VRF-Aware IPsec

Last Updated: May 20, 2011

The VRF-Aware IPsec feature introduces IP Security (IPsec) tunnel mapping to Multiprotocol Label Switching (MPLS) Virtual Private Networks (VPNs). By using the VRF-Aware IPsec feature, you can map IPsec tunnels to Virtual Routing and Forwarding (VRF) instances using a single public-facing address.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the Feature Information Table at the end of this document.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for VRF-Aware IPsec

- If you are configuring VRF-Aware IPsec using a crypto map configuration and the Inside VRF (IVRF) is not the same as the Front Door VRF (FVRF), this feature is not interoperable with unicast reverse path forwarding (uRPF) if uRPF is enabled on the crypto map interface. If your network requires uRPF, it is recommended that you use Virtual Tunnel Interface (VTI) for IPsec instead of crypto maps.

- The VRF-Aware IPsec feature does not allow IPsec tunnel mapping between VRFs. For example, it does not allow IPsec tunnel mapping from VRF vpn1 to VRF vpn2.
- When the VRF-Aware IPsec feature is used with a crypto map, this crypto map cannot use the global VRF as the IVRF and a non-global VRF as the FVRF. However, configurations based on virtual tunnel interfaces do not have that limitation.

Information About VRF-Aware IPsec

VRF Instance

A VRF instance is a per-VPN routing information repository that defines the VPN membership of a customer site attached to the Provider Edge (PE) router. A VRF comprises an IP routing table, a derived Cisco Express Forwarding (CEF) table, a set of interfaces that use the forwarding table, and a set of rules and routing protocol parameters that control the information that is included in the routing table. A separate set of routing and CEF tables is maintained for each VPN customer.

MPLS Distribution Protocol

The MPLS distribution protocol is a high-performance packet-forwarding technology that integrates the performance and traffic management capabilities of data link layer switching with the scalability, flexibility, and performance of network-layer routing.

VRF-Aware IPsec Functional Overview

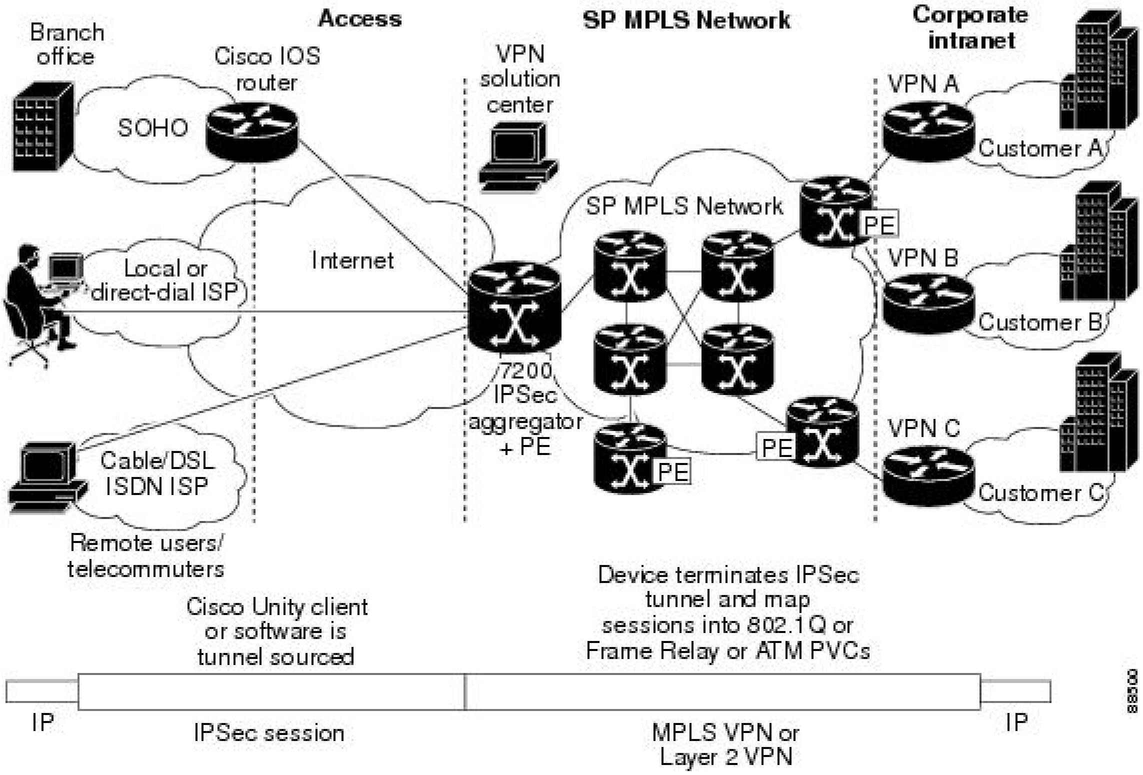
Front Door VRF (FVRF) and Inside VRF (IVRF) are central to understanding the feature.

Each IPsec tunnel is associated with two VRF domains. The outer encapsulated packet belongs to one VRF domain, which we shall call the FVRF, while the inner, protected IP packet belongs to another domain called the IVRF. Another way of stating the same thing is that the local endpoint of the IPsec tunnel belongs to the FVRF while the source and destination addresses of the inside packet belong to the IVRF.

One or more IPsec tunnels can terminate on a single interface. The FVRF of all these tunnels is the same and is set to the VRF that is configured on that interface. The IVRF of these tunnels can be different and depends on the VRF that is defined in the Internet Security Association and Key Management Protocol (ISAKMP) profile that is attached to a crypto map entry.

GUID-640C027D-F465-4B58-93B1-1DC0CDC5A91AE is an illustration of a scenario showing IPsec to MPLS and Layer 2 VPNs.

Figure 1: IPsec to MPLS and Layer 2 VPNs



- Packet Flow into the IPsec Tunnel, page 3
- Packet Flow from the IPsec Tunnel, page 3

Packet Flow into the IPsec Tunnel

- A VPN packet arrives from the Service Provider MPLS backbone network to the PE and is routed through an interface facing the Internet.
- The packet is matched against the Security Policy Database (SPD), and the packet is IPsec encapsulated. The SPD includes the IVRF and the access control list (ACL).
- The IPsec encapsulated packet is then forwarded using the FVRF routing table.

Packet Flow from the IPsec Tunnel

- An IPsec-encapsulated packet arrives at the PE router from the remote IPsec endpoint.
- IPsec performs the Security Association (SA) lookup for the Security Parameter Index (SPI), destination, and protocol.
- The packet is decapsulated using the SA and is associated with IVRF.
- The packet is further forwarded using the IVRF routing table.

How to Configure VRF-Aware IPsec

Configuring Crypto Keyrings

A crypto keyring is a repository of preshared and Rivest, Shamir, and Adelman (RSA) public keys. There can be zero or more keyrings on the Cisco IOS router.

Perform the following optional task to configure a crypto keyring.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto keyring** *keyring-name* [**vrf** *fvrf-name*
4. **description** *string*
5. **pre-shared-key** {**address** *address* [*mask*] | **hostname** *hostname*} **key** *key*
6. **rsa-pubkey** {**address** *address* | **name** *fqdn*} [**encryption** | **signature**]
7. **address** *ip-address*
8. **serial-number** *serial-number*
9. **key-string**
10. **text**
11. **quit**
12. **exit**
13. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. - Enter your password if prompted.
Step 2	**configure terminal**	Enters global configuration mode.

	Command or Action	Purpose
	Example: `Router# configure terminal`	
Step 3	**crypto keyring** *keyring-name* [**vrf** *fvrf-name* **Example:** `Router (config)# crypto keyring VPN1`	Defines a keyring with *keyring-name* as the name of the keyring and enters keyring configuration mode. • (Optional) The **vrf** keyword and *fvrf-name* argument imply that the keyring is bound to Front Door Virtual Routing and Forwarding (FVRF). The key in the keyring is searched if the local endpoint is in FVRF. If **vrf** is not specified, the keyring is bound to the global.
Step 4	**description** *string* **Example:** **Example:** `Router (config-keyring)# description The keys for VPN1`	(Optional) Specifies a one-line description of the keyring.
Step 5	**pre-shared-key** {**address** *address* [*mask*] | **hostname** *hostname*} **key** *key* **Example:** `Router (config-keyring)# pre-shared-key address 10.72.23.11 key VPN1`	(Optional) Defines a preshared key by address or host name.
Step 6	**rsa-pubkey** {**address** *address* | **name** *fqdn*} [**encryption** | **signature**] **Example:** `Router(config-keyring)# rsa-pubkey name host.vpn.com`	(Optional) Defines a Rivest, Shamir, and Adelman (RSA) public key by address or host name and enters rsa-pubkey configuration mode. • By default, the key is used for signature. • The optional **encryption** keyword specifies that the key should be used for encryption. The optional **signature** keyword specifies that the key should be used for signature. By default, the key is used for signature.
Step 7	**address** *ip-address* **Example:** `Router(config-pubkey-key)# address 10.5.5.1`	(Optional) Defines the RSA public key IP address.
Step 8	**serial-number** *serial-number* **Example:** `Router(config-pubkey-key)# serial-number 1000000`	(Optional) Specifies the serial number of the public key. The value is from 0 through infinity.
Step 9	**key-string**	Enters into the text mode in which you define the public key.

