEv2 initiator, while the ***ca trust−point*** command is optional for the IKEv1 initiator. Under some circumstances (multiple trust−points under one profile), the previously described problems might occur. For this reason, Cisco recommends that you use symmetric trust−point configurations for both sides of the connection (the same trust−points configured under both of the IKEv2 profiles).

Topology

This is a generic topology that is used for all of the examples in this document.

Note: Router 1 (R1) and Router 2 (R2) use Virtual Tunnel Interfaces (VTIs) in order to access the loopbacks. These VTIs are protected by IPSec.



For this IKEv1 example, each router has two trust−points for each Certificate Authority (CA), and the certificates for each of the trust−points are enrolled.

When R1 is the ISAKMP initiator, the tunnel negotiates correctly and traffic is protected. This is expected behavior. When R2 is the ISAKMP initiator, the Phase1 negotiation fails.

Note: For the IKEv2 examples in this document, the topology and addressing is the same as that shown the IKEv1 example.

Packet Exchange Process

This section describes the IKEv1 and the IKEv2 configuration variations that are used for the packet exchange process, and the possible problems that might arise.

IKEv1 with Multiple Certificates

Here is the R1 network and VPN configuration for IKEv1 with multiple certificates:

```
crypto isakmp policy 10
 encr 3des
 hash md5
 group 2

crypto isakmp profile prof1
   self-identity fqdn
   ca trust-point IOSCA1
   match identity host R2.cisco.com
!
crypto ipsec transform-set TS esp-aes esp-sha256-hmac
 mode tunnel
!
crypto ipsec profile prof1
 set transform-set TS
 set isakmp-profile prof1
!
interface Loopback0
 description Simulate LAN
 ip address 192.168.100.1 255.255.255.0
!
interface Tunnel1
 ip address 10.0.0.1 255.255.255.0
 tunnel source Ethernet0/0
 tunnel destination 192.168.0.2
 tunnel protection ipsec profile prof1
!
interface Ethernet0/0
 ip address 192.168.0.1 255.255.255.0

ip route 192.168.200.0 255.255.255.0 10.0.0.2
```

Here is the R2 network and VPN configuration for IKEv1 with multiple certificates:

```
crypto isakmp policy 10
 encr 3des
 hash md5
 group 2

crypto isakmp profile prof1
   self-identity fqdn
   match identity host R1.cisco.com
!
crypto ipsec transform-set TS esp-aes esp-sha256-hmac
 mode tunnel
!
crypto ipsec profile prof1
 set transform-set TS
 set isakmp-profile prof1
!
interface Loopback0
 ip address 192.168.200.1 255.255.255.0
!
interface Tunnel1
 ip address 10.0.0.2 255.255.255.0
 tunnel source Ethernet0/0
 tunnel destination 192.168.0.1
 tunnel protection ipsec profile prof1
!
interface Ethernet0/0
 ip address 192.168.0.2 255.255.255.0

ip route 192.168.100.0 255.255.255.0 10.0.0.1
```

In this example, R1 has two trust-points: one uses ***IOSCA1*** and the second uses ***IOSCA2***:

```
crypto pki trustpoint IOSCA1
 enrollment url http://192.168.0.3:80
 serial-number
 fqdn R1.cisco.com
 ip-address 192.168.0.1
 subject-name CN=R1,OU=IT,O=cisco,O=com
 revocation-check crl
!
crypto pki trustpoint IOSCA2
 enrollment url http://192.168.0.4:80
 serial-number
 fqdn R1.cisco.com
 ip-address 192.168.0.1
 subject-name CN=R1,OU=IT,O=cisco,O=com
 revocation-check crl
```

In this example, R2 also has two trust-points: one uses ***IOSCA1*** and the second uses ***IOSCA2***:

```
crypto pki trustpoint IOSCA1
 enrollment url http://192.168.0.3:80
 serial-number
```

```
 fqdn R2.cisco.com
 ip-address 192.168.0.2
 subject-name CN=R2,OU=IT,O=cisco,O=com
 revocation-check crl
!
crypto pki trustpoint IOSCA2
 enrollment url http://192.168.0.4:80
 serial-number
 fqdn R2.cisco.com
 ip-address 192.168.0.2
 subject-name CN=R2,OU=IT,O=cisco,O=com
 revocation-check crl
```

It is important to note the single difference in these configurations: the R1 ISAKMP profile uses the ***ca trust-point*** command for the ***IOSCA1*** trust-point, which indicates that R1 trusts only the certificates that are validated by that specific trust-point. In contrast, R2 trusts all of the certificates that are validated by all of the globally-defined trust-points.

R1 as the IKEv1 Initiator

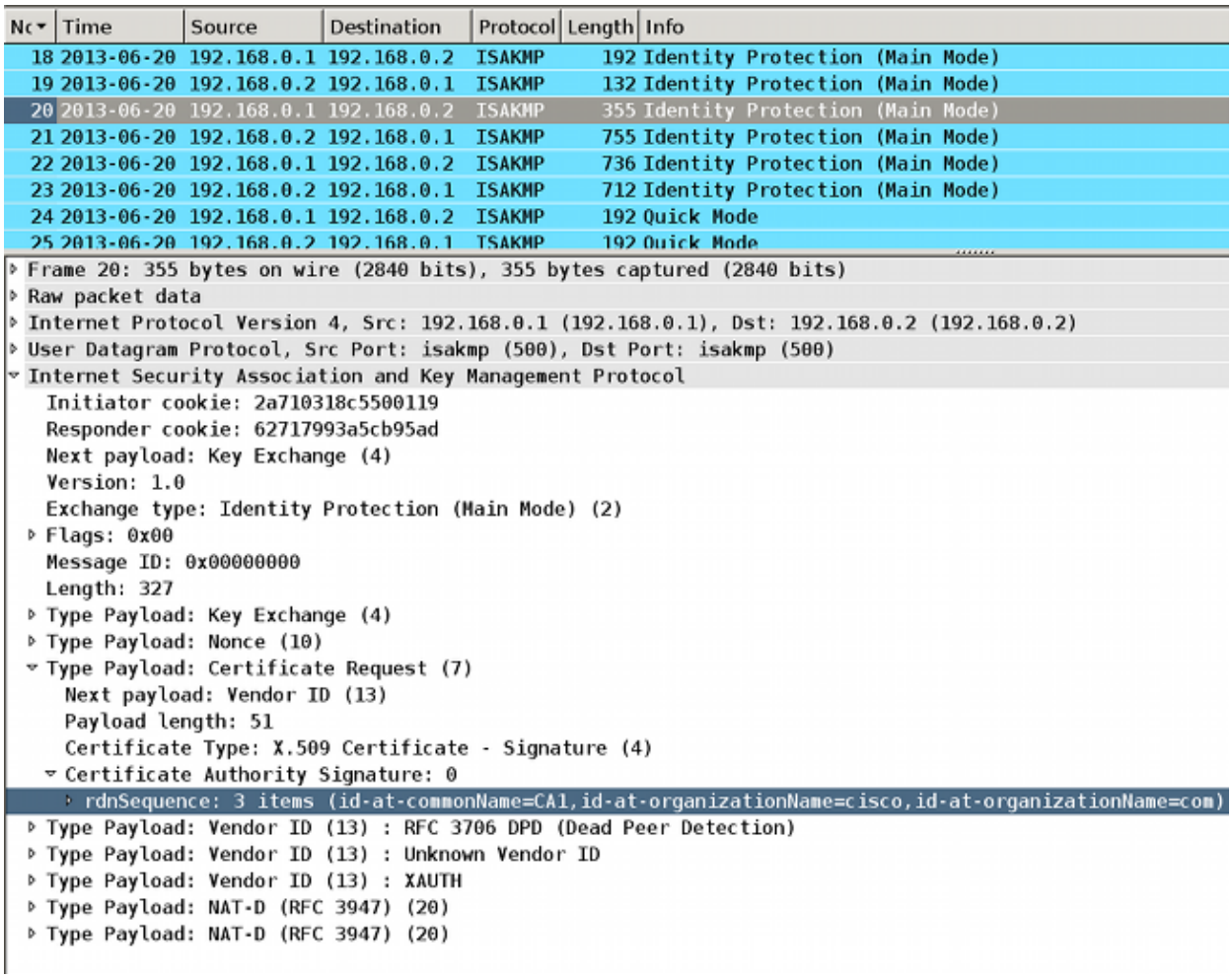
Here are the debugs commands for both R1 and R2:

