Implementing IPSec Network Security on Cisco IOS XR Software

IP Security (IPSec) provides security for transmission of sensitive information over unprotected networks such as the Internet. IPSec acts at the network layer, protecting and authenticating IP packets between participating IPSec devices ("peers"), such as Cisco routers.

With IPSec, data can be sent across a public network without observation, modification, or spoofing, which enables applications, such as Virtual Private Networks (VPNs), including intranets, extranets, and remote user access.

IPSec for Cisco IOS XR software supports the following two types of traffic:

- IPSec for locally sourced traffic or traffic terminated on the router. Either tunnel-ipsec interfaces or a transport entity are used. This type is also called *software-based IPSec*.
- IPSec for transit traffic. This mode is also called *hardware-based IPSec*. Both service-ipsec and service-gre interfaces are used for this type.

This module describes the tasks that you need to implement IPSec network security on your Cisco IOS XR network.

Note For a complete description of the IPSec network security commands used in this chapter, see the *IPSec Network Security Commands on Cisco IOS XR Software* module of *Cisco IOS XR System Security Command Reference*. To locate documentation of other commands that appear in this chapter, use the command reference master index, or search online.

Feature History for Implementing IPSec Network Security on Cisco XR 12000 Series Router Contents

Release	Modification
Release 3.2	This feature was introduced on the Cisco XR 12000 Series Router.
Release 3.3.0	No modification.
Release 3.4.0	Support was added for Cisco XR 12000 Series Router IPSec VPN SPA. The **crypto ipsec chkpt-disabled** command was removed; therefore, the *Configuring Checkpointing* section was removed.
Release 3.5.0	The Multiprotocol Label Switching (MPLS) Encapsulated Packets on Inbound Direction feature was added. IPSec—SNMP support feature was added on the Cisco XR 12000 Series Router IPSec VPN SPA.
Release 3.6.0	Information was added about the use of object tracking in IPSec. Information was added about new functionality that enables dynamically learned routes in reverse-route injection to take precedence over static routes. Distance and route tag parameters were added to the reverse-route injection feature. The upper range value of the *sa-id* argument in the **show crypto ipsec sa** and **clear crypto ipsec sa** commands was increased from 16500 to 64500. Information was added about implementing IPSec in site-to-site and remote VPN topologies. Information, including examples, were added about the use of object tracking in an IPSec-enabled network. Some information was reorganized to increase readability.
Release 3.7.0	No modification.
Release 3.8.0	Information was edited to make clearer which features are supported on the Cisco XR 12000 Series Router exclusively.
Release 3.9.0	No modification.

Prerequisites for Implementing IPSec Network Security

The following prerequisites are required to implement IPSec network security:

- You must be in a user group associated with a task group that includes the proper task IDs for security commands. The command reference guides include the task IDs required for each command.
- If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.
- You must install and activate the Package Installation Envelope (PIE) for the security software.

  For detailed information about optional PIE installation, see *Cisco IOS XR System Management Configuration Guide*.
- For Cisco XR 12000 Series Router IPSec VPN SPA, you must install a service software PIE.
- You must configure Internet Key Exchange (IKE), as described in the *Implementing Internet Key Exchange Security Protocol on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide*.

Restrictions for Implementing IPSec Network Security

If you use Network Address Translation (NAT), you must configure static NAT translations so that IPSec works properly. In general, NAT translation should occur before the router performs IPSec encapsulation; in other words, IPSec should be working with global addresses.

Note Use static crypto profiles only.

Restrictions for Implementing a Cisco IPSec VPN SPA Within an IPSec Network

The following restrictions exist with regard to implementing a Cisco XR 12000 Series Router IPSec VPN SPA within an IPSec network:

- Clear generic routing encapsulation (GRE) is not supported. Only secure GRE is supported by the Cisco XR 12000 Series Router IPSec VPN SPA. To configure the Cisco XR 12000 Series Router IPSec VPN SPA, you can use either service-ipsec or service-gre interfaces.
- Dynamic virtual interfaces are not supported.
- Dynamic Multipoint VPN (DMVPN) is not supported.
- Multicast is supported on interfaces in global VRF.

The following restrictions are known when implementing IPSec with the Cisco XR 12000 Series Router IPSec VPN SPA for Internet Security Association Key Management Protocol (ISAKMP) and IPSec profile configurations:

- One IPSec profile is configured on a virtual interface (for example, service ipsec or service-gre). A profile has one or more access control lists (ACLs). Each ACL has one or more access control entry (ACE). ACLs and ACEs cannot intersect.

- With both tunnel source and tunnel destination defined, a dynamic profile configuration on a virtual interface is rejected.
- Multiple virtual interfaces are attached to ISAKMP profiles; however, a virtual interface is referenced from a single ISAKMP profile only. The constraint is required to uniquely identify the virtual interface, ISAKMP profile, and IPSec profile as an IKE initiator and IKE responder.

The following restrictions are known to implement Cisco XR 12000 Series Router IPSec VPN SPA for the tunnel source address:

- Virtual interfaces that use the same tunnel source and FVRF are configured on the same Cisco XR 12000 Series Router IPSec VPN SPA.
- When NAT traversal is active for two tunnels that share the same source and destination address and FVRF under two different virtual interfaces, IP packets that require fragmentation are dropped.

Information About Implementing IPSec Networks

To implement IP network security, you should understand the following concepts:

Note For information about IPSec quality of service (QoS), refer to *Cisco IOS XR Modular Quality of Service Configuration Guide*.

Crypto Profiles

Crypto profile entries created for IPSec combine the various parts used to set up IPSec security associations (SAs), including the following:

- Traffic that should be protected by IPSec (per a crypto access list)
- Granularity of the flow to be protected by a set of SAs
- IPSec security that should be applied to this traffic (selecting from a list of one or more transform sets)

- Other parameters that might be necessary to define an IPSec SA

Crypto profiles are applied to IPSec interfaces (for example, tunnel-ipsec, service-ipsec, and service) or crypto transport.

If the access control lists (ACLs) specified within the profile match any outbound IP traffic, the IP traffic is protected by IPSec. The SA is established with the remote peer by IKE.

When using service-gre interfaces, the profile, which is attached to the interface, is not configured with an explicit ACL. Instead, all traffic, which is destined to the GRE tunnel, is protected by IPSec.

The policy described in the crypto profile entries is used during the negotiation of SAs. If the local router initiates the negotiation, it uses the policy specified in the static crypto profile entries to create the offer to be sent to the specified IPSec peer. If the IPSec peer initiates the negotiation, the local router checks the policy associated with the interface or profile associated with the identity specified in the ISAKMP profile, which is being used to decide whether to accept or reject the peer's request (offer).

For IPSec to succeed between two IPSec peers, both peers' crypto profile entries must contain compatible configuration statements. When two peers try to establish an SA, each must have at least one crypto profile entry that is compatible with one of the other peer's crypto profile entries. For two crypto profile entries to be compatible, they must at least meet the following criteria:

- The crypto profile entries must contain compatible crypto access lists. In the case where the responding peer is using dynamic crypto profiles, the entries in the local crypto access list must be "permitted" by the peer's crypto access list.
- The crypto profile entries must have at least one transform set in common.

Note

- Crypto profiles cannot be shared, that is, the same profile cannot be attached to multiple tunnel-IPSec interfaces or an interface and transport mode IPSec.
- The restriction is only for ipsec-tunnel interface or transport and not service-ipsec or service-gre interfaces.

Dynamic Crypto Profiles

A dynamic crypto profile entry is essentially a crypto profile entry without all the parameters configured. It acts as a policy template in which the missing parameters are later dynamically configured (as the result of an IPSec negotiation) to match the requirements of a remote peer. This allows remote peers to exchange IPSec traffic with the router even if the router does not have a crypto profile entry specifically configured to meet all of the remote peer's requirements.

Dynamic crypto profiles are not used by the router to initiate new IPSec SAs with remote peers. Dynamic crypto profiles are used when a remote peer tries to initiate an IPSec SA with the router. Dynamic crypto profiles are also used in evaluating traffic.

If the router accepts the peer's request, at the point that it installs the new IPSec SAs it implicitly installs a temporary crypto profile entry. This entry is filled in with the results of the negotiation. At this point, the router performs normal processing, using this temporary crypto profile entry as a normal entry, even requesting new SAs if the current ones are expiring (based upon the policy specified in the temporary crypto profile entry). After the flow expires (that is, all of the corresponding SAs expire), the temporary crypto profile entry is then removed.

Static Crypto Profiles

When static crypto profile entries exist, if outbound traffic matches a **permit** statement in an access list and the corresponding SA is not yet established, the router initiates new SAs with the remote peer. In the case of dynamic crypto profile entries, if no SA existed, the traffic would be dropped because dynamic crypto profiles are not used for initiating new SAs.

Crypto Access Lists

Crypto access lists are used to define all IP traffic whether or not it is protected by crypto. For example, access lists can be created to protect all IP traffic between Subnet A and Subnet Y or Telnet traffic between Host A and Host B.

The access lists themselves are not specific to IPSec. It is the crypto profile entry referencing the specific access list that defines whether IPSec processing is applied to the traffic matching a **permit** in the access list.

Crypto access lists associated with IPSec crypto profile entries have four primary functions:

- Select outbound traffic to be protected by IPSec (permit = protect).
- Indicate the data flow to be protected by the new SAs (specified by a single **permit** entry) when initiating negotiations for IPSec SAs.
- Process inbound traffic to filter and discard traffic that should have been protected by IPSec.
- Determine whether to accept requests for IPSec SAs on behalf of the requested data flows when processing IKE negotiation from the IPSec peer. (Negotiation is done only for **ipsec-isakmp** crypto profile entries.) To be accepted, the peer initiating the IPSec negotiation must specify a data flow that is “permitted” by a crypto access list associated with an **ipsec-isakmp** crypto profile entry.

If you want certain traffic to receive one combination of IPSec protection (for example, authentication only) and other traffic to receive a different combination of IPSec protection (for example, both authentication and encryption), you must create two different crypto access lists to define the two different types of traffic.

Transform Sets

A transform set represents a certain combination of security protocols and algorithms. During the IPSec SA negotiation, the peers agree to use a particular transform set for protecting a particular data flow.

You can specify multiple transform sets and then one or more of these transform sets in a crypto profile entry. The transform set defined in the crypto profile entry is used in the IPSec SA negotiation to protect the data flows specified by that crypto profile entry’s access list.

During IPSec SA negotiations with IKE, the peers search for a transform set that is the same at both peers. When such a transform set is found, it is selected and applied to the protected traffic as part of both peers’ IPSec SAs.

If you change a transform set definition, the change is applied only to crypto profile entries that reference the transform set. The change will not be applied to existing SAs, but is used in subsequent negotiations to establish new SAs. If you want the new settings to take effect sooner, you can clear all or part of the SA database by using the **clear crypto ipsec sa** command.

Global Lifetimes for IPSec Security Associations

You can change the global lifetime values that are used when negotiating new IPSec SAs.

Two lifetimes exist: a "timed" lifetime and "traffic-volume" lifetime. An SA expires after the first of these lifetimes is reached. The default lifetimes are 3600 seconds (1 hour) and 4,194,303 kilobytes (10 MBps for 1 hour).

A lifetime per profile is also supported. If a profile is configured with a lifetime, it overrides the global definition.

If you change a global lifetime, the new lifetime value is not applied to currently existing SAs, but is used in the negotiation of subsequently established SAs. If you want to use the new values immediately, you can clear all or part of the SA database. For more information, see the documentation of the **clear crypto ipsec sa** command in *Cisco IOS XR System Security Configuration Guide*.

IPSec SAs use one or more shared secret keys. These keys and their SAs time out together.

Assuming that the particular crypto profile entry does not have lifetime values configured, when the router requests new SAs it specifies its global lifetime values in the request to the peer; it uses this value as the lifetime of the new SAs. When the router receives a negotiation request from the peer, it uses the smaller of either the lifetime value proposed by the peer or the locally configured lifetime value as the lifetime of the new SAs.

The SA (and corresponding keys) expire according to whichever comes sooner, either after the number of seconds has passed (specified by the **seconds** keyword) or amount of traffic in kilobytes is passed (specified by the **kilobytes** keyword).

A new SA is negotiated *before* the lifetime threshold of the existing SA is reached, to ensure that a new SA is ready for use when the old one expires. The new SA is negotiated approximately 30 seconds before the **seconds** lifetime expires or when the volume of traffic through the tunnel reaches 300 kilobytes less than the **kilobytes** lifetime (whichever comes first).

If no traffic has passed through the tunnel during the entire life of the SA, a new SA is not negotiated when the lifetime expires. Instead, a new SA is negotiated only when IPSec sees another packet that should be protected.

Manual IPSec Security Associations

The use of manual IPSec security associations is a result of a prior arrangement between the users of the local router and the IPSec peer. The two parties can begin with manual security associations, but must move to using security associations established through IKE, or the system belonging to the remote party cannot support IKE. If IKE is not used to establish security associations, there is no negotiation of security associations, so the configuration information in both systems must be the same for traffic to be processed successfully by IPSec.

The local router can simultaneously support manual and IKE-established security associations, even within a single crypto profile entry. There is little reason to disable IKE on the local router unless the router supports only manual security associations.

Perfect Forward Secrecy

Perfect Forward Secrecy (PFS) ensures that a given key of an IPSec security association (SA) is not derived from any other secret, such as some other keys. In other words, if someone broke a key, PFS would ensure that the attacker would not be able to derive any other key. If PFS is not enabled, someone

can hypothetically break the IKE SA secret key, copy all the IPSec-protected data, and use knowledge of the IKE SA secret to compromise the IPSec SAs set up by this IKE SA. With PFS, breaking IKE does not give an attacker immediate access to IPSec. The attacker needs to break each IPSec SA individually.

During negotiation, the **set pfs** command causes IPSec to request PFS when requesting new security associations for the crypto profile entry. If the **set pfs** command statement does not specify a group, the default (group1) is sent. If the peer initiates the negotiation and the local configuration specifies PFS, the remote peer must perform a PFS exchange or the negotiation fails. If the local configuration does not specify a group, a default of group1 is assumed, and an offer of either group1, group2, or group5 is accepted. If the local configuration specifies group2 or group5, the group must be part of the offer from the peer or the negotiation fails. If the local configuration does not specify PFS, the configuration accepts any offer of PFS from the peer.

Checkpointing

IPSec checkpoints SAs in the local database. If an IPSec process restarts, SAs are retrieved from the local database and need not be re-established with remote peers.

DF Bit Override Functionality with IPSec Tunnels

Note This IPSec feature is supported only on the Cisco IPSec VPN SPA.