	Command or Action	Purpose
	Example: `Router (config-pubkey-key)# key-string`	
Step 10	**text** **Example:** `Router (config-pubkey)# 00302017 4A7D385B 1234EF29 335FC973`	Specifies the public key. **Note**Only one public key may be added in this step.
Step 11	**quit** **Example:** `Router (config-pubkey)# quit`	Quits to the public key configuration mode.
Step 12	**exit** **Example:** `Router (config-pubkey)# exit`	Exits to the keyring configuration mode.
Step 13	**exit** **Example:** `Router(config-keyring)# exit#`	Exits to global configuration mode.

Configuring ISAKMP Profiles

An ISAKMP profile is a repository for IKE Phase 1 and IKE Phase 1.5 configuration for a set of peers. An ISAKMP profile defines items such as keepalive, trustpoints, peer identities, and XAUTH AAA list during the IKE Phase 1 and Phase 1.5 exchange. There can be zero or more ISAKMP profiles on the Cisco IOS router.

Note If traffic from the router to a certification authority (CA) (for authentication, enrollment, or for obtaining a certificate revocation list [CRL]) or to an Lightweight Directory Access Protocol (LDAP) server (for obtaining a CRL) needs to be routed via a VRF, the **vrf**command must be added to the trustpoint. Otherwise, such traffic will use the default routing table.

- If a profile does not specify one or more trustpoints, all trustpoints in the router will be used to attempt to validate the certificate of the peer (Internet Key Exchange (IKE) main mode or signature authentication). If one or more trustpoints are specified, only those trustpoints will be used.

Note 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 will be 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.)

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto isakmp profile** *profile-name*
4. **description** *string*
5. **vrf** *ivrf-name*
6. **keepalive** *seconds* **retry** *retry-seconds*
7. **self-identity** {**address** | **fqdn**| **user-fqdn** *user-fqdn*}
8. **keyring** *keyring-name*
9. **ca trust-point** {*trustpoint-name*}
10. **match identity** {**group** *group-name* | **address** *address* [*mask*] [*fvrf*] | **host** *host-name* | **host domain** *domain-name* | **user** *user-fqdn* | **user domain** *domain-name*}
11. **client configuration address** {**initiate** | **respond**}
12. **client authentication list** *list-name*
13. **isamkp authorization list** *list-name*
14. **initiate mode aggressive**
15. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto isakmp profile** *profile-name*	Defines an Internet Security Association and Key Management Protocol (ISAKMP) profile and enters into isakmp profile configuration mode.

	Command or Action	Purpose
	Example: `Router (config)# crypto isakmp profile vpnprofile`	
Step 4	**description** *string* **Example:** **Example:** `Router (config-isa-prof)# description configuration for VPN profile`	(Optional) Specifies a one-line description of an ISAKMP profile.
Step 5	**vrf** *ivrf-name* **Example:** `Router (config-isa-prof)# vrf VPN1`	(Optional) Maps the IPsec tunnel to a Virtual Routing and Forwarding (VRF) instance. **Note** The VRF also serves as a selector for matching the Security Policy Database (SPD). If the VRF is not specified in the ISAKMP profile, the IVRF of the IPsec tunnel will be the same as its FVRF.
Step 6	**keepalive** *seconds* **retry** *retry-seconds* **Example:** `Router (config-isa-prof)# keepalive 60 retry 5`	(Optional) Allows the gateway to send dead peer detection (DPD) messages to the peer. • If not defined, the gateway uses the globally configured value. • *seconds* --Number of seconds between DPD messages. The range is from 10 to 3600 seconds. • **retry** *retry-seconds* --Number of seconds between retries if the DPD message fails. The range is from 2 to 60 seconds.

Step 7	**self-identity** {**address** | **fqdn**| **user-fqdn** *user-fqdn*}
	Example:
	`Router (config-isa-prof)# self-identity address`
		(Optional) Specifies the identity that the local Internet Key Exchange (IKE) should use to identify itself to the remote peer.
		• If not defined, IKE uses the global configured value.
		• **address** --Uses the IP address of the egress interface.
		• **fqdn--** Uses the fully qualified domain name (FQDN) of the router.
		• **user-fqdn** --Uses the specified value.

Step 8	**keyring** *keyring-name* **Example:** **Example:** `Router (config-isa-prof)# keyring VPN1`	(Optional) Specifies the keyring to use for Phase 1 authentication. • If the keyring is not specified, the global key definitions are used.
Step 9	**ca trust-point** {*trustpoint-name*}	(Optional) Specifies a trustpoint to validate a Rivest, Shamir, and Adelman (RSA) certificate.

	Command or Action	Purpose
	Example: `Router (config-isa-prof)# ca trustpoint VPN1-trustpoint`	• If no trustpoint is specified in the ISAKMP profile, all the trustpoints that are configured on the Cisco IOS router are used to validate the certificate.

Step 10	**match identity** {**group** *group-name* | **address** *address* [*mask*] [*fvrf*] | **host** *host-name* | **host domain** *domain-name* | **user** *user-fqdn* | **user domain** *domain-name*}
	Example:
	`Router (config-isa-prof)# match identity address 10.1.1.1`
		Specifies the client IKE Identity (ID) that is to be matched.
		• **group** *group-name* --Matches the *group-name* with the ID type ID_KEY_ID. It also matches the *group-name* with the Organizational Unit (OU) field of the Distinguished Name (DN).·
		• **address** *address* [*mask*] *fvrf*--Matches the *address*with the ID type ID_IPV4_ADDR. The *mask* argument can be used to specify a range of addresses. The *fvrf* argument specifies that the address is in Front Door Virtual Routing and Forwarding (FVRF)
		• **host** *hostname* --Matches the *hostname* with the ID type ID_FQDN.
		• **host domain** *domain-name* --Matches the *domain-name*to the ID type ID_FQDN whose domain name is the same as the *domain-name*. Use this command to match all the hosts in the domain.
		• **user** *username* --Matches the *username* with the ID type ID_USER_FQDN·
		• **user domain** *domain-name* --Matches the ID type ID_USER_FQDN whose domain name matches the *domain-name*.

Step 11	**client configuration address** {**initiate** | **respond**}
	Example:
	`Router (config-isa-prof)# client configuration address initiate`
		(Optional) Specifies whether to initiate the mode configuration exchange or responds to mode configuration requests.

Step 12	**client authentication list** *list-name* **Example:** `Router (config-isa-prof)# client authentication list xauthlist`	(Optional) Authentication, authorization, and accounting (AAA) to use for authenticating the remote client during the extended authentication (XAUTH) exchange.
Step 13	**isamkp authorization list** *list-name* **Example:** `Router (config-isa-prof)# isakmp authorization list ikessaaalist`	(Optional) Network authorization server for receiving the Phase 1 preshared key and other attribute-value (AV) pairs.
Step 14	**initiate mode aggressive** **Example:** `Router (config-isa-prof)# initiate mode aggressive`	(Optional) Initiates aggressive mode exchange. • If not specified, IKE always initiates Main Mode exchange.
Step 15	**exit**	Exits to global configuration mode.

	Command or Action	Purpose
	Example: `Router (config-isa-prof)# exit`	

- What to Do Next, page 10

What to Do Next

Go to the section “Configuring an ISAKMP Profile on a Crypto Map, page 10.”

Configuring an ISAKMP Profile on a Crypto Map

An ISAKMP profile must be applied to the crypto map. The IVRF on the ISAKMP profile is used as a selector when matching the VPN traffic. If there is no IVRF on the ISAKMP profile, the IVRF will be equal to the FVRF. Perform this required task to configure an ISAKMP profile on a crypto map.

Before configuring an ISAKMP profile on a crypto map, you must first have configured your router for basic IPsec.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto map** *map-name* **isakmp-profile** *isakmp-profile-name*
4. **set isakmp-profile** *profile-name*
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto map** *map-name* **isakmp-profile** *isakmp-profile-name*	(Optional) Specifies the Internet Key Exchange and Key Management Protocol (ISAKMP) profile for the crypto map set and enters crypto map configuration mode. • The ISAKMP profile will be used during IKE exchange.

	Command or Action	Purpose
	Example: `Router (config)# crypto map vpnmap isakmp-profile vpnprofile`	
Step 4	**set isakmp-profile** *profile-name* **Example:** `Router (config-crypto-map)# set isakmp-profile vpnprofile`	(Optional) Specifies the ISAKMP profile to use when the traffic matches the crypto map entry.
Step 5	**exit** **Example:** `Router (config-crypto-map)# exit`	Exits to global configuration mode.

Configuring to Ignore Extended Authentication During IKE Phase 1 Negotiation

To ignore XAUTH during an IKE Phase 1 negotiation, use the **no crypto xauth** command. Use the **no crypto xauth** command if you do not require extended authentication for the Unity clients.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no crypto xauth** *interface*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**no crypto xauth** *interface*	Ignores XAUTH proposals for requests that are destined to the IP address of the interface. By default, Internet Key Exchange (IKE) processes XAUTH proposals.