- ***R1# debug crypto isakmp***
- ***R1# debug crypto ipsec***
- ***R1# debug crypto pki validation***

Here, R1 initiates the tunnel and sends the certificate requestin the MM3:

```
*Jun 20 13:00:37.609: ISAKMP:(0): SA request profile is prof1
*Jun 20 13:00:37.610: ISAKMP (0): constructing CERT_REQ for issuer
 cn=CA1,o=cisco,o=com
*Jun 20 13:00:37.610: ISAKMP:(0): sending packet to 192.168.0.2
 my_port 500 peer_port 500 (I) MM_SA_SETUP
*Jun 20 13:00:37.610: ISAKMP:(0):Old State = IKE_I_MM2  New State = IKE_I_MM3
```

It is important to notice that the packet contains only one certificate request, which is only for the ***IOSCA1*** trust-point. This is expected behavior with the current configuration of the ISAKMP profile (***CN=CA1, O=cisco, O=com***). No other certificate requests are sent, which you can verify with the Embedded Packet Capture feature:



When R2 receives the packet, it begins to process the certificate request, which creates a match that determines the trust−point and the associated certificate that is used for authentication in the MM5. The process order is the same as the certificate request payload in the ISAKMP packet. This means that the first match is used. In this scenario, there is only one match since R1 is configured with a specific trust−point and sends only one certificate request that is associated with the trust−point.

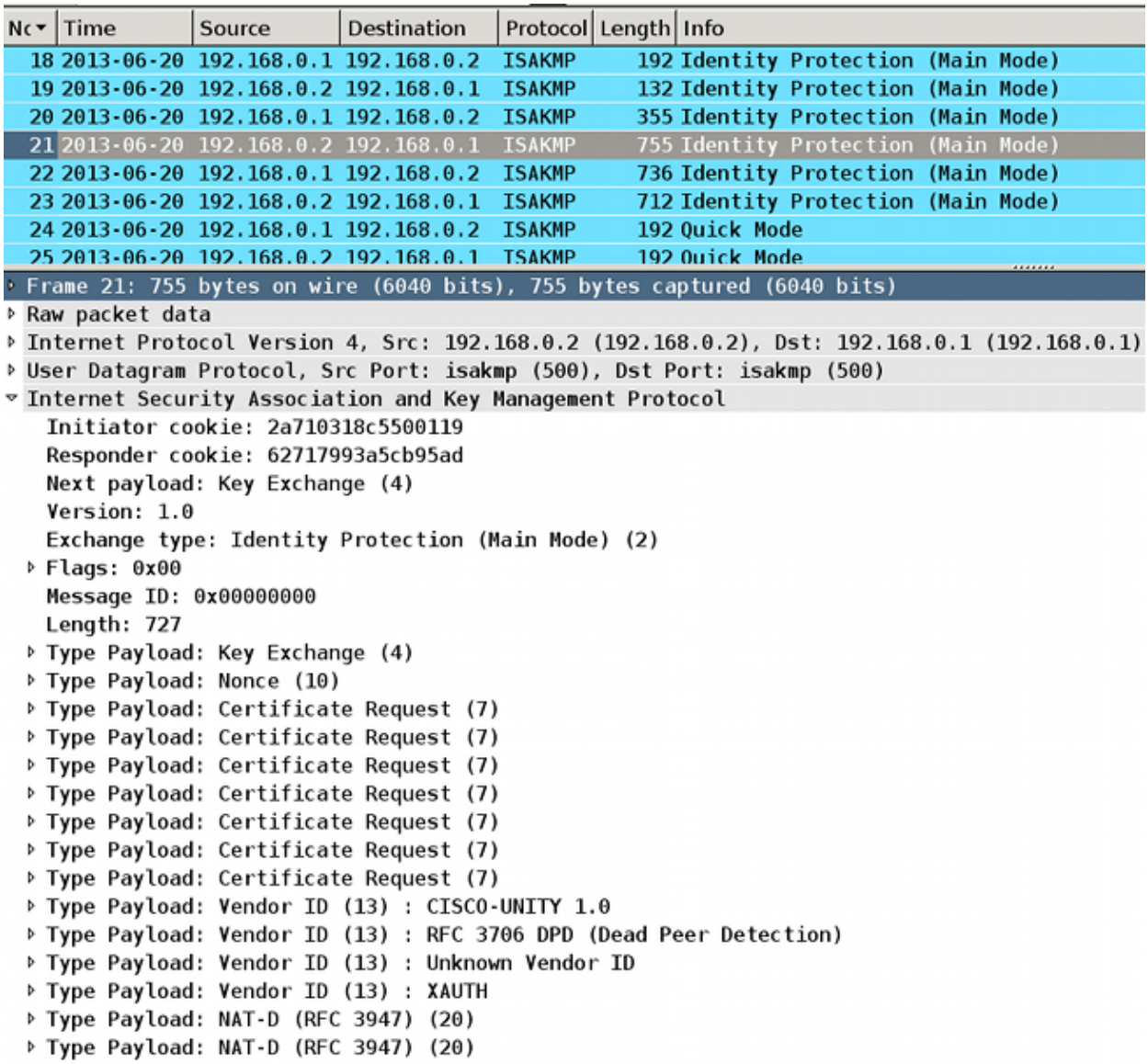
```
*Jun 20 13:00:37.617: ISAKMP:(1010): peer wants a CT_X509_SIGNATURE cert
*Jun 20 13:00:37.617: ISAKMP:(1010): peer wants cert issued
 by cn=CA1,o=cisco,o=com
*Jun 20 13:00:37.617:  Choosing trustpoint IOSCA1 as issuer
```

Afterwards, R2 prepares the MM4. This is the packet that contains the certificate request for all of the trusted trust−points. Since R2 is the ISAKMP responder, all of the globally−defined trust−points are trusted (the ***ca trust−point*** configuration is not checked). Two of the trust−points are defined manually (***IOSCA1*** and ***IOSCA2***), and the rest are predefined.

```
*Jun 20 13:00:37.617: ISAKMP (1010): constructing CERT_REQ
 for issuer cn=CA1,o=cisco,o=com
*Jun 20 13:00:37.617: ISAKMP (1010): constructing CERT_REQ
```

```
 for issuer cn=CA2,o=cisco,o=com
*Jun 20 13:00:37.617: ISAKMP (1010): constructing CERT_REQ
 for issuer ou=Class 3 Public Primary Certification Authority,
 o=VeriSign, Inc.,c=US
*Jun 20 13:00:37.617: ISAKMP (1010): constructing CERT_REQ
 for issuer cn=Cisco SSCA2,o=Cisco Systems
*Jun 20 13:00:37.617: ISAKMP (1010): constructing CERT_REQ
 for issuer cn=Cisco Manufacturing CA,o=Cisco Systems
*Jun 20 13:00:37.617: ISAKMP (1010): constructing CERT_REQ
 for issuer cn=Cisco Root CA 2048,o=Cisco Systems
*Jun 20 13:00:37.617: ISAKMP (1010): constructing CERT_REQ
 for issuer cn=Cisco Root CA M1,o=Cisco
*Jun 20 13:00:37.617: ISAKMP:(1010): sending packet to
 192.168.0.1 my_port 500 peer_port 500 (R) MM_KEY_EXCH
*Jun 20 13:00:37.617: ISAKMP:(1010):Sending an IKE IPv4 Packet.
*Jun 20 13:00:37.617: ISAKMP:(1010):Input = IKE_MESG_INTERNAL,
 IKE_PROCESS_COMPLETE
*Jun 20 13:00:37.617: ISAKMP:(1010):Old State = IKE_R_MM3
 New State = IKE_R_MM4
```

You can verify the packet with Wireshark. The MM4 packet from R2 contains seven certificate request entries:



Then, R1 receives the MM4 from R2 with multiple certificate request fields:

```
*Jun 20 13:00:37.623: ISAKMP:(1010): processing CERT_REQ payload. message ID = 0
*Jun 20 13:00:37.623: ISAKMP:(1010): peer wants a CT_X509_SIGNATURE cert
*Jun 20 13:00:37.623: ISAKMP:(1010): peer wants cert issued by
 cn=CA1,o=cisco,o=com
*Jun 20 13:00:37.623: ISAKMP: Examining profile list for trustpoint IOSCA1
*Jun 20 13:00:37.623: ISAKMP: Found matching profile for IOSCA1
*Jun 20 13:00:37.623:  Choosing trustpoint IOSCA1 as issuer
*Jun 20 13:00:37.623: ISAKMP:(1010): processing CERT_REQ payload. message ID = 0
*Jun 20 13:00:37.623: ISAKMP:(1010): peer wants a CT_X509_SIGNATURE cert
*Jun 20 13:00:37.623: ISAKMP:(1010): peer wants cert issued by
 cn=CA2,o=cisco,o=com
*Jun 20 13:00:37.623: ISAKMP:(1010): processing CERT_REQ payload. message ID = 0
*Jun 20 13:00:37.623: ISAKMP:(1010): peer wants a CT_X509_SIGNATURE cert
*Jun 20 13:00:37.623: ISAKMP:(1010): peer wants cert issued by ou=Class 3
 Public Primary Certification Authority,o=VeriSign, Inc.,c=US
*Jun 20 13:00:37.623: ISAKMP:(1010): processing CERT_REQ payload. message ID = 0
*Jun 20 13:00:37.623: ISAKMP:(1010): peer wants a CT_X509_SIGNATURE cert
*Jun 20 13:00:37.623: ISAKMP:(1010): peer wants cert issued by
```

```
 cn=Cisco SSCA2,o=Cisco Systems
*Jun 20 13:00:37.623: ISAKMP:(1010): processing CERT_REQ payload. message ID = 0
*Jun 20 13:00:37.623: ISAKMP:(1010): peer wants a CT_X509_SIGNATURE cert
*Jun 20 13:00:37.623: ISAKMP:(1010): peer wants cert issued by
 cn=Cisco Manufacturing CA,o=Cisco Systems
*Jun 20 13:00:37.623: ISAKMP:(1010): processing CERT_REQ payload. message ID = 0
*Jun 20 13:00:37.623: ISAKMP:(1010): peer wants a CT_X509_SIGNATURE cert
*Jun 20 13:00:37.623: ISAKMP:(1010): peer wants cert issued by
 cn=Cisco Root CA 2048,o=Cisco Systems
*Jun 20 13:00:37.623: ISAKMP:(1010): processing CERT_REQ payload. message ID = 0
*Jun 20 13:00:37.623: ISAKMP:(1010): peer wants a CT_X509_SIGNATURE cert
*Jun 20 13:00:37.623: ISAKMP:(1010): peer wants cert issued by
 cn=Cisco Root CA M1,o=Cisco
```

The first−match rule on R1 matches the first certificate request with the ***IOSCA1*** trust−point. This determines that R1 uses the certificate that is associated with trust−point ***IOSCA1*** for authentication in the MM5. The Fully Qualified Domain Name (FQDN) is used as the IKE ID. This is due to the ***self−identity fqdn*** configuration in the ISAKMP profile:

```
*Jun 20 13:00:37.624: ISAKMP (1010): constructing CERT payload for serialNumber=
 100+ipaddress=192.168.0.1+hostname=R1.cisco.com,cn=R1,ou=IT,o=cisco,o=com
*Jun 20 13:00:37.624: ISAKMP:(1010): using the IOSCA1 trustpoint's
 keypair to sign
```

The MM5 is received and processed by R2. The received IKE ID (***R1.cisco.com***) matches the ISAKMP profile ***prof1***. The received certificate is then validated and authentication is successful:

```
*Jun 20 13:00:37.625: ISAKMP:(1010): processing ID payload. message ID = 0
*Jun 20 13:00:37.625: ISAKMP (1010): ID payload
        next−payload : 6
        type         : 2
        FQDN name    : R1.cisco.com
        protocol     : 17
        port         : 500
        length       : 20
*Jun 20 13:00:37.625: ISAKMP:(0):: peer matches prof1 profile
.........
*Jun 20 13:00:37.626: CRYPTO_PKI: (A0013) Certificate validation succeeded
.........
*Jun 20 13:00:37.626: ISAKMP:(1010):SA authentication status:
        authenticated
```

Then, R2 prepares the MM6 with the certificate that is associated with ***IOSCA1***:

```
*Jun 20 13:00:37.627: ISAKMP (1010): constructing CERT payload for serialNumber=
 101+ipaddress=192.168.0.2+hostname=R2.cisco.com,cn=R2,ou=IT,o=cisco,o=com
*Jun 20 13:00:37.627: ISAKMP:(1010): using the IOSCA1 trustpoint's keypair to sign
*Jun 20 13:00:37.632: ISAKMP:(1010): sending packet to 192.168.0.1
 my_port 500 peer_port 500 (R) MM_KEY_EXCH
```

The packet is received by R1, and R1 verifies the certificate and authentication:

```
*Jun 20 13:00:37.632: ISAKMP (1010): received packet from 192.168.0.2
 dport 500 sport 500 Global (I) MM_KEY_EXCH
*Jun 20 13:00:37.632: ISAKMP:(1010): processing ID payload. message ID = 0
*Jun 20 13:00:37.632: ISAKMP (1010): ID payload
        next-payload : 6
        type         : 2
        FQDN name    : R2.cisco.com
        protocol     : 17
        port         : 500
        length       : 20
....
*Jun 20 13:00:37.632: ISAKMP:(0): Creating CERT validation list: IOSCA1
....
*Jun 20 13:00:37.633: CRYPTO_PKI: (80013) Certificate validation succeeded
....
*Jun 20 13:00:37.637: ISAKMP:(1010):SA authentication status:
        authenticated
*Jun 20 13:00:37.637: ISAKMP:(1010):Old State = IKE_I_MM6
 New State = IKE_P1_COMPLETE
```

This completes Phase 1. Phase 2 is negotiated as usual. The tunnel is established successfully and traffic is protected.

R2 as the IKEv1 Initiator

This example describes the process when R2 initiates the same IKEv1 tunnel and explains why it is not established.

Note: Portions of the logs are removed in order to focus only on the differences in relation to the example presented in the previous section.

R2 sends the MM3 with seven certificate request payloads because R2 does not have a trust−point associated with the ISAKMP profile (all trust−points are trusted):

```
*Jun 17 18:08:44.321: ISAKMP (0): constructing CERT_REQ for
 issuer cn=CA1,o=cisco,o=com
*Jun 17 18:08:44.321: ISAKMP (0): constructing CERT_REQ for
 issuer cn=CA2,o=cisco,o=com
*Jun 17 18:08:44.321: ISAKMP (0): constructing CERT_REQ for
 issuer ou=Class 3 Public Primary Certification Authority,
 o=VeriSign, Inc.,c=US
*Jun 17 18:08:44.321: ISAKMP (0): constructing CERT_REQ for
 issuer cn=Cisco SSCA2,o=Cisco Systems
```

```
*Jun 17 18:08:44.321: ISAKMP (0): constructing CERT_REQ for
 issuer cn=Cisco Manufacturing CA,o=Cisco Systems
*Jun 17 18:08:44.321: ISAKMP (0): constructing CERT_REQ for
 issuer cn=Cisco Root CA 2048,o=Cisco Systems
*Jun 17 18:08:44.321: ISAKMP (0): constructing CERT_REQ for
 issuer cn=Cisco Root CA M1,o=Cisco
*Jun 17 18:08:44.321: ISAKMP (0): sending packet to 192.168.0.1
 my_port 500 peer_port 500 (I) MM_SA_SETUP
```

When R1 receives the packet from R2, it processes the certificate request and matches the ***IOSCA1*** trust−point, which determines the certificate that is sent in the MM6:

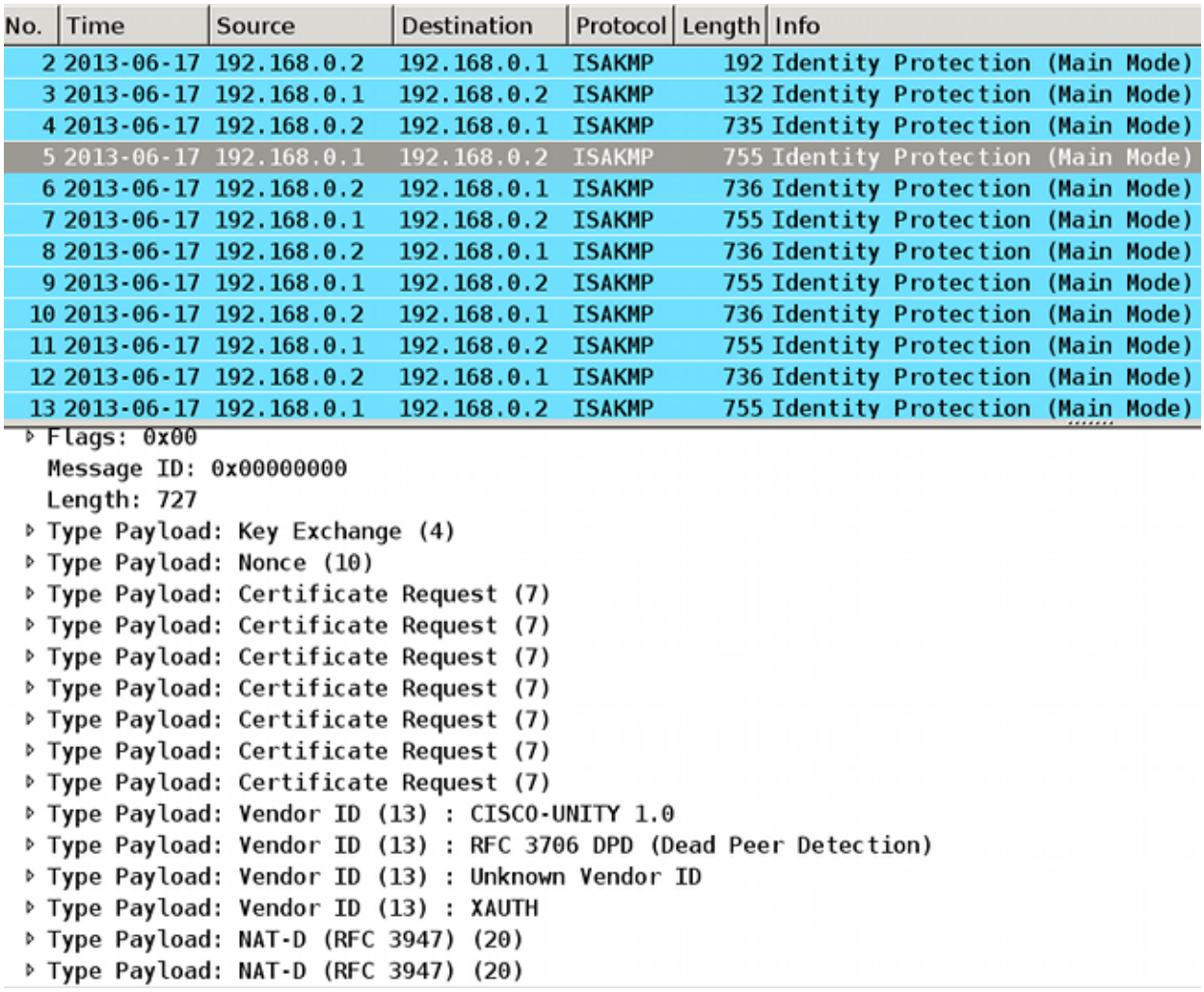
```
*Jun 17 18:08:14.321: ISAKMP:(1099): processing CERT_REQ payload.
 message ID = 0
*Jun 17 18:08:14.321: ISAKMP:(1099): peer wants a CT_X509_SIGNATURE cert
*Jun 17 18:08:14.321: ISAKMP:(1099): peer wants cert issued by
 cn=CA1,o=cisco,o=com
*Jun 17 18:08:14.321:  Choosing trustpoint IOSCA1 as issuer
*Jun 17 18:08:14.321: ISAKMP:(1099): processing CERT_REQ payload.
 message ID = 0
*Jun 17 18:08:14.321: ISAKMP:(1099): peer wants a CT_X509_SIGNATURE cert
*Jun 17 18:08:14.321: ISAKMP:(1099): peer wants cert issued by
 cn=CA2,o=cisco,o=com
*Jun 17 18:08:14.321: ISAKMP:(1099): processing CERT_REQ payload.
 message ID = 0
*Jun 17 18:08:14.321: ISAKMP:(1099): peer wants a CT_X509_SIGNATURE cert
*Jun 17 18:08:14.321: ISAKMP:(1099): peer wants cert issued by
 ou=Class 3 Public Primary Certification Authority,o=VeriSign, Inc.,c=US
*Jun 17 18:08:14.321: ISAKMP:(1099): processing CERT_REQ payload.
 message ID = 0
*Jun 17 18:08:14.321: ISAKMP:(1099): peer wants a CT_X509_SIGNATURE cert
*Jun 17 18:08:14.321: ISAKMP:(1099): peer wants cert issued by
 cn=Cisco SSCA2,o=Cisco Systems
*Jun 17 18:08:14.321: ISAKMP:(1099): processing CERT_REQ payload.
 message ID = 0
*Jun 17 18:08:14.321: ISAKMP:(1099): peer wants a CT_X509_SIGNATURE cert
*Jun 17 18:08:14.321: ISAKMP:(1099): peer wants cert issued by
 cn=Cisco Manufacturing CA,o=Cisco Systems
*Jun 17 18:08:14.321: ISAKMP:(1099): processing CERT_REQ payload.
 message ID = 0
*Jun 17 18:08:14.321: ISAKMP:(1099): peer wants a CT_X509_SIGNATURE cert
*Jun 17 18:08:14.321: ISAKMP:(1099): peer wants cert issued by
 cn=Cisco Root CA 2048,o=Cisco Systems
*Jun 17 18:08:14.321: ISAKMP:(1099): processing CERT_REQ payload.
 message ID = 0
*Jun 17 18:08:14.321: ISAKMP:(1099): peer wants a CT_X509_SIGNATURE cert
*Jun 17 18:08:14.321: ISAKMP:(1099): peer wants cert issued by
 cn=Cisco Root CA M1,o=Cisco
```

Afterwards, R1 prepares the MM4 packet with the certificate request payload. Now there are multiple certificate request payloads:

```
*Jun 17 18:08:14.321: ISAKMP (1099): constructing CERT_REQ for issuer
 cn=CA2,o=cisco,o=com
*Jun 17 18:08:14.321: ISAKMP (1099): constructing CERT_REQ for issuer
```

```
 cn=CA1,o=cisco,o=com
*Jun 17 18:08:14.322: ISAKMP (1099): constructing CERT_REQ for issuer
 ou=Class 3 Public Primary Certification Authority,
 o=VeriSign, Inc.,c=US
*Jun 17 18:08:14.322: ISAKMP (1099): constructing CERT_REQ for issuer
 cn=Cisco SSCA2,o=Cisco Systems
*Jun 17 18:08:14.322: ISAKMP (1099): constructing CERT_REQ for issuer
 cn=Cisco Manufacturing CA,o=Cisco Systems
*Jun 17 18:08:14.322: ISAKMP (1099): constructing CERT_REQ for issuer
 cn=Cisco Root CA 2048,o=Cisco Systems
*Jun 17 18:08:14.322: ISAKMP (1099): constructing CERT_REQ for issuer
 cn=Cisco Root CA M1,o=Cisco
*Jun 17 18:08:14.322: ISAKMP:(1099): sending packet to 192.168.0.2
 my_port 500 peer_port 500 (R) MM_KEY_EXCH
```

Verify the logs with Embedded Packet Capture (EPC) and Wireshark:



No.	Time	Source	Destination	Protocol	Length	Info
2	2013-06-17	192.168.0.2	192.168.0.1	ISAKMP	192	Identity Protection (Main Mode)
3	2013-06-17	192.168.0.1	192.168.0.2	ISAKMP	132	Identity Protection (Main Mode)
4	2013-06-17	192.168.0.2	192.168.0.1	ISAKMP	735	Identity Protection (Main Mode)
5	2013-06-17	192.168.0.1	192.168.0.2	ISAKMP	755	Identity Protection (Main Mode)
6	2013-06-17	192.168.0.2	192.168.0.1	ISAKMP	736	Identity Protection (Main Mode)
7	2013-06-17	192.168.0.1	192.168.0.2	ISAKMP	755	Identity Protection (Main Mode)
8	2013-06-17	192.168.0.2	192.168.0.1	ISAKMP	736	Identity Protection (Main Mode)
9	2013-06-17	192.168.0.1	192.168.0.2	ISAKMP	755	Identity Protection (Main Mode)
10	2013-06-17	192.168.0.2	192.168.0.1	ISAKMP	736	Identity Protection (Main Mode)
11	2013-06-17	192.168.0.1	192.168.0.2	ISAKMP	755	Identity Protection (Main Mode)
12	2013-06-17	192.168.0.2	192.168.0.1	ISAKMP	736	Identity Protection (Main Mode)
13	2013-06-17	192.168.0.1	192.168.0.2	ISAKMP	755	Identity Protection (Main Mode)

```
▸ Flags: 0x00
  Message ID: 0x00000000
  Length: 727
▸ Type Payload: Key Exchange (4)
▸ Type Payload: Nonce (10)
▸ Type Payload: Certificate Request (7)
▸ Type Payload: Certificate Request (7)
▸ Type Payload: Certificate Request (7)
▸ Type Payload: Certificate Request (7)
▸ Type Payload: Certificate Request (7)
▸ Type Payload: Certificate Request (7)
▸ Type Payload: Certificate Request (7)
▸ Type Payload: Vendor ID (13) : CISCO-UNITY 1.0
▸ Type Payload: Vendor ID (13) : RFC 3706 DPD (Dead Peer Detection)
▸ Type Payload: Vendor ID (13) : Unknown Vendor ID
▸ Type Payload: Vendor ID (13) : XAUTH
▸ Type Payload: NAT-D (RFC 3947) (20)
▸ Type Payload: NAT-D (RFC 3947) (20)
```

Even though R1 is configured for a single trust−point (***IOSCA1***) in the ISAKMP profile, there are multiple certificate requests sent. This occurs because the ***ca trust−point*** command in the ISAKMP profile determines the certificate request payload, but only when the router is the initiator of the ISAKMP session. If the router is the responder, there are multiple certificate request payloads for all of the globally−defined trust−points

because R1 does not yet know the ISAKMP profile that is used for the IKE session.

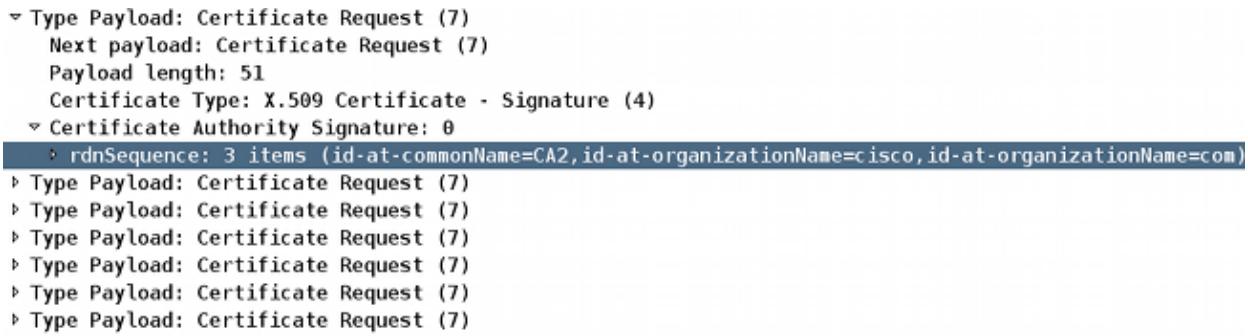
The inbound IKE session is bound to a specific ISAKMP profile after the reception of the MM5, which includes the IKE ID. Afterwards, the ***match identity*** command for the specific profile binds the IKE session to the profile. However, the router cannot determine this until now. There might be multiple ISAKMP profiles with different ***ca trust−point*** commands configured for each profile.

For this reason, R1 must send the certificate request for all of the globally−configured trust−points.

Refer to the command reference for the ***ca trust−point*** command:

> ***A router initiating IKE and a router responding to the IKE request should have symmetrical trustpoint configurations***. For example, a responding router (in IKE Main Mode) performing RSA signature encryption and authentication might use trustpoints that were defined in the global configuration when sending the CERT−REQ payloads. However, the router might use a restricted list of trustpoints that were defined in the ISAKMP profile for the certificate verification. If the peer (the IKE initiator) is configured to use a certificate whose trustpoint is in the global list of the responding router but not in ISAKMP profile of the responding router, the certificate is rejected. (However, if the initiating router does not know about the trustpoints in the global configuration of the responding router, the certificate can still be authenticated.)

Now verify the MM4 packet details in order to discover the first certificate request payload:



The MM4 packet that is sent from R1 includes the ***IOSCA2*** trust−point in the first certificate request payload because of the order in which the certificates are installed; the first one is signed by the ***IOSCA2*** trust−point:

```
R1#sh crypto pki certificates
Certificate
  Status: Available
  Certificate Serial Number (hex): 03
  Certificate Usage: General Purpose
  Issuer:
    cn=CA2
    o=cisco
    o=com
  Subject:
    Name: R1.cisco.com
    IP Address: 192.168.0.1
    Serial Number: 100
```

```
    serialNumber=100+ipaddress=192.168.0.1+hostname=R1.cisco.com
    cn=R1
    ou=IT
    o=cisco
    o=com
  Validity Date:
    start date: 13:25:01 CET Jun 17 2013
    end   date: 13:25:01 CET Jun 17 2014
  Associated Trustpoints: IOSCA2
...
<output omitted, 1 more R1 cert signed by CA1, 2 more CA certs>
```

Make a comparison with the MM3 packet that is sent from R2 when the ***IOSCA1*** trust−point is included in the first certificate request payload:

```
R2#sh crypto pki certificates
Certificate
  Status: Available
  Certificate Serial Number (hex): 02
  Certificate Usage: General Purpose
  Issuer:
    cn=CA1
    o=cisco
    o=com
  Subject:
    Name: R2.cisco.com
    IP Address: 192.168.0.2
    Serial Number: 101
    serialNumber=101+ipaddress=192.168.0.2+hostname=R2.cisco.com
    cn=R2
    ou=IT
    o=cisco
    o=com
  Validity Date:
    start date: 13:23:49 CET Jun 17 2013
    end   date: 13:23:49 CET Jun 17 2014
  Associated Trustpoints: IOSCA1
  Storage: nvram:CA1#2.cer
...
<output omitted, 1 more R2 cert signed by CA2, 2 more CA certs>
```

Now R2 receives the MM4 packet from R1 and begins to process the certificate request. The first certificate request payload matches the ***IOSCA2*** trust−point:

```
*Jun 17 18:08:44.335: ISAKMP:(1100): processing CERT_REQ payload.
 message ID = 0
*Jun 17 18:08:44.335: ISAKMP:(1100): peer wants a CT_X509_SIGNATURE cert
*Jun 17 18:08:44.335: ISAKMP:(1100): peer wants cert issued by
 cn=CA2,o=cisco,o=com
*Jun 17 18:08:44.335:  Choosing trustpoint IOSCA2 as issuer
*Jun 17 18:08:44.335: ISAKMP:(1100): processing CERT_REQ payload.
 message ID = 0
*Jun 17 18:08:44.335: ISAKMP:(1100): peer wants a CT_X509_SIGNATURE cert
*Jun 17 18:08:44.335: ISAKMP:(1100): peer wants cert issued by
 cn=CA1,o=cisco,o=com
```

```
*Jun 17 18:08:44.335: ISAKMP:(1100): processing CERT_REQ payload.
 message ID = 0
*Jun 17 18:08:44.335: ISAKMP:(1100): peer wants a CT_X509_SIGNATURE cert
*Jun 17 18:08:44.335: ISAKMP:(1100): peer wants cert issued by
 ou=Class 3 Public Primary Certification Authority,o=VeriSign, Inc.,c=US
*Jun 17 18:08:44.335: ISAKMP:(1100): processing CERT_REQ payload.
 message ID = 0
*Jun 17 18:08:44.335: ISAKMP:(1100): peer wants a CT_X509_SIGNATURE cert
*Jun 17 18:08:44.335: ISAKMP:(1100): peer wants cert issued by
 cn=Cisco SSCA2,o=Cisco Systems
*Jun 17 18:08:44.335: ISAKMP:(1100): processing CERT_REQ payload.
 message ID = 0
*Jun 17 18:08:44.335: ISAKMP:(1100): peer wants a CT_X509_SIGNATURE cert
*Jun 17 18:08:44.335: ISAKMP:(1100): peer wants cert issued by
 cn=Cisco Manufacturing CA,o=Cisco Systems
*Jun 17 18:08:44.335: ISAKMP:(1100): processing CERT_REQ payload.
 message ID = 0
*Jun 17 18:08:44.335: ISAKMP:(1100): peer wants a CT_X509_SIGNATURE cert
*Jun 17 18:08:44.335: ISAKMP:(1100): peer wants cert issued by
 cn=Cisco Root CA 2048,o=Cisco Systems
*Jun 17 18:08:44.335: ISAKMP:(1100): processing CERT_REQ payload.
 message ID = 0
*Jun 17 18:08:44.335: ISAKMP:(1100): peer wants a CT_X509_SIGNATURE cert
*Jun 17 18:08:44.335: ISAKMP:(1100): peer wants cert issued by
 cn=Cisco Root CA M1,o=Cisco
```