A *Don't Fragment (DF) bit* is a bit within the IP header that determines whether a router is allowed to fragment a packet. The DF Bit Override Functionality with IPSec Tunnels feature allows you to specify whether your router can clear, set, or copy the DF bit from the encapsulated header.

Some configurations have hosts that perform the following functions:

- Set the DF bit in packets they send.
- Use firewalls that block Internet Control Message Protocol (ICMP) errors from outside the firewall, preventing hosts from learning about the maximum transmission unit (MTU) size outside the firewall.
- Use IPSec to encapsulate packets to reduce the available MTU size.

If your configurations have hosts that prevent them from learning about the available MTU size, you can configure your router to clear the DF bit and fragment the packet.

The DF Bit Override Functionality with IPSec Tunnels feature allows you to configure the setting of the DF bit when encapsulating IPSec tunnels for IPSec traffic on a global or per-interface level. Thus, if the DF bit is set to clear, routers can fragment packets regardless of the original DF bit setting.

IPSec Antireplay Window

Note This IPSec feature is supported only on the Cisco IPSec VPN SPA.

Cisco IPSec authentication provides antireplay protection against an attacker duplicating encrypted packets, by assigning a unique sequence number to each encrypted packet. (Security association [SA] antireplay is a security service in which the receiver can reject old or duplicate packets to protect itself

against replay attacks.) The decryptor checks off the sequence numbers that it has seen before. The encryptor assigns sequence numbers in an increasing order. The decryptor remembers the value *X* of the highest sequence number that it has already seen. *N* is the window size, and the decryptor also remembers whether it has seen packets having sequence numbers from *X-N*+1 through *X*. Any packet with the sequence number smaller than *X-N* is discarded. Currently, *N* is set at 64, so only 64 packets can be kept in the memory of the decryptor.

At times, however, the 64-packet window size is not sufficient. For example, Cisco quality of service (QoS) gives priority to high-priority packets, which could cause some low-priority packets to be discarded even though they could be one of the last 64 packets received by the decryptor. The IPSec Antireplay Window: Expanding and Disabling feature allows you to expand the window size, allowing the decryptor to keep more than 64 packets in its memory.

IPSec NAT Transparency

Note This IPSec feature is supported only on the Cisco IPSec VPN SPA.

Previously, a standard IPSec Virtual Private Network (VPN) tunnel does not work if there were one or more Network Address Translator (NAT) or Point Address Translation (PAT) points in the delivery path of the IPSec packet. The IPSec NAT transparency feature makes NAT IPSec-aware; therefore, allowing remote access users to build IPSec tunnels to home gateways.

IPSec Security Association Idle Timers

Note This IPSec feature is supported only on the Cisco IPSec VPN SPA.

When a router running Cisco IOS XR software creates an IPSec SA for a peer, resources must be allocated to maintain the SA. The SA requires both memory and several managed timers. For idle peers, these resources are wasted. If enough resources are wasted by idle peers, the counter could be prevented from creating new SAs with other peers. The IPSec security feature introduces a configurable idle timer to monitor SAs for activity, allowing SAs for idle peers to be deleted. The idle timers are configured either globally or on a crypto profile basis.

Prefragmentation for Cisco IPSec VPN SPAs

Note This IPSec feature is supported only on the Cisco IPSec VPN SPA.

When a packet is nearly the size of the maximum transmission unit (MTU) of the outbound link of the encrypting router and it is encapsulated with IPSec headers, the packet is likely to exceed the MTU of the outbound link. The packet causes packet fragmentation after encryption, which makes the decrypting router reassemble in the process path. Prefragmentation for Cisco IPSec VPN SPAs increases the decrypting router's performance by enabling it to operate in the high-performance CEF path instead of the process path.

This feature allows an encrypting router to predetermine the encapsulated packet size from information available in transform sets, which are configured as part of the IPSec SA. If it is predetermined that the packet exceeds the MTU of the output interface, the packet is fragmented before encryption. This function avoids process-level reassembly before decryption and helps improve decryption performance and overall IPSec traffic throughput.

Prefragmentation for the Cisco IPSec VPN SPA functionality depends on the service-ipsec interface from the **crypto ipsec df-bit** command configuration and the incoming packet "do not fragment" (DF) bit state (see Table 1).

Table 1 Pre-Fragmentation for Cisco IPSec VPN SPA Dependencies

Pre-Fragmentation for IPSec VPN SPAs Feature State (Enabled or Disabled)	Service IPSec Interface "crypto ipsec df-bit" Configuration	Incoming Packet DF Bit State	Result
Enabled	crypto ipsec df-bit copy	0	Fragmentation occurs before encryption.
Enabled	crypto ipsec df-bit copy	1	Packets are dropped.
Enabled	crypto ipsec df-bit clear	0	Fragmentation occurs before encryption.
Enabled	crypto ipsec df-bit clear	1	Packets are sent to egress interfaces (not fragmented before encryption).[1]
Enabled	crypto ipsec df-bit set	0	Fragmentation occurs before encryption.
Enabled	crypto ipsec df-bit set	1	Packets are dropped.
Disabled	crypto ipsec df-bit copy	0	Packets are sent to egress interfaces.[1]
Disabled	crypto ipsec df-bit copy	1	Packets are dropped.
Disabled	crypto ipsec df-bit clear	0	Packets are sent to egress interfaces.[1]
Disabled	crypto ipsec df-bit clear	1	Packets are sent to egress interfaces.[1]
Disabled	crypto ipsec df-bit set	0	Packets are sent to egress interfaces.[1]
Disabled	crypto ipsec df-bit set	1	Packets are dropped.

1. A packet that is sent to egress interfaces can get fragmented on an egress LC under the following conditions: Packet exceeds the MTU of the egress physical interface. The df-bit is not set on the outer IP header.

IPSec—SNMP Support

Note This IPSec feature is supported only on the Cisco IPSec VPN SPA.

The IPSec SNMP support feature allows you to specify the desired size of a tunnel history table by using the Cisco IOS XR CLI. The history table archives attribute and statistic information about the tunnel. A tunnel history table does not accompany every failure table, because every failure does not correspond to a tunnel. Thus, supported setup failures are recorded in the failure table, but an associated history table is not recorded because a tunnel was never set up.

Information About IPSec Networks and the Cisco IPSec VPN SPA Using Cisco IOS XR Software

To implement an IPSec network with a Cisco IPSec VPN SPA, you should understand the following concepts:

Cisco IPSec VPN SPA Overview

Cisco IOS XR Software supports security protocols such as Authentication Header (AH), Encapsulating Security Payload (ESP), and Internet Key Exchange (IKE). The resources consumed by these activities are significant and make it difficult to achieve line-rate transmission speeds over VPNs. Using the Cisco IPSec VPN SPA enables you to send all VPN traffic coming from or going to the Internet through the SPA hardware. The SPA supports all IPSec-related processing. Packets coming from the trusted LAN are encrypted and sent through the Internet. Packets that are received from the WAN routers pass through the Cisco IPSec VPN SPA for IPSec processing (for example, decryption and validation of the packet before the packet is sent onto the trusted LAN).

The following three SIP card types are supported:

- SIP-401
- SIP-501
- SIP-601

Displaying the SPA Hardware Type

To verify that the SPA hardware type is installed, use the **show diags** command. In addition, you can use the **show platform** command to verify SPA hardware information.

For an example of the **show diags** command output for the Cisco XR 12000 Series Router IPSec VPN SPA, see the "Displaying the SPA Hardware Type: Example" section on page 139.

Information About Security for VPNs with IPSec

To implement security for VPNs with IPSec, you should understand the following concepts:

For information about configuring security for VPNs with IPSec, see How to Implement IPSec Network Security for VPNs, page 128.

IPSec Virtual Interfaces

IPSec virtual interfaces simplify configuration of IPSec for protection of remote links, support multicast, and simplify network management and load balancing.

Generic Routing Encapsulation (GRE) is a tunneling protocol that can encapsulate a wide variety of protocol packet types inside IP tunnels, creating a virtual point-to-point link to routers at remote points over an IP network.

When you configure an IPSec virtual interface, you must associate the service entity by using the **service-location** command. The association is done statically. The **service-location** command is mandatory. A virtual interface configuration is not fully verified until the service location is specified.

The following IPSec virtual interfaces are supported:

- Static IPSec service virtual interface (SVI)—Use the **interface service-ipsec** command to create a static IPSec SVI.

  The tunnel endpoints are defined by the **tunnel source** command and **tunnel destination** command. The tunnel source can be shared between interfaces. Although the address is internal to the router, the address is used only as a tunnel source. Unlike other internal router IP addresses, the tunnel source is not used to serve routing protocols, or other applications that are terminated on the router. The sole purpose for a tunnel source is tunneling.

  If a dynamic IPSec profile is attached to a static IPSec SVI, tunnel destination should not be configured and is negotiated when a new tunnel is created. In this case, it is possible that multiple IPSec tunnels can terminate on the same IPSec SVI and contain a different tunnel destination.

  A virtual interface must be associated with an IPSec service SPA. The **service-location** command specifies both active and standby locations for the interface. All interfaces that share the same tunnel source and tunnel VRF (FVRF) must be associated to the same location. The only event in which virtual interfaces share the same source address, destination address, and FVRF, is when NAT traversal is in effect.

- Static IPSec-protected GRE virtual interface—Use the **interface service-gre** command to create a static IPSec-protected GRE interface. When GRE is used with IPSec, only transport mode is supported. Only one IPSec profile can be attached to a GRE interface in which case one IPSec SA is created. Only point-to-point GRE interfaces are supported.

IPSec Load Balancing

Load balancing is the mechanism by which IPSec handling is distributed between Cisco IPSec VPN SPAs. You configure a traffic diversion policy to effect automatic switchover from one SPA to another, designating one SPA as the active (primary) SPA and another as the standby (secondary). If the active SPA goes down, the standby takes on the active role and traffic is diverted to it.

Another option is to configure auto-revert. Using this policy, when the previously active location again becomes functional, traffic is diverted back to the original, preferred active location. If you do not configure auto-revert, after a failure in location A, location B (originally configured as the standby) becomes active and location A becomes the standby.

Note The preferred active and standby SPAs cannot reside on the same line card.

Reverse-route injection (RRI) is designed to simplify network design for Virtual Private Networks (VPNs) where redundancy or load balancing are required. For more information about RRI, see "IPSec—SNMP Support" section on page 89.

VRF-Aware IPSec

Each IPSec tunnel is associated with two VRF domains. The outer encapsulated domain belongs to one VRF domain, which is called the *front door VRF (FVRF)*, while the inner, protected IP packet belongs to another domain called *inside VRF (IVRF)*. Therefore, 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.

Clear IP traffic is forwarded from an internal VRF to a remote site or host within the VRF over IPSec tunnels. The IVRF is determined on the SVI by using the **vrf** command. The encrypted packets going over the IPSec tunnel are forwarded over the FVRF, which is configured on one or more switched virtual interfaces (SVI) by using the **tunnel vrf** command. The tunnel source and destination are addresses of the FVRF. The encapsulated packets and the ACLs, which are configured in the IPSec profile, are all part of the IVRF.

MPLS Encapsulated Packets on Inbound Direction

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

The IPSec packet arrives from the Internet and is destined for the provider edge (PE) 2, which is also called the *IPSec terminator*. If the packet arrives at a PE1 outside of a VRF (for example, in the global table), the ingress PE1 pushes a label switched path (LSP) label onto the IPSec packet. The LSP packet is used to tunnel the IPSec packet to the egress PE, which is the IPSec terminator.

Reverse-Route Injection

Reverse-routes provide a scalable mechanism to dynamically learn and advertise the IP address and subnets that belong to a remote site that connects through an IPSec Virtual Private Network (VPN) tunnel. These static routes can then be redistributed into other dynamic routing protocols, so that they can be advertised to other parts of the network (usually done by redistributing the routes into dynamic routing protocols on the core side). Routes learned during the establishment of an IPSec tunnel are removed when the IPSec tunnel is torn down.

Reverse-route injection (RRI) refers to the ability to insert static routes automatically into the routing of networks and hosts protected by a remote tunnel endpoint. These protected hosts and networks are known as *remote proxy identities*.

RRI simplifies network design in cases where VPNs require load balancing or redundancy. RRI also integrates route availability to the IPSec state. Without RRI, routing decisions would be made independent from the IPSec peer.

A route to the secure subnet of a remote peer must exist. The route might be statically configured or derived through some form of end-to-end dynamic routing exchange.

Each route is created on the basis of the remote proxy network and mask, with the next hop to this network being the remote tunnel endpoint. By using the remote tunnel endpoint as the next hop, the traffic is forced through the crypto process to be encrypted.

After a static route is created on the VPN router, this information is propagated to upstream devices, allowing them to determine the appropriate VPN router in which to send returning traffic to maintain IPSec state flows.

Being able to determine the appropriate VPN router is particularly useful when multiple VPN routers are used to provide load balancing, automatic switchover, or when the remote VPN devices are not accessible through a default route.

Routes are created in either the global routing table or the appropriate VPN routing and forwarding (VRF) table. For information about configuring reverse-route injection, see Configuring Reverse-Route Injection in a Crypto Profile, page 123.

Information About Implementing IPSec in a Site-to-Site or Remote VPN Topology

The following topics describe the tasks required to enable IPSec on a VPN:

- Restrictions to Implementing IPSec in a Site-to-Site or Remote VPN Topology, page 93
- Configuration of IPSec in a Site-to-Site VPN Topology, page 93
- Configuration of IPSec in a Remote-Access VPN Topology, page 96

For examples, see Configuration Examples for Implementing IPSec in a Site-to-Site or Remote VPN Topology, page 147.

Restrictions to Implementing IPSec in a Site-to-Site or Remote VPN Topology

This functionality can only be implemented in a Cisco XR 12000 Series Router network, using the IPSec VPN SPA.

Configuration of IPSec in a Site-to-Site VPN Topology

Site-to-site VPNs are deployed in static configurations that change very infrequently. Both endpoints of the VPN are generally routers (or, occasionally, high-end security devices). IKE uses both phases of negotiation and the resulting connection typically has a maximum lifetime to prevent excessive protocol churn.

Site-to-site VPN deployment requires that you complete the following procedures:

1. Enable ISAKMP.

   For detailed information, see the "Enabling or Disabling IKE" section on page 171 in the *Implementing Internet Key Exchange Security Protocol on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide*.

2. Configure the ISAKMP policy by defining the parameters to be used during Phase-1 IKE negotiation.

   For detailed information, see the "Configuring IKE Policies" section on page 172 in the *Implementing Internet Key Exchange Security Protocol on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide*.