	Command or Action	Purpose
	Example: `Router(config)# no crypto xauth ethernet0`	

Verifying VRF-Aware IPsec

To verify your VRF-Aware IPsec configurations, use the following **show** commands. These **show** commands allow you to list configuration information and security associations (SAs):

SUMMARY STEPS

1. **enable**
2. **show crypto ipsec sa** [**map** *map-name* | **address** | **identity** | **interface** *interface* | **peer** [**vrf** *fvrf-name*] **address** | **vrf** *ivrf-name*] [**detail**]
3. **show crypto isakmp key**
4. **show crypto isakmp profile**
5. **show crypto key pubkey-chain rsa**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show crypto ipsec sa** [**map** *map-name* | **address** | **identity** | **interface** *interface* | **peer** [**vrf** *fvrf-name*] **address** | **vrf** *ivrf-name*] [**detail**] **Example:** `Router# show crypto ipsec sa vrf vpn1`	Allows you to view the settings used by current security associations (SAs).
Step 3	**show crypto isakmp key** **Example:** `Router# show crypto isakmp key`	Lists all the keyrings and their preshared keys. • Use this command to verify your crypto keyring configuration.
Step 4	**show crypto isakmp profile** **Example:** `Router# show crypto isakmp profile`	Lists all ISAKMP profiles and their configurations.
Step 5	**show crypto key pubkey-chain rsa**	Views the Rivest, Shamir, and Adelman (RSA) public keys of the peer that are stored on your router.

	Command or Action	Purpose
	Example: `Router# show crypto key pubkey-chain rsa`	• The output is extended to show the keyring to which the public key belongs.

Clearing Security Associations

The following **clear** commands allow you to clear SAs.

SUMMARY STEPS

1. **enable**
2. **clear crypto sa** [**counters** | **map** *map-name* | **peer**[**vrf** *fvrf-name*] *address* | **spi** *address* {**ah** | **esp**} *spi* | **vrf** *ivrf-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear crypto sa** [**counters** | **map** *map-name* | **peer**[**vrf** *fvrf-name*] *address* | **spi** *address* {**ah** | **esp**} *spi* | **vrf** *ivrf-name*] **Example:** `Router# clear crypto sa vrf VPN1`	Clears the IPsec security associations (SAs).

Troubleshooting VRF-Aware IPsec

To troubleshoot VRF-Aware IPsec, use the following **debug**commands:

SUMMARY STEPS

1. **enable**
2. **debug crypto ipsec**
3. **debug crypto isakmp**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	Example: `Router> enable`	
Step 2	**debug crypto ipsec** **Example:** `Router# debug crypto ipsec`	Displays IP security (IPsec) events.
Step 3	**debug crypto isakmp** **Example:** `Router(config)# debug crypto isakmp`	Displays messages about Internet Key Exchange (IKE) events.

- Debug Examples for VRF-Aware IPsec, page 14

Debug Examples for VRF-Aware IPsec

The following sample debug outputs are for a VRF-aware IPsec configuration:

IPsec PE

```
Router# debug crypto ipsec
Crypto IPSEC debugging is on
IPSEC-PE#debug crypto isakmp
Crypto ISAKMP debugging is on
IPSEC-PE#debug crypto isakmp d
04:31:28: ISAKMP (0:12): purging SA., sa=6482B354, delme=6482B354
04:31:28: ISAKMP: Unlocking IKE struct 0x63C142F8 for declare_sa_dead(), count 0
IPSEC-PE#debug crypto isakmp detail
Crypto ISAKMP internals debugging is on
IPSEC-PE#
IPSEC-PE#
IPSEC-PE#
04:32:07: ISAKMP: Deleting peer node by peer_reap for 10.1.1.1: 63C142F8
04:32:55: ISAKMP cookie gen for src 172.16.1.1 dst 10.1.1.1
04:32:55: ISAKMP cookie 3123100B DC887D4E
04:32:55: ISAKMP cookie gen for src 10.1.1.1 dst 172.68.1.1
04:32:55: ISAKMP cookie AA8F7B41 49A60E88
04:32:55: ISAKMP cookie gen for src 172.16.1.1 dst 10.1.1.1
04:32:55: ISAKMP cookie 3123100B DBC8E125
04:32:55: ISAKMP cookie gen for src 10.1.1.1 dst 172.16.1.1
04:32:55: ISAKMP cookie AA8F7B41 B4BDB5B7
04:32:55: ISAKMP (0:0): received packet from 10.1.1.1 dport 500 sport 500 Global (N) NEW
SA
04:32:55: ISAKMP: local port 500, remote port 500
04:32:55: ISAKMP: hash from 729FA94 for 619 bytes
04:32:55: ISAKMP: Packet hash:
64218CC0:                   B91E2C70 095A1346          9.,p.Z.F
64218CD0: 0EDB4CA6 8A46784F B314FD3B 00        .[L&.FxO.};.
04:32:55: ISAKMP cookie gen for src 10.1.1.1 dst 172.18.1.1
04:32:55: ISAKMP cookie AA8F7B41 F7ACF384
04:32:55: ISAKMP cookie gen for src 10.1.1.1 dst 172.18.1.1
04:32:55: ISAKMP cookie AA8F7B41 0C07C670
04:32:55: ISAKMP: insert sa successfully sa = 6482B354
04:32:55: ISAKMP (0:13): processing SA payload. message ID = 0
04:32:55: ISAKMP (0:13): processing ID payload. message ID = 0
```

```
04:32:55: ISAKMP (0:13): peer matches vpn2-ra profile
04:32:55: ISAKMP: Looking for a matching key for 10.1.1.1 in default
04:32:55: ISAKMP: Created a peer struct for 10.1.1.1, peer port 500
04:32:55: ISAKMP: Locking peer struct 0x640BBB18, IKE refcount 1 for
crypto_ikmp_config_initialize_sa
04:32:55: ISAKMP (0:13): Setting client config settings 648252B0
04:32:55: ISAKMP (0:13): (Re)Setting client xauth list  and state
04:32:55: ISAKMP (0:13): processing vendor id payload
04:32:55: ISAKMP (0:13): vendor ID seems Unity/DPD but major 157 mismatch
04:32:55: ISAKMP (0:13): vendor ID is NAT-T v3
04:32:55: ISAKMP (0:13): processing vendor id payload
04:32:55: ISAKMP (0:13): vendor ID seems Unity/DPD but major 123 mismatch
04:32:55: ISAKMP (0:13): vendor ID is NAT-T v2
04:32:55: ISAKMP (0:13) Authentication by xauth preshared
04:32:55: ISAKMP (0:13): Checking ISAKMP transform 1 against priority 1 policy
04:32:55: ISAKMP:      encryption 3DES-CBC
04:32:55: ISAKMP:      hash SHA
04:32:55: ISAKMP:      default group 2
04:32:55: ISAKMP:      auth XAUTHInitPreShared
04:32:55: ISAKMP:      life type in seconds
04:32:55: ISAKMP:      life duration (VPI) of  0x0 0x20 0xC4 0x9B
04:32:55: ISAKMP (0:13): atts are acceptable. Next payload is 3
04:32:55: ISAKMP (0:13): processing vendor id payload
04:32:55: ISAKMP (0:13): vendor ID seems Unity/DPD but major 157 mismatch
04:32:55: ISAKMP (0:13): vendor ID is NAT-T v3
04:32:55: ISAKMP (0:13): processing vendor id payload
04:32:55: ISAKMP (0:13): vendor ID seems Unity/DPD but major 123 mismatch
04:32:55: ISAKMP (0:13): vendor ID is NAT-T v2
04:32:55: ISAKMP (0:13): processing KE payload. message ID = 0
04:32:55: ISAKMP (0:13): processing NONCE payload. message ID = 0
04:32:55: ISAKMP (0:13): processing vendor id payload
04:32:55: ISAKMP (0:13): vendor ID is DPD
04:32:55: ISAKMP (0:13): processing vendor id payload
04:32:55: ISAKMP (0:13): vendor ID seems Unity/DPD but major 175 mismatch
04:32:55: ISAKMP (0:13): vendor ID is XAUTH
04:32:55: ISAKMP (0:13): processing vendor id payload
04:32:55: ISAKMP (0:13): claimed IOS but failed authentication
04:32:55: ISAKMP (0:13): processing vendor id payload
04:32:55: ISAKMP (0:13): vendor ID is Unity
04:32:55: ISAKMP (0:13): Input = IKE_MESG_FROM_PEER, IKE_AM_EXCH
04:32:55: ISAKMP (0:13): Old State = IKE_READY  New State = IKE_R_AM_AAA_AWAIT
04:32:55: ISAKMP cookie gen for src 11.1.1.1 dst 172.16.1.1
04:32:55: ISAKMP cookie AA8F7B41 7AE6E1DF
04:32:55: ISAKMP:        isadb_post_process_list: crawler: 4 AA 31 (6482B354)
04:32:55:       crawler my_cookie AA8F7B41 F7ACF384
04:32:55:        crawler his_cookie E46E088D F227FE4D
04:32:55: ISAKMP: got callback 1
04:32:55: ISAKMP (0:13): SKEYID state generated
04:32:55: ISAKMP: Unity/DPD ID: vendor_id_payload:
        next: 0xD, reserved: 0x0, len 0x14
04:32:55: ISAKMP: Unity/DPD ID payload dump:
63E66D70:                            0D000014              ....
63E66D80: 12F5F28C 457168A9 702D9FE2 74CC0100  .ur.Eqh)p-.btL..
63E66D90: 00                                   .
04:32:55: ISAKMP: Unity/DPD ID: vendor_id_payload:
        next: 0xD, reserved: 0x0, len 0x14
04:32:55: ISAKMP: Unity/DPD ID payload dump:
63E66D90: 0D000014 AFCAD713 68A1F1C9 6B8696FC  ..../JW.h!qIk..|
63E66DA0: 77570100 00                          wW...
04:32:55: ISAKMP (0:13): constructed NAT-T vendor-03 ID
04:32:55: ISAKMP (0:13): SA is doing pre-shared key authentication plus XAUTH using id
type ID_IPV4_ADDR
04:32:55: ISAKMP (13): ID payload
        next-payload : 10
        type         : 1
        addr         : 172.16.1.1
        protocol     : 17
        port         : 0
        length       : 8
04:32:55: ISAKMP (13): Total payload length: 12
04:32:55: ISAKMP (0:13): constructed HIS NAT-D
04:32:55: ISAKMP (0:13): constructed MINE NAT-D
04:32:55: ISAKMP (0:13): sending packet to 10.1.1.1 my_port 500 peer_port 500 (R)
```

```
AG_INIT_EXCH
04:32:55: ISAKMP (0:13): Input = IKE_MESG_FROM_AAA, PRESHARED_KEY_REPLY
04:32:55: ISAKMP (0:13): Old State = IKE_R_AM_AAA_AWAIT  New State = IKE_R_AM2
04:32:55: ISAKMP cookie gen for src 172.16.1.1 dst 10.1.1.1
04:32:55: ISAKMP cookie 3123100B D99DA70D
04:32:55: ISAKMP cookie gen for src 10.1.1.1 dst 172.16.1.1
04:32:55: ISAKMP cookie AA8F7B41 9C69F917
04:32:55: ISAKMP:        isadb_post_process_list: crawler: 5 21FF 1 (6482B354)
04:32:55:       crawler my_cookie AA8F7B41 F7ACF384
04:32:55:        crawler his_cookie E46E088D F227FE4D
04:32:55: ISAKMP cookie gen for src 172.16.1.1 dst 10.1.1.1
04:32:55: ISAKMP cookie 3123100B 00583224
04:32:55: ISAKMP cookie gen for src 10.1.1.1 dst 172.16.1.1
04:32:55: ISAKMP cookie AA8F7B41 C1B006EE
04:32:55: ISAKMP:        isadb_post_process_list: crawler: 5 21FF 1 (6482B354)
04:32:55:       crawler my_cookie AA8F7B41 F7ACF384
04:32:55:        crawler his_cookie E46E088D F227FE4D
04:32:55: ISAKMP (0:13): received packet from 10.1.1.1 dport 500 sport 500 Global (R)
AG_INIT_EXCH
04:32:55: ISAKMP: hash from 7003A34 for 132 bytes
04:32:55: ISAKMP: Packet hash:
64218CC0:                   D1202D99 2BB49D38          Q -.+4.8
64218CD0: B8FBB1BE 7CDC67D7 4E26126C 63        8{1>|\gWN&.lc
04:32:55: ISAKMP (0:13): processing HASH payload. message ID = 0
04:32:55: ISAKMP:received payload type 17
04:32:55: ISAKMP (0:13): Detected NAT-D payload
04:32:55: ISAKMP (0:13): recalc my hash for NAT-D
04:32:55: ISAKMP (0:13): NAT match MINE hash
04:32:55: ISAKMP:received payload type 17
04:32:55: ISAKMP (0:13): Detected NAT-D payload
04:32:55: ISAKMP (0:13): recalc his hash for NAT-D
04:32:55: ISAKMP (0:13): NAT match HIS hash
04:32:55: ISAKMP (0:13): processing NOTIFY INITIAL_CONTACT protocol 1
        spi 0, message ID = 0, sa = 6482B354
04:32:55: ISAKMP (0:13): Process initial contact,
bring down existing phase 1 and 2 SA's with local 172.16.1.1 remote 10.1.1.1 remote port
500
04:32:55: ISAKMP (0:13): returning IP addr to the address pool
04:32:55: ISAKMP cookie gen for src 10.1.1.1 dst 172.16.1.1
04:32:55: ISAKMP cookie AA8F7B41 05D315C5
04:32:55: ISAKMP cookie gen for src 172.16.1.1 dst 10.1.1.1
04:32:55: ISAKMP cookie 3123100B 041A85A6
04:32:55: ISAKMP (0:13): SA has been authenticated with 10.1.1.1
04:32:55: ISAKMP: Trying to insert a peer 172.16.1.1/10.1.1.1/500/,  and inserted
successfully.
04:32:55: ISAKMP: set new node -803402627 to CONF_XAUTH
04:32:55: IPSEC(key_engine): got a queue event...
04:32:55: ISAKMP (0:13): sending packet to 10.1.1.1 my_port 500 peer_port 500 (R)
QM_IDLE
04:32:55: ISAKMP (0:13): purging node -803402627
04:32:55: ISAKMP: Sending phase 1 responder lifetime 86400
04:32:55: ISAKMP (0:13): Input = IKE_MESG_FROM_PEER, IKE_AM_EXCH
04:32:55: ISAKMP (0:13): Old State = IKE_R_AM2  New State = IKE_P1_COMPLETE
04:32:55: ISAKMP cookie gen for src 10.1.1.1 dst 172.168.1.1
04:32:55: ISAKMP cookie AA8F7B41 25EEF256