When R2 prepares the MM5 packet, it uses the certificate that is associated with the ***IOSCA2*** trust−point:

```
*Jun 17 18:08:44.335: ISAKMP:(1100):SA is doing RSA signature authentication
 using id type ID_FQDN
*Jun 17 18:08:44.335: ISAKMP (1100): ID payload
        next-payload : 6
        type         : 2
        FQDN name    : R2.cisco.com
        protocol     : 17
        port         : 500
        length       : 20
*Jun 17 18:08:44.335: ISAKMP:(1100):Total payload length: 20
*Jun 17 18:08:44.335: ISAKMP:(1100): IKE->PKI Get CertificateChain to be sent
 to peer state (I) MM_KEY_EXCH (peer 192.168.0.1)
*Jun 17 18:08:44.335: ISAKMP:(1100): PKI->IKE Got CertificateChain to be sent
 to peer state (I) MM_KEY_EXCH (peer 192.168.0.1)
*Jun 17 18:08:44.336: ISAKMP (1100): constructing CERT payload for
 serialNumber=101+ipaddress=192.168.0.2+hostname=R2.cisco.com,cn=R2,
 ou=IT,o=cisco,o=com
R2#
*Jun 17 18:08:44.336: ISAKMP:(1100): using the IOSCA2 trustpoint's
 keypair to sign
*Jun 17 18:08:44.336: ISAKMP:(1100): sending packet to 192.168.0.1
 my_port 500 peer_port 500 (I) MM_KEY_EXCH
*Jun 17 18:08:44.336: ISAKMP:(1100):Sending an IKE IPv4 Packet.
```

The MM5 packet is received by R1. Because R1 trusts only the ***IOSCA1*** trust−point (for ISAKMP profile ***prof1***), the certificate validation fails:

```
*Jun 17 18:08:44.337: ISAKMP (1100): received packet from 192.168.0.2
 dport 500 sport 500 Global (R) MM_KEY_EXCH
*Jun 17 18:08:44.337: ISAKMP:(1100):Input = IKE_MESG_FROM_PEER, IKE_MM_EXCH
*Jun 17 18:08:44.337: ISAKMP:(1100):Old State =IKE_R_MM4  New State = IKE_R_MM5

*Jun 17 18:08:44.337: ISAKMP:(1100): processing ID payload. message ID = 0
*Jun 17 18:08:44.337: ISAKMP (1100): ID payload
        next-payload : 6
        type         : 2
        FQDN name    : R2.cisco.com
        protocol     : 17
        port         : 500
        length       : 20
*Jun 17 18:08:44.337: ISAKMP:(0):: peer matches prof1 profile
*Jun 17 18:08:44.337: ISAKMP:(1100): processing CERT payload. message ID = 0
*Jun 17 18:08:44.337: ISAKMP:(1100): processing a CT_X509_SIGNATURE cert
*Jun 17 18:08:44.337: ISAKMP:(1100): IKE->PKI Add peer's certificate state
 (R) MM_KEY_EXCH (peer 192.168.0.2)
*Jun 17 18:08:44.337: CRYPTO_PKI: (900C5) Adding peer certificate
*Jun 17 18:08:44.337: ISAKMP:(1100): PKI->IKE Added peer's certificate state
 (R) MM_KEY_EXCH (peer 192.168.0.2)
*Jun 17 18:08:44.337: ISAKMP:(1100): IKE->PKI Get PeerCertificateChain state
 (R) MM_KEY_EXCH (peer 192.168.0.2)
*Jun 17 18:08:44.337: ISAKMP:(1100): PKI->IKE Got PeerCertificateChain state
 (R) MM_KEY_EXCH (peer 192.168.0.2)
*Jun 17 18:08:44.337: ISAKMP:(1100): peer's pubkey isn't cached
*Jun 17 18:08:44.337: ISAKMP:(1100):Profile has no keyring, aborting key search
*Jun 17 18:08:44.337: ISAKMP:(0): Creating CERT validation list: IOSCA1,
*Jun 17 18:08:44.337: ISAKMP:(1100): IKE->PKI Validate certificate chain state
 (R) MM_KEY_EXCH (peer 192.168.0.2)
*Jun 17 18:08:44.337: CRYPTO_PKI:ip-ext-val:IP extension validation not required
*Jun 17 18:08:44.341: CRYPTO_PKI: (900C5) Check for identical certs
*Jun 17 18:08:44.341: CRYPTO_PKI: (900C5) Create a list of suitable trustpoints
*Jun 17 18:08:44.341: CRYPTO_PKI: (900C5) No suitable trustpoints found
*Jun 17 18:08:44.341: ISAKMP:(1100): PKI->IKE Validate certificate chain state
 (R) MM_KEY_EXCH (peer 192.168.0.2)
*Jun 17 18:08:44.341: %CRYPTO-5-IKMP_INVAL_CERT: Certificate received from
 192.168.0.2 is bad: unknown error returned in certificate validation
R1#
*Jun 17 18:08:44.341: ISAKMP:(1100): Unknown error in cert validation, -1
```

This configuration works if the order of the certificate enrollment on R1 is different because the first displayed certificate is signed by the ***IOSCA1*** trust-point. Also, the first certificate request payload in the MM4 is the ***IOSCA1*** trust-point, which is then chosen by R2 and validated successfully on R1 in the MM6.

IKEv1 without a *ca trust-point* Command in the Profile

For scenarios with multiple profiles and trust-points but without a specific trust-point configuration in the profiles, there are no issues because there is no validation of specific trust-points determined by a ***ca trust-point*** command configuration. However, the selection process might not be obvious. Dependent upon the router that is the initiator, the different certificates are selected for the authentication process in relation to the order of certificate enrollment.

Sometimes a certificate can be supported by only one side of the connection, such as in x509 Version 1, which is not a typical hash function that is used in order to sign. The VPN tunnel might be established only from one side of the connection.

RFC Reference for IKEv1

Here is a snip from RFC4945:

> 3.2.7.1. Specifying Certification Authorities
>
> When ***requesting*** in−band exchange of keying materials, implementations SHOULD generate CERTREQs for every peer trust anchor that ***local policy explicitly*** deems trusted during a given exchange.

The RFC is not clear. The ***local policy explicitly*** might relate to the ***ca trust−point*** command that is configured in the crypto ISAKMP profile. The problem is that at the MM3 and MM4 stage of the process, you cannot select an ISAKMP profile unless you use an IP address for the identity and the trust−points because the authentication in the MM5 and the MM6 stage of the process must occur first. For this reason, ***local policy explicitly*** relates to all of the trust−points that are configured on the device.

Note: This information is not Cisco−specific, but it is IKEv1−specific.

IKEv2 Profile Selection with Identities that Overlap

Before multiple certificates for IKEv2 is described, it is important to know the way that the profiles are selected when match identity is used, which is satisfied for all the profiles. This is not a recommended scenario because the results of the IKEv2 negotiation depends on multiple factors. The same problems exist for IKEv1 when profiles that overlap are used.

Here is an example IKEv2 initiator configuration:

```
crypto ikev2 proposal prop−1
 encryption 3des
 integrity md5
 group 2
!
crypto ikev2 policy pol−1
 match fvrf any
 proposal prop−1
!
crypto ikev2 profile profile1
 match identity remote address 192.168.0.2 255.255.255.255
 identity local address 192.168.0.1
 authentication remote rsa−sig
 authentication local rsa−sig
 pki trustpoint TP1

crypto ipsec transform−set trans esp−3des esp−sha−hmac
 mode tunnel
!
crypto ipsec profile profile1
 set transform−set trans
 set ikev2−profile profile1
!
```

```
interface Loopback0
 ip address 192.168.100.1 255.255.255.255
!
interface Tunnel1
 ip address 10.0.0.1 255.255.255.0
 tunnel source Ethernet0/0
 tunnel destination 192.168.0.2
 tunnel protection ipsec profile profile1
!
interface Ethernet0/0
 ip address 192.168.0.1 255.255.255.0

ip route 192.168.200.1 255.255.255.255 10.0.0.2
```

The identity type address is used for both sides of the connection. Authentication via certificates (can also be pre−shared keys) is not important for this example. The responder has multiple profiles that all match the inbound IKEv2 traffic:

```
crypto ikev2 proposal prop−1
 encryption 3des
 integrity md5
 group 2
!
crypto ikev2 policy pol−1
 match fvrf any
 proposal prop−1
!
crypto ikev2 profile profile1
 match identity remote address 192.168.0.1 255.255.255.255
 identity local address 192.168.0.2
 authentication remote rsa−sig
 authentication local rsa−sig
 pki trustpoint TP1
!
crypto ikev2 profile profile2
 match identity remote address 192.168.0.1 255.255.255.255
 identity local address 192.168.0.2
 authentication remote rsa−sig
 authentication local rsa−sig
 pki trustpoint TP1
!
crypto ikev2 profile profile3
 match identity remote address 192.168.0.1 255.255.255.255
 identity local address 192.168.0.2
 authentication remote rsa−sig
 authentication local rsa−sig
 pki trustpoint TP1

crypto ipsec transform−set trans esp−3des esp−sha−hmac
 mode tunnel
!
crypto ipsec profile profile1
 set transform−set trans
 set ikev2−profile profile1
!
interface Loopback0
 ip address 192.168.200.1 255.255.255.255
!
interface Tunnel1
 ip address 10.0.0.2 255.255.255.0
```

```
 tunnel source Ethernet0/0
 tunnel destination 192.168.0.1
 tunnel protection ipsec profile profile1
!
interface Ethernet0/0
 ip address 192.168.0.2 255.255.255.0

ip route 192.168.100.1 255.255.255.255 10.0.0.1
```

The initiator sends the third IKEv2 packet, and the responder must choose the profile based on the identity that is received. The identity is an IPv4 address (***192.168.0.1***):

```
IKEv2:(SA ID = 1):Searching policy based on peer's identity '192.168.0.1' of
 type 'IPv4 address'
```

All of the profiles satisfy this identity because of the ***match identity*** command that is configured. The IOS chooses the last one in the configuration, which is ***profile3*** in this example:

```
IKEv2:found matching IKEv2 profile 'profile3'
```

In order to verify the order, enter the ***show crypto ikev2 profile*** command.

Note: Even when there is a generic address (0.0.0.0) in the profile, it is still selected. The IOS does not attempt to find a best match; it tries to find the first match. However, this only occurs because all of the profiles have the same ***match identity remote*** command configured. For the IKEv1 and the IKEv2 profiles that have different match identity rules, the most specific one is always used. Cisco recommends that you not have the profiles configured with the ***overlapping match identity*** command because it is difficult to predict the profile that is selected.

In this scenario, ***profile3*** is selected by the responder, but ***profile1*** is used for the tunnel interface. This causes an error to appear when the proxy ID is negotiated:

```
*Jul 17 09:23:48.187: map_db_check_isakmp_profile profile did not match
*Jul 17 09:23:48.187: map_db_find_best did not find matching map
*Jul 17 09:23:48.187: IPSEC(ipsec_process_proposal):
 proxy identities not supported
*Jul 17 09:23:48.187: IKEv2:(SA ID = 1):There was no
 IPSEC policy found for received TS
*Jul 17 09:23:48.187: IKEv2:(SA ID = 1):
*Jul 17 09:23:48.187: IKEv2:(SA ID = 1):Sending TS unacceptable notify
```

IKEv2 Flow when Certificates are Used

When certificates are used for IKEv2 in order to authenticate, the initiator does not send the certificate request payload in the first packet:

```
IKEv2 IKE_SA_INIT Exchange REQUEST
Payload contents:
 SA KE N VID VID NOTIFY(NAT_DETECTION_SOURCE_IP)
 NOTIFY(NAT_DETECTION_DESTINATION_IP)
```

The responder answers with the certificate request payload (second packet) and all of the CAs because the responder has no knowledge of the profile that should be used at this stage. The packet that contains the information is sent to the initiator:

```
IKEv2 IKE_SA_INIT Exchange RESPONSE
Payload contents:
 SA KE N VID VID NOTIFY(NAT_DETECTION_SOURCE_IP) NOTIFY
 (NAT_DETECTION_DESTINATION_IP) CERTREQ NOTIFY(HTTP_CERT_LOOKUP_SUPPORTED)
```

The initiator processes the packet and chooses a trust−point that matches the proposed CA:

```
IKEv2:(SA ID = 1):[IKEv2 −> PKI] Retrieving trustpoint(s) from
 received certificate hash(es)
IKEv2:(SA ID = 1):[PKI −> IKEv2] Retrieved trustpoint(s): 'TP1'
```

The initiator then sends the third packet with both the certificate request and the certificate payload. This packet is already encrypted with keying material from the Diffie−Hellman (DH) phase:

```
IKEv2:(SA ID = 1):Building packet for encryption.
Payload contents:
 VID IDi CERT CERTREQ NOTIFY(HTTP_CERT_LOOKUP_SUPPORTED) AUTH CFG SA TSi
 TSr NOTIFY(INITIAL_CONTACT) NOTIFY(SET_WINDOW_SIZE) NOTIFY(ESP_TFC_NO_SUPPORT)
 NOTIFY(NON_FIRST_FRAGS)
```

The fourth packet is sent from the responder to the initiator and contains only the certificate payload:

```
IKEv2 IKE_AUTH Exchange RESPONSE
```

```
Payload contents:
 VID IDr CERT AUTH SA TSi TSr NOTIFY(SET_WINDOW_SIZE) NOTIFY(ESP_TFC_NO_SUPPORT)
 NOTIFY(NON_FIRST_FRAGS)
```

The flow described here is similar to the IKEv1 flow. The responder must send the certificate request payload up front without knowledge of the profile that should be used, which creates the same problems that are previously described for IKEv1 (from a protocol perspective). However, the implementation on the IOS is better for the IKEv2 than for the IKEv1.

IKEv2 Mandatory Trust−point for the Initiator

Here is an example of when an IKEv2 initiator attempts to use a profile with certificate authentication and has no trust−point configured under that profile:

```
crypto ikev2 profile profile1
 match identity remote address 192.168.0.2 255.255.255.255
 identity local address 192.168.0.1
 authentication remote rsa−sig
 authentication local rsa−sig
```

The first packet is sent without any certificate request payload, as previously described. The response from the responder includes the certificate request payload for all of the trust−points that are defined in Global Configuration mode. This is received by the initiator:

```
*Jul 17 17:40:43.183: IKEv2:(SA ID = 1):[IKEv2 −> PKI] Retrieving trustpoint(s)
 from received certificate hash(es)
*Jul 17 17:40:43.183: IKEv2:(SA ID = 1):[PKI −> IKEv2] Retrieved
 trustpoint(s): 'TP1'
*Jul 17 17:40:43.183: CRYPTO_PKI: crypto_pki_get_cert_record_by_subject()
*Jul 17 17:40:43.183: CRYPTO_PKI: Found a subject match
*Jul 17 17:40:43.183: CRYPTO_PKI: crypto_pki_get_cert_record_by_subject()
*Jul 17 17:40:43.183: CRYPTO_PKI: Found a subject match
*Jul 17 17:40:43.183: CRYPTO_PKI: Trust−Point TP1 picked up
*Jul 17 17:40:43.183: CRYPTO_PKI: 1 matching trustpoints found
*Jul 17 17:40:43.183: IKEv2:(SA ID = 1):[IKEv2 −> PKI] Retrieving trustpoint(s)
 from received certificate hash(es)
*Jul 17 17:40:43.183: IKEv2:(SA ID = 1):[PKI −> IKEv2] Retrieved
 trustpoint(s): 'TP2'
*Jul 17 17:40:43.183: CRYPTO_PKI: Trust−Point TP2 picked up
*Jul 17 17:40:43.183: CRYPTO_PKI: crypto_pki_get_cert_record_by_subject()
*Jul 17 17:40:43.183: CRYPTO_PKI: Found a subject match
*Jul 17 17:40:43.183: CRYPTO_PKI: crypto_pki_get_cert_record_by_subject()
*Jul 17 17:40:43.183: CRYPTO_PKI: Found a subject match
*Jul 17 17:40:43.183: CRYPTO_PKI: 1 matching trustpoints found
*Jul 17 17:40:43.183: IKEv2:(SA ID = 1):Failed to build certificate payload
```

The initiator does not know the trust−point that should be used in order to sign. This is the main difference

when the IKEv2 implementation is compared to the IKEv1. The IKEv2 initiator must have the trust−point configured under the IKEv2 initiator profile, but it is not necessary for the IKEv2 responder.

Here is an excerpt from the command reference:

> If there is no trustpoint defined in the IKEv2 profile configuration, the default is to ***validate the certificate*** using all the trustpoints that are defined in the global configuration

It is possible to define different trust−points; one in order to sign and a different one in order to validate. Unfortunately, the mandatory trust−point that is configured under the IKEv2 profile does not solve all of the problems.

R2 as the IKEv2 Initiator

In this example, R2 is the IKEv2 initiator:

```
crypto ikev2 profile profile1
 match identity remote address 192.168.0.1 255.255.255.255
 identity local address 192.168.0.2
 authentication remote rsa−sig
 authentication local rsa−sig
 pki trustpoint TP1
 pki trustpoint TP2
```

In this example, R1 is the IKEv2 responder:

```
crypto ikev2 profile profile1
 match identity remote address 192.168.0.2 255.255.255.255
 identity local address 192.168.0.1
 authentication remote rsa−sig
 authentication local rsa−sig
 pki trustpoint TP1
```

Here, R2 sends the first packet without any certificate request. The responder responds with a certificate request for all of the configured trust−points. The order of the payloads is similar to the IKEv1 and is dependent upon the certificates that are installed:

```
R1#show crypto pki certificates
Certificate
  Status: Available
  Certificate Serial Number (hex): 04
  Certificate Usage: General Purpose
  Issuer:
    cn=CA2
 ....
 Associated Trustpoints: TP2
```

The first configured certificate on R1 is associated with the ***TP2*** trust−point, so the first certificate request payload is for the CA that is associated with the ***TP2*** trust−point. Thus, R2 selects it for authentication (first match rule):

```
R2#
*Jul 17 18:09:04.542: IKEv2:(SA ID = 1):Processing IKE_SA_INIT message
*Jul 17 18:09:04.542: IKEv2:(SA ID = 1):[IKEv2 −> PKI] Retrieving trustpoint(s)
 from received certificate hash(es)
*Jul 17 18:09:04.542: IKEv2:(SA ID = 1):[PKI −> IKEv2] Retrieved
 trustpoint(s): 'TP2'
*Jul 17 18:09:04.542: IKEv2:(SA ID = 1):[IKEv2 −> PKI] Getting cert chain for
 the trustpoint TP2
*Jul 17 18:09:04.542: IKEv2:(SA ID = 1):[PKI −> IKEv2] Getting of cert chain
 for the trustpoint PASSED
```

Then, R2 prepares a response (packet 3) with the certification request payload that is associated with ***TP2***. R1 cannot trust the certificate since it is configured for validation against the ***TP1*** trust−point:

```
*Jul 17 18:09:04.550: IKEv2:(SA ID = 1):[IKEv2 −> PKI] Retrieving trustpoint(s)
 from received certificate hash(es)
*Jul 17 18:09:04.550: IKEv2:(SA ID = 1):[PKI −> IKEv2] Retrieved
 trustpoint(s): 'TP1'
*Jul 17 18:09:04.550: IKEv2:(SA ID = 1):[IKEv2 −> PKI] Getting cert chain for
 the trustpoint TP1
*Jul 17 18:09:04.550: IKEv2:(SA ID = 1):[PKI −> IKEv2] Getting of cert chain
 for the trustpoint PASSED
*Jul 17 18:09:04.550: IKEv2:(SA ID = 1):Get peer's authentication method
*Jul 17 18:09:04.550: IKEv2:(SA ID = 1):Peer's authentication method is 'RSA'
*Jul 17 18:09:04.550: IKEv2:(SA ID = 1):[IKEv2 −> PKI] Validating
 certificate chain
*Jul 17 18:09:04.554: IKEv2:(SA ID = 1):[PKI −> IKEv2] Validation of certificate
 chain FAILED
*Jul 17 18:09:04.554: IKEv2:(SA ID = 1):Verification of peer's authentication
 data FAILED
*Jul 17 18:09:04.554: IKEv2:(SA ID = 1):Sending authentication failure notify
*Jul 17 18:09:04.554: IKEv2:(SA ID = 1):Building packet for encryption.
Payload contents:
 NOTIFY(AUTHENTICATION_FAILED)
```

As previously mentioned, Cisco recommends that you do not use multiple trust−points under one IKEv2 profile. When you use multiple trust−points, it is necessary to ensure that both sides trust exactly the same trust−points. For example, both R1 and R2 have both TP1 and TP2 configured in their profiles.

Summary

This section provides a brief summary of the information that is described in the document.

The certificate request payload content depends on the configuration. If a specific trust−point is configured for the ISAKMP profile and the router is the ISAKMP initiator, then the certificate request in the MM3 contains only the CA that is associated with the trust−point. However, if the same router is the ISAKMP responder, then the MM4 packet that is sent by the router includes multiple certificate request payloads for all of the globally−defined trust−points (when the ***ca trust−point*** command is not taken into consideration). This occurs because the ISAKMP responder can determine the ISAKMP profile that should be used only after it receives the MM5 and the certificate request that is included in the MM4.

The certificate request payload in the MM3 and the MM4 is important because of the first match rule. The first match rule determines the trust−point that is used for the certificate selection, which is needed for authentication in the MM5 and the MM6.

The order of certificate request payload depends on the order of the certificates that are installed. The issuer of the first certificate that appears in the output of the ***show crypto pki certificate*** command is sent first. This first certificate is the last one that is enrolled.

It is possible to configure multiple trust−points for an ISAKMP profile. If this is performed, then all the previous rules still apply.

All of the problems and caveats that are described in this document are due to the IKEv1 protocol design. The authentication stage occurs in the MM5 and the MM6, while the proposals for the authentication (certificate requests) must be sent at an earlier stage (up front) without knowledge of the ISAKMP profile that should be used. This is not a Cisco−specific problem and is related to the limitations of the IKEv1 protocol design.

The IKEv2 protocol is similar to the IKEv1 in regards to the certificate negotiation process. However, the implementation on the IOS forces the use of specific trust−points for the initiator. This does not solve all of the issues. When multiple trust−points are configured for a single profile and a single trust−point is configured on the other side, it is still possible to encounter problems with authentication. Cisco recommends that you use symmetric trust−point configurations for both sides of the connection (the same trust−points configured for both of the IKEv2 profiles).

Here are some important notes about the information that is described in this document:

- With asymmetric trust−point configurations for the IKEv1 profiles of peers, the tunnel might initiate from only one side of the tunnel. The trust−point configuration for the IKEv1 profile is optional.
- With asymmetric trust−point configurations for the IKEv2 profiles of peers, the tunnel might initiate from only one side of the tunnel. The trust−point configuration for the IKEv2 profile is mandatory for the initiator.
- The certificate request payload order depends on the order of the certificates that appear in the output of the ***show crypto pki certificate*** command (first match).
- The certificate request payload order determines the certificate that is selected by the responder (first match).
- When you use multiple profiles for the IKEv1 and the IKEv2 and have the same match identity rules configured, it is difficult to predict the results (too many factors involved).
- Cisco recommends that you use symmetric trust−point configurations for both the IKEv1 and the IKEv2.

Related Information

- ***Internet Key Exchange for IPsec VPNs Configuration Guide, Cisco IOS Release 15M&T − Certificate to ISAKMP Profile Mapping***
- ***Cisco IOS Security Command Reference: Commands A to C − ca trust−point through clear eou***
- ***Technical Support & Documentation − Cisco Systems***

Updated: Apr 23, 2014 Document ID: 117633