3. Configure a crypto access control List (ACL) to enable communication between subnets.

For detailed information, see "Defining Group Policy Information for Mode Configuration" section on page 174 of the *Implementing Internet Key Exchange Security Protocol on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide*.

4. Configure the IPSec transform set that protects the data flows specified in the crypto profile access list.

   During the negotiation, the peers search for a transform set that is the same at both peers. When a matching transform set is found, IKE selects it and applies it to the protected traffic as part of the IPSec SAs for both peers.

   For details, see Defining Transform Sets, page 103.

5. Configure a crypto keyring.

   A keyring is a repository of preshared keys and RSA public keys. The peer keys defined in the keyring are used by the ISAKMP profile to authenticate the remote side during IKE negotiation.

   For details, see the "Configuring Crypto Keyrings" section on page 197 of the *Implementing Internet Key Exchange Security Protocol on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide*.

6. Configure the IPSec profile.

   For details, see Configuring Crypto Profiles, page 104.

   If configured, the destination address from the ACL is added as a static route pointing to the SVI. While this command is optional in site-to-site configurations, it is on by default for remote access topologies. (See Configuring IPSec in a Remote-Access VPN Topology: Example, page 151.)

7. Configure the IPSec virtual interface (SVI) by specifying its physical location (the IPSec SPA), IP address, and the source and destination addresses of the IPSec tunnel between the two nodes.

   The interface can be either of the following:

   - **service-ipsec**
   - **service-gre**

   The **service-ipsec** interface supports only **tunnel** mode, while the **service-gre** interface supports only **transport** mode, because the generic routing encapsulation (GRE) already provides tunneling. Each of these modes is configured using the **transform-set** command referenced in Step 6. "Configure the IPSec profile." on page 148.

   For details about configuring a tunnel source and destination, see the "Applying Crypto Profiles to tunnel-ipsec Interfaces" section on page 126.

   Note If the SVI has a standby SPA, it would also be configured with the **preferred-standby** keyword. The standby IPSec SPA must reside on a different SIP-x01 than the primary.

   Multiple SVIs on the same SPA may be configured with the same tunnel source address, which is useful for conserving addresses.

8. Configure crypto ISAKMP profile.

   The **crypto isakmp profile** command defines an ISAKMP profile and audits IPSec user sessions. Peers are mapped to an ISAKMP profile when their identities are matched (Step 7.), as given in the ID payload of IKE. The ISAKMP profile successfully completes authentication of peers if the peer keys are defined in the keyring attached to this profile.

   When the XR router is the IKE *responder,* this configuration line allows the router to determine the proper SVI for sending its encrypted traffic.

When the XR router is the IKE *initiator,* the correct SVI has already been chosen by doing a lookup in the forwarding table. The remote endpoint is then determined by the tunnel destination of that SVI.

For an example, see Configuring IPSec in a Site-to-Site VPN Topology: Example, page 147.

Site-to-Site Encrypted Generic Routing Encapsulation Tunnels

Generic routing encapsulation (GRE) is a tunneling protocol developed by Cisco Systems. This protocol supports:

- Encapsulation of a wide variety of protocol packet types inside IP tunnels.
- Creation of a virtual point-to-point link between Cisco routers at remote points over an IP network.

For details, see Configuring Crypto Profiles, page 104.

Identity Verification

Preshared keys, which are insecure, provide no verification of the identity of the remote endpoint.To mitigate this risk, the Public Key Infrastructure (PKI) protocol provides the following options:

- RSA keys
- Certificates

For information about Public Key Infrastructure commands, see the *Public Key Infrastructure Commands on Cisco IOS XR Software* module in *Cisco IOS XR System Security Command Reference*.

For information about the implementation of Certificate Authority Interoperability configurations, see the *Implementing Certificate Authority Interoperability on Cisco IOS XR Software* module in *Cisco IOS XR System Security Configuration Guide*.

Encryption of Public Key Infrastructure with RSA

PKI with RSA encryption consists of the following configuration tasks:

1. Configure a router hostname and IP domain name.

   For more information about the hostname command, see *Cisco IOS XR System Management Command Reference*. For more information about the domain name command, see *Cisco IOS XR IP Addresses and Services Command Reference.*

2. Generate RSA key pairs that are required to sign and encrypt IKE key management messages.
3. Configure the RSA public key information into the keyring.

   This configuration takes place under the keyring, in other words, the key is manually exchanged similar to a pre-shared key. See "Manually Configuring RSA Keys" section on page 184 of the *Implementing Internet Key Exchange Security Protocol on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide*.

4. Configure the ISAKMP policy by using the RSA-ENCR authentication method.
5. Update the ISAKMP profile.

   The RSA key authentication must match the identity that was specified in the crypto keyring command. See again Configuring Crypto Keyrings, page 197.

   For configuration details, see the *Implementing Certification Authority Interoperability on Cisco IOS XR Software* module in the *Cisco IOS XR System Security Configuration Guide*

Certificate Configuration

For a certification authority (CA), you do not need to configure keys between all the encrypting routers. Instead, enroll each participating router with the CA and request a certificate for the router.

The certificate enrollment, which takes place before establishment of a tunnel, specifies either the URL of the CA or a manual cut-and-paste of the enrollment (terminal).

Certificate configuration consists of the following tasks:

1. Declare a CA and configure a trusted point.
2. Authenticate the CA to your router.

   The router must authenticate the CA by obtaining the self-signed certificate of the CA, which also contains its public key.
3. Determine which method you want to use to enroll the certificates of the requesting router:
   - Simple Cisco Enrollment Protocol (SCEP).
   - Manual enrollment by terminal.
4. Enroll the certificate (SCEP) by means of the CA URL, or generate the certificate request for the public key of the router (manual method), depending on which method of enrollment you chose.

   Note If using the SCEP certificate enrollment method, this step completes the procedure. If you are using the manual method, continue to the next step.
5. Configure the certificate enrollment by cutting and pasting the retrieved certificate.

For details, see the *Implementing Certification Authority Interoperability on Cisco IOS XR Software* module in the *Cisco IOS XR System Security Configuration Guide*.

Configuration of IPSec in a Remote-Access VPN Topology

Configuration of IPSec for a remote-access topology differs from configuration for a site-to-site topology in the following ways:

- No tunnel destination is included as part of in the SVI configuration.
- You configure set type dynamic in the IPSec profile.
- The router cannot initiate sessions in dynamic mode.
- No keyring, because remote access is unknown.

Use of the following provides addresses for remote clients:

- Configuration of a local pool on the router (30,000 maximum, in other words, 30720 addresses).
- Use of extended DHCP.
- Definition of a static IP in RADIUS (TACACS not supported).

The SVI can terminate many site-to-site tunnels, which is useful for remote access configurations where all remote devices may share the same parameters.

Remote access uses the following procedure:

1. Configure AAA authorization and authentication.

   AAA is required to be configured for users and/or groups. The keys are stored in AAA and referenced from the ISAKMP profile, in Step 4.

See the *Configuring AAA Services on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide*.

2. Configure IP pool address on the router.

   See the example under Configuring IPSec in a Remote-Access VPN Topology: Example, page 151.

3. Configure the ISAKMP client pool.

   For details, see Defining Group Policy Information for Mode Configuration, page 174 of the module *Implementing Internet Key Exchange Security Protocol on Cisco IOS XR Software* in *Cisco IOS XR System Security Configuration Guide*.

4. Configure the ISAKMP profile.

   For details, see Configuring IKE Policies, page 172 of the module *Implementing Internet Key Exchange Security Protocol on Cisco IOS XR Software* in *Cisco IOS XR System Security Configuration Guide*.

5. Configure the IPSec profile.

   For details, see Configuring IKE Policies, page 172.

6. Configure the policy set.

   For details, see Configuring IKE Policies, page 172, Limiting an IKE Peer to Use a Specific Policy Set, page 178, and Configuring Static IPSec Virtual Interfaces, page 129.

See also Configuring IPSec in a Remote-Access VPN Topology: Example, page 151.

Information About Object Tracking in IPSec

The purpose of object tracking in Cisco IOS XR and in IPSec is to prevent the dropping of packets routed to destinations that have become unreachable due to network changes (also called “blackholing”).

This is achieved by tracking the state of the router interfaces or by tracking the prefixes used by the routing protocols. Any change to the state changes the value of the tracked object, as configured by the user, shutting previously defined service-IPSec interfaces. This causes the IPSec tunnel to be torn down and stops the flow of traffic to the black hole.

Cisco IOS XR currently supports tracking of the following:

- Physical interfaces
- Logical interfaces
- Routing protocol prefixes
- Host routes

The *Cisco IOS XR System Management Configuration Guide* provides detailed configuration information about each type of object tracking, while the “Configuration Examples for the Use of Object Tracking in IPSec” section on page 144 of this chapter illustrates how to use these same object tracking procedures in the context of IPSec.

How to Implement General IPSec Configurations for IPSec Networks

This section contains the following implementation procedures:

Setting Global Lifetimes for IPSec Security Associations

This task sets global lifetimes for IPSec security associations.

SUMMARY STEPS

1. **configure**
2. **crypto ipsec security-association lifetime** {**seconds** *seconds* | **kilobytes** *kilobytes*}
3. **end**
   or
   commit
4. **clear crypto ipsec sa** {*sa-id* | **all**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**crypto ipsec security-association lifetime** {**seconds** *seconds* | **kilobytes** *kilobytes*} **Example:** `RP/0/0/CPU0:router(config)# crypto ipsec security-association lifetime seconds 2700`	Changes global lifetime values used when negotiating IPSec SAs. • The **seconds** *seconds* keyword and argument change the global "timed" lifetime for IPSec SAs. This form of the command causes the SA to time out after the specified number of seconds have passed. • The **kilobytes** *kilobytes* keyword and argument change the global "traffic-volume" lifetime for IPSec SAs. This form of the command causes the SA to time out after the specified amount of traffic (in kilobytes) has passed through the IPSec "tunnel" using the SA.

	Command or Action	Purpose

Step 3	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config)# end`
	or
	`RP/0/0/CPU0:router(config)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 4	**clear crypto ipsec sa {*sa-id* | all}** **Example:** `RP/0/0/CPU0:router# clear crypto ipsec sa 100`	(Optional) Clears existing security associations, which causes any existing SAs to expire immediately. • Future SAs use the new lifetimes. • Any existing SAs expire according to the previously configured lifetimes. **Note** Using the **clear crypto ipsec sa** command with the **all** keyword clears the full SA database, which clears active security sessions. You may also specify the *sa-id* argument to clear an SA with a specific ID. For more information, see the **clear crypto ipsec sa** command.

Creating Crypto Access Lists

This task creates a crypto access list.

SUMMARY STEPS

1. **configure**
2. **ipv4 access-list** *name*
3. [*sequence-number*] **permit** {**ipv4** | *ipv4-protocol-number*} {**any** | **host** *source-ip* | *source-ip/prefix* | *source-ip source-wildcard*} {**any** | **host** *destination-ip* | *destination-ip/prefix* | *destination-ip destination-wildcard*}

   or

   [*sequence-number*] **permit** {**tcp** | **udp**}{ **any** | **host** *source-ip* | *source-ip/prefix* | *source-ip source-wildcard*}[**eq** *port-number* | **gt** *port-number* | **lt** *port-number* | **neq** *port-number* | **range** *port-number port-number*] {**any** | **host** *destination-ip* | *destination-ip/prefix* | *destination-ip destination-wildcard*} [**eq** *port-number*| **gt** *port-number* | **lt** *port-number* | **neq** *port-number* | **range** *port-number port-number*]
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**ipv4 access-list** *name*
	
	Example:
	`RP/0/0/CPU0:router(config)# ipv4 access-list InternetFilter`
	`RP/0/0/CPU0:router(config-ipv4-acl)#`
		Creates an access list named “InternetFilter” and enters IPv4 access list configuration mode.
		
		Note Only IPv4 access list configuration mode is relevant to creation of a crypto access list, not IPv6 access list configuration mode.

Command or Action	Purpose
Step 3 [*sequence-number*] **permit** {**ipv4** | *ipv4-protocol-number*} {**any** | **host** *source-ip* | *source-ip/prefix* | *source-ip source-wildcard*} {**any** | **host** *destination-ip* | *destination-ip/prefix* | *destination-ip destination-wildcard*} or [*sequence-number*] **permit** {**tcp** | **udp**}{**any** | **host** *source-ip* | *source-ip/prefix* | *source-ip source-wildcard*}[**eq** *port-number* | **gt** *port-number* | **lt** *port-number* | **neq** *port-number* | **range** *port-number port-number*] {**any** | **host** *destination-ip* | *destination-ip/prefix* | *destination-ip destination-wildcard*} [**eq** *port-number*| **gt** *port-number* | **lt** *port-number* | **neq** *port-number* | **range** *port-number port-number*] **Example:** `RP/0/0/CPU0:router(config-ipv4-acl)# 10 permit tcp 100.0.1.0 0.0.0.255 lt 15 30.0.0.0/16 range 2000 2050`	Specifies conditions to determine which IP packets are protected. • Enables crypto for traffic that matches these conditions. • In the first version of this step, any IPv4 protocol together with source and destination IP addresses can be used to define crypto traffic. • In the second version, either TCP or UDP protocol can be used to define crypto traffic, together with source and destination IP addresses, and optional selection of port numbers. ⚠ **Caution** Use the **any** keyword with caution. For details, see the "About Use of the any Keyword in Crypto Access Lists" section on page 126. **Note** Only those keywords that have a relationship to crypto access list creation are referenced here. For this reason, for example, the **deny** command has been omitted. This is because Cisco IOS XR software ignores an ACL if configured with the **deny** command and associated with an IPSec profile. • *sequence-number*—Specifies *a* sequence number to be associated with the protocol used to define crypto traffic. Range is from 1-2147483646 • *ipv4-protocol-number*— Specifies an IPv4 protocol number to be used to define crypto traffic. Range is from 0-255. • *port-number* —Specifies a port number used to define crypto traffic. You can define a range of port numbers using the **gt, lt, neq,** or **range** keyword. Range is from 0-65535. • **range** keyword—Specifies a range of port numbers. Range is from 0-65535. • In the example, an ACL is defined for traffic of a TCP protocol with a source address in the range of from 100.0.1.0 to 100.0.1.255, using a source port number of from 0 to 14, with a destination address in the range of from 30.0.0.0 to 30.0.255.255, and using any destination port in the range of from 2000 to 2050.

	Command or Action	Purpose

Step 4	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config)# end`
	or
	`RP/0/0/CPU0:router(config)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Defining Transform Sets

This task defines a transform set.