04:32:55: ISAKMP:        isadb_post_process_list: crawler: 9 27FF 2 (6482B354)
04:32:55:       crawler my_cookie AA8F7B41 F7ACF384
04:32:55:        crawler his_cookie E46E088D F227FE4D
04:32:55: ISAKMP (0:13): Need XAUTH
04:32:55: ISAKMP (0:13): Input = IKE_MESG_INTERNAL, IKE_PHASE1_COMPLETE
04:32:55: ISAKMP (0:13): Old State = IKE_P1_COMPLETE  New State =
IKE_XAUTH_AAA_START_LOGIN_AWAIT
04:32:55: ISAKMP cookie gen for src 10.1.1.1 dst 172.16.1.1
04:32:55: ISAKMP cookie AA8F7B41 2CCFA491
04:32:55: ISAKMP:        isadb_post_process_list: crawler: B 27FF 12 (6482B354)
04:32:55:       crawler my_cookie AA8F7B41 F7ACF384
04:32:55:        crawler his_cookie E46E088D F227FE4D
04:32:55: ISAKMP: got callback 1
04:32:55: ISAKMP: set new node -1447732198 to CONF_XAUTH
04:32:55: ISAKMP/xauth: request attribute XAUTH_USER_NAME_V2
04:32:55: ISAKMP/xauth: request attribute XAUTH_USER_PASSWORD_V2
04:32:55: ISAKMP (0:13): initiating peer config to 10.1.1.1. ID = -1447732198
04:32:55: ISAKMP (0:13): sending packet to 10.1.1.1 my_port 500 peer_port 500 (R)
CONF_XAUTH
04:32:55: ISAKMP (0:13): Input = IKE_MESG_FROM_AAA, IKE_AAA_START_LOGIN
04:32:55: ISAKMP (0:13): Old State = IKE_XAUTH_AAA_START_LOGIN_AWAIT  New State =
IKE_XAUTH_REQ_SENT
04:33:00: ISAKMP (0:13): retransmitting phase 2 CONF_XAUTH    -1447732198 ...
04:33:00: ISAKMP (0:13): incrementing error counter on sa: retransmit phase 2
04:33:00: ISAKMP (0:13): incrementing error counter on sa: retransmit phase 2
04:33:00: ISAKMP (0:13): retransmitting phase 2 -1447732198 CONF_XAUTH
04:33:00: ISAKMP (0:13): sending packet to 10.1.1.1 my_port 500 peer_port 500 (R)
CONF_XAUTH
04:33:03: ISAKMP cookie gen for src 172.18.1.1 dst 10.1.1.1
04:33:03: ISAKMP cookie 3123100B 124D4618
04:33:03: ISAKMP cookie gen for src 10.1.1.1 dst 172.18.1.1
04:33:03: ISAKMP cookie AA8F7B41 B0C91917
04:33:03: ISAKMP:        isadb_post_process_list: crawler: B 27FF 2 (6482B354)
04:33:03:       crawler my_cookie AA8F7B41 F7ACF384
04:33:03:        crawler his_cookie E46E088D F227FE4D
04:33:03: ISAKMP cookie gen for src 172.18.1.1 dst 10.1.1.1
04:33:03: ISAKMP cookie 3123100B 0E294692
04:33:03: ISAKMP cookie gen for src 10.1.1.1 dst 172.18.1.1
04:33:03: ISAKMP cookie AA8F7B41 091A7695
04:33:03: ISAKMP:        isadb_post_process_list: crawler: B 27FF 2 (6482B354)
04:33:03:       crawler my_cookie AA8F7B41 F7ACF384
04:33:03:        crawler his_cookie E46E088D F227FE4D
04:33:03: ISAKMP (0:13): received packet from 10.1.1.1 dport 500 sport 500 Global (R)
CONF_XAUTH
04:33:03: ISAKMP: hash from 7292D74 for 92 bytes
04:33:03: ISAKMP: Packet hash:
64218CC0:                   84A1AF24 5D92B116          .!/$].1.
64218CD0: FC2C6252 A472C5F8 152AC860 63        |,bR$rEx.*H`c
04:33:03: ISAKMP (0:13): processing transaction payload from 11.1.1.1. message ID =
-1447732198
04:33:03: ISAKMP: Config payload REPLY
04:33:03: ISAKMP/xauth: reply attribute XAUTH_USER_NAME_V2
04:33:03: ISAKMP/xauth: reply attribute XAUTH_USER_PASSWORD_V2
04:33:03: ISAKMP (0:13): deleting node -1447732198 error FALSE reason "done with xauth
request/reply exchange"
04:33:03: ISAKMP (0:13): Input = IKE_MESG_FROM_PEER, IKE_CFG_REPLY
04:33:03: ISAKMP (0:13): Old State = IKE_XAUTH_REQ_SENT  New State =
IKE_XAUTH_AAA_CONT_LOGIN_AWAIT
04:33:03: ISAKMP cookie gen for src 10.1.1.1 dst 172.18.1.1
04:33:03: ISAKMP cookie AA8F7B41 A1B3E684
04:33:03: ISAKMP:        isadb_post_process_list: crawler: B 27FF 12 (6482B354)
04:33:03:       crawler my_cookie AA8F7B41 F7ACF384
04:33:03:        crawler his_cookie E46E088D F227FE4D
04:33:03: ISAKMP: got callback 1
04:33:03: ISAKMP: set new node 524716665 to CONF_XAUTH
04:33:03: ISAKMP (0:13): initiating peer config to 10.1.1.1. ID = 524716665
04:33:03: ISAKMP (0:13): sending packet to 10.1.1.1 my_port 500 peer_port 500 (R)
CONF_XAUTH
04:33:03: ISAKMP (0:13): Input = IKE_MESG_FROM_AAA, IKE_AAA_CONT_LOGIN
04:33:03: ISAKMP (0:13): Old State = IKE_XAUTH_AAA_CONT_LOGIN_AWAIT  New State =
IKE_XAUTH_SET_SENT
004:33:03: ISAKMP cookie gen for src 172.18.1.1 dst 10.1.1.1
04:33:03: ISAKMP cookie 3123100B 5C83A09D
04:33:03: ISAKMP cookie gen for src 10.1.1.1 dst 172.18.1.1
04:33:03: ISAKMP cookie AA8F7B41 2BEBEFD4
04:33:03: ISAKMP:        isadb_post_process_list: crawler: B 27FF 2 (6482B354)
04:33:03:       crawler my_cookie AA8F7B41 F7ACF384
04:33:03:        crawler his_cookie E46E088D F227FE4D
04:33:03: ISAKMP cookie gen for src 172.18.1.1 dst 10.1.1.1
04:33:03: ISAKMP cookie 3123100B DA00A46B
04:33:03: ISAKMP cookie gen for src 10.1.1.1 dst 172.18.1.1
04:33:03: ISAKMP cookie AA8F7B41 FDD27773
04:33:03: ISAKMP:        isadb_post_process_list: crawler: B 27FF 2 (6482B354)
04:33:03:       crawler my_cookie AA8F7B41 F7ACF384
04:33:03:        crawler his_cookie E46E088D F227FE4D
04:33:03: ISAKMP (0:13): received packet from 10.1.1.1 dport 500 sport 500 Global (R)
CONF_XAUTH
04:33:03: ISAKMP: hash from 7292A34 for 68 bytes
04:33:03: ISAKMP: Packet hash:
64218CC0:                   5034B99E B8BA531F          P49.8:S.
64218CD0: 6267B8BD F3006989 DC118796 63        bg8=s.i.\...c
```

```
04:33:03: ISAKMP (0:13): processing transaction payload from 11.1.1.1. message ID =
524716665
04:33:03: ISAKMP: Config payload ACK
04:33:03: ISAKMP (0:13):        XAUTH ACK Processed
04:33:03: ISAKMP (0:13): deleting node 524716665 error FALSE reason "done with
transaction"
04:33:03: ISAKMP (0:13): Input = IKE_MESG_FROM_PEER, IKE_CFG_ACK
04:33:03: ISAKMP (0:13): Old State = IKE_XAUTH_SET_SENT  New State = IKE_P1_COMPLETE
04:33:03: ISAKMP cookie gen for src 10.1.1.1 dst 172.18.1.1
04:33:03: ISAKMP cookie AA8F7B41 E0BB50E9
04:33:03: ISAKMP:        isadb_post_process_list: crawler: 9 27FF 2 (6482B354)
04:33:03:       crawler my_cookie AA8F7B41 F7ACF384
04:33:03:        crawler his_cookie E46E088D F227FE4D
04:33:03: ISAKMP (0:13): Input = IKE_MESG_INTERNAL, IKE_PHASE1_COMPLETE
04:33:03: ISAKMP (0:13): Old State = IKE_P1_COMPLETE  New State = IKE_P1_COMPLETE
04:33:03: ISAKMP cookie gen for src 172.18.1.1 dst 10.1.1.1
04:33:03: ISAKMP cookie 3123100B 7794EF6E
04:33:03: ISAKMP cookie gen for src 10.1.1.1 dst 172.18.1.1
04:33:03: ISAKMP cookie AA8F7B41 C035AAE5
04:33:03: ISAKMP:        isadb_post_process_list: crawler: 9 27FF 2 (6482B354)
04:33:03:       crawler my_cookie AA8F7B41 F7ACF384
04:33:03:        crawler his_cookie E46E088D F227FE4D
04:33:03: ISAKMP cookie gen for src 172.18.1.1 dst 10.1.1.1
04:33:03: ISAKMP cookie 3123100B F1FCC25A
04:33:03: ISAKMP cookie gen for src 10.1.1.1 dst 172.18.1.1
04:33:03: ISAKMP cookie AA8F7B41 31744F44
04:33:03: ISAKMP:        isadb_post_process_list: crawler: 9 27FF 2 (6482B354)
04:33:03:       crawler my_cookie AA8F7B41 F7ACF384
04:33:03:        crawler his_cookie E46E088D F207FE4D
04:33:03: ISAKMP (0:13): received packet from 10.1.1.1 dport 500 sport 500 Global (R)
QM_IDLE
04:33:03: ISAKMP: set new node -1639992295 to QM_IDLE
04:33:03: ISAKMP: hash from 7293A74 for 100 bytes
04:33:03: ISAKMP: Packet hash:
64218CC0:                   9D7DF4DF FE3A6403          .}t_~:d.
64218CD0: 3F1D1C59 C5D138CE 50289B79 07        ?..YEQ8NP(.y.
04:33:03: ISAKMP (0:13): processing transaction payload from 10.1.1.1. message ID =
-1639992295
04:33:03: ISAKMP: Config payload REQUEST
04:33:03: ISAKMP (0:13): checking request:
04:33:03: ISAKMP:    IP4_ADDRESS
04:33:03: ISAKMP:    IP4_NETMASK
04:33:03: ISAKMP:    IP4_DNS
04:33:03: ISAKMP:    IP4_DNS
04:33:03: ISAKMP:    IP4_NBNS
04:33:03: ISAKMP:    IP4_NBNS
04:33:03: ISAKMP:    SPLIT_INCLUDE
04:33:03: ISAKMP:    DEFAULT_DOMAIN
04:33:03: ISAKMP (0:13): Input = IKE_MESG_FROM_PEER, IKE_CFG_REQUEST
04:33:03: ISAKMP (0:13): Old State = IKE_P1_COMPLETE  New State =
IKE_CONFIG_AUTHOR_AAA_AWAIT
04:33:03: ISAKMP cookie gen for src 10.1.1.1 dst 172.18.1.1
04:33:03: ISAKMP cookie AA8F7B41 B02E0D67
04:33:03: ISAKMP:        isadb_post_process_list: crawler: C 27FF 12 (6482B354)
04:33:03:       crawler my_cookie AA8F7B41 F7ACF384
04:33:03:        crawler his_cookie E46E088D F227FE4D
04:33:03: ISAKMP: got callback 1
04:33:03: ISAKMP (0:13): attributes sent in message:
04:33:03:         Address: 10.2.0.0
04:33:03: ISAKMP (0:13): allocating address 10.4.1.4
04:33:03: ISAKMP: Sending private address: 10.4.1.4
04:33:03: ISAKMP: Sending DEFAULT_DOMAIN default domain name: vpn2.com
04:33:03: ISAKMP (0:13): responding to peer config from 10.1.1.1. ID = -1639992295
04:33:03: ISAKMP (0:13): sending packet to 10.1.1.1 my_port 500 peer_port 500 (R)
CONF_ADDR
04:33:03: ISAKMP (0:13): deleting node -1639992295 error FALSE reason ""