SUMMARY STEPS

1. **configure**
2. **crypto ipsec transform-set** *name*
   transform-set submode transform protocol
   transform-set submode **mode** {**transport** | **tunnel**}
3. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**crypto ipsec transform-set** *name*
	transform-set submode **transform** *protocol*
	transform-set submode **mode** {**transport** | **tunnel**}
	
	Example:
	`RP/0/0/CPU0:router(config)# crypto ipsec transform-set new`
	`RP/0/0/CPU0:router(config-transform-set new)# transform esp-sha-hmac`
		Defines a transform set.
		• Complex rules define which entries you can use for the transform arguments. These rules are explained in the command description for the **crypto ipsec transform-set** command.

Step 3	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config-transform-set new)# end`
	or
	`RP/0/0/CPU0:router(config-transform-set new)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Crypto Profiles

This task configures static or dynamic crypto profiles.

SUMMARY STEPS

1. **configure**
2. **crypto ipsec profile** *name*
3. **match** *acl-name* **transform-set** *transform-set-name*
4. **set pfs** {**group1** | **group2** | **group5**}
5. **set type** {**static** | **dynamic**}

6. **set transform-set** *transform-set-name*
7. **reverse-route**
8. **set security-association idle-time** *seconds*
9. **set security-association lifetime seconds** *seconds* **kilobytes** *kilobytes*
10. **set security-association replay disable**
11. **set session-key inbound ah** *spi hex-key-data*
12. **set session-key inbound esp** *spi* {**cipher** *hex-key-data* **authentication** *hex-key-data*}
13. **set session-key outbound ah** *spi hex-key-data*
14. **set session-key outbound esp** *spi* {**cipher** *hex-key-data* **authentication** *hex-key-data*}
15. **exit**
16. **end**
    or
    commit
17. **show crypto ipsec sa** [**sa-id** | **peer** *ip-address* | **profile** *profile-name* | **detail** | **fvrf** *fvrf-name* | **ivrf** *ivrf-name* | **location** *location*]
18. **show crypto ipsec summary**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**crypto ipsec profile** *name* **Example:** `RP/0/0/CPU0:router(config)# crypto ipsec profile new`	Creates the IPSec profile and enters profile configuration mode.
Step 3	**match** *acl-name* **transform-set** *transform-set-name* **Example:** `RP/0/0/CPU0:router(config-new)# match sampleacl transform-set tset1`	Configures the ACL to use for packet classification, and if the packets need protecting, the transform set to use for IPSec processing. **Note** You can configure up to five different transform-sets. The **match transform-set** command is used in profiles that are attached to service-ipsec interfaces, tunnel-ipsec interfaces, and transport. The description for this command is similar to the **set transform-set** command but used on a different interface.
Step 4	**set pfs** {**group1** | **group2** | **group5**} **Example:** `RP/0/0/CPU0:router(config-new)# set pfs group5`	(Optional) Specifies that IPSec should ask for perfect forward secrecy (PFS) when requesting new security associations for this crypto profile entry, or should demand PFS in requests received from the IPSec peer.

	Command or Action	Purpose
Step 5	**set type {static | dynamic}** **Example:** `RP/0/0/CPU0:router(config-new)# set type dynamic`	(Optional) Sets the profile mode type. • Default is static mode, which means that the peer is identified in the configuration. • Dynamic mode lets the profile be dynamic, which means that SA negotiation from any authenticated peer is allowed.
Step 6	**set transform-set** *transform-set-name* **Example:** `RP/0/0/CPU0:router(config-new)# set transform-set ts1`	Specifies a list of transform sets in priority order. The **set transform-set** command is used in profiles that are attached to service-gre interfaces. The description for this command is similar to the **match transform-set** command but used on a different interface. **Note** You can configure up to five different transform-sets. Use the *transform-set-name* argument to name the transform-set. The maximum characters is 32.
Step 7	**reverse-route** **Example:** `RP/0/0/CPU0:router(config-new)# reverse-route`	Creates source proxy information for a crypto profile entry.
Step 8	**set security-association idle-time** *seconds* **Example:** `RP/0/0/CPU0:router(config-new)# set security-association idle-time 800`	Specifies the maximum time in which the current peer can be idle before the default peer is used. • Use the *seconds* argument to specify the number of seconds for which the current peer can be idle before the default peer is used. The valid values are 600 to 86400.

Step 9	**set security-association lifetime seconds** *seconds* **kilobytes** *kilobytes*
	
	Example:
	`RP/0/0/CPU0:router(config-new)# set security-association lifetime seconds 2700`
	`RP/0/0/CPU0:router(config-new)# set security-association lifetime kilobytes 2304000`
		(Optional) Overrides (for a particular crypto profile entry) the global lifetime value, which is used when negotiating IP Security security associations.
		The example shows how to shorten lifetimes to reduce the risk that the keys could be compromised. The timed lifetime is shortened to 2700 seconds (45 minutes), and the traffic-volume lifetime is shortened to 2,304,000 KB (10 MBps for 30 minutes).
		• Use the **seconds** keyword to specify the number of seconds a security association lives before expiring. The range is from 120 to 86400.
		• Use the **kilobytes** keyword to specify the volume of traffic (in kilobytes) that can pass between IPSec peers using a given security association before that security association expires. The range is from 2560 to 536870912.

	Command or Action	Purpose
Step 10	**set security-association replay disable** **Example:** `RP/0/0/CPU0:router(config-new)# set security-association replay disable`	(Optional) Disables replay checking for a particular crypto profile.
Step 11	**set session-key inbound ah** *spi hex-key-data* **Example:** `RP/0/0/CPU0:router(config-new)# set session-key inbound ah 300 11111111111111111111111111111111111111111`	(Optional) Manually specifies the IP Security session keys to set the inbound IPSec session key for the Authentication Header (AH) protocol. The length of the keys should match the encryption or authentication method that is specified in the transform-set. • Use the *spi* argument to specify the security parameter index (SPI), a number that uniquely identifies a security association. The SPI is an arbitrary number you assign in the range of 256 to 4,294,967,295 (FFFF FFFF). • Use the *hex-key-data* argument to specify the session key; enter in hexadecimal format. This is an arbitrary hexadecimal string of 8, 16, or 20 bytes.
Step 12	**set session-key inbound esp** *spi* {**cipher** *hex-key-data* **authentication** *hex-key-data*} **Example:** `RP/0/0/CPU0:router(config-new)# set session-key inbound esp 300 cipher 0123456789012345 authentication 00001111222233334444555566667777888899999`	(Optional) Manually specifies the IP Security session key to set the inbound IPSec session key for Encapsulation Security Protocol (ESP). The length of the keys should match the encryption or authentication method that is specified in the transform-set. • Use the *spi* argument to specify the SPI, a number that is used to uniquely identify a security association. The SPI is an arbitrary number you assign in the range of 256 to 4,294,967,295 (FFFF FFFF). • Use the **cipher** keyword to specify the key string to be used with the ESP encryption transform. • Use the *hex-key-data* argument to specify the session key; enter in hexadecimal format. This is an arbitrary hexadecimal string of 8, 16, or 20 bytes. • Use the **authentication** keyword to specify that the key string is used with the ESP authentication transform. The **authentication** keyword is required only when the transform set includes an ESP authentication transform.

	Command or Action	Purpose
Step 13	**set session-key outbound ah** *spi hex-key-data* **Example:** `RP/0/0/CPU0:router(config-new)# set session-key outbound ah 300 fedcbafedcbafedcbafedcbafedcbafedcbafedc`	(Optional) Manually specifies the IP Security session key to set the outbound IPSec session key for the AH protocol. The length of the keys should match the encryption or authentication method that is specified in the transform-set. • Use the *spi* argument to specify the security parameter index (SPI), a number that uniquely identifies a security association. The SPI is an arbitrary number you assign in the range of 256 to 4,294,967,295 (FFFF FFFF). • Use the *hex-key-data* argument to specify the session key; enter in hexadecimal format. This is an arbitrary hexadecimal string of 8, 16, or 20 bytes.
Step 14	**set session-key outbound esp** *spi* {**cipher** *hex-key-data* **authentication** *hex-key-data*} **Example:** `RP/0/0/CPU0:router(config-new)# set session-key outbound esp 300 cipher abcdefabcdefabcd authentication 99998888777766665555444433332222111110000`	(Optional) Manually specifies the IP Security session key to set the outbound IPSec session key for ESP. The length of the keys should match the encryption or authentication method that is specified in the transform-set. • Use the *spi* argument to specify the SPI, a number that is used to uniquely identify a security association. The SPI is an arbitrary number you assign in the range of 256 to 4,294,967,295 (FFFF FFFF). • Use the **cipher** keyword to specify the key string to be used with the ESP encryption transform. • Use the *hex-key-data* argument to specify the session key; enter in hexadecimal format. This is an arbitrary hexadecimal string of 8, 16, or 20 bytes. • Use the **authentication** keyword to specify that the key string is used with the ESP authentication transform. The **authentication** keyword is required only when the transform set includes an ESP authentication transform.
Step 15	**exit** **Example:** `RP/0/0/CPU0:router(config-new)# exit`	Exits profile configuration mode.

	Command or Action	Purpose

Step 16	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config)# end`
	or
	`RP/0/0/CPU0:router(config)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 17	**show crypto ipsec sa** [**sa-id** | **peer** *ip-address* | **profile** *profile-name* | **detail** | **fvrf** *fvrf-name* | **ivrf** *ivrf-name* | **location** *location*] **Example:** `RP/0/0/CPU0:router# show crypto ipsec sa peer 172.19.72.120`	(Optional) Displays SA information based on the rack/slot/instance location. • Use the optional **detail** keyword to display additional dynamic SA information. The **detail** keyword is used only for software-based SAs. SAs that are configured under the tunnel-ipsec interface or crypto transport.
Step 18	**show crypto ipsec summary** **Example:** `RP/0/0/CPU0:router# show crypto ipsec summary`	(Optional) Displays IPSec summary information.

Configuring the DF Bit for the Encapsulating Header in IPSec Tunnels

This task configures the DF bit for the encapsulating header in IPSec tunnels. The DF bit configuration is also specified for both service-ipsec and service-gre interfaces.

Note This IPSec feature is supported only on the Cisco XR 12000 Series Router using the Cisco IPSec VPN SPA.

SUMMARY STEPS

1. **configure**
2. **crypto ipsec df-bit** {**clear** | **set** | **copy**}
3. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**crypto ipsec df-bit** {**clear** | **set** | **copy**} **Example:** `RP/0/0/CPU0:router(config)# crypto ipsec df-bit clear` or	Sets the DF bit for the encapsulating header in IPSec tunnels to all interfaces. You must specify at least one option for the **crypto ipsec df-bit** command. If no global setting is set, the default value is set to clear.

	Command or Action	Purpose

	Example:
	`RP/0/0/CPU0:router(config)# interface service-ipsec 5`
	`RP/0/0/CPU0:router(config-if)# crypto ipsec df-bit clear`
	`RP/0/0/CPU0:router(config-if)# crypto ipsec df-bit clear`
		Use the **crypto ipsec df-bit** command in global configuration mode and service-ipsec interface configuration mode.
		• (Optional) Use the **clear** keyword to specify that the outer IP header has the DF bit cleared and the router can fragment the packet to add the IPSec encapsulation.
		• (Optional) Use the **set** keyword to specify that the outer IP header has the DF bit set; however, the router can fragment the packet if the original packet had the DF bit cleared.
		• (Optional) Use the **copy** keyword to specify that the router looks in the original packet for the outer DF bit setting. The **copy** keyword is the default setting.

Step 3	**`end`**
	or
	`commit`
	
	Example:
	`RP/0/0/CPU0:router(config-if)# end`
	or
	`RP/0/0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring the IPSec Antireplay Window

This section describes the configuration of the IPSec antireplay window:

- Restrictions to Configuring the IPSec Antireplay Window, page 112
- Configuring the IPSec Antireplay Window: Expanding and Disabling Globally, page 112 (optional)
- Disabling IPSec Replay Checking on a Crypto Profile, page 113 (optional)

Restrictions to Configuring the IPSec Antireplay Window

This IPSec feature is supported only on the Cisco XR 12000 Series Router using the Cisco IPSec VPN SPA.

Configuring the IPSec Antireplay Window: Expanding and Disabling Globally

This task globally expands and disables the IPSec Antireplay Window globally.

SUMMARY STEPS

1. **configure**
2. **crypto ipsec security-association replay window-size** [*N*]
3. c**rypto ipsec security-association replay disable**
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**crypto ipsec security-association replay window-size** [*N*] **Example:** `RP/0/0/CPU0:router(config)# crypto ipsec security-association replay window-size 256`	Sets the size of the SA replay window globally. Use the *N* argument to specify the size of the window. Values are 64, 128, 256, 512, and 1024. This value becomes the default value. **Note** Configure this command or the **crypto ipsec security-association replay disable** command. The two commands are not used at the same time.

	Command or Action	Purpose
Step 3	**crypto ipsec security-association replay disable** **Example:** `RP/0/0/CPU0:router(config)# crypto ipsec security-association replay disable`	Disables checking globally. **Note** Configure this command or the **crypto ipsec security-association replay window-size** command. The two commands are not used at the same time.

Step 4	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config)# end`
	or
	`RP/0/0/CPU0:router(config)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Disabling IPSec Replay Checking on a Crypto Profile

This task disables replay checking on a crypto profile.

SUMMARY STEPS

1. **configure**
2. **crypto ipsec profile** *name*
3. **set security-association replay disable**
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**crypto ipsec profile** *name* **Example:** `RP/0/0/CPU0:router(config)# crypto ipsec profile myprofile`	Creates or modifies a crypto profile entry and enters profile configuration mode. • Use the *name* argument to specify the name of an IPSec profile. The maximum length is 32 characters.
Step 3	**set security-association replay disable** **Example:** `RP/0/0/CPU0:router(config-myprofile)# set security-association replay disable`	Disables replay checking for a particular crypto profile.

Step 4	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config-myprofile)# end`
	or
	`RP/0/0/CPU0:router(config-myprofile)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring IPSec NAT Transparency

Network Address Translator (NAT) is automatically detected by the Cisco IPSec VPN SPA. If both VPN devices are NAT-transparency capable, NAT transparency is automatically detected and negotiated. No configuration steps are needed to enable IPSec NAT transparency.

Note This IPSec feature is supported only on the Cisco XR 12000 Series Router using the Cisco IPSec VPN SPA.

Disabling IPSec NAT Transparency

This task disables NAT transparency if you already know that your network uses IPSec-awareness NAT (spi-matching scheme).

SUMMARY STEPS

1. **configure**
2. **crypto nat-transparency disable**
3. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**crypto nat-transparency disable** **Example:** `RP/0/0/CPU0:router(config)# crypto ipsec nat-transparency disable`	Disables the NAT transparency capability.

Step 3	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config)# end`
	or
	`RP/0/0/CPU0:router(config)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring IPSec Security Association Idle Timers

To configure the IPSec Security Association Idle Timers feature, you must understand the following concepts and tasks:

Restrictions to Configuring IPSec SA Idle Timers

This feature is configurable only on the Cisco XR 12000 Series Router using the Cisco IPSec VPN SPA.

Lifetimes for IPSec Security Associations

Cisco IOS XR software currently allows the configuration of lifetimes for IPSec SAs. Lifetimes can be configured globally or for each crypto profile. Two lifetimes exist: a "timed" lifetime and a "traffic-volume" lifetime. A security association expires after the first of these lifetimes is reached.

IPSec Security Association Idle Timers

The IPSec SA idle timers are different from the global lifetimes for IPSec SAs. The expiration of the global lifetime is independent of peer activity. The IPSec SA idle timer allows SAs associated with inactive peers to be deleted before the global lifetime has expired.

If the IPSec SA idle timers are not configured, only the global lifetimes for IPSec SAs are applied. SAs are maintained until the global timers expire, regardless of peer activity.

Note If the last IPSec SA to a given peer is deleted because of idle timer expiration, the Internet Key Exchange (IKE) SA to that peer is also deleted.

Configuring the IPSec SA Idle Timer Globally

This task configures IPSec security association (SA) idle timers globally.

SUMMARY STEPS

1. **configure**
2. **crypto ipsec security-association idle-time** *seconds*
3. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

	Command or Action	Purpose
Step 2	**crypto ipsec security-association idle-time** *seconds* **Example:** `RP/0/0/CPU0:router(config)# crypto ipsec security-association idle-time 600`	Configures the IPSec SA idle timer globally. • Use the *seconds* argument to specify the time, in seconds, that the idle timer allows an inactive peer to maintain an SA. Valid values for the *seconds* argument range from 600 to 86400.

Step 3	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config)# end`
	or
	`RP/0/0/CPU0:router(config)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring the IPSec SA Idle Timer for Each Crypto Profile

This task configures the IPSec SA idle timer for a specified crypto profile. The idle timer configuration is applied to all SAs under the specified crypto profile. If no idle time is specified under the profile and idle time is configured in global mode, the idle time of the global mode is applied.

SUMMARY STEPS

1. **configure**
2. **crypto ipsec profile** *name*
3. **set security-association idle-time** *seconds*
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**crypto ipsec profile** *name*
	
	Example:
	`RP/0/0/CPU0:router(config)# crypto ipsec profile myprofile`
	`RP/0/0/CPU0:router(config-myprofile)#`
		Creates or modifies a crypto profile entry and enters profile configuration mode.
		• Use the *name* argument to specify the name of an IPSec profile. The maximum length is 32 characters.

Step 3	**set security-association idle-time** *seconds* **Example:** `RP/0/0/CPU0:router(config-myprofile)# set security-association idle-time 800`	Specifies the maximum time in which the current peer can be idle before the default peer is used. • Use the *seconds* argument to specify the number of seconds in which the current peer can be idle before the default peer is used. Valid values are 60 to 86400.

Step 4	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config-myprofile)# end`
	or
	`RP/0/0/CPU0:router(config-myprofile)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Disabling Prefragmentation for Cisco IPSec VPN SPAs

This section provides the following procedures to disable prefragmentation for Cisco IPSec VPN SPAs on the Cisco XR 12000 Series Router:

- Disabling Prefragmentation for service-ipsec Interfaces, page 120
- Disabling Prefragmentation for service-gre Interfaces, page 121

Note This IPSec feature is supported only on the Cisco IPSec VPN SPA.

Disabling Prefragmentation for service-ipsec Interfaces

This task disables prefragmentation for service-ipsec interfaces.

SUMMARY STEPS

1. **configure**
2. **crypto ipsec pre-fragmentation disable**
3. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose

Step 1	`configure`
	
	Example:
	`RP/0/0/CPU0:router# configure`
		Enters global configuration mode.

	Command or Action	Purpose

Step 2	**crypto ipsec pre-fragmentation disable**
	
	Example:
	`RP/0/0/CPU0:router(config)# crypto ipsec pre-fragmentation disable`
	
	or
	
	Example:
	`RP/0/0/CPU0:router(config)# interface service-ipsec 5`
	`RP/0/0/CPU0:router(config-if)# crypto ipsec pre-fragmentation disable`
		Specifies the handling of fragmentation for the near-MTU-sized packets.
		• Use the **disable** keyword to disable the fragmentation of large packets before IPSec encapsulation.
		You can use the **crypto ipsec pre-fragmentation** command in global configuration mode or service-ipsec interface configuration mode.

Step 3	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config)# end`
	or
	`RP/0/0/CPU0:router(config)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Disabling Prefragmentation for service-gre Interfaces

The fragmentation for a service-gre interface is done based on the original IP packet size, which does not include the GRE overhead.

This task disables prefragmentation for service-gre interfaces.

SUMMARY STEPS

1. **configure**
2. **crypto ipsec pre-fragmentation disable**

3. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**crypto ipsec pre-fragmentation disable**
	
	Example:
	`RP/0/0/CPU0:router(config)# crypto ipsec pre-fragmentation disable`
	
	or
	
	Example:
	`RP/0/0/CPU0:router(config)# interface service-gre 500`
	`RP/0/0/CPU0:router(config-if)# crypto ipsec pre-fragmentation disable`
		Specifies the handling of fragmentation for the near-MTU-sized packets.
		• Use the **disable** keyword to disable the fragmentation of large packets before IPSec encapsulation.
		You can use the **crypto ipsec pre-fragmentation** command in global configuration mode or service-gre interface configuration mode.

Step 3	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config)# end`
	or
	`RP/0/0/CPU0:router(config)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Reverse-Route Injection in a Crypto Profile

This task configures reverse-route injection in a crypto profile, allowing insertion of either static or dynamically learned routes to take routing precedence primarily in networks and hosts protected by a remote tunnel endpoint. The ability to use dynamically learned routes is, for example, key to effecting failover policy in load balancing.

Note Routes learned during IPSec tunnel establishment are removed when the IPSec tunnel is torn down.

For information about reverse-route injection, see IPSec—SNMP Support, page 89.

Note This IPSec feature is supported only on the Cisco XR 12000 Series Router using the Cisco IPSec VPN SPA.

SUMMARY STEPS

1. **configure**
2. **crypto ipsec profile** *name*
3. **reverse-route** {[**distance** *value*] [**tag** *value*]}
4. **end**
   or
   commit
5. **show route** [**vrf** *vrf name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**crypto ipsec profile** *name*
	
	Example:
	`RP/0/0/CPU0:router(config)# crypto ipsec profile myprofile`
	`RP/0/0/CPU0:router(config-myprofile)#`
		Creates or modifies a crypto profile entry and enters profile configuration mode.
		
		Use the *name* argument to specify the name of an IPSec profile. Maximum length is 32 characters.

	Command or Action	Purpose
Step 3	**reverse-route** {[**distance** *value*] [**tag** *value*]} **Example:** `RP/0/0/CPU0:router(config-myprofile)# reverse-route distance 255 tag 200`	Configures the administrative distance or identifies a routing policy value associated with a predefined group, or both: • **distance—**Configures an administrative distance of from 1 to 255. A static route always takes precedence. The default is 0. • **tag—**Configures the route tag, which can be from 1 to 497777. When you add a tag to a route, you associate a value with a predefined group that allows you to manipulate the routing policy on all the routes that share the same tag value. **Note** You may configure either the **distance** value, the **tag** value, or both.

Step 4	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config-myprofile)# end`
	or
	`RP/0/0/CPU0:router(config-myprofile)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	**show route** [**vrf** *vrf name*] **Example:** `RP/0/0/CPU0:router#`	Displays the proxy information that has been added as static routes. The SAs, which are created on these interfaces, are also called software-based SAs.

Configuring IPSec Failure History Table Size

This task changes the size of the failure history table.

Note This IPSec feature is supported only on the Cisco XR 12000 Series Router using Cisco IPSec VPN SPA.

SUMMARY STEPS

1. **configure**
2. **crypto mib ipsec flowmib history failure size** *number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**crypto mib ipsec flowmib history failure size** *number* **Example:** `RP/0/0/CPU0:router(config)# crypto mib ipsec flowmib history failure size 140`	Sets the size of the failure history table.

How to Implement IPSec Network Security for Locally Sourced and Destined Traffic

Locally sourced and terminated traffic are evaluated against IPSec profiles that are attached to tunnel-ipsec interfaces or crypto transport.

Note

- Multiple profiles can be attached to a tunnel-ipsec interface or crypto transport.
- For locally sourced traffic or terminated traffic, we discourage the use of the **any** keyword to specify source or destination addresses in the crypto profiles, which are attached to the tunnel-ipsec interface or transport. This recommendation is only for locally sourced traffic for VPN transit traffic. You can encrypt all the traffic going through the interface. Therefore, ACLs in profiles, which are attached to service-ipsec interfaces, can use the **any** keyword).

This section contains the following procedures:

- About Use of the any Keyword in Crypto Access Lists, page 126
- Applying Crypto Profiles to tunnel-ipsec Interfaces, page 126
- Applying Crypto Profiles to Crypto Transport, page 127

About Use of the any Keyword in Crypto Access Lists

When you create crypto access lists, using the **any** keyword could cause problems. We discourage the use of the **any** keyword to specify source or destination addresses. The **any** keyword is relevant only to locally sourced or terminated traffic.

No concept of default access lists exists for IPSec.

The **permit any any** statement is strongly discouraged, because it causes all outbound traffic to be protected (and all protected traffic to be sent to the peer specified in the corresponding crypto profile entry) and requires protection for all inbound traffic. Then, all inbound packets that lack IPSec protection are silently dropped, including packets for routing protocols, NTP, echo, and echo response.

Be sure to define which packets to protect. If you *must* use the **any** keyword in a **permit** statement, you must preface that statement with a series of **deny** statements to filter any traffic (that would otherwise fall within that **permit** statement) that you do not want to be protected.

Applying Crypto Profiles to tunnel-ipsec Interfaces

This task applies a crypto IPsec profile to a tunnel-ipsec interface.

You must apply a crypto profile to each tunnel-ipsec interface through which IPSec traffic flows. Applying the crypto profile set to a tunnel-ipsec interface instructs the router to evaluate all the interface's traffic against the crypto profile set and to use the specified policy during connection or SA negotiation on behalf of traffic to be protected by crypto.

SUMMARY STEPS

1. **configure**
2. **interface tunnel-ipsec** *interface-number*
3. **profile** *profile-name*
4. **tunnel source** *ip-address*
5. **tunnel destination** *ip-address*
6. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure interface`	Enters global configuration mode.
Step 2	**interface tunnel-ipsec** *interface-number* **Example:** `RP/0/0/CPU0:router(config)# interface tunnel-ipsec 0`	Identifies the IPSec interface to which the crypto profile is attached. You can use the **interface tunnel-ipsec** command to enter tunnel-ipsec interface configuration mode.

	Command or Action	Purpose
Step 3	**profile** *profile-name* **Example:** `RP/0/0/CPU0:router(config-if)# profile sample1`	Specifies the crypto profile to use in IPSec processing. • The same crypto profile cannot be shared in different IPSec modes.
Step 4	**tunnel source** *ip-address* **Example:** `RP/0/0/CPU0:router(config-if)# tunnel source 10.0.0.2`	Specifies the tunnel source IP address. • This command is required for both static and dynamic profiles.
Step 5	**tunnel destination** *ip-address* **Example:** `RP/0/0/CPU0:router(config-if)# tunnel destination 10.0.0.5`	Specifies the tunnel destination IP address. • This command is not required if the profile is dynamic.

Step 6	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config-if)# end`
	or
	`RP/0/0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Applying Crypto Profiles to Crypto Transport

This task applies a crypto profile to crypto transport.

You need to apply a crypto profile to transport mode to make the profile active. Applying the crypto profile set to transport instructs the router to evaluate all of the locally sourced traffic against the crypto profile set and use the specified policy during connection or SA negotiation on behalf of traffic to be protected by crypto.

SUMMARY STEPS

1. **configure**
2. **crypto ipsec transport**

3. **profile** *profile-name*
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**crypto ipsec transport** **Example:** `RP/0/0/CPU0:router(config)# crypto ipsec transport`	Enters IPSec transport configuration mode. • In the IPSec transport configuration mode, IPSec protects the Upper Layer Protocol (ULP) header and the payload. IPSec transport configuration mode is used when security is desired end to end. That is, security endpoints are the same as host endpoints.
Step 3	**profile** *profile-name* **Example:** `RP/0/0/CPU0:router(config-transport)# profile sample2`	Specifies the crypto profile to use in IPSec processing.

Step 4	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config-transport)# end`
	or
	`RP/0/0/CPU0:router(config-transport)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

How to Implement IPSec Network Security for VPNs

To implent IPSec network security for VPNs, review the following information:

Restrictions to Implementing IPSec Network Security for VPNs

This feature is supported only on the Cisco XR 12000 Series Router using the Cisco IPSec VPN SPA.

Configuring IPSec Service Virtual Interfaces

These tasks configure IPSec virtual interfaces:

Configuring Static IPSec Virtual Interfaces

This task configures static IPSec service virtual interfaces (SVIs).

SUMMARY STEPS

1. **configure**
2. **interface service-ipsec** *number*
3. **profile** *profile-name*
4. **tunnel source** *ip-address*
5. **tunnel destination** *ip-address*
6. **tunnel vrf** *vrf-name*
7. **vrf** *vrf-name*
8. **ipv4 address** *ipv4-address mask* [**secondary**]
9. **service-location preferred-active** *location* [**preferred-standby** *location* [**auto-revert**]]
10. **end**
    or
    commit
11. **show route** [**vrf** *vrf name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**interface service-ipsec** *number*
	
	Example:
	`RP/0/0/CPU0:router(config)# interface service-ipsec 2`
	`RP/0/0/CPU0:router(config-if)#`
		Creates a static IPSec SVI.
		You can use the **interface service-ipsec** command to enter service-ipsec interface configuration mode.