04:33:03: ISAKMP (0:13): Input = IKE_MESG_FROM_AAA, IKE_AAA_GROUP_ATTR
04:33:03: ISAKMP (0:13): Old State = IKE_CONFIG_AUTHOR_AAA_AWAIT  New State =
IKE_P1_COMPLETE
04:33:03: ISAKMP cookie gen for src 172.18.1.1 dst 10.1.1.1
04:33:03: ISAKMP cookie 3123100B 881D5411
04:33:03: ISAKMP cookie gen for src 11.1.1.1 dst 172.18.1.1
04:33:03: ISAKMP cookie AA8F7B41 6FD82541
```

```
04:33:03: ISAKMP:        isadb_post_process_list: crawler: 9 27FF 2 (6482B354)
04:33:03:       crawler my_cookie AA8F7B41 F7ACF384
04:33:03:        crawler his_cookie E46E088D F227FE4D
04:33:03: ISAKMP cookie gen for src 172.18.1.1 dst 10.1.1.1
04:33:03: ISAKMP cookie 3123100B 8A94C1BE
04:33:03: ISAKMP cookie gen for src 10.1.1.1 dst 172.18.1.1
04:33:03: ISAKMP cookie AA8F7B41 F3BA766D
04:33:03: ISAKMP:        isadb_post_process_list: crawler: 9 27FF 2 (6482B354)
04:33:03:       crawler my_cookie AA8F7B41 F7ACF384
04:33:03:        crawler his_cookie E46E088D F207FE4D
04:33:03: ISAKMP (0:13): received packet from 10.1.1.1 dport 500 sport 500 Global (R)
QM_IDLE
04:33:03: ISAKMP: set new node 17011691 to QM_IDLE
04:33:03: ISAKMP: hash from 70029F4 for 540 bytes
04:33:03: ISAKMP: Packet hash:
64218CC0:                   AFBA30B2 55F5BC2D           /:02Uu<-
64218CD0: 3A86B1C9 00D2F5BA 77BF5589 07        :.1I.Ru:w?U..
04:33:03: ISAKMP (0:13): processing HASH payload. message ID = 17011691
04:33:03: ISAKMP (0:13): processing SA payload. message ID = 17011691
04:33:03: ISAKMP (0:13): Checking IPSec proposal 1
04:33:03: ISAKMP: transform 1, ESP_3DES
04:33:03: ISAKMP:   attributes in transform:
04:33:03: ISAKMP:      encaps is 1
04:33:03: ISAKMP:      SA life type in seconds
04:33:03: ISAKMP:      SA life duration (VPI) of  0x0 0x20 0xC4 0x9B
04:33:03: ISAKMP:      SA life type in kilobytes
04:33:03: ISAKMP:      SA life duration (VPI) of  0x0 0x46 0x50 0x0
04:33:03: ISAKMP:      authenticator is HMAC-SHA
04:33:03: ISAKMP (0:13): atts are acceptable.
04:33:03: IPSEC(validate_proposal_request): proposal part #1,
  (key eng. msg.) INBOUND local= 172.18.1.1, remote= 10.1.1.1,
    local_proxy= 0.0.0.0/0.0.0.0/0/0 (type=4),
    remote_proxy= 10.4.1.4/255.255.255.255/0/0 (type=1),
    protocol= ESP, transform= esp-3des esp-sha-hmac ,
    lifedur= 0s and 0kb,
    spi= 0x0(0), conn_id= 0, keysize= 0, flags= 0x2
04:33:03: IPSEC(kei_proxy): head = ra, map->ivrf = vpn1, kei->ivrf = vpn2
04:33:03: IPSEC(kei_proxy): head = ra, map->ivrf = vpn2, kei->ivrf = vpn2
04:33:03: IPSEC(validate_transform_proposal): transform proposal not supported for
identity:
    {esp-3des esp-sha-hmac }
04:33:03: ISAKMP (0:13): IPSec policy invalidated proposal
04:33:03: ISAKMP (0:13): Checking IPSec proposal 2
04:33:03: ISAKMP: transform 1, ESP_3DES
04:33:03: ISAKMP:   attributes in transform:
04:33:03: ISAKMP:      encaps is 1
04:33:03: ISAKMP:      SA life type in seconds
04:33:03: ISAKMP:      SA life duration (VPI) of  0x0 0x20 0xC4 0x9B
04:33:03: ISAKMP:      SA life type in kilobytes
04:33:03: ISAKMP:      SA life duration (VPI) of  0x0 0x46 0x50 0x0
04:33:03: ISAKMP:      authenticator is HMAC-MD5
04:33:03: ISAKMP (0:13): atts are acceptable.
04:33:03: IPSEC(validate_proposal_request): proposal part #1,
  (key eng. msg.) INBOUND local= 172.18.1.1, remote= 10.1.1.1,
    local_proxy= 0.0.0.0/0.0.0.0/0/0 (type=4),
    remote_proxy= 10.4.1.4/255.255.255.255/0/0 (type=1),
    protocol= ESP, transform= esp-3des esp-md5-hmac ,
    lifedur= 0s and 0kb,
    spi= 0x0(0), conn_id= 0, keysize= 0, flags= 0x2
04:33:03: IPSEC(kei_proxy): head = ra, map->ivrf = vpn1, kei->ivrf = vpn2
04:33:03: IPSEC(kei_proxy): head = ra, map->ivrf = vpn2, kei->ivrf = vpn2
04:33:03: ISAKMP (0:13): processing NONCE payload. message ID = 17011691
04:33:03: ISAKMP (0:13): processing ID payload. message ID = 17011691
04:33:03: ISAKMP (0:13): processing ID payload. message ID = 17011691
04:33:03: ISAKMP (0:13): asking for 1 spis from ipsec
04:33:03: ISAKMP (0:13): Node 17011691, Input = IKE_MESG_FROM_PEER, IKE_QM_EXCH
04:33:03: ISAKMP (0:13): Old State = IKE_QM_READY  New State = IKE_QM_SPI_STARVE
04:33:03: IPSEC(key_engine): got a queue event...
04:33:03: IPSEC(spi_response): getting spi 2749516541 for SA
        from 172.18.1.1     to 10.1.1.1         for prot 3
04:33:03: ISAKMP: received ke message (2/1)
04:33:04: ISAKMP (13): ID payload
        next-payload : 5
        type         : 1
        addr         : 10.4.1.4
        protocol     : 0
        port         : 0
04:33:04: ISAKMP (13): ID payload
        next-payload : 11
        type         : 4
        addr         : 0.0.0.0
        protocol     : 0
        port         : 0
04:33:04: ISAKMP (0:13): sending packet to 10.1.1.1 my_port 500 peer_port 500 (R)
QM_IDLE
04:33:04: ISAKMP (0:13): Node 17011691, Input = IKE_MESG_FROM_IPSEC, IKE_SPI_REPLY
04:33:04: ISAKMP (0:13): Old State = IKE_QM_SPI_STARVE  New State = IKE_QM_R_QM2
04:33:04: ISAKMP cookie gen for src 172.18.1.1 dst 10.1.1.1
04:33:04: ISAKMP cookie 3123100B 93DE46D2
04:33:04: ISAKMP cookie gen for src 10.1.1.1 dst 172.18.1.1
04:33:04: ISAKMP cookie AA8F7B41 088A0A16
04:33:04: ISAKMP:        isadb_post_process_list: crawler: 9 27FF 2 (6482B354)
04:33:04:       crawler my_cookie AA8F7B41 F7ACF384
04:33:04:        crawler his_cookie E46E088D F227FE4D
04:33:04: ISAKMP cookie gen for src 172.18.1.1 dst 10.1.1.1
04:33:04: ISAKMP cookie 3123100B A8F23F73
04:33:04: ISAKMP cookie gen for src 10.1.1.1 dst 172.18.1.1
04:33:04: ISAKMP cookie AA8F7B41 93D8D879
04:33:04: ISAKMP:        isadb_post_process_list: crawler: 9 27FF 2 (6482B354)
04:33:04:       crawler my_cookie AA8F7B41 F7ACF384
04:33:04:        crawler his_cookie E46E088D F227FE4D
04:33:04: ISAKMP (0:13): received packet from 10.1.1.1 dport 500 sport 500 Global (R)
QM_IDLE
04:33:04: ISAKMP: hash from 7290DB4 for 60 bytes
04:33:04: ISAKMP: Packet hash:
64218CC0:                   4BB45A92 7181A2F8          K4Z.q."x
64218CD0: 73CC12F8 091875C0 054F77CD 63       sL.x..u@.OwMc
04:33:04: ISAKMP: Locking peer struct 0x640BBB18, IPSEC refcount 1 for for stuff_ke
04:33:04: ISAKMP (0:13): Creating IPSec SAs
04:33:04:         inbound SA from 10.1.1.1 to 172.18.1.1 (f/i)  0/ 2
        (proxy 10.4.1.4 to 0.0.0.0)
04:33:04:         has spi 0xA3E24AFD and conn_id 5127 and flags 2
04:33:04:         lifetime of 2147483 seconds
04:33:04:         lifetime of 4608000 kilobytes
04:33:04:         has client flags 0x0
04:33:04:         outbound SA from 172.18.1.1     to 10.1.1.1        (f/i)  0/ 2 (proxy
0.0.0.0         to 10.4.1.4       )
04:33:04:         has spi 1343294712 and conn_id 5128 and flags A
04:33:04:         lifetime of 2147483 seconds
04:33:04:         lifetime of 4608000 kilobytes
04:33:04:         has client flags 0x0
04:33:04: ISAKMP (0:13): deleting node 17011691 error FALSE reason "quick mode done
(await)"
04:33:04: ISAKMP (0:13): Node 17011691, Input = IKE_MESG_FROM_PEER, IKE_QM_EXCH
04:33:04: ISAKMP (0:13): Old State = IKE_QM_R_QM2  New State = IKE_QM_PHASE2_COMPLETE
04:33:04: IPSEC(key_engine): got a queue event...
04:33:04: IPSEC(initialize_sas): ,
  (key eng. msg.) INBOUND local= 172.18.1.1, remote= 10.1.1.1,
    local_proxy= 0.0.0.0/0.0.0.0/0/0 (type=4),
    remote_proxy= 10.4.1.4/0.0.0.0/0/0 (type=1),
    protocol= ESP, transform= esp-3des esp-md5-hmac ,
    lifedur= 2147483s and 4608000kb,
    spi= 0xA3E24AFD(2749516541), conn_id= 5127, keysize= 0, flags= 0x2
04:33:04: IPSEC(initialize_sas): ,
  (key eng. msg.) OUTBOUND local= 172.18.1.1, remote= 10.1.1.1,
    local_proxy= 0.0.0.0/0.0.0.0/0/0 (type=4),
    remote_proxy= 10.4.1.4/0.0.0.0/0/0 (type=1),
    protocol= ESP, transform= esp-3des esp-md5-hmac ,
    lifedur= 2147483s and 4608000kb,
    spi= 0x50110CF8(1343294712), conn_id= 5128, keysize= 0, flags= 0xA
04:33:04: IPSEC(kei_proxy): head = ra, map->ivrf = vpn1, kei->ivrf = vpn2
04:33:04: IPSEC(kei_proxy): head = ra, map->ivrf = vpn2, kei->ivrf = vpn2
04:33:04: IPSEC(rte_mgr): VPN Route Added 10.4.1.4 255.255.255.255 via 10.1.1.1 in vpn2
04:33:04: IPSEC(add mtree): src 0.0.0.0, dest 10.4.1.4, dest_port 0
04:33:04: IPSEC(create_sa): sa created,
  (sa) sa_dest= 172.18.1.1, sa_prot= 50,
    sa_spi= 0xA3E24AFD(2749516541),
    sa_trans= esp-3des esp-md5-hmac , sa_conn_id= 5127
04:33:04: IPSEC(create_sa): sa created,
  (sa) sa_dest= 10.1.1.1, sa_prot= 50,
    sa_spi= 0x50110CF8(1343294712),
    sa_trans= esp-3des esp-md5-hmac , sa_conn_id= 5128
04:33:53: ISAKMP (0:13): purging node -1639992295
04:33:54: ISAKMP (0:13): purging node 17011691
```