Step 3	**profile** *profile-name* **Example:** `RP/0/0/CPU0:router(config-if)# profile ipsec_profa`	Specifies the crypto profile to use for IPSec processing. • Use the *profile-name* argument to define the previous crypto profile to use. The character range is from 1 to 32 characters.
Step 4	**tunnel source** {*ip-address*} **Example:** `RP/0/0/CPU0:router(config-if)# tunnel source 172.19.72.92`	Specifies the source address for a tunnel-ipsec interface. • Use the *ip-address* argument to set the IP address to use as the source address for packets in the tunnel.
Step 5	**tunnel destination** *ip-address* **Example:** `RP/0/0/CPU0:router(config-if)# tunnel destination 172.19.72.120`	Identifies the IP address of the tunnel destination. • Use the *ip-address* argument to set the IP address of the host destination. If the IPSec profile is a dynamic, the tunnel destination should not be configured.
Step 6	**tunnel vrf** *vrf-name* **Example:** `RP/0/0/CPU0:router(config-if)# tunnel vrf internet`	Associates a VRF instance with the tunnel source or destination of the interfaces. The tunnel VRF specifies in which VRF the tunneled traffic is forwarded (FVRF). Tunnel VRF is not required if FVRF is the global VRF. • Use the *vrf-name* argument to assign the name of a VRF.
Step 7	**vrf** *vrf-name* **Example:** `RP/0/0/CPU0:router(config-if)# vrf vpn_a`	Assigns a VRF to the interface. VRF is specified to clear traffic that is forwarded for the internal VRF (IVRF). In addition, VRF is not required if IVRF is a global VRF. • Use the *vrf-name* argument to assign the name of a VRF.

	Command or Action	Purpose
Step 8	**ipv4 address** *ipv4-address mask* [**secondary**] **Example:** `RP/0/0/CPU0:router(config-if)# ipv4 address 192.168.1.27 255.255.255.0`	Sets a primary or secondary IPv4 address for an interface, for example, POS interface. • Use the *ipv4-address* argument to set the IPv4 address. • Use the *mask* argument to set the mask for the associated IP subnet. The network mask is specified in either of two ways: – The network mask is a four-part dotted decimal address. For example, 255.0.0.0 indicates that each bit equal to 1 means the corresponding address bit belongs to the network address. – The network mask is indicated as a slash (/) and number. For example, /8 indicates that the first 8 bits of the mask are ones, and the corresponding bits of the address are network address. • (Optional) Use the **secondary** keyword to specify that the configured address is a secondary IPv4 address. If this keyword is omitted, the configured address is the primary IPv4 address.
Step 9	**service-location preferred-active** *location* [**preferred-standby** *location* [**auto-revert**]] **Example:** `RP/0/0/CPU0:router(config-if)# service-location preferred-active 0/0/0 preferred-standby 0/1/0`	Specifies both active and standby locations for the interface. • Use the **preferred-active** keyword to specify that the SPA in this location serves all traffic going through the interface. The location argument is expressed in *rack/slot/module* notation. • (Optional) Use the **preferred-standby** keyword to specify that if a SPA fails, the interface is served by the SPA in this location. The location argument is expressed in *rack/slot/module* notation. • (Optional) Use the **auto-revert** keyword to revert to the preferred-active location if the **auto-revert** keyword is configured. **Note** The **auto-revert** keyword is specified only if the **preferred-standby** keyword with the *location* argument is configured.

	Command or Action	Purpose

Step 10	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config-if)# end`
	or
	`RP/0/0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 11	**show route** [**vrf** *vrf name*] **Example:** `RP/0/0/CPU0:router# show route vrf`	Displays the proxy information that was added as static routes.

Configuring IPSec-Protected GRE Virtual Interfaces

This task configures IPSec-protected GRE service virtual interfaces.

SUMMARY STEPS

1. **configure**
2. **interface service-gre** *number*
3. **profile** *profile-name*
4. **tunnel source** {*ip-address*}
5. **tunnel destination** *ip-address*
6. **tunnel vrf** *vrf-name*
7. **vrf** *vrf-name*
8. **ipv4 address** *ipv4-address mask* [**secondary**]
9. **service-location preferred-active** *location* [**preferred-standby** *location*] [**auto-revert**]

10. **end**
    or
    commit
11. **show route** [**vrf** *vrf name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**interface service-gre** *number*
	
	Example:
	`RP/0/0/CPU0:router(config)# interface service-gre 2`
	`RP/0/0/CPU0:router(config-if)#`
		Creates a GRE service virtual interface.
		You can use the **interface service-gre** command to enter service-gre interface configuration mode

Step 3	**profile** *profile-name* **Example:** `RP/0/0/CPU0:router(config-if)# profile ipsec_profa`	Specifies the crypto profile to use for IPSec processing. For the service-gre interface, the IPSec profile must be static. • Use the *profile-name* argument to define the previous crypto profile to use. The character range is from 1 to 32 characters.
Step 4	**tunnel source** {ip-address} **Example:** `RP/0/0/CPU0:router(config-if)# tunnel source 172.19.72.92`	Specifies the source address for a tunnel-ipsec interface. • Use the *ip-address* argument to set the IP address to use as the source address for packets in the tunnel.
Step 5	**tunnel destination** *ip-address* **Example:** `RP/0/0/CPU0:router(config-if)# tunnel destination 172.19.72.120`	Identifies the IP address of the tunnel destination. • Use the *ip-address* argument to set the IP address of the host destination. If dynamic, the destination IP address is optional.
Step 6	**tunnel vrf** *vrf-name* **Example:** `RP/0/0/CPU0:router(config-if)# tunnel vrf internet`	Associates a VRF instance with the tunnel source or destination of the interfaces. The tunnel VRF specifies in which VRF the tunneled traffic is forwarded (FVRF). Tunnel VRF is not required if FVRF is the global VRF. • Use the *vrf-name* argument to assign the name of a VRF.
Step 7	**vrf** *vrf-name* **Example:** `RP/0/0/CPU0:router(config-if)# vrf vpn_a`	Assigns a VRF to the interface. VRF is specified to clear traffic that is forwarded for the internal VRF (IVRF). In addition, VRF is not required if IVRF is a global VRF. • Use the *vrf-name* argument to assign the name of a VRF.

	Command or Action	Purpose
Step 8	**ipv4 address** *ipv4-address mask* [**secondary**] **Example:** `RP/0/0/CPU0:router(config-if)# ipv4 address 192.168.1.27 255.255.255.0`	Sets a primary or secondary IPv4 address for an interface, for example, a POS interface. • Use the *ipv4-address* argument to set the IPv4 address. • Use the *mask* argument to set the mask for the associated IP subnet. The network mask is specified in either of two ways: – The network mask is a four-part dotted decimal address. For example, 255.0.0.0 indicates that each bit equal to 1 means that the corresponding address bit belongs to the network address. – The network mask is indicated as a slash (/) and number. For example, /8 indicates that the first 8 bits of the mask are ones, and the corresponding bits of the address are the network address. • (Optional) Use the **secondary** keyword to specify that the configured address is a secondary IPv4 address. If this keyword is omitted, the configured address is the primary IPv4 address.
Step 9	**service-location preferred-active** *location* [**preferred-standby** *location* [**auto-revert**] **Example:** `RP/0/0/CPU0:router(config-if)# service-location preferred-active 0/0/0 preferred-standby 0/1/0`	Specifies both active and standby locations for the interface. • Use the **preferred-active** keyword to specify that the SPA in this location serves all traffic going through the interface. The location argument is expressed in *rack/slot/module* notation. • (Optional) Use the **preferred-standby** keyword to specify that if a SPA fails, the interface is served by the SPA in this location. The location argument is expressed in *rack/slot/module* notation. • (Optional) Use the **auto-revert** keyword to revert to the preferred-active location if the **auto-revert** keyword is configured. **Note** The **auto-revert** keyword is specified only if the **preferred-standby** keyword with the *location* argument is configured.

	Command or Action	Purpose

Step 10	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config-if)# end`
	or
	`RP/0/0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 11	**show route** [**vrf** *vrf name*] **Example:** `RP/0/0/CPU0:router# show route vrf`	Displays the proxy information that was added as static routes.

Configuring the Default Path Maximum Transmission Unit for the SA

This task configures the default path maximum transmission unit (MTU) for the SA.

SUMMARY STEPS

1. **configure**
2. **interface service-ipsec** *number*
3. **crypto ipsec pmtu** *pmtu*
4. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**interface service-ipsec** *number*
	
	Example:
	`RP/0/0/CPU0:router(config)# interface service-ipsec 2`
	`RP/0/0/CPU0:router(config-if)#`
		Creates a static IPSec SVI.
		You can use the **interface service-ipsec** command to enter service-ipsec interface configuration mode.

Step 3	**crypto ipsec pmtu** *pmtu* **Example:** `RP/0/0/CPU0:router(config-if)# crypto ipsec pmtu 1500`	Specifies the default path MTU for the SAs that are created under the interface. • Use the *pmtu* argument to specify the value of MTU in bytes. The range is from 68 to 9216.

Step 4	**end**
	or
	commit
	
	Example:
	`RP/0/0/CPU0:router(config-if)# end`
	or
	`RP/0/0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuration Examples for Implementing IPSec Network Security for Locally Sourced and Destined Traffic

This section provides the following configuration examples:

Configuring a Static Profile and Attaching to a tunnel-ipsec Interface: Example

The following example shows a minimal IPSec configuration where a static crypto profile is created and attached to a tunnel-ipsec interface.

An IPSec access list named sample1 defines which traffic to protect:

```
ipv4 access-list sample1 permit ip 10.0.0.0 0.0.0.255 10.2.2.0 0.0.0.255
```

A transform set defines how the traffic is protected. In this example, transform set myset1 uses Data Encryption Standard (DES) encryption and Secure Hash Algorithm (SHA) for data packet authentication:

```
crypto ipsec transform-set myset1
   transform esp-des esp-sha
```

Another transform set example is myset2, which uses 3DES encryption and the Message Digest 5 (MD5) (Hashed Message Authentication Code [HMAC] variant) algorithm for data packet authentication:

```
crypto ipsec transform-set myset2
   transform esp-3des esp-md5-hmac
```

A crypto profile named toRemoteSite is created and joins the IPSec access list and transform set:

```
crypto ipsec profile toRemoteSite
    match sample1 transform-set myset1
    end
```

The toRemoteSite crypto profile is then applied to a tunnel-ipsec interface:

```
interface tunnel-ipsec0
    profile toRemoteSite
    tunnel source 10.0.0.2
    tunnel destination 10.0.0.5
```

Configuring a Dynamic Profile and Attaching It to a tunnel-ipsec Interface: Example

The following example shows a minimal IPSec configuration where a dynamic crypto profile is created and attached to a tunnel-ipsec interface.

An IPSec access list named sample2 defines which traffic to protect:

```
ipv4 access-list sample2 permit ip 10.0.0.0 0.0.0.255 10.2.2.0 0.0.0.255
```

A transform set defines how the traffic is protected. In this example, transform set myset2 uses DES encryption and SHA for data packet authentication:

```
crypto ipsec transform-set myset2
   transform esp-des esp-sha
```

Another transform set example is myset3, which uses 3DES encryption and MD5 (HMAC variant) for data packet authentication:

```
crypto ipsec transform-set myset3
   transform esp-3des esp-md5-hmac
```

A dynamic crypto profile named toRemoteSite is created and joins the IPSec access list and transform set:

```
crypto ipsec profile toRemoteSite
    match sample2 transform-set myset3
    set type dynamic
    end
```

The toRemoteSite profile is applied to a tunnel-ipsec interface:

```
interface tunnel-ipsec0
    profile toRemoteSite
    tunnel source 10.0.0.2
```

The tunnel destination is not required when the profile is dynamic.

Configuring a Static Profile and Attaching to Transport: Example

The following example shows a minimal IPSec configuration in which a static profile is created and attached to a transport.

An IPSec access list named sample3 defines which traffic to protect:

```
ipv4 access-list sample3 permit ip 10.0.0.0 0.0.0.255 10.2.2.0 0.0.0.255
```

A transform set defines how the traffic is protected. In this example, transform set myset1 uses DES encryption and SHA for data packet authentication:

```
crypto ipsec transform-set myset1
   transform esp-des esp-sha
```

Another transform set example is myset2, which uses 3DES encryption and the MD5 (HMAC variant) for data packet authentication:

```
crypto ipsec transform-set myset2
   transform esp-3des esp-md5-hmac
```

A crypto profile named toRemoteSite is created and joins the IPSec access list and transform set:

```
crypto ipsec profile toRemoteSite
    match sample3 transform-set myset2
    end
```

The toRemoteSite profile is applied to a transport:

```
crypto ipsec transport
    profile toRemoteSite
    end
```

Configuration Examples for an IPSec Network with a Cisco IPSec VPN SPA

This section provides the following configuration examples:

Displaying the SPA Hardware Type: Example

The following sample output is from the **show diags** command on the Cisco XR 12000 Series Router IPSec VPN SPA installed in slot 1:

```
RP/0/0/CPU0:router# show diags

SLOT  1 (RP/LC 1): Cisco 12000 Series SPA Interface Processor-600
  MAIN: type 117,  800-26102-01 rev B0 dev 0
        HW config: 0x01    SW key: 00-00-00
  PCA:  73-9863-02 rev B0 ver 8
        HW version 1.0  S/N SAD092306A5
  MBUS: Embedded Agent
        Test hist: 0x00    RMA#: 00-00-00    RMA hist: 0x00
  DIAG: Test count: 0x00000000    Test results: 0x00000000
  FRU:  Linecard/Module: 12000-SIP-600
        Route Memory: MEM-LC5-1024=
        Packet Memory: MEM-LC5-PKT-512=
  L3 Engine: 5 - ISE OC192 (10 Gbps)
  MBUS Agent Software version 2.49 (RAM) (ROM version is 3.6)
  Using CAN Bus A
  ROM Monitor version 17.1
  Fabric Downloader version used 3.9 (ROM version is 3.9)
  Primary clock is CSC1
  Board State is IOS-XR RUN
  Insertion time: Thu Jun 29 23:23:48 2006 (1w0d ago)
  DRAM size: 1073741824 bytes
  FrFab SDRAM size: 268435456 bytes
  ToFab SDRAM size: 268435456 bytes
  0 crashes since restart/fault forgive

  SPA Information:
        subslot 0/1/0: SPA-IPSEC-2G (0x3d7), status is ok
```

Table 2 lists the hardware description that appears in the **show diags** command output for a Cisco XR 12000 Series Router IPSec VPN SPA.

Table 2 SPA Hardware Description in show diag Command

SPA	Description in show diag Command
SPA-IPSEC-2G	SPA-IPSEC-2G (0x3d7)

For more information about the **show diag** command, see *Cisco IOS XR Interface and Hardware Component Command Reference*. For information about the related **show platform** command, see *Cisco IOS XR System Management Command Reference.*

Configuring IPSec for a VRF-Aware service-ipsec Interface: Example

The following example shows an IPSec configuration of two VRF-aware service-ipsec interfaces with a crypto IPSec profile that uses RRI.

The **interface service-ipsec** command is set to 1 and is part of the customer_1 VRF. FVRF is the global VRF (default). Clear traffic is coming from customer_1 VRF with a source IP address 100.0.1.0/24 and is destined to 30.0.1.0/24, which is encrypted and sent over to the global VRF. Respectively, the encrypted traffic from 30.0.1.0/24 is destined to 100.0.1.0/24 and is encrypted on the remote site or host and decrypted on the router.

Configuring VRF

```
vrf customer_1
 address-family ipv4 unicast
  import route-target
   100:1000
  !
  export route-target
   100:1000
  !
 !
!
```

Configuring ACL That Is Used by the IPSec Profile

```
ipv4 access-list acl1
 10 permit ipv4 100.0.1.0 0.0.0.255 30.0.1.0 0.0.0.255
!
```

Configuring the Service-ipsec Interface

```
interface service-ipsec1
 vrf customer_1  <------------- IVRF
 ipv4 address 40.40.41.41 255.255.255.0
 profile prof1   <--------- the ipsec profile
 tunnel source 4.0.1.1
 tunnel destination 5.0.1.1
 service-location preferred-active 0/1/1 preferred-standby 0/2/0 <----------- The IPSec
SPA is located on the 0/1/1 and the standby SPA on  0/2/0
!
```

Configuring IKE

```
crypto isakmp
crypto isakmp policy 1
 authentication pre-share
 encryption 3des
 lifetime 86400
!
crypto keyring kr1 vrf default
 pre-shared-key address 5.0.1.1 255.255.255.255 key aBrAkAdAbRa

crypto isakmp profile a_prof
 keyring kr1
 match identity address 5.0.1.1/32 vrf default
  set interface service-ipsec1
 !
```

Configuring IPSec

The following example shows that the transform-set is set to esp-256-aes:

```
crypto ipsec transform-set ts1
 transform esp-256-aes
!
```

The following example shows that the IPSec profile uses acl1 as the traffic proxy and transform-set is ts1. In addition, RRI is configured.

```
crypto ipsec profile prof1
 set pfs group1
 set type static
 match acl1 transform-set ts1
 reverse-route
!
```

The following example shows that the IPSec SA is created from the **show crypto ipsec summary** command and **show crypto ipsec sa** command:

```
RP/0/0/CPU0:router# show crypto ipsec summary

# * Attached to a transform indicates a bundle

# Active IPSec Sessions: 1

SA    Local Peer      Remote Peer     FVRF      Profile   Transform    Lifetime
-------------------------------------------------------------------------------
502   4.0.1.1         5.0.1.1         default   prof1     esp-256-aes  3600/4194303

RP/0/0/CPU0:router# show crypto ipsec sa

SA id:          502
Node id:        0/1/1 0/2/0
SA Type:        ISAKMP
interface:      service-ipsec1
profile  :      prof1
local  ident (addr/mask/prot/port) : (100.0.1.0/255.255.255.0/0/0)
remote ident (addr/mask/prot/port) : (30.0.1.0/255.255.255.0/0/0)
local crypto endpt: 4.0.1.1, remote crypto endpt: 5.0.1.1, vrf default

 #pkts tx         :0                     #pkts rx         :0
 #bytes tx        :0                     #bytes rx        :0
 #pkts encrypt    :0                     #pkts decrypt    :0
 #pkts digest     :0                     #pkts verify     :0
 #pkts encrpt fail:0                     #pkts decrpt fail:0
 #pkts digest fail:0                     #pkts verify fail:0
 #pkts replay fail:0
 #pkts tx errors  :0                     #pkts rx errors  :0

outbound esp sas:
        spi: 0x3482d5c8(880989640)
        transform: esp-256-aes
        in use settings = Tunnel
        sa agreed lifetime: 3600s, 4194303kb
        sa timing: remaining key lifetime (sec/kb): (3525/4194303)
        sa DPD disabled
        sa idle timeout: disable, 0s
        sa anti-replay (HW accel): enable, window 64
inbound esp sas:
        spi: 0x3c9869ee(1016621550)
        transform: esp-256-aes
        in use settings = Tunnel
```

```
        sa agreed lifetime: 3600s, 4194303kb
        sa timing: remaining key lifetime (sec/kb): (3525/4194303)
        sa DPD disabled
        sa idle timeout: disable, 0s
        sa anti-replay (HW accel): enable, window 64
```

The following example shows that RRI was configured so the proxy is added to the routing table of the VRF:

```
RP/0/0/CPU0:router# show route vrf customer_1

Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
       U - per-user static route, o - ODR, L - local

Gateway of last resort is not set

S    30.0.1.0/24 is directly connected, 00:02:09, service-ipsec1
C    40.40.41.0/24 is directly connected, 00:02:09, service-ipsec1
L    40.40.41.41/32 is directly connected, 00:02:09, service-ipsec1
C    100.100.100.0/24 is directly connected, 00:01:26, GigabitEthernet0/0/0/3
L    100.100.100.1/32 is directly connected, 00:01:26, GigabitEthernet0/0/0/3
```

The following example shows that the **interface service-ipsec** command is set to 1 and is part of the customer_1 VRF:

```
RP/0/0/CPU0:router# show crypto ipsec interface service-ipsec 1

--------------- IPSec interface ----------------
Interface service-ipsec1, mode Tunnel, intf_handle 0x5000180
Locations 0/1/1 0/2/0,  VRF customer_1 (60000002)
Number of profiles 1, number of flows 1
Tunnel: source 4.0.1.1, destination 5.0.1.1, tunnel VRF default
DF-bit: copy, pre-fragmentation enable
default pmtu: 9216
1 connected flows:
502
```

Configuring a service-gre Interface: Example

The following example shows a basic configuration of a service-gre interface and an IPSec SA that is created on the interface.

Configuring the Transform-set to Use Transport Mode

```
crypto ipsec transform-set tsfm2
 transform esp-3des esp-md5-hmac
 mode transport
!
```

Configuring the IPSec Profile to Use the Set Transform-set Format

```
crypto ipsec profile gre
 set transform-set tsfm2
!
```

Configuring the Service-gre Interface

```
interface service-gre1
 ipv4 address 11.2.6.6 255.255.255.0
 profile gre
 tunnel source 50.50.50.2
 tunnel destination 40.40.40.2
 service-location preferred-active 0/1/1
!
```

The following example shows the sample output from the **show crypto ipsec summary** command:

```
RP/0/0/CPU0:router# show crypto ipsec summary

# * Attached to a transform indicates a bundle

# Active IPSec Sessions: 2

SA    Local Peer      Remote Peer     FVRF      Profile   Transform    Lifetime
-------------------------------------------------------------------------------
503   50.50.50.2      40.40.40.2      default   gre       esp-3des esp 120/4194303
```

The following example shows that the service-gre interface is set to 1 with a profile gre:

```
RP/0/0/CPU0:router# show crypto ipsec sa 503

SA id:          503
Node id:        0/1/1
SA Type:        ISAKMP
interface:      service-gre1
profile  :      gre
local  ident (addr/mask/prot/port) : (50.50.50.2/255.255.255.255/47/0)
remote ident (addr/mask/prot/port) : (40.40.40.2/255.255.255.255/47/0)
local crypto endpt: 50.50.50.2, remote crypto endpt: 40.40.40.2, vrf default

 #pkts tx         :0                     #pkts rx         :0
 #bytes tx        :0                     #bytes rx        :0
 #pkts encrypt    :0                     #pkts decrypt    :0
 #pkts digest     :0                     #pkts verify     :0
 #pkts encrpt fail:0                     #pkts decrpt fail:0
 #pkts digest fail:0                     #pkts verify fail:0
 #pkts replay fail:0
 #pkts tx errors  :0                     #pkts rx errors  :0

outbound esp sas:
        spi: 0x5aeffcbd(1525677245)
        transform: esp-3des esp-md5-hmac
        in use settings = Transport
        sa agreed lifetime: 120s, 4194303kb
        sa timing: remaining key lifetime (sec/kb): (108/4194303)
        sa DPD disabled
        sa idle timeout: disable, 0s
        sa anti-replay (HW accel): enable, window 64
inbound esp sas:
        spi: 0x54373dd3(1412906451)
        transform: esp-3des esp-md5-hmac
        in use settings = Transport
        sa agreed lifetime: 120s, 4194303kb
        sa timing: remaining key lifetime (sec/kb): (108/4194303)
        sa DPD disabled
        sa idle timeout: disable, 0s
        sa anti-replay (HW accel): enable, window 64
```

The following example shows that the **interface service-gre** command is set to 1:

```
RP/0/0/CPU0:router# show crypto ipsec interface service-gre 1

--------------- IPSec interface ----------------
Interface service-gre1, mode Transport, intf_handle 0x5000880
Locations 0/1/1,  VRF default (60000000)
Number of profiles 1, number of flows 1
Tunnel: source 50.50.50.2, destination 40.40.40.2, tunnel VRF default
DF-bit: copy, pre-fragmentation enable
default pmtu: 9216
1 connected flows:
503
```

Configuration Examples for the Use of Object Tracking in IPSec

The example scenarios in this section illustrate how you would use object tracking in IPSec, depending on whether you were tracking connectivity from site to site or in a remote network:

- Tracking Connectivity From Site to Site, Based on a Physical or Logical Interface: Example, page 144
- Tracking Connectivity on a Remote Site, Using Routing Protocol Prefixes: Example, page 145

For a discussion of the use of object tracking in IPSec, see the "Information About Object Tracking in IPSec" section on page 97 of this chapter. For general configuration procedures specific to each type of object tracking see *Cisco IOS XR System Management Configuration Guide*.

Tracking Connectivity From Site to Site, Based on a Physical or Logical Interface: Example

When the connectivity between the physical or logical interface between the IPSec router (the router starting or ending with the IPSec tunnel) and other routers goes down, traffic arriving from the IPSec tunnel is dropped, because there is suddenly no next hop.

To avoid this situation, configure tracking of the state of the interfaces and tear down the IPSec tunnel when the value of the interface state changes.

There are two options to dealing with this scenario.

Option 1: Track Physical Interfaces

In this example, traffic arrives from interface service-ipsec 1 and exits through interface GigabitEthernet 0/0/0/3. For configuration steps for this type of object tracking, see "Tracking the Line Protocol State of an Interface" in the *Cisco IOS XR System Management Configuration Guide*.

1. Configure object tracking:

```
track IPSec1
 type line-protocol state
  interface gigabitethernet0/0/0/3
```

2. Associate service IPSec interface whose state should be tracked (ipsec1).

```
interface service-ipsec1
 ipv4 address 70.0.0.1 255.255.255.0
 profile vrf1_profile_ipsec
 line-protocol track ipsec1
```

```
 tunnel source 80.0.0.1
 tunnel destination 80.0.0.2
 service-location preferred-active 0/0/1
```

3. Check the state of the tracking object:

```
RP/0/0/CPU0:Router# show track

Track IPSec1
Interface GigabitEthernet0_0_0_3 line-protocol
        Line protocol is UP
        1 change, last change 10:37:32 UTC Thu Sep 20 2007
        Tracked by:
                service-ipsec1
```

Option 2: Track a List of Logical Interfaces

Traffic is arriving from interface service-ipsec 1 and should exit through interface GigabitEthernet 0/0/0/3 and interface ATM 0/2/0/0.1. This configuration example uses a Boolean OR expression to indicate that the IPSec tunnel should be torn down as soon as both interfaces go down. For configuration steps for this type of object tracking, see "Building a Track Based on a List of Objects" in the *Cisco IOS XR System Management Configuration Guide*.

```
track list1
 type list boolean or
  object IPSec1
  object IPSec2
 !
!
track IPSec1
 type line-protocol state
  interface GigabitEthernet0/0/0/3
 !
!
track IPSec2
 type line-protocol state
  interface ATM0/2/0.1
!
interface service-ipsec1 << Associates the track with an IPSec interface.
 ipv4 address 70.0.0.1 255.255.255.0
 profile vrf1_profile_ipsec
 line-protocol track list1
 tunnel source 80.0.0.1
 tunnel destination 80.0.0.2
 service-location preferred-active 0/0/1
```

Tracking Connectivity on a Remote Site, Using Routing Protocol Prefixes: Example

When a host or a network goes down on a remote site, routing protocols notify the router and the routing table is updated accordingly. You can track a route by configuring the routing process to notify the tracking process when the route state changes due to a routing update.

Option 1: Track the Routing Prefix in the Routing Table

In this example, traffic arriving from interface service-ipsec1 has its destination in network 7.0.0.0/24. We want to track when the state of the routing protocol prefix changes in the routing table. For configuration steps for this type of object tracking, see "Tracking IP Route Reachability" in the *Cisco IOS XR System Management Configuration Guide*.

```
track PREFIX1
 type route reachability
  route ipv4 7.0.0.0/24
!
interface service-ipsec1
 vrf 1
 ipv4 address 70.0.0.2 255.255.255.0
 profile vrf_1_ipsec
 line-protocol track PREFIX1
 tunnel source 80.0.0.2
 tunnel destination 80.0.0.1
 service-location preferred-active 0/2/0
```

Option 2: Track a List of Objects

In this example, traffic arriving from interface service-ipsec 1 exits through interface GigabitEthernet 0/0/0/3 and interface ATM 0/2/0/0.1. The destination of the traffic is at network 7.0.0.0/24.

We want to stop the flow of traffic if either one of the interfaces or the remote network goes down. To do this, we use a Boolean AND expression. For configuration steps for this type of object tracking, see "Building a Track Based on a List of Interfaces" in the *Cisco IOS XR System Management Configuration Guide*.

```
track LIST2
 type list boolean and
  object IPSec1
  object IPsec2
  object PREFIX1
 !
track IPSec1
 type line-protocol state
  interface GigabitEthernet0/0/0/3
 !
track IPSec2
 type line-protocol state
  interface ATM0/2/0.1
 !
track PREFIX1
 type route reachability
  route ipv4 7.0.0.0/24
!
interface service-ipsec1
 vrf 1
 ipv4 address 70.0.0.2 255.255.255.0
 profile vrf_1_ipsec
 line-protocol track LIST2
 tunnel source 80.0.0.2
 tunnel destination 80.0.0.1
 service-location preferred-active 0/2/0
```

Configuration Examples for Implementing IPSec in a Site-to-Site or Remote VPN Topology

This section contains the following sections:

For related information, see Information About Implementing IPSec in a Site-to-Site or Remote VPN Topology, page 93.

Configuring IPSec in a Site-to-Site VPN Topology: Example

To configure a site-to-site VPN, complete the following tasks. For details about each task example, follow the links to the associated configuration procedures.