Configuration Examples for VRF-Aware IPsec

Static IPsec-to-MPLS VPN Example

The following sample shows a static configuration that maps IPsec tunnels to MPLS VPNs. The configurations map IPsec tunnels to MPLS VPNs “VPN1” and “VPN2.” Both of the IPsec tunnels terminate on a single public-facing interface.

IPsec PE Configuration

```
ip vrf vpn1
 rd 100:1
 route-target export 100:1
 route-target import 100:1
!
ip vrf vpn2
 rd 101:1
 route-target export 101:1
 route-target import 101:1
!
crypto keyring vpn1
  pre-shared-key address 172.16.1.1 key vpn1
crypto keyring vpn2
  pre-shared-key address 10.1.1.1 key vpn2
!
crypto isakmp policy 1
 encr 3des
 authentication pre-share
 group 2
!
crypto isakmp profile vpn1
 vrf vpn1
 keyring vpn1
 match identity address 172.16.1.1 255.255.255.255
!
crypto isakmp profile vpn2
 vrf vpn2
 keyring vpn2
 match identity address 10.1.1.1 255.255.255.255
!
crypto ipsec transform-set vpn1 esp-3des esp-sha-hmac
crypto ipsec transform-set vpn2 esp-3des esp-md5-hmac
!
crypto map crypmap 1 ipsec-isakmp
 set peer 172.16.1.1
 set transform-set vpn1
 set isakmp-profile vpn1
 match address 101
```

```
crypto map crypmap 3 ipsec-isakmp
 set peer 10.1.1.1
 set transform-set vpn2
 set isakmp-profile vpn2
 match address 102
!
interface Ethernet1/1
 ip address 172.17.1.1 255.255.0.0
 tag-switching ip
!
interface Ethernet1/2
 ip address 172.18.1.1 255.255.255.0
 crypto map crypmap
!
ip route 172.16.1.1 255.255.255.255 172.18.1.2
ip route 10.1.1.1 255.255.255.255 172.18.1.2
ip route vrf vpn1 10.2.0.0 255.255.0.0 172.18.1.2 global
ip route vrf vpn2 10.2.0.0 255.255.0.0 172.18.1.2 global
!
access-list 101 permit ip 10.1.0.0 0.0.255.255 10.2.0.0 0.0.255.255
access-list 102 permit ip 10.1.0.0 0.0.255.255 10.2.0.0 0.0.255.255
```

IPsec Customer Provided Edge (CPE) Configuration for VPN1

```
crypto isakmp policy 1
 encr 3des
 authentication pre-share
 group 2
crypto isakmp key vpn1 address 172.18.1.1
!
!
crypto ipsec transform-set vpn1 esp-3des esp-sha-hmac
!
crypto map vpn1 1 ipsec-isakmp
 set peer 172.18.1.1
 set transform-set vpn1
 match address 101
!
interface FastEthernet1/0
 ip address 172.16.1.1 255.255.255.0
 crypto map vpn1
!
interface FastEthernet1/1
 ip address 10.2.1.1 255.255.0.0
!
access-list 101 permit ip 10.2.0.0 0.0.255.255 10.1.0.0 0.0.255.255
!
```

IPsec CPE Configuration for VPN2

```
crypto isakmp policy 1
 encr 3des
 authentication pre-share
 group 2
!
crypto isakmp key vpn2 address 172.18.1.1
!
!
crypto ipsec transform-set vpn2 esp-3des esp-md5-hmac
!
crypto map vpn2 1 ipsec-isakmp
 set peer 172.18.1.1
 set transform-set vpn2
 match address 101
!
interface FastEthernet0
 ip address 10.1.1.1 255.255.255.0
 crypto map vpn2
!
interface FastEthernet1
```

```
 ip address 10.2.1.1 255.255.0.0
!
access-list 101 permit ip 10.2.0.0 0.0.255.255 10.1.0.0 0.0.255.255
```

IPsec-to-MPLS VPN Using RSA Encryption Example

The following example shows an IPsec-to-MPLS configuration using RSA encryption:

PE Router Configuration

```
ip vrf vpn1
 rd 100:1
 route-target export 100:1
 route-target import 100:1
!
crypto isakmp policy 10
 authentication rsa-encr
!
crypto keyring vpn1
 rsa-pubkey address 172.16.1.1 encryption
  key-string
   305C300D 06092A86 4886F70D 01010105 00034B00 30480241 00DBF381 00DDECC8
   DC4AA490 40320C52 9912D876 EB36717C 63DCA95C 7E5EC02A 84F276CE 292B42D7
   D664F324 3726F4E0 39D33093 ECB81B95 482511A5 F064C4B3 D5020301 0001
   quit
!
crypto isakmp profile vpn1
 vrf vpn1
 keyring vpn1
 match identity address 172.16.1.1 255.255.255.255
!
crypto ipsec transform-set vpn1 esp-3des esp-sha-hmac
!
crypto map crypmap 1 ipsec-isakmp
 set peer 172.16.1.1
 set transform-set vpn1
 set isakmp-profile vpn1
 match address 101
!
interface Ethernet1/1
 ip address 172.17.1.1 255.255.0.0
 tag-switching ip
!
interface Ethernet1/2
 ip address 172.18.1.1 255.255.255.0
 crypto map crypmap
!
ip route 172.16.1.1 255.255.255.255 172.18.1.2
ip route vrf vpn1 10.2.0.0 255.255.0.0 172.18.1.2 global
!
access-list 101 permit ip 10.1.0.0 0.0.255.255 10.2.0.0 0.0.255.255
```

IPsec CPE Configuration for VPN1

```
crypto isakmp policy 10
 authentication rsa-encr
!
crypto key pubkey-chain rsa
 addressed-key 172.18.1.1 encryption
 key-string
  3082011B 300D0609 2A864886 F70D0101 01050003 82010800 30820103 0281FB00
  C90CC78A 6002BDBA 24683396 B7D7877C 16D08C47 E00C3C10 63CF13BC 4E09EA23
  92EB8A48 4113F5A4 8796C8BE AD7E2DC1 3B0742B6 7118CE7C 1B0E21D1 AA9724A4
  4D74FCEA 562FF225 A2B11F18 E53C4415 61C3B741 3A06E75D B4F9102D 6163EE40
  16C68FD7 6532F660 97B59118 9C8DE3E5 4E2F2925 BBB87FCB 95223D4E A5E362DB
  215CB35C 260080805 17BBE1EF C3050E13 031F3D5B 5C22D16C FC8B1EC5 074F07A5
  D050EC80 7890D9C5 EC20D6F0 173FE2BA 89F5B5F9 2EADC9A6 D461921E 3D5B60016
  ABB8B6B9 E2124A21 93F0E4AE B487461B E7F1F1C4 032A0B0E 80DC3E15 CB268EC9
  5D76B9BD 3C78CB75 CE9F68C6 484D6573 CBC3EB59 4B5F3999 8F9D0203 010001
  quit
!
crypto ipsec transform-set vpn1 esp-3des esp-sha-hmac
!
crypto map vpn1 1 ipsec-isakmp
 set peer 172.18.1.1
 set transform-set vpn1
 match address 101
!
interface FastEthernet1/0
 ip address 172.16.1.1 255.255.255.0
 crypto map vpn1
!
interface FastEthernet1/1
 ip address 10.2.1.1 255.255.0.0
!
access-list 101 permit ip 10.2.0.0 0.0.255.255 10.1.0.0 0.0.255.255
!
```

IPsec-to-MPLS VPN with RSA Signatures Example

The following shows an IPsec-to-MPLS VPN configuration using RSA signatures:

PE Router Configuration

```
ip vrf vpn1
 rd 100:1
 route-target export 100:1
 route-target import 100:1
!
crypto ca trustpoint bombo
 enrollment url http://172.31.68.59:80
 crl optional
!
crypto ca certificate chain bombo
 certificate 03C0
 308203BF 308202A7 A0030201 02020203 C0300D06 092A8648 86F70D01 01050500
 . . .
 quit
 certificate ca 01
 30820379 30820261 A0030201 02020101 300D0609 2A864886 F70D0101 05050030
 . . .
 quit
!
crypto isakmp profile vpn1
 vrf vpn1
 ca trust-point bombo
 match identity address 172.16.1.1 255.255.255.255
!
crypto ipsec transform-set vpn1 esp-3des esp-sha-hmac
!
crypto map crypmap 1 ipsec-isakmp
 set peer 172.16.1.1
 set transform-set vpn1
 set isakmp-profile vpn1
 match address 101
!
interface Ethernet1/1
 ip address 172.31.1.1 255.255.0.0
 tag-switching ip
!
interface Ethernet1/2
 ip address 172.18.1.1 255.255.255.0
 crypto map crypmap
!
ip route 172.16.1.1 255.255.255.255 172.18.1.2
ip route vrf vpn1 10.2.0.0 255.255.0.0 172.18.1.2 global
!
```

```
access-list 101 permit ip 10.1.0.0 0.0.255.255 10.2.0.0 0.0.255.255
!
```

IPsec CPE Configuration for VPN1

```
crypto ca trustpoint bombo
 enrollment url http://172.31.68.59:80
 crl optional
 !
crypto ca certificate chain bombo
 certificate 03BF
  308203BD 308202A5 A0030201 02020203 BF300D06 092A8648 86F70D01 01050500
  . . .
  quit
 certificate ca 01
  30820379 30820261 A0030201 02020101 300D0609 2A864886 F70D0101 05050030
  . . .
  quit
!
crypto ipsec transform-set vpn1 esp-3des esp-sha-hmac
!
crypto map vpn1 1 ipsec-isakmp
 set peer 172.18.1.1
 set transform-set vpn1
 match address 101
!
interface FastEthernet1/0
 ip address 172.16.1.1 255.255.255.0
 crypto map vpn1
!
interface FastEthernet1/1
 ip address 10.2.1.1 255.255.0.0
!
access-list 101 permit ip 10.2.0.0 0.0.255.255 10.1.0.0 0.0.255.255
!
```

Upgrade from Previous Versions of the Cisco Network-Based IPsec VPN Solution

The VRF-Aware IPsec feature in the Cisco network-based IPsec VPN solution release 1.5 requires that you change your existing configurations.

The following sample configuration indicates the changes you must make to your existing configuration:

- Site-to-Site Configuration Upgrade, page 25

Site-to-Site Configuration Upgrade

The following configurations show the changes that are necessary for a site-to-site configuration upgrade from a previous version of the network-based IPsec VPN solution to the Cisco network-based IPsec VPN solution release 1.5:

- Previous Version Site-to-Site Configuration, page 25
- New Version Site-to-Site Configuration, page 26

Previous Version Site-to-Site Configuration

```
crypto isakmp key VPN1 address 172.21.25.74
 crypto isakmp key VPN2 address 172.21.21.74
!
 crypto ipsec transform-set VPN1 esp-des esp-sha-hmac
 crypto ipsec transform-set VPN2 esp-3des esp-sha-hmac
```

```
!
 crypto map VPN1 10 ipsec-isakmp
 set peer 172.21.25.74
 set transform-set VPN1
 match address 101
!
 crypto map VPN2 10 ipsec-isakmp
 set peer 172.21.21.74
 set transform-set VPN2
 match address 102
!
 interface FastEthernet0/0.1
 encapsulation dot1Q 1 native
 ip vrf forwarding VPN1
 ip address 172.21.25.73 255.255.255.0
 crypto map VPN1
!
 interface FastEthernet0/0.2
  encapsulation dot1Q 2 native
 ip vrf forwarding VPN2
 ip address 172.21.21.74 255.255.255.0
 crypto map VPN2
```

New Version Site-to-Site Configuration

The following is an upgraded version of the same site-to-site configuration to the Cisco network-based IPsec VPN solution release 1.5 solution:

Note You must change to keyrings. The VRF-Aware IPsec feature requires that keys be associated with a VRF if the IKE local endpoint is in the VRF.

```
crypto keyring VPN1-KEYS vrf VPN1
 pre-shared-key address 172.21.25.74 key VPN1
!
 crypto keyring VPN2-KEYS vrf VPN2
  pre-shared-key address 172.21.21.74 key VPN2
!
 crypto ipsec transform-set VPN1 esp-des esp-sha-hmac
 crypto ipsec transform-set VPN2 esp-3des esp-sha-hmac
!
 crypto map VPN1 10 ipsec-isakmp
 set peer 172.21.25.74
  set transform-set VPN1
 match address 101
!
 crypto map VPN2 10 ipsec-isakmp
 set peer 172.21.21.74
  set transform-set VPN2
 match address 102
!
 interface FastEthernet0/0.1
 encapsulation dot1Q 1 native
 ip vrf forwarding VPN1
 ip address 172.21.25.73 255.255.255.0
 crypto map VPN1
!
 interface FastEthernet0/0.2
 encapsulation dot1Q 2 native
 ip vrf forwarding VPN2
 ip address 172.21.21.74 255.255.255.0
 crypto map VPN2
```

Additional References

The following sections provide references related to the VRF-Aware IPsec feature.

Related Documents

Related Topic	Document Title
IPsec configuration tasks	The chapter "Configuring Security for VPNs with IPsec" in the *Cisco IOS Security Configuration Guide: Secure Connectivity*
IPsec commands	*Cisco IOS Security Command Reference*
IKE Phase 1 and Phase 2, agressive mode, and main mode	The chapter "Configuring Internet Key Exchange for IPsec VPNs" in the *Cisco IOS Security Configuration Guide: Secure Connectivity*
IKE dead peer detection	Easy VPN Server

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for VRF-Aware IPsec

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 1: Feature Information for VRF-Aware IPsec

Feature Name	Releases	Feature Information
VRF-Aware IPsec	XE 2.6	The VRF-Aware IPsec feature introduces IP Security (IPsec) tunnel mapping to Multiprotocol Label Switching (MPLS) Virtual Private Networks (VPNs). Using the VRF-Aware IPsec feature, you can map IPsec tunnels to Virtual Routing and Forwarding (VRF) instances using a single public-facing address. This feature was introduced in Cisco IOS Release 12.2(15)T,. The following commands were introduced or modified: **address, ca trust-point, client authentication list, client configuration address, crypto isakmp profile, crypto keyring, crypto map isakmp-profile**, **initiate-mode, isakmp authorization list, keepalive (isakmp profile), keyring, key-string, match identity, no crypto xauth, pre-shared-key, quit, rsa-pubkey, self-identity, serial-number, set isakmp-profile, show crypto isakmp key, show crypto isakmp profile, vrf, clear crypto sa, crypto isakmp peer, crypto map isakmp-profile, show crypto ipsec sa, show crypto isakmp sa, show crypto map (IPsec)**.

Glossary

CA --certification authority. CA is an entity that issues digital certificates (especially X.509 certificates) and vouches for the binding between the data items in a certificate.

CLI --command-line-interface. CLI is an interface that allows the user to interact with the operating system by entering commands and optional arguments. The UNIX operating system and DOS provide CLIs.

client --Corresponding IPsec IOS peer of the UUT in the Multi Protocol Label Switching (MPLS) network.

dead peer --IKE peer that is no longer reachable.

DN --Distinguished Name. A DN is the global, authoritative name of an entry in the Open System Interconnection (OSI Directory [X.500]).

FQDN --fully qualified domain name. A FQDN is the full name of a system rather than just its host name. For example, aldebaran is a host name, and aldebaran.interop.com is an FQDN.

FR --Frame Relay. FR is an industry-standard, switch-data-link-layer protocol that handles multiple virtual circuits using high-level data link (HDLC) encapsulation between connected devices. Frame Relay is more efficient than X.25, the protocol for which it generally is considered a replacement.

FVRF --Front Door Virtual Routing and Forwarding (VRF) repository. FVRF is the VRF used to route the encrypted packets to the peer.

IDB --Interface descriptor block. An IDB subblock is an area of memory that is private to an application. This area stores private information and states variables that an application wants to associate with an IDB or an interface. The application uses the IDB to register a pointer to its subblock, not to the contents of the subblock itself.

IKE --Internet Key Exchange. IKE establishes a shared security policy and authenticates keys for services (such as IPsec) that require keys. Before any IPsec traffic can be passed, each router, firewall, and host must verify the identity of its peer. This can be done by manually entering preshared keys into both hosts or by a CA service.

IKE keepalive --Bidirectional mechanism for determining the liveliness of an IKE peer.

IPsec --Security protocol for IP.

IVRF --Inside Virtual Routing and Forwarding. IVRF is the VRF of the plaintext packets.

MPLS --Multiprotocol Label Switching. MPLS is a switching method that forwards IP traffic using a label. This label instructs the routers and the switches in the network where to forward the packets based on preestablished IP routing information.

RSA --Rivest, Shamir, and Adelman are the inventors of the RSA technique. The RSA technique is a public-key cryptographic system that can be used for encryption and authentication.

SA --Security Association. SA is an instance of security policy and keying material applied to a data flow.

VPN --Virtual Private Network. A VPN enables IP traffic to travel securely over a public TCP or IP network by encrypting all traffic from one network to another. A VPN uses "tunneling" to encrypt all information at the IP level.

VRF --Virtual Route Forwarding. VRF is A VPN routing and forwarding instance. A VRF consists of an IP routing table, a derived forwarding table, a set of interfaces that use the forwarding table, and a set of rules and routing protocols that determine what goes into the forwarding table. In general, a VRF includes the routing information that defines a customer VPN site that is attached to a PE router.

XAUTH --Extended authentication. XAUTH is an optional exchange between IKE Phase 1 and IKE Phase 2, in which the router demands additional authentication information in an attempt to authenticate the actual user (as opposed to authenticating the peer).

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.