1. Enable ISAKMP.

```
RP/0/0/CPU0:Router(config)# crypto isakmp
RP/0/0/CPU0:Router(config)# commit
```

2. Configure the ISAKMP policy by defining the parameters to be used during Phase-1 IKE negotiation.

   SHA-1 specifies the has algorithm. The other supported algorithm is MD-5.

```
RP/0/0/CPU0:Router(config)# crypto isakmp policy 1
RP/0/0/CPU0:Router(config-isakmp)#  description PSK_SHA_AES
RP/0/0/CPU0:Router(config-isakmp)# authentication pre-share

RP/0/0/CPU0:Router(config-isakmp)# hash sha
RP/0/0/CPU0:Router(config-isakmp)# group 2


RP/0/0/CPU0:Router(config-isakmp)# encryption aes

RP/0/0/CPU0:Router(config-isakmp)# exit
RP/0/0/CPU0:Router(config)# commit
```

Note SHA-1 and MD-5 can be used to calculate HMAC.

3. Configure a crypto access control List (ACL) to enable communication between subnets.

   This example configures an ACL to permit communication between subnets 192.102.1.0 and 192.101.1.0. These are the subnet addresses on for both routers on a private network.

```
RP/0/0/CPU0:Router(config)# ipv4 access-list crypto-1
(config-ipv4-acl)# permit ipv4 192.102.1.0 0.0.0.255 192.101.1.0 0.0.0.255
```

Note If you configure reverse-route injection (RRI) in the IPSec profile that references this ACL, the ACL destination address is added as a static route pointing to the SVI. RRI will be configured in a later step.

4. Configure the IPSec transform set that protects the data flows specified in the crypto profile access list.

   During the negotiation, the peers search for a transform set that is the same at both peers. When a matching transform set is found, IKE selects it and applies it to the protected traffic as part of the IPSec SAs for both peers.

   ```
   RP/0/0/CPU0:Router(config)#  crypto IPSec transform-set transform-1
   ```

   The following command specifies the mode. Supported modes are tunnel (default) and transport.

   ```
   RP/0/0/CPU0:Router(config-transform-set transform-1)# mode tunnel
   ```

   The following command defines the encryption and mac algorithms to use to protect traffic. In this example, Advanced Encryption Standard (AES) cipher (128 bits) algorithm was used.

   ```
   RP/0/0/CPU0:Router(config-transform-set transform-1)# transform esp-aes esp-sha-hmac
   ```

   Other supported encryption and mac algorithms consist of the following:

   - oah-md5-hmac
   - oah-sha-hmac
   - oesp-192-aes
   - oesp-256-aes
   - oesp-3des
   - oesp-aes
   - oesp-des
   - oesp-md5-hmac
   - oesp-sha-hmac

   ESP encapsulation normally uses both an encryption and hash algorithm. Because AH is not encrypted, it uses only a hash algorithm.

5. Configure crypto keyring.

   A keyring is a repository of preshared keys and RSA public keys. The peer keys defined in the keyring are used by the ISAKMP profile to authenticate the remote side during IKE negotiation.

   This configuration also allows the user to specify the fVRF referenced by the tunnel for various endpoints.

   ```
   RP/0/0/CPU0:Router(config)# crypto keyring keyring-1 vrf default
   RP/0/0/CPU0:Router(config-keyring)# pre-shared-key address 201.201.201.1
   255.255.255.255 key cisco
   ```

   The following command specifies the pre-shared key that is associated with the remote endpoint of the tunnel (IPSec tunnel IP address and mask). If the remote endpoint of the tunnel resides in a VRF other than the global table, it can be specified on this line.

   ```
   RP/0/0/CPU0:Router(config-keyring)# commit
   ```

6. Configure the IPSec profile.

   The following commands define the cryptographic behavior of the IPSec transport and IPSec-enabled interfaces.

   ```
   RP/0/0/CPU0:Router(config)#  crypto IPSec profile profile-1
   ```

Th e following command defines the DH group to use when renegotiating IPSec SAs. In this case, the perfect forward secrecy (PFS) setting is for IKE to handle group 5 (1536-bit Diffie-Hellman prime modulus group) negotiation.

```
RP/0/0/CPU0:Router(config-profile-1)#  set pfs group5
```

The following command defines which traffic (passing through ACL *crypto-1*) to encrypt and how to encrypt it (through **transform set** *transform-1*). It also defines which set of parameters to negotiate (**transform-set**) during IKE Phase 2.

```
RP/0/0/CPU0:Router(config-profile-1)# match crypto-1 transform-set transform-1
```

Note You can define up to five transform-sets for match.

For details, see Configuring Crypto Profiles, page 104.

If configured, the destination address from the ACL is added as a static route pointing to the SVI. While this command is optional in site-to-site configurations, it is on by default for remote access topologies. (See Configuring IPSec in a Remote-Access VPN Topology: Example, page 151.)

```
RP/0/0/CPU0:Router(config-profile-1)#  reverse-route distance 255
RP/0/0/CPU0:Router(config-profile-1)#  commit
```

For details, see the "Configuring Reverse-Route Injection in a Crypto Profile" section on page 123.

7. Configure the IPSec virtual interface (SVI) by specifying its physical location (the IPSec SPA), IP address, and the source and destination addresses of the IPSec tunnel between the two nodes.

   The interface can be either of the following:

   - **service-ipsec**
   - **service-gre**

   The **service-ipsec** interface supports only **tunnel** mode, while the **service-gre** interface supports only **transport** mode, because the generic routing encapulation (GRE) already provides tunneling. Each of these modes is configured using the **transform-set** command referenced in Step 6. "Configure the IPSec profile." on page 148.

   This example uses **service-ipsec:**

   ```
   RP/0/0/CPU0:Router(config)#  interface service-ipsec1
   RP/0/0/CPU0:Router(config-if)#  ipv4 address 202.202.202.1 255.255.255.252
   ```

   Normally, you set the IPv4 address to be unnumbered, because the SVI itself is rarely a traffic destination. If absolutely needed for management or application purposes, the IP address must not have a /32 mask.

   The following command specifies the IPSec profile configured in Step 6. "Configure the IPSec profile." on page 148.

   ```
   RP/0/0/CPU0:Router(config-if)#  profile profile-1
   ```

   The tunnel destination in the command below is the remote IPSec tunnel end point IP address:

   ```
   RP/0/0/CPU0:Router(config-if)#  tunnel source 201.201.201.2
   RP/0/0/CPU0:Router(config-if)#  tunnel destination 201.201.201.1

   RP/0/0/CPU0:Router(config-profile-1)#  commit
   ```

   The following command provides the physical location (rack/slot/module) of the IPSec where this SVI should reside.

```
RP/0/0/CPU0:Router(config-if)#  service-location preferred-active 0/2/1
```

If the SVI has a standby SPA, it would also be configured with the **preferred-standby** keyword:

```
RP/0/0/CPU0:Router(config-if)#  service-location preferred-active 0/2/1
preferred-standby 0/3/1

RP/0/0/CPU0:Router(config-if)#  commit
```

Note The standby IPSec SPA must reside on a different SIP-x01 than the primary.

Multiple SVIs on the same SPA may be configured with the same tunnel source address, which is useful for conserving addresses. For example:

```
interface service-ipsec1
 tunnel source 1.1.1.1
 tunnel destination 2.2.2.2

interface service-ipsec2
 tunnel source 1.1.1.1
 tunnel destination 3.3.3.3
```

8. Configure crypto ISAKMP profile.

   The **crypto isakmp profile** command defines an ISAKMP profile and audits IPSec user sessions. Peers are mapped to an ISAKMP profile when their identities are matched (Step 7.), as given in the ID payload of IKE. The ISAKMP profile successfully completes authentication of peers if the peer keys are defined in the keyring attached to this profile.

   The following command specifies the keyring defined in Step 5. Configure crypto keyring.

   ```
   RP/0/0/CPU0:Router(config)# crypto isakmp profile isakmp_profile-1
   RP/0/0/CPU0:Router(config-isa-prof)# keyring keyring-1
   ```

   The following command defines the remote endpoint of the tunnel. This address and VRF should match an entry in the keyring specified in Step 5. Configure crypto keyring.

   ```
   RP/0/0/CPU0:Router(config-isa-prof)# match identity address 201.201.201.1/32 vrf
   default
   ```

   The following command determines the SVI to use for encryption/decryption when a matching peer is found:

   ```
   RP/0/0/CPU0:Router(config-isa-prof-match)# set interface service-IPSec1
   RP/0/0/CPU0:Router(config-isa-prof-match)# commit
   ```

   When the XR router is the IKE *responder,* this configuration line allows the router to determine the proper SVI for sending its encrypted traffic.

   When the XR router is the IKE *initiator,* the correct SVI has already been chosen by doing a lookup in the forwarding table. The remote endpoint is then determined by the tunnel destination of that SVI.

Configuring IPSec in a Remote-Access VPN Topology: Example

The SVI can terminate many site-to-site tunnels, which is useful for remote access configurations where all remote devices may share the same parameters.

Configuration of IPSec for a remote-access topology differs from configuration for a site-to-site topology in the following ways:

- No tunnel destination is included as part of in the SVI configuration
- You configure set type dynamic in the IPSec profile
- The router cannot initiate sessions in dynamic mode
- No keyring, because remote access is unknown

You can provide addresses for remote clients by using one of the following methods:

- Configure a local pool on the router (30,000 maximum, in other words, 30,720 addresses).
- Use extended DHCP.
- Define a static IP in RADIUS (TACACS not supported).

Configuring Addresses for Remote Clients, Using a Local Pool

The following example illustrates configuration of a local pool.

1. Configure AAA authorization and authentication.

   AAA is required to be configured for users and/or groups. The keys are stored in AAA and referenced from the ISAKMP profile, in Step 4.

   ```
   RP/0/0/CPU0:Router# aaa authorization network net23 local
   RP/0/0/CPU0:Router# aaa authentication login net23 local
   ```

2. Configure an IP address pool on the router.

   ```
   RP/0/0/CPU0:Router# Local pool ipv4 pool-101 210.210.0.1 210.210.0.254
   ```

3. Configure the ISAKMP client pool.

   ```
   RP/0/0/CPU0:Router# crypto isakmp client configuration group group-1
   RP/0/0/CPU0:Router(config-isa-client)# key poolkey
   RP/0/0/CPU0:Router(config-isa-client)# pool pool-101
   ```

4. Configure the ISAKMP profile.

   ```
   crypto isakmp profile ra-1
    client authentication list foo
    match identity group group-1
     set interface service-ipsec1
    !
    !
    isakmp authorization list author2
   !
   ```

5. Configure the IPSec profile.

   ```
   RP/0/0/CPU0:Router# crypto ipsec profile ra-1 set type dynamic
   RP/0/0/CPU0:Router(config-profile-1)# match acl transform-set transform-set-name
   RP/0/0/CPU0:Router(config-profile-1)# reverse-route
   ```

6. Configure the policy set.

   ```
   RP/0/0/CPU0:IOX2(config)# crypto isakmp policy 1
   ```

```
RP/0/0/CPU0:IOX2(config-isakmp)#encryption 3des
RP/0/0/CPU0:IOX2(config-isakmp)#group 2
RP/0/0/CPU0:IOX2(config-isakmp)#exit
RP/0/0/CPU0:IOX2(config)#
RP/0/0/CPU0:IOX2(config)# crypto isakmp policy 2
RP/0/0/CPU0:IOX2(config-isakmp)# encryption des
RP/0/0/CPU0:IOX2(config-isakmp)# group 2
RP/0/0/CPU0:IOX2(config-isakmp)# authentication pre-share
RP/0/0/CPU0:IOX2(config-isakmp)# exit
RP/0/0/CPU0:IOX2(config)#
RP/0/0/CPU0:IOX2(config)# crypto isakmp policy 3
RP/0/0/CPU0:IOX2(config-isakmp)# authentication pre-share
RP/0/0/CPU0:IOX2(config-isakmp)# authentication rsa-sig
RP/0/0/CPU0:IOX2(config-isakmp)# encryption des
RP/0/0/CPU0:IOX2(config-isakmp)# exit
RP/0/0/CPU0:IOX2(config)#
RP/0/0/CPU0:IOX2(config)# crypto isakmp policy-set pol-set1
RP/0/0/CPU0:IOX2(config-isakmp-pol-set)# policy 2 3
RP/0/0/CPU0:IOX2(config-isakmp-pol-set)# match identity local-address 1.1.1.1
RP/0/0/CPU0:IOX2(config-isakmp-pol-set)# exit
RP/0/0/CPU0:IOX2(config)#
RP/0/0/CPU0:IOX2(config)#interface service-ipsec 1
RP/0/0/CPU0:IOX2(config-if)# ipv4 address 20.1.1.1 255.255.255.0
RP/0/0/CPU0:IOX2(config-if)# tunnel source 1.1.1.1
RP/0/0/CPU0:IOX2(config-if)# profile prof1
RP/0/0/CPU0:IOX2(config-if)# service-location preferred-active 0/1/0
RP/0/0/CPU0:IOX2(config-if)#
RP/0/0/CPU0:IOX2(config-if)# commit
```

Additional References

The following sections provide references related to implementing IPSec network security.

Related Documents

Related Topic	Document Title
IPSec network security commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*IPSec Network Security Commands on Cisco IOS XR Software* module in *Cisco IOS XR System Security Command Reference*
Internet Key Exchange (IKE) security protocol commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Internet Key Exchange Security Protocol Commands on Cisco IOS XR Software* module in *Cisco IOS XR System Security Command Reference*
IP-Sec-related object tracking commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS XR System Management Command Reference*
Object tracking configuration procedures, including examples	*Cisco IOS XR System Management Configuration Guide*
IPSec in Quality of Service (QoS)	*Cisco IOS XR Modular Quality of Service Configuration Guide*
MPLS distribution protocol	*Cisco IOS XR MPLS Configuration Guide*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
—	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
RFC 2401	*Security Architecture for the Internet Protocol*
RFC 2402	*IP Authentication Header*
RFC 2403	*The Use of HMAC-MD5-96 within ESP and AH*
RFC 2404	*The Use of HMAC-SHA-1-96 within ESP and AH*
RFC 2405	*The ESP DES-CBC Cipher Algorithm With Explicit IV*
RFC 2406	*IP Encapsulating Security Payload (ESP)*
RFC 2407	*The Internet IP Security Domain of Interpretation for ISAKMP*
RFC 2408	*Internet Security Association and Key Management Protocol (ISAKMP)*
RFC 2409	*The Internet Key Exchange (IKE)*

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport