IPSec Reference, StarOS Release 21.15

First Published: 2019-08-29

Last Modified: 2019-08-29

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS

About this Guide

This preface describes the *IPSec Reference*, how it is organized and its document conventions.

This guide describes configuration requirements for IP Security services. IPSec provides security for transmission of sensitive information over unprotected networks such as the Internet. IPSec is a standards-based robust solution that provides data authentication and anti-replay services in addition to data confidentiality services.

Important IPSec is a suite of standard and licensed Cisco features. Contact your Cisco account representative for detailed information on specific licensing requirements. For information on installing and verifying licenses, refer to the *System Administration Guide*.

Important This guide documents IPSec features that appear in the StarOS command line interface (CLI). IPSec features are not universally supported across all StarOS products. Support for IPSec features varies per platform, service type and StarOS release. Refer to the gateway administration guides and StarOS *Release Notes* for additional information.

Conventions Used

The following tables describe the conventions used throughout this documentation.

Notice Type	Description
Information Note	Provides information about important features or instructions.
Caution	Alerts you of potential damage to a program, device, or system.
Warning	Alerts you of potential personal injury or fatality. May also alert you of potential electrical hazards.

Typeface Conventions	Description
Text represented as a `screen display`	This typeface represents displays that appear on your terminal screen, for example: `Login:`
Text represented as **commands**	This typeface represents commands that you enter, for example: **show ip access-list** This document always gives the full form of a command in lowercase letters. Commands are not case sensitive.
Text represented as a **command** *variable*	This typeface represents a variable that is part of a command, for example: **show card** *slot_number* *slot_number* is a variable representing the desired chassis slot number.
Text represented as menu or sub-menu names	This typeface represents menus and sub-menus that you access within a software application, for example: Click the **File** menu, then click **New**

Command Syntax Conventions	Description
{ **keyword** or *variable* }	Required keyword options and variables are those components that are required to be entered as part of the command syntax. Required keyword options and variables are surrounded by grouped braces { }. For example: **sctp-max-data-chunks { limit** *max_chunks* **| mtu-limit }** If a keyword or variable is not enclosed in braces or brackets, it is mandatory. For example: **snmp trap link-status**
[**keyword** or *variable*]	Optional keywords or variables, or those that a user may or may not choose to use, are surrounded by brackets.

Command Syntax Conventions	Description
|	Some commands support multiple options. These are documented within braces or brackets by separating each option with a vertical bar. These options can be used in conjunction with required or optional keywords or variables. For example: **action activate-flow-detection { intitiation | termination }** or **ip address [count** *number_of_packets* **| size** *number_of_bytes* **]**

Related Documentation

Common Documents (All Products)

The most up-to-date information for this product is available in the product *Release Notes* provided with each product release.

The following common documents are available:

- Installation Guide (platform dependent)
- *AAA Interface Administration Reference*
- *Command Line Interface Reference*
- *GTPP Interface Administration Reference*
- *Release Change Reference*
- *Statistics and Counters Reference*
- *System Administration Guide* (platform dependent)
- *Thresholding Configuration Guide*

Product Documents (Gateways)

The following product documents are also available and include information regarding IPSec configuration:

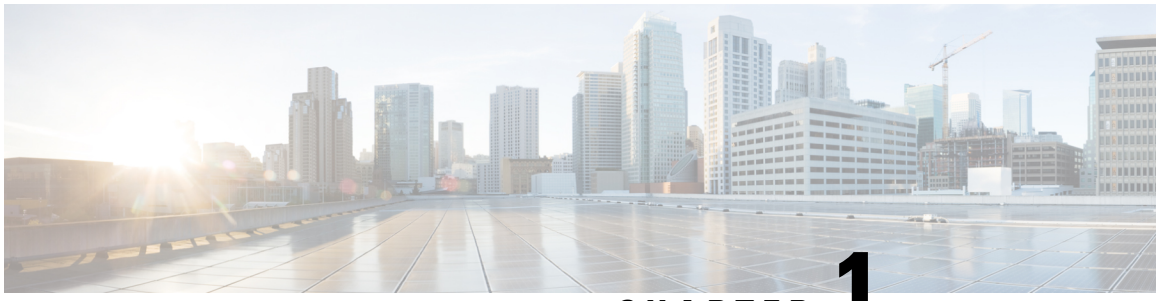
- *ePDG Administration Guide*
- *GGSN Administration Guide*
- *HA Administration Guide*
- *HNB-GW Administration Guide*
- *HSGW Administration Guide*
- *MME Administration Guide*
- *P-GW Administration Guide*
- *PDIF Administration Guide*
- *PDSN Administration Guide*

- *S-GW Administration Guide*
- *SAEGW Administration Guide*
- *SecGW Administration Guide*

Contacting Customer Support

Use the information in this section to contact customer support.

Refer to the support area of http://www.cisco.com for up-to-date product documentation or to submit a service request. A valid username and password are required to access this site. Please contact your Cisco sales or service representative for additional information.



CHAPTER 1

Introduction to IP Security (IPSec)

This chapter briefly describes IPSec functionality and associated terminology.

Important IPSec is a suite of standard and licensed Cisco features. Contact your Cisco account representative for detailed information on specific licensing requirements. For information on installing and verifying licenses, refer to the *System Administration Guide*.

The following topics are discussed in this chapter:

Overview

IPSec is a suite of protocols that interact with one another to provide secure private communications across IP networks. These protocols allow the system to establish and maintain secure tunnels with peer security gateways. IPSec provides confidentiality, data integrity, access control, and data source authentication to IP datagrams.

IPSec AH and ESH

Authentication Header (AH) and Encapsulating Security Payload (ESP) are the two main wire-level protocols used by IPSec. They authenticate (AH) and encrypt-plus-authenticate (ESP) the data flowing over that connection.

- **AH** is used to authenticate – but not encrypt – IP traffic.Authentication is performed by computing a cryptographic hash-based message authentication code over nearly all the fields of the IP packet (excluding those which might be modified in transit, such as TTL or the header checksum), and stores this in a newly-added AH header that is sent to the other end. This AH header is injected between the original IP header and the payload.

- **ESP** provides encryption and optional authentication. It includes header and trailer fields to support the encryption and optional authentication. Encryption for the IP payload is supported in transport mode and for the entire packet in the tunnel mode. Authentication applies to the ESP header and the encrypted data.

IPSec Transport and Tunnel Mode

Transport Mode provides a secure connection between two endpoints as it encapsulates IP payload, while Tunnel Mode encapsulates the entire IP packet to provide a virtual "secure hop" between two gateways.

Tunnel Mode forms the more familiar VPN functionality, where entire IP packets are encapsulated inside another and delivered to the destination. It encapsulates the full IP header as well as the payload.

Security Associations (SAs) and Child SAs

An Internet Key Exchange-Security Association (IKE-SA) is used to secure IKE comicality. SA is identified by two, eight-byte Security Parameter Indices (SPIs) shared by each peer during the initial IKE exchange. Both SPIs are carried in all subsequent messages.

A Child-SA is created by IKE for use in AH or ESP security. Two Child-SAs are created as a result of one exchange – Inbound and Outbound. A Child-SA is identified by a single four-byte SPI, Protocol and Gateway IP Address and is carried in each AH/ESP packet.

Each SA (IKE or Child) has an associated lifetime. After the expiry of lifetime, SAs are deleted. To proactively establish an SA before the last one expires, SAs are rekeyed on soft lifetime expiry. Both IKE and Child SAs may be rekeyed.

Anti-Replay (IKEv2)

Anti-replay is a sub-protocol of IPSec (RFC 4303) that is supported for IKEv1 and IKEv2 tunnels. Its main goal is to prevent hackers injecting or making changes in packets that travel from a source to a destination. Anti-replay protocol employs a unidirectional security association to establish a secure connection between two nodes in the network.

Once a secure connection is established, the anti-replay protocol uses a sequence number or a counter. When the source sends a message, it adds a sequence number to its packet starting at 0 and increments every time it sends another message. At the destination end, the protocol receives the message and keeps a history of the number and shifts it as the new number. If the next message has a lower number, the destination drops the packet, and, if the number is larger than the previous one, it keeps and shifts it as the new number.

The anti-replay feature may be enabled or disabled via the StarOS CLI. Anti-Replay Window Sizes of 32, 64, 128, 256, 384 and 512 bits are supported (default = 64 bits).

Behavior for ACL-based calls differs from Subscriber-based calls.

- **ACL-based.** An anti-replay configuration change in the CLI will not be propagated until a call is cleared and re-established.
- **Subscriber-based.** An anti-replay configuration change in the CLI will not affect established calls but new calls will utilize the new anti-replay configuration.

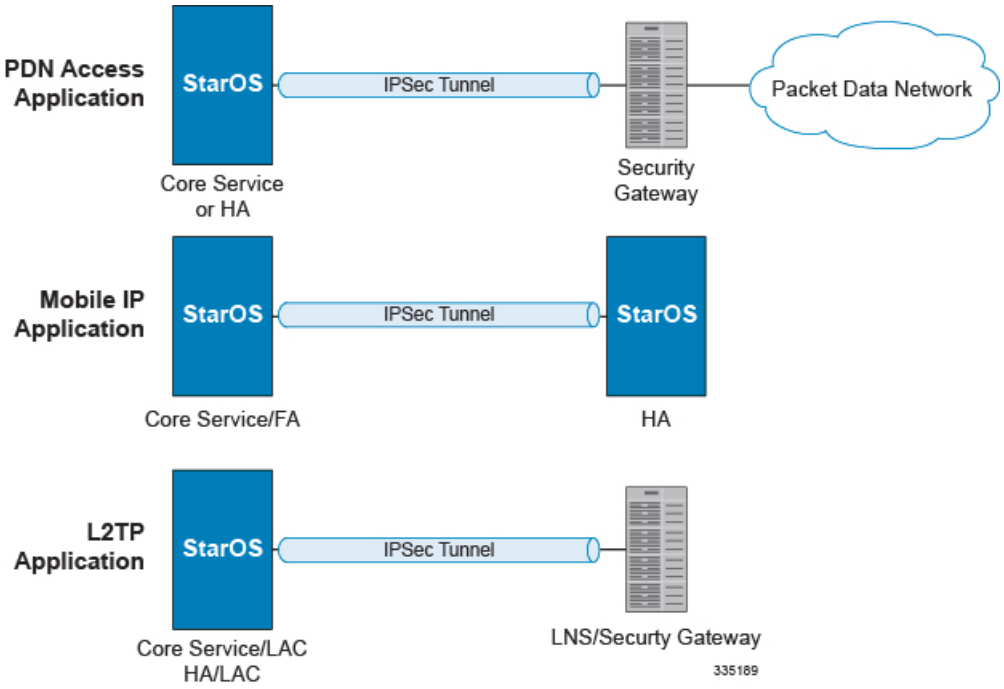
IPSec Applications

Important Support for IPSec features varies per platform, service type and StarOS release. Refer to the gateway administration guide and StarOS *Release Notes* for additional information.

IPSec can be implemented via StarOS for the following applications:

- **PDN Access:** Subscriber IP traffic is routed over an IPSec tunnel from the system to a secure gateway on the packet data network (PDN) as determined by access control list (ACL) criteria. This application can be implemented for both core network service and HA-based systems. The following figure shows several IPSec configurations.

Figure 1: IPSec Applications



- **Mobile IP:** Mobile IP (MIP) control signals and subscriber data is encapsulated in IPSec tunnels that are established between foreign agents (FAs) and home agents (HAs) over the Pi interfaces.

Important Once an IPSec tunnel is established between an FA and HA for a particular subscriber, all new Mobile IP sessions using the same FA and HA are passed over the tunnel regardless of whether or not IPSec is supported for the new subscriber sessions. Data for existing Mobile IP sessions is unaffected.

- **L2TP:** L2TP-encapsulated packets are routed from the system to an LNS/secure gateway over an IPSec tunnel.

Note that: IPSec can be implemented for both attribute-based and compulsory tunneling applications for 3GPP2 services.

IPSec Terminology

There are several items related to IPSec support under StarOS that must be understood prior to beginning configuration. They include:

Crypto Access Control List (ACL)

Access Control Lists define rules, usually permissions, for handling subscriber data packets that meet certain criteria. Crypto ACLs, however, define the criteria that must be met in order for a subscriber data packet to be routed over an IPSec tunnel.

Unlike other ACLs that are applied to interfaces, contexts, or one or more subscribers, crypto ACLs are matched with crypto maps. In addition, crypto ACLs contain only a single rule while other ACL types can consist of multiple rules.

Prior to routing, the system examines the properties of each subscriber data packet. If the packet properties match the criteria specified in the crypto ACL, the system will initiate the IPSec policy dictated by the crypto map.

For additional information refer to the *Access Control* chapter of this guide. There you will find a discussion of blacking and whitelisting, as well as IKE Call Admission Control (CAC).

Transform Set

Transform Sets are used to define IPSec security associations (SAs). IPSec SAs specify the IPSec protocols to use to protect packets.

Transform sets are used during Phase 2 of IPSec establishment. In this phase, the system and a peer security gateway negotiate one or more transform sets (IPSec SAs) containing the rules for protecting packets. This negotiation ensures that both peers can properly protect and process the packets.

For additional information refer to the *Transform Set Configuration* chapter of this guide,

ISAKMP Policy

Internet Security Association Key Management Protocol (ISAKMP) policies are used to define Internet Key Exchange (IKE) SAs. The IKE SAs dictate the shared security parameters (such as which encryption parameters to use, how to authenticate the remote peer, etc.) between the system and a peer security gateway.

During Phase 1 of IPSec establishment, the system and a peer security gateway negotiate IKE SAs. These SAs are used to protect subsequent communications between the peers including the IPSec SA negotiation process.

For additional information refer to the *ISAKMP Policy Configuration* chapter of this guide.

Crypto Map

Crypto Maps define the tunnel policies that determine how IPSec is implemented for subscriber data packets.

There are several types of crypto maps supported by StarOS. They are:

- Manual crypto maps
- IKEv2 crypto maps
- Dynamic crypto maps

Important The **map ip pool** command is not supported within crypto maps on the ASR 5500.

Manual Crypto Maps (IKEv1)

These are static tunnels that use pre-configured information (including security keys) for establishment. Because they rely on statically configured information, once created, the tunnels never expire; they exist until their configuration is deleted.

Manual crypto maps define the peer security gateway to establish a tunnel with, the security keys to use to establish the tunnel, and the IPSec SA to be used to protect data sent/received over the tunnel. Additionally, manual crypto maps are applied to specific system interfaces.

Important Because manual crypto map configurations require the use of static security keys (associations), they are not as secure as crypto maps that rely on dynamically configured keys. Therefore, it is recommended that they only be configured and used for testing purposes.

IKEv2 Crypto Maps

These tunnels are similar to manual crypto maps in that they require some statically configured information such as the IP address (IPv4 or IPv6) of a peer security gateway and that they are applied to specific system interfaces.

However, IKEv2 crypto maps offer greater security because they rely on dynamically generated security associations through the use of the Internet Key Exchange (IKE) protocol.

When IKEv2 crypto maps are used, the system uses the pre-shared key configured for the map as part of the Diffie-Hellman (D-H) exchange with the peer security gateway to initiate Phase 1 of the establishment process. Once the exchange is complete, the system and the security gateway dynamically negotiate IKE SAs to complete Phase 1. In Phase 2, the two peers dynamically negotiate the IPSec SAs used to determine how data traversing the tunnel will be protected.

Dynamic Crypto Maps (IKEv1)

These tunnels are used for protecting L2TP-encapsulated data between the system and an LNS/security gateway or Mobile IP data between an FA service configured on one system and an HA service configured on another.

The system determines when to implement IPSec for L2TP-encapsulated data either through attributes returned upon successful authentication for attribute based tunneling, or through the configuration of the L2TP Access Concentrator (LAC) service used for compulsory tunneling.

The system determines when to implement IPSec for Mobile IP based on RADIUS attribute values as well as the configurations of the FA and HA service(s).

For additional information, refer to the *Crypto Maps* chapter of this guide

Crypto Template

A Crypto Template configures an IKEv2 IPSec policy. It includes most of the IPSec parameters and IKEv2 dynamic parameters for cryptographic and authentication algorithms. A security gateway service will not function without a configured crypto template.

Only one crypto template can be configured per service. However, a single StarOS instance can run multiple instances of the same service with each associated with that crypto template.

For additional information, refer to the *Crypto Templates* chapter of this guide.

IKEv1 versus IKEv2

StarOS supports features associated with:

- IKEv1 as defined in RFC 2407, RFC 2408 and RFC 2409
- IKEv2 as defined in RFC 4306, RFC 4718 and RFC 5996

The table below compares features supported by IKEv1 and IKEv2.

Table 1: IKEv1 versus IKEv2 Features

IKEv1	IKEv2
IPSec Security Associations (SAs)	Child Security Associations (Child SAs)
Exchange modes: • Main mode • Aggressive mode	Only one exchange mode is defined. Exchange modes were obsoleted.
Number of exchanged messages required to establish a VPN: • Main mode = 9 messages • Aggressive mode = 6 messages	Only 4 messages are required to establish a VPN.

IKEv1	IKEv2
Authentication methods: • Pre-Shared Key (PSK) • Digital Signature (RSA-Sig) • Public Key Encryption • Revised mode of public Key Encryption	Authentication methods: • Pre-Shared Key (PSK) • Digital Signature (RSA-Sig)
Traffic Selector: • Only a combination of a source IP range, a destination IP range, a source port and a destination port is allowed per IPSec SA. • Exact agreement of the traffic selection between peers is required.	Traffic Selector: • Multiple combinations of of a source IP range, a destination IP range, a source port and a destination port are allowed per Child SA. IPv4 and IPv6 addresses can be configured for the same Child SA. • Narrowing traffic selectors between peers is allowed.
Lifetime for SAs requires negotiation between peers.	Lifetime for SAs is not negotiated. Each peer can delete SAs by exchanging DELETE payloads.
Multihosting is not supported	Multihosting is supported by using multiple IDs on a single IP address and port pair.
Rekeying is not defined.	Rekeying is defined and supported.
Dead peer Detection (DPD) for SAs is defined as an extension.	DPD is supported by default.
NAT Transversal (NATT) is not supported.	NAT Transversal (NATT) is supported only for subscriber mode.
Remote Access VPN is not defined, but is supported by vendor-specific implementations for Mode config and XAUTH.	Remote Access VPN is supported by default: • Extensible Authentication Protocol (EAP) • User authentication via EAP is associated with IKE authentication • Configuration payload (CP)
Multihoming is not supported.	Multihoming is supported by MOBIKE (IKEv2 Mobility and Multihoming Protocol, RFC 4555)
Mobile Clients are not supported.	Mobile Clients are supported by MOBIKE.
Denial of Service (DoS) protections are not supported.	DoS protections include an anti-replay function.

Supported Algorithms

IPSec supports the protocols in the table below, which are specified in RFC 5996.

Table 2: Supported Algorithms

Protocol	Type	Supported Options
Internet Key Exchange version 2	IKEv2 Encryption	DES-CBC, 3DES-CBC, AES-CBC-128, AES-CBC-256
	IKEv2 Pseudo Random Function	PRF-HMAC-SHA1, PRF-HMAC-MD5, AES-XCBC-PRF-128
	IKEv2 Integrity	HMAC-SHA1-96, HMAC-SHA2-256-128, HMAC-SHA2-384-192. HMAC-SHA2-512-256, HMAC-MD5-96, AES-XCBC-96
	IKEv2 Diffie-Hellman Group	Group 1 (768-bit), Group 2 (1024-bit), Group 5 (1536-bit), Group 14 (2048-bit)

Protocol	Type	Supported Options
IP Security	IPSec Encapsulating Security Payload Encryption	NULL, DES-CBC, 3DES-CBC, AES-CBC-128, AES-CBC-256, AES-128-GCM-128, AES-128-GCM-64, AES-128-GCM-96, AES-256-GCM-128, AES-256-GCM-64, AES-256-GCM-96 **Note** AES-GCM algorithms are supported only on vPC-DI and vPC-SI Platform.
	Extended Sequence Number	Value of 0 or off is supported (ESN itself is not supported)
	IPSec Integrity	NULL, HMAC-SHA1-96, HMAC-MD5-96, AES-XCBC-96, HMAC-SHA2-256-128, HMAC-SHA2-384-192, HMAC-SHA2-512-256 **Important** HMAC-SHA2-384-192 and HMAC-SHA2-512-256 are not supported on vPC-DI and vPC-SI platforms if the hardware doesn't have crypto hardware.

Boost Crypto Performance

An **require ipsec-large** command boosts IPSec crypto performance by enabling the resource manager (RM) task to assign additional IPSec managers to packet processing cards that have sufficient processing capacity.

```
configure
    require ipsec-large
    end
```

This command works with ePDG, PDIF and other StarOS applications.

Important When IPSec large and demux on MIO are configured together, enable the IPsec large feature (using the **require ipsec-large** command) before enabling the demux on MIO (using the **require demux management-card** command).

Refer to the *Release Notes* accompanying each StarOS build for the latest information on supported products and packet processing cards.

Multiple Authentication Configuration

List of authentication methods are defined and associated in Crypto Template. The basic sample configuration required for OCSP and Certificate based authentication is as follows. For backward compatibility, the configuration for auth method inside Crypto Template will be working.

The following are the configuration considerations:

- A maximum of three sets of authentication methods in the list can be associated.
- Each set can have only one local and one remote authentication method configuration.
- The existing configuration inside the Crypto Template takes precedence over the new auth-method-set defined, in case same authentication method is configured at both places.

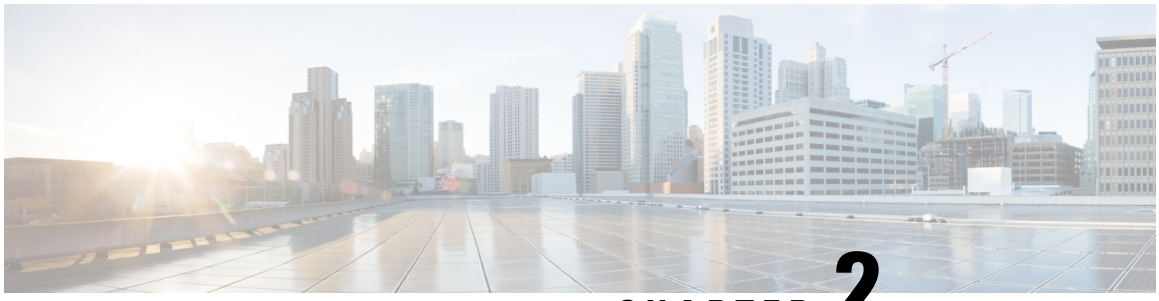
configure

CA Certificate for device certificate authentication:

```
ca-certificate name <ca-name> pem url file: <ca certificate path>
```

Gateway Certificate:

```
ca-certificate name <gateway-name> pem url file: <gateway certificate path>
private-key pem url file:<gateway private key path>
            eap-profile <profile name>
                 mode authenticator-pass-through
                 exit
            ikev2-ikesa auth-method-set <list-name-1>
                 authentication remote certificate
                 authentication local certificate
                 exit
            ikev2-ikesa auth-method-set <list-name-2>
                 authentication eap-profile eap1
                 exit
            crypto template boston ikev2-subscriber
                 ikev2-ikesa auth-method-set list <list-name-2> <list-name-2>
                 ca-certificate list ca-cert-name <ca-name>
                 exit
```



CHAPTER 2

IPSec to Product Feature Mapping

The IPSec feature is supported for various products. This chapter indicates the products on which IPSec is supported, as well as the relevant sections within this guide that pertain to that product.

Important This guide documents IPSec features that appear in the StarOS command line interface (CLI). IPSec features are not universally supported across all StarOS products and platforms. Refer to the Administration Guide for individual products for IPSec limitations.

IPSec support is outlined for the following products:

PDSN, FA and HA

The following chapters (in bold) and sections apply to PDSN (Packet Data Serving Node), FA (Foreign Agent) and HA (Home Agent) gateway products:

- **Introduction to IP Security (IPSec)**
- **IPSec Network Applications**
  - IPSec for PDN Access Applications
  - IP Sec for Mobile IP Applications
  - RADIUS Attributes for IPSec-based Mobile IP Applications
- **Transform Set Configuration**
- **ISAKMP Policy Configuration**
- **Crypto Maps**
  - Dynamic Crypto Map Configuration (IKEv1 only)
  - Manual Crypto MAp Configuration (IKEv1 only)
  - Crypto Map and Interface Association

- **Service Configurations**
  - FA Services Configuration to Support IPSec
  - HA Services Configuration to Support IPSec
  - PDSN Services Configuration to L2TP Support
  - LAC Service Configuration to Support IPSec
  - RADIUS and Subscriber Attributes for L2TP Application IPSec Support
- **Redundant IPSec Tunnel Fail-Over**
  - Redundant IPSec Tunnel Fail-Over (IKEv1 only)
  - Dead Peer Detection (DPD) Configuration
- **IKEv2 RFC 5996 Compliance**

GGSN, FA and HA

The following chapters (in bold) and sections apply to GGSN (Gateway GPRS Support Node), FA (Foreign Agent) and HA (Home Agent) gateway products:

- **Introduction to IP Security (IPSec)**
- **IPSec Network Applications**
  - IPSec for PDN Access Applications
  - IPSec for Mobile IP Applications
  - IPSec for L2TP Applications
  - RADIUS Attributes for IPSec-based Mobile IP Applications
- **Transform Set Configuration**
- **ISAKMP Policy Configuration**
- **Crypto Maps**
  - Dynamic Crypto Map Configuration (IKEv1 only)
  - Manual Crypto Map Configuration (IKEv1 only)
  - Crypto Map and Interface Association
- **Service Configurations**
  - FA Services Configuration to Support IPSec
  - HA Services Configuration to Support IPSec
  - LAC Service Configuration to Support IPSec
  - RADIUS and Subscriber Attributes for L2TP Application IPSec Support
- **Redundant IPSec Tunnel Fail-Over**
  - Redundant IPSec Tunnel Fail-Over (IKEv1 only)
  - Dead Peer Detection (DPD) Configuration
- **IKEv2 RFC 5996 Compliance**

HeNBGW, HNBGW and HSGW

The following chapters (in bold) and sections apply to HeNBGW (Home evolved Node B Gateway), HNBGW (Home node B Gateway), and HRPD Serving Gateway (HSGW):

Important HNBGW is not supported from Release 20 and later, and HeNBGW is not supported in Releases 20, 21.0 and 21.1. For more information, contact your Cisco account representative.

- **Introduction to IP Security (IPSec)**
- **IPSec Network Applications**
  - IPSec for PDN Access Applications
  - IPSec for Mobile IP Applications
  - IPSec for L2TP Applications
  - RADIUS Attributes for IPSec-based Mobile IP Applications
- **Transform Set Configuration**
- **ISAKMP Policy Configuration**
- **Crypto Maps**
  - Dynamic Crypto Map Configuration (IKEv1 only)
  - Manual Crypto Map Configuration (IKEv1 only)
  - Crypto Map and Interface Association
- **Service Configurations**
  - FA Services Configuration to Support IPSec
  - HA Services Configuration to Support IPSec
  - LAC Service Configuration to Support IPSec
  - RADIUS and Subscriber Attributes for L2TP Application IPSec Support
- **Redundant IPSec Tunnel Fail-Over**
  - Redundant IPSec Tunnel Fail-Over (IKEv1 only)
  - Dead Peer Detection (DPD) Configuration
- **IKEv2 RFC 5996 Compliance**

ePDG

The following chapters (in bold) and sections apply to an evolved Packet Data Gateway (ePDG):

- **Introduction to IP Security (IPSec)**
- **IPSec Network Applications**
  - IPSec for Mobile IP Applications
  - IPSec for L2TP Applications
  - RADIUS Attributes for IPSec-based Mobile IP Applications
- **Transform Set Configuration**
- **ISAKMP Policy Configuration**
- **Crypto Templates**
- **Redundant IPSec Tunnel Fail-Over**
  - Redundant IPSec Tunnel Fail-Over (IKEv1 only)
  - Dead Peer Detection (DPD) Configuration
- **IKEv2 RFC 5996 Compliance**

MME, S-GW, P-GW and SAE-GW

The following chapters (in bold) and sections apply to LTE components, including Mobile Management Entity (MME), Serving Gateway (S-GW), PDN Gateway (P-GW) and System Architecture Evolution Gateway (SAE-GW):

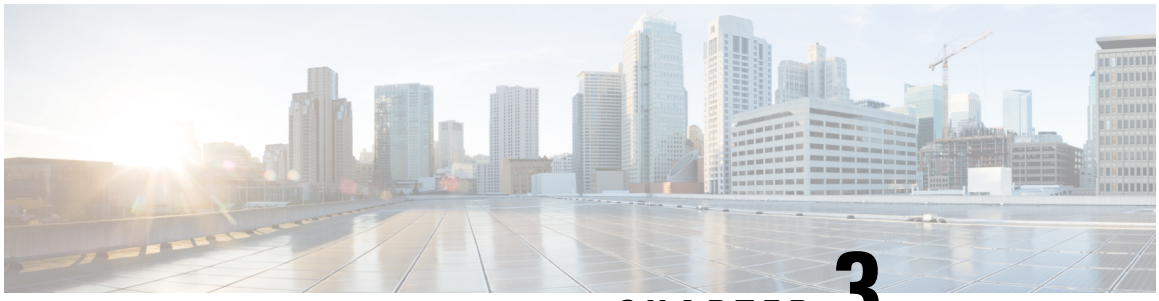
- **Introduction to IP Security (IPSec)**
- **IPSec Network Applications**
  - IPSec for PDN Access Applications
  - IPSec for Mobile IP Applications
  - IPSec for L2TP Applications
  - IPSec for LTE/SAE Networks
  - RADIUS Attributes for IPSec-based Mobile IP Applications
- **Transform Set Configuration**
- **ISAKMP Policy Configuration**
- **Crypto Maps**
  - Dynamic Crypto Map Configuration (IKEv1 only)
  - Manual Crypto Map Configuration (IKEv1 only)
  - Crypto Map and Interface Association
- **Crypto Templates** (MME, S-GW)
- **Service Configurations**
  - LAC Service Configuration to Support IPSec
  - RADIUS and Subscriber Attributes for L2TP Application IPSec Support
- **Redundant IPSec Tunnel Fail-Over**

  - Redundant IPSec Tunnel Fail-Over (IKEv1 only)
  - Dead Peer Detection (DPD) Configuration
- **IKEv2 RFC 5996 Compliance**

SecGW

The following chapters (in bold) and sections apply to a Security Gateway (SecGW, WSG service) running within a Virtualized Packet Core-Standalone Instance (VPC-SI) in a virtual machine on an ASR 9000 Virtualized Service Module (VSM).

- **Introduction to IP Security (IPSec)**
- **IPSec Network Applications**
  - IPSec for PDN Access Applications
  - IPSec for Mobile IP Applications
  - IPSec for L2TP Applications
  - RADIUS Attributes for IPSec-based Mobile IP Applications
  - IPSec for consumer and enterprise small cell
  - IPSec for Macro Cell
  - IPSec for Unlicensed Mobile Access (UMA)
- **Transform Set Configuration**
- **ISAKMP Policy Configuration**
- **Service Configurations**
  - WSG Service
- **Redundant IPSec Tunnel Fail-Over**
  - Redundant IPSec Tunnel Fail-Over
  - Dead Peer Detection (DPD) Configuration
- **IPSec X.509 Certificates**
- **Rekeying SAs**
- **Access Control**
- **Remote Secrets**
- **IKEv2 RFC 5996 Compliance**
- **Duplicate Session Detection**
- **Extended Sequence Number**
- **Security Gateway as Intiator**



CHAPTER 3

IPSec Network Applications

This chapter describes several methods for implementing IPSec within various network applications.

Topics discussed in this chapter include:

Implementing IPSec for PDN Access Applications

This section provides information on the following topics:

This section assumes that ISAKMP crypto maps are used as opposed to manual crypto maps.

How IPSec-based PDN Access Configuration Works

The following figure and the text that follows describe how sessions accessing a PDN using IPSec are processed by StarOS.

Table 3: IPSec PDN Access Processing

Step	Description
1	A subscriber session or PDP context Request, in GGSN service, arrives at the system.
2	The system processes the subscriber session or request as it would typically.
3	Prior to routing the session packets, the system compares them against configured Access Control Lists (ACLs).

Step	Description
4	The system determines that the packet matches the criteria of an ACL that is associated with a configured crypto map.
5	From the crypto map, the system determines the following: • The map type, in this case ISAKMP • The pre-shared key used to initiate the Internet Key Exchange (IKE) and the IKE negotiation mode • The IP address of the security gateway • Whether perfect forward secrecy (PFS) should be enabled for the IPSec SA and if so, what group should be used • IPSec SA lifetime parameters • The name of a configured transform set defining the IPSec SA
6	To initiate the IKE SA negotiation, the system performs a Diffie-Hellman exchange of the pre-shared key specified in the crypto map with the specified peer security gateway.
7	The system and the security gateway negotiate an ISAKMP policy (IKE SA) to use to protect further communications.
8	Once the IKE SA has been negotiated, the system negotiates an IPSec SA with the security gateway using the transform method specified in the transform sets.
9	Once the IPSec SA has been negotiated, the system protects the data according to the IPSec SAs established during step 8 and sends it over the IPSec tunnel.

Configuring IPSec Support for PDN Access

This section provides a list of the steps required to configure IPSec functionality on the system in support of PDN access. Each step listed refers to a different section containing the specific instructions for completing the required procedure.

Important These instructions assume that the system was previously configured to support subscriber data sessions either as a core service or an HA. In addition, parameters configured using this procedure must be configured in the same destination context on the system.

Step 1 Configure one or more IP access control lists (ACLs) according to the information and instructions located in the *IP Access Control Lists* chapter of the product Administration Guide.

Step 2 Configure one or more transform sets according to the instructions located in the *Transform Set Configuration* chapter of this guide.

Step 3 Configure one or more ISAKMP policies according to the instructions located in the *ISAKMP Policy Configuration* chapter of this guide.

Step 4 Configure an ipsec-isakmp crypto map according to the instructions located in the *ISAKMP Crypto Map Configuration* section of the *Crypto Maps* chapter in this guide.

Step 5 Apply the crypto map to an interface on the system according to the instructions located in the *Crypto Map and Interface Association* section of the *Crypto Maps* chapter in this guide.

Step 6 Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command **save configuration**. For additional information on how to verify and save configuration files, refer to the *System Administration Guide* and the *Command Line Interface Reference*.

Implementing IPSec for Mobile IP Applications

This section provides information on the following topics:

How IPSec-based Mobile IP Configuration Works

The following figure and the text that follows describe how Mobile IP sessions using IPSec are processed by the system.

Figure 2: IPSec-based Mobile IP Session Processing

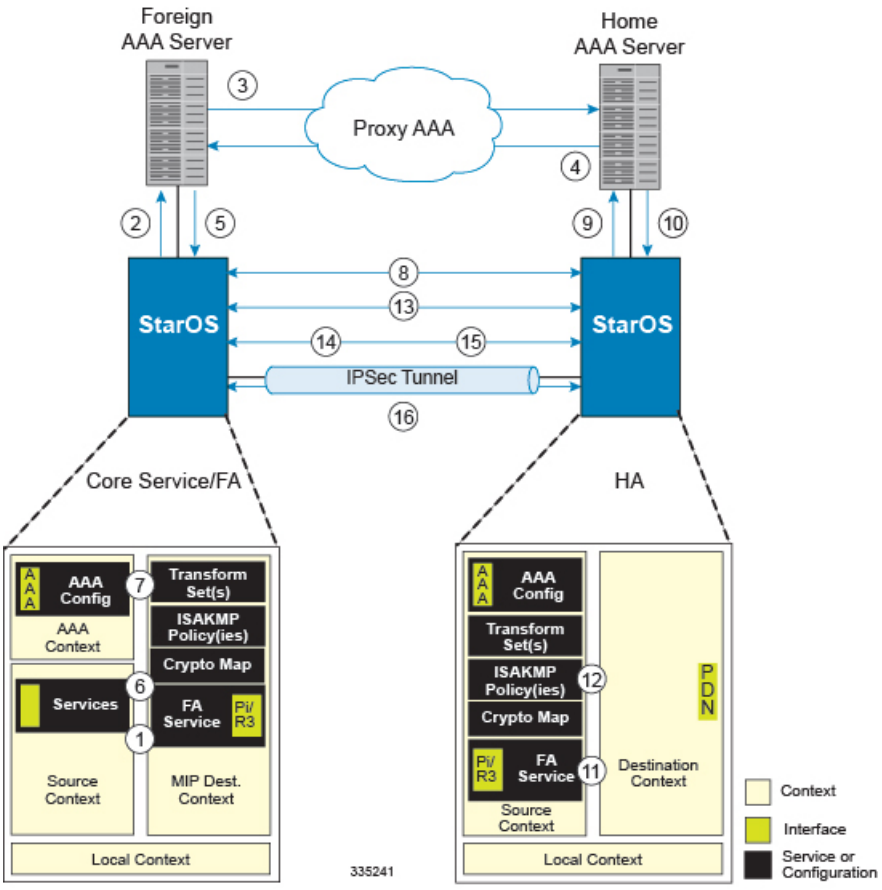


Table 4: IPSec-based Mobile IP Session Processing 0

Step	Description
1	FA service receives a Mobile IP registration request from the mobile node.
2	FA sends an Access-Request to the FAAA server with the 3GPP2-IKE-Secret-Request attribute equal to yes.
3	The FAAA proxies the request to the HAAA.

Step	Description
4	The HAAA returns an Access-Accept message including the following attributes: • 3GPP2-Security-Level set to 3 for IPSec tunnels and registration messages • 3GPP2-MIP-HA-Address indicating the IP address of the HA with which the FA is to communicate • 3GPP2-KeyId providing an identification number for the IKE secret (alternatively, the keys may be statically configured for the FA and/or HA) • 3GPP2-IKE-Secret indicating the pre-shared secret to use to negotiate the IKE SA
5	The FAAA passes the accept message to the FA with all of the attributes.
6	The FA determines if an IPSec SA already exists based on the HA address supplied. If so, that SA will be used. If not, a new IPSec SA will be negotiated.
7	The FA determines the appropriate crypto map to use for IPSec protection based on the HA address attribute. It does this by comparing the address received to those configured using the isakmp peer-ha command. From the crypto map, the system determines the following: • The map type, in this case dynamic • Whether perfect forward secrecy (PFS) should be enabled for the IPSec SA and if so, what group should be used • IPSec SA lifetime parameters • The name of one or more configured transform set defining the IPSec SA
8	To initiate the IKE SA negotiation, the FA performs a Diffie-Hellman (D-H) exchange of the ISAKMP secret specified in the IKE secret attribute with the peer HA dictated by the HA address attribute. Included in the exchange is the Key ID received from the HAAA.

Step	Description
9	Upon receiving the exchange, the HA sends an access request to the HAAA with the following attributes: • 3GPP2-S-Request (note that this attribute is not used if the IPSec keys are statically configured) • 3GPP2-User-name (the username specified is the IP addresses of the FA and HA). The password used in the access request is the RADIUS shared secret.
10	The HAAA returns an Access-Accept message to the HA with the following attributes: • 3GPP2-S indicating the "S" secret used to generate the HA's response to the D-H exchange • 3GPP2-S-Lifetime indicating the length of time that the "S" secret is valid • 3GPP2-Security-Level set to 3 for IPSec tunnels and registration messages (optional)
11	The HA determines the appropriate crypto map to use for IPSec protection based on the FA's address. It does this by comparing the address received to those configured using the isakmp peer-fa command. From the crypto map, the system determines the following: • The map type, in this case dynamic • Whether perfect forward secrecy (PFS) should be enabled for the IPSec SA and if so, what group should be used • IPSec SA lifetime parameters • The name of one or more configured transform set defining the IPSec SA
12	The HA creates a response to the D-H exchange using the "S" secret and the Key ID sent by the FA.
13	The HA sends IKE SA negotiation D-H exchange response to the FA.
14	The FA and the HA negotiate an ISAKMP (IKE) policy to use to protect further communications.

Step	Description
15	Once the IKE SA has been negotiated, the system negotiates an IPSec SA with the security gateway using the transform method specified in the transform sets.
16	Once the IPSec SA has been negotiated, the system protects the data according to the IPSec SAs established during step 15 and sends it over the IPSec tunnel.

Important Once an IPSec tunnel is established between an FA and HA for a particular subscriber, all new Mobile IP sessions using the same FA and HA are passed over the tunnel regardless of whether or not IPSec is supported for the new subscriber sessions. Data for existing Mobile IP sessions is unaffected Configuring IPSec

Configuring IPSec Support for Mobile IP

This section provides a list of the steps required to configure IPSec functionality on the system in support of Mobile IP. Each step listed refers to a different section containing the specific instructions for completing the required procedure.

Important These instructions assume that the systems were previously configured to support subscriber data sessions either as an FA or an HA.

Step 1 Configure one or more transform sets for the FA system according to the instructions located in the *Transform Set Configuration* chapter of this guide.

The transform set(s) must be configured in the same context as the FA service.

Step 2 Configure one or more ISAKMP policies or the FA system according to the instructions located in the *ISAKMP Policy Configuration* chapter of this guide.

The ISAKMP policy(ies) must be configured in the same context as the FA service.

Step 3 Configure an ipsec-isakmp crypto map or the FA system according to the instructions located in the *Dynamic Crypto Map Configuration* section of the *Crypto Maps* chapter of this guide.

The crypto map(s) must be configured in the same context as the FA service.

Step 4 Optional. Configure DPD for the FA to help prevent IPSec tunnel state mismatches between the FA and HA according to the instructions located in the *Dead Peer Detection (DPD) Configuration* section of the *Redundant IPSec Tunnel Fail-Over* chapter of this guide.

Important Though the use of DPD is optional, it is recommended in order to ensure service availability.

Step 5 Configure the FA Service or the FA system according to the instructions located in the *FA Services Configuration to Support IPSec* section of the *Service Configurations* chapter in this guide.

Step 6 Configure one or more transform sets for the HA system according to the instructions located in the *Transform Set Configuration* chapter of this guide.

The transform set(s) must be configured in the same context as the HA service.

Step 7 Configure one or more ISAKMP policies or the HA system according to the instructions located in the *ISAKMP Policy Configuration* chapter of this guide.

The ISAKMP policy(ies) must be configured in the same context as the HA service.

Step 8 Configure an ipsec-isakmp crypto map or the HA system according to the instructions located in the *Dynamic Crypto Map Configuration* section of the *Crypto Maps* chapter of this guide.

The crypto map(s) must be configured in the same context as the HA service.

Step 9 Optional. Configure DPD for the HA to help prevent IPSec tunnel state mismatches between the FA and HA according to the instructions located in the *Dead Peer Detection (DPD) Configuration* section of the *Redundant IPSec Tunnel Fail-Over* chapter of this guide.

Important Though the use of DPD is optional, it is recommended in order to ensure service availability.

Step 10 Configure the HA Service or the HA system according to the instructions located in the *HA Service Configuration to Support IPSec* section in the *Service Configurations* chapter of this guide.

Step 11 Configure the required attributes for RADIUS-based subscribers according to the information located in the *RADIUS Attributes for IPSec-based Mobile IP Applications* chapter of this guide.

Step 12 Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command **save configuration**. For additional information on how to verify and save configuration files, refer to the *System Administration Guide* and the *Command Line Interface Reference*.

RADIUS Attributes for IPSec-based Mobile IP Applications

StarOS uses attributes stored in a subscriber's RADIUS profile to determine how IPSec should be implemented.

The table below lists the attributes that must be configured in the subscriber's RADIUS attributes to support IPSec for Mobile IP. These attributes are contained in the following dictionaries:

- 3GPP2
- 3GPP2-835
- Starent
- Starent-835
- Starent-VSA1
- Starent-VSA1-835

Table 5: Attributes Used for Mobile IP IPSec Support

Attribute	Description	Variable
3GPP2-Security-Level	Indicates the type of security that the home network mandates on the visited network.	Integer value: **3** – Enables IPSec for tunnels and registration messages **4** – Disables IPSec
3GPP2 -KeyId	Contains the opaque IKE Key Identifier for the FA/HA shared IKE secret.	Supported value for the first eight bytes is the network-order FA IP address in hexadecimal characters. Supported value for the next eight bytes is the network-order HA IP address in hexadecimal characters. Supported value for the final four bytes is a timestamp in network order, indicating when the key was created, and is the number of seconds since January 1, 1970, UTC.
3GPP2-IKE-Secret	Contains the FA/HA shared secret for the IKE protocol. This attribute is salt-encrypted.	A binary string of 1 to 127 bytes.
3GPP2-S	Contains the "S" secret parameter used to make the IKE pre-shared secret.	A binary string of the value of "S" consisting of 1 to 127 characters
3GPP2- S-Lifetime	Contains the lifetime of the "S" secret parameter used to make the IKE pre-shared secret.	An integer in network order, indicating the time in seconds since January 1, 1970 00:00 UTC Note that this is equivalent to the Unix operating system expression of time.

Implementing IPSec for L2TP Applications

This section provides information on the following topics:

How IPSec is Used for Attribute-based L2TP Configurations

The following figure and the text that follows describe how IPSec-encrypted attribute-based L2TP sessions are processed by the system.

Table 6: Attribute-based L2TP, IPSec-Encrypted Session Processing 1

Step	
1	A subscriber session arrives at the system.
2	The system attempts to authenticate the subscriber with the AAA server.
3	The profile attributes returned upon successful authentication by the AAA server indicate that session data is to be tunneled using L2TP. In addition, attributes specifying a crypto map name and ISAKMP secret are also supplied indicating that IP security is also required.
4	The system determines that the crypto map name supplied matches a configured crypto map.
5	From the crypto map, the system determines the following: • The map type, in this case dynamic • Whether perfect forward secrecy (PFS) should be enabled for the IPSec SA and if so, what group should be used • IPSec SA lifetime parameters • The name of one or more configured transform set defining the IPSec SA
6	To initiate the IKE SA negotiation, the system performs a Diffie-Hellman exchange of the ISAKMP secret specified in the profile attribute with the specified peer LNS (L2TP Network Server) or security gateway.
7	The system and the LNS or security gateway negotiate an ISAKMP (IKE) policy to use to protect further communications.
8	Once the IKE SA has been negotiated, the system negotiates an IPSec SA with the LNS or security gateway using the transform method specified in the transform sets.

Step	
9	Once the IPSec SA has been negotiated, the system protects the L2TP encapsulated data according to the IPSec SAs established during step 9 and sends it over the IPSec tunnel.

Configuring Support for L2TP Attribute-based Tunneling with IPSec

This section provides a list of the steps required to configure IPSec functionality on the system in support of attributebasedL2TP tunneling. Each step listed refers to a different section containing the specific instructions for completing the required procedure.

Important These instructions assume that the system was previously configured to support subscriber data sessions and L2TP tunneling either as a PDSN or an HA. In addition, with the exception of subscriber attributes, all other parameters configured using this procedure must be configured in the same destination context on the system as the LAC service.

Step 1 Configure one or more transform sets according to the instructions located in the *Transform Set Configuration* chapter of this guide.

Step 2 Configure one or more ISAKMP policies according to the instructions located in the *ISAKMP Policy Configuration* chapter of this guide.

Step 3 Configure an ipsec-isakmp crypto map according to the instructions located in the *Dynamic Crypto Map Configuration* section of the *Crypto Maps* chapter of this guide.

Step 4 Configure the subscriber profile attributes according to the instructions located in the Subscriber Attributes for *L2TP Application IPSec Support* section of the *RADIUS Attributes for IPSec-Based Mobile IP* chapter of this guide.

Step 5 Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command **save configuration**. For additional information on how to verify and save configuration files, refer to the *System Administration Guide* and the *Command Line Interface Reference*.

How IPSec is Used for PDSN Compulsory L2TP Configuration

The following figure and the text that follows describe how IPSec-encrypted PDSN compulsory L2TP sessions are processed by the system.

Table 7: PDSN Compulsory L2TP, IPSec-Encrypted Session Processing 2

Step	Description
1	A subscriber session arrives at a PDSN service on the system that is configured to perform compulsory tunneling. The system uses the LAC service specified in the PDSN service's configuration.

Step	Description
2	The LAC service dictates the peer LNS (L2TP Network Server) to use and also specifies the following parameters indicating that IP security is also required: • Crypto map name • ISAKMP secret
3	The system determines that the crypto map name supplied matches a configured crypto map.
4	From the crypto map, the system determines the following: • The map type, in this case dynamic • Whether perfect forward secrecy (PFS) should be enabled for the IPSec SA and if so, what group should be used • IPSec SA lifetime parameters • The name of one or more configured transform set defining the IPSec SA
5	To initiate the IKE SA negotiation, the system performs a Diffie-Hellman exchange of the ISAKMP secret specified by the attribute with the specified peer LNS or security gateway.
6	The system and the LNS or security gateway negotiate an ISAKMP policy (IKE SA) to use to protect further communications.
7	Once the IKE SA has been negotiated, the system negotiates an IPSec SA with the LNS or security gateway.
8	Once the IPSec SA has been negotiated, the system protects the L2TP encapsulated data according to the rules specified in the transform set and sends it over the IPSec tunnel.

Configuring Support for L2TP PDSN Compulsory Tunneling with IPSec

This section provides a list of the steps required to configure IPSec functionality on the system in support of PDSN compulsory L2TP tunneling. Each step listed refers to a different section containing the specific instructions for completing the required procedure.

Important These instructions assume that the system was previously configured to support PDSN compulsory tunneling subscriber data sessions. In addition, all parameters configured using this procedure must be configured in the same destination context on the system as the LAC service.

Step 1 Configure one or more transform sets according to the instructions located in the *Transform Set Configuration* chapter of this guide.

Step 2 Configure one or more ISAKMP policies according to the instructions located in the *ISAKMP Policy Configuration* chapter of this guide.

Step 3 Configure an ipsec-isakmp crypto map according to the instructions located in the *Dynamic Crypto Map Configuration* section of the *Crypto Maps* chapter of this guide.

Step 4 Configure the subscriber profile attributes according to the instructions located in the Subscriber Attributes for *L2TP Application IPSec Support* section of the *RADIUS Attributes for IPSec-Based Mobile IP* chapter of this guide.

Step 5 Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command **save configuration**. For additional information on how to verify and save configuration files, refer to the *System Administration Guide* and the *Command Line Interface Reference*.

How IPSec is Used for L2TP Configurations on the GGSN

The following figure and the text that follows describe how IPSec-encrypted attribute-based L2TP sessions are processed by the system.

Table 8: GGSN PDP Context Processing with IPSec-Encrypted L

Step	Description
1	A subscriber session/PDP Context Request arrives at the system.
2	The configuration of the APN accessed by the subscriber indicates that session data is to be tunneled using L2TP. In addition, attributes specifying a crypto map name and ISAKMP secret are also supplied indicating that IP security is also required.
3	The system determines that the crypto map name supplied matches a configured crypto map.

Step	Description
4	From the crypto map, the system determines the following: • The map type, in this case dynamic • Whether perfect forward secrecy (PFS) should be enabled for the IPSec SA and if so, what group should be used • IPSec SA lifetime parameters • The name of one or more configured transform set defining the IPSec SA
5	To initiate the IKE SA negotiation, the system performs a Diffie-Hellman exchange of the ISAKMP secret specified in the profile attribute with the specified peer LNS or security gateway.
6	The system and the LNS or security gateway negotiate an ISAKMP (IKE) policy to use to protect further communications.
7	Once the IKE SA has been negotiated, the system negotiates an IPSec SA with the LNS or security gateway using the transform method specified in the transform sets.
8	Once the IPSec SA has been negotiated, the system protects the L2TP encapsulated data according to the IPSec SAs established during step 9 and sends it over the IPSec tunnel.

Configuring GGSN Support for L2TP Tunneling with IPSec

This section provides a list of the steps required to configure the GGSN to encrypt L2TP tunnels using IPSEC. Each step listed refers to a different section containing the specific instructions for completing the required procedure.

Important These instructions assume that the system was previously configured to support subscriber PDP contexts and L2TP tunneling either as a GGSN. In addition, all parameters configured using this procedure must be configured in the same destination context on the system as the LAC service.

Step 1 Configure one or more transform sets according to the instructions located in the *Transform Set Configuration* chapter of this guide.

Step 2 Configure one or more ISAKMP policies according to the instructions located in the *ISAKMP Policy Configuration* chapter of this guide.

Step 3 Configure an ipsec-isakmp crypto map according to the instructions located in the *Dynamic Crypto Map Configuration* section of the *Crypto Maps* chapter of this guide.

Step 4 Configure APN support for encrypting L2TP tunnels using IPSec according to the instructions located in the *APN Template Configuration to Support L2TP* chapter of this guide.

Step 5 Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command **save configuration**. For additional information on how to verify and save configuration files, refer to the *System Administration Guide* and the *Command Line Interface Reference*.

IPSec for LTE/SAE Networks

The Cisco MME (Mobility Management Entity), S-GW (Serving Gateway), and P-GW (Packet Data Network Gateway) support IPSec and IKEv2 encryption using IPv4 and IPv6 addressing in LTE/SAE (Long Term Evolution/System Architecture Evolution) networks. IPSec and IKEv2 encryption enables network domain security for all IP packet switched networks, providing confidentiality, integrity, authentication, and anti-replay protection via secure IPSec tunnels.

Encryption Algorithms

IPSec for LTE/SAE supports the following control and data path encryption algorithms:

- AES-CBC-128 (Advanced Encryption Standard-Cipher Block Chaining-128)
- AES-CBC-256 (Advanced Encryption Standard-Cipher Block Chaining-256)
- DES-CBC (Data Encryption Standard-Cipher Block Chaining)
- 3DES-CBC (Triple Data Encryption Standard-Cipher Bock Chaining)

HMAC Functions

IPSec for LTE/SAE supports the following data path HMAC (Hash-based Message Authentication Code) functions:

- AES-XCBC-MAC-96 (Advanced Encryption Standard-X Cipher Block Chaining-Message Authentication Code-96)
- MD5-96 (Message Digest 5-96)
- SHA1-96 (Secure Hash Algorithm 1-96)

IPSec for LTE/SAE supports the following control path HMAC (Hash-based Message Authentication Code) functions:

- AES-XCBC-MAC-96 (Advanced Encryption Standard-X Cipher Block Chaining-Message Authentication Code-96)
- MD5-96 (Message Digest 5-96)
- SHA1-96 (Secure Hash Algorithm 1-96)
- SHA2-256-128 (Secure Hash Algorithm 2-256-128)

- SHA2-384-192 (Secure Hash Algorithm 2-384-192)
- SHA2-512-256 (Secure Hash Algorithm 2-512-256)

Diffie-Hellman Groups

IPSec for LTE/SAE supports the following Diffie-Hellman groups for IKE and Child SAs (Security Associations):

- Diffie-Hellman Group 1: 768-bit MODP (Modular Exponential) Group
- Diffie-Hellman Group 2: 1024-bit MODP Group
- Diffie-Hellman Group 5: 1536-bit MODP Group
- Diffie-Hellman Group 14: 2048-bit MODP Group
- None: No Diffie-Hellman Group (no perfect forward secrecy)

Dynamic Node-to-Node IPSec Tunnels

IPSec for LTE/SAE enables network nodes to initiate an IPSec tunnel with another node for secure signaling and data traffic between the nodes, enabling up to 64K dynamic, service-integrated IPSec tunnels per chassis. Once established, a dynamic node-to-node IPSec tunnel continues to carry all of the signaling and/or bearer traffic between the nodes. Dynamic node-to-node IPSec for LTE/SAE is supported on the S1-MME interface for signaling traffic between the eNodeB and the MME, on the S1-U interface for data traffic between the eNodeB and the S-GW, and on the S5 interface for data traffic between the S-GW and the P-GW.

Dynamic node-to-node IPSec gets configured using dynamic IKEv2 crypto templates, which are used to specify common cryptographic parameters for the IPSec tunnels such as the encryption algorithm, HMAC function, and Diffie-Hellman group. Additional information necessary for creating node-to-node IPSec tunnels such as revocation lists are fetched dynamically from the IPSec tunnel requests.

For configuration instructions for dynamic node-to-node IPSec, see the configuration chapter in the administration guides for the MME, S-GW, and P-GW.

ACL-based Node-to-Node IPSec Tunnels

Node-to-node IPSec for LTE/SAE can also be configured using crypto ACLs (Access Control Lists), which define the matching criteria used for routing subscriber data packets over an IPSec tunnel. ACL-based node-to-node IPSec tunnels are supported on the S1-MME interface for signaling traffic between the eNodeB and the MME, on the S1-U interface for data traffic between the eNodeB and the S-GW, and on the S5 interface for data traffic between the S-GW and the PGW.

Unlike other ACLs that are applied to interfaces, contexts, or to one or more subscribers, crypto ACLs are applied via matching criteria to crypto maps, which define tunnel policies that determine how IPSec is implemented for subscriber data packets. Prior to routing, the system examines the properties of each subscriber data packet. If the packet properties match the criteria specified in the crypto ACL, the system initiates the IPSec policy dictated by the crypto map. ACL-based node-to-node IPSec tunnels are configured using either IKEv2-IPv4 or IKEv2-IPv6 crypto maps for IPv4 or IPv6 addressing.

Up to 150 ACL-based node-to-node IPSec tunnels are supported on the system, each with one SA bundle that includes one Tx and one Rx endpoint. However, to avoid significant performance degradation, dynamic

node-to-node IPSec tunnels are recommended. If ACL-based node-to-node IPSec tunnels are used, a limit of about ten ACL-based node-to-node IPSec tunnels per system is recommended.

For configuration instructions for ACL-based node-to-node IPSec, see the configuration chapter in the administration guides for the MME, S-GW, and P-GW.

For more information on ACLs, see the *System Administration Guide*.

Traffic Selectors

Per RFC 4306, when a packet arrives at an IPSec subsystem and matches a 'protect' selector in its Security Policy Database (SPD), the subsystem must protect the packet via IPSec tunneling. Traffic selectors enable an IPSec subsystem to accomplish this by allowing two endpoints to share information from their SPDs. Traffic selector payloads contain the selection criteria for packets being sent over IPSec security associations (SAs). Traffic selectors can be created on the P-GW, S-GW, and MME for dynamic node-to-node IPSec tunnels during crypto template configuration by specifying a range of peer IPv4 or IPV6 addresses from which to carry traffic over IPSec tunnels.

For example, consider an eNodeB with an IP address of 1.1.1.1 and an S-GW with a service address of 2.2.2.2. The S-GW is registered to listen for IKE requests from the eNodeBs in the network using the following information:

- Local Address: 2.2.2.2
- Peer Address Network: 1.1.0.0 Mask: 255.255.0.0
- Payload ACL (Access Control List): udp host 2.2.2.2 eq 2123 1.1.0.0 0.0.255.255

When an IKE request arrives the S-GW from eNodeB address 1.1.1.1, the IPSec subsystem converts the payload ACL to: udp host 2.2.2.2 eq 2123 host 1.1.1.1, and this payload becomes the traffic selector for the IPSec tunnel being negotiated.

To properly accommodate control traffic between IPSec nodes, each child SA must include at least two traffic selectors: one with a well-known port in the source address, and one with a well-known port in the destination address. Continuing the example above, the final traffic selectors would be:

- Destination port as well-known port: udp host 2.2.2.2 1.1.0.0 0.0.255.255 eq 2123
- Source port as well-known port: udp host 2.2.2.2 eq 2123 1.1.0.0 0.0.255.255

For ACL-based node-to-node IPSec tunnels, the configured crypto ACL becomes the traffic selector with no modification.

If a TSr (Traffic Selector responder) configuration exists in the crypto template, traffic selector negotiation automatically occurs for the TSr in accordance with RFC 5996 (see exception the note below). If no TSr is configured, the gateway simply respects received traffic selectors and responds with the received traffic selectors. In either case, the gateway can send a maximum of four traffic selectors per TSr.

The negotiation process respects a UE request for a smaller range of IP addresses. Packets are then sent to the target server over the negotiated range.

For additional information on TSr configuration, refer to the *Crypto Template IKEv2-Dynamic Payload Parameters* section in the *Crypto Templates* chapter.

Important For Wireless Security Gateway (WSG) remote access service (RAS), incoming traffic selectors are honored and sent back in the response without negotiation. This exception applies to Security Gateways (SecGWs).

Authentication Methods

IPSec for LTE/SAE includes the following authentication methods:

- **PSK (Pre-Shared Key) Authentication.** A pre-shared key is a shared secret that was previously shared between two network nodes. IPSec for LTE/SAE supports PSK such that both IPSec nodes must be configured to use the same shared secret.
- **X.509 Certificate-based Peer Authentication.** IPSec for LTE/SAE supports X.509 certificate-based peer authentication and CA (Certificate Authority) certificate authentication as described below.

X.509 Certificate-based Peer Authentication

X.509 specifies standard formats for public key certificates, certificate revocation lists, attribute certificates, and a certification path validation algorithm. X.509 certificates are configured on each IPSec node so that it can send the certificate as part of its IKE_AUTH_REQ for the remote node to authenticate it. These certificates can be in PEM (Privacy Enhanced Mail) or DER (Distinguished Encoding Rules) format, and can be fetched from a repository via HTTP or FTP.

CA certificate authentication is used to validate the certificate that the local node receives from a remote node during an IKE_AUTH exchange.

A maximum of sixteen certificates and sixteen CA certificates are supported per system. One certificate is supported per service, and a maximum of four CA certificates can be bound to one crypto template.

For configuration instructions for X.509 certificate-based peer authentication, see the configuration chapter in the administration guides for the MME, S-GW, and P-GW.

The figure below shows the message flow during X.509 certificate-based peer authentication. The table that follows the figure describes each step in the message flow.

For additional information refer to the *IPSec Certificates* chapter of this guide.

Figure 3: X.509 Certificate-based Peer Authentication

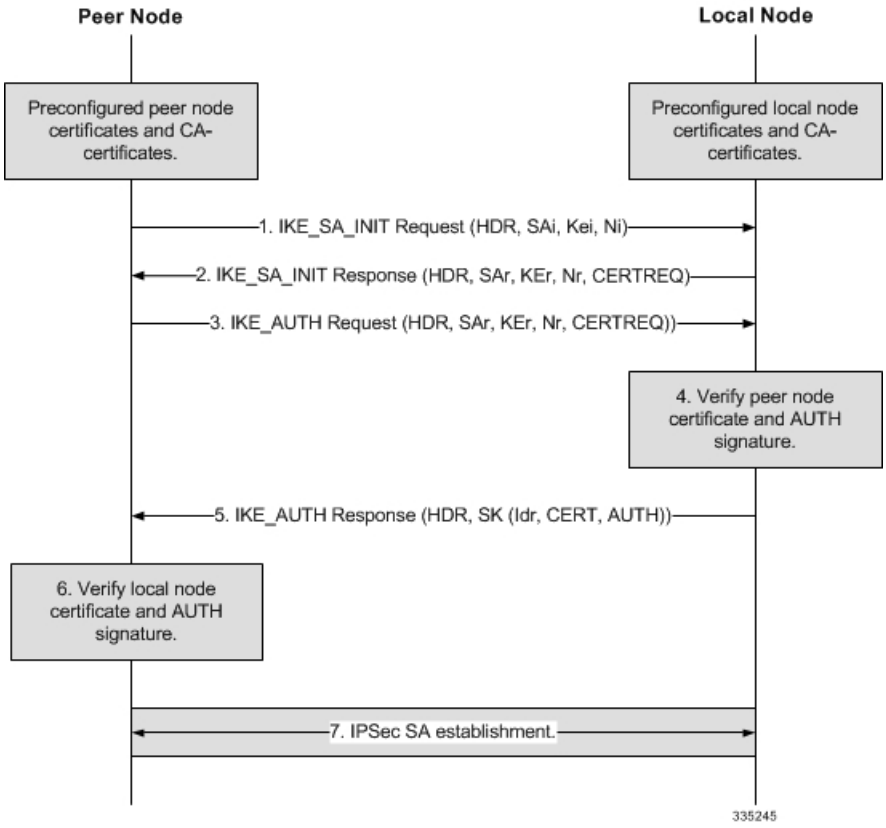


Table 9: X.509 Certificate-based Peer Authentication

Step	Description
1	The peer node initiates an IKEv2 exchange with the local node, known as the IKE_SA_INIT exchange, by issuing an IKE_SA_INIT Request to negotiate cryptographic algorithms, exchange nonces, and perform a Diffie-Hellman exchange with the local node.

Step	Description
2	The local node responds with an IKE_SA_INIT Response by choosing a cryptographic suite from the initiator's offered choices, completing the Diffie-Hellman and nonce exchanges with the peer node. In addition, the local node includes the list of CA certificates that it will accept in its CERTREQ payload. For successful peer authentication, the CERTREQ payload must contain at least one CA certificate that is in the trust chain of the peer certificate. At this point in the negotiation, the IKE_SA_INIT exchange is complete and all but the headers of all the messages that follow are encrypted and integrity-protected.
3	The peer node initiates an IKE_AUTH exchange with the local node by including the IDi payload, setting the CERT payload to the peer certificate, and including the AUTH payload containing the signature of the previous IKE_SA_INIT Request message (in step 1) generated using the private key of the peer certificate. The authentication algorithm used to generate the AUTH payload is also included in the AUTH payload. The peer node also includes the CERTREQ payload containing the list of SHA-1 hash algorithms for local node authentication. For successful server authentication, the CERTREQ payload must contain at least one CA certificate that is in the trust chain of the peer certificate.
4	Using the CA certificate corresponding to the peer certificate, the local node first verifies that the peer certificate in the CERT payload has not been modified and the identity included in the IDi corresponds to the identity in the peer certificate. If the verification is successful, using the public key of the peer certificate, the local node generates the expected AUTH payload and compares it with the received AUTH payload. If they match, the authentication of the peer node is successful. Otherwise, the local node sends an IKEv2 Notification message indicating authentication failure.
5	The local node responds with the IKE_AUTH Response, including the IDr payload, setting the CERT payload to the local node certificate, and including the AUTH payload containing the signature of the IKE_SA_INIT Response message (in step 2) generated using the private key of the local node certificate. The authentication algorithm used to generate the AUTH payload is also included in the AUTH payload.

Step	Description
6	Using the CA certificate corresponding to the local node certificate, the peer node first verifies that the local node certificate in the CERT payload has not been modified. If the verification is successful, using the public key of the local node certificate, the peer generates the expected AUTH payload and compares it with the received AUTH payload. If they match, the local node authentication is successful. This completes the IKE_AUTH exchange.
7	An IPSec SA gets established between the peer node and the local node. If more IPSec SAs are needed, either the peer or local node can initiate the creation of additional Child SAs using a CREATE_CHILD_SA exchange.

Certificate Revocation Lists

Certificate revocation lists track certificates that have been revoked by the CA (Certificate Authority) and are no longer valid. Per RFC 3280, during certificate validation, IPSec for LTE/SAE checks the certificate revocation list to verify that the certificate the local node receives from the remote node has not expired and hence is still valid.

During configuration via the system CLI, one certificate revocation list is bound to each crypto template and can be fetched from its repository via HTTP or FTP.

For additional information refer to the *CRL Fetching* section of the *IPSec Certificates* chapter of this guide.

Child SA Rekey Support

Rekeying of an IKEv2 Child Security Association (SA) occurs for an already established Child SA whose lifetime (either time-based or data-based) is about to exceed a maximum limit. The IPSec subsystem initiates rekeying to replace the existing Child SA. During rekeying, two Child SAs exist momentarily (500ms or less) to ensure that transient packets from the original Child SA are processed by the IPSec node and not dropped.

Child SA rekeying is disabled by default, and rekey requests are ignored. This feature gets enabled in the Crypto Configuration Payload Mode of the system's CLI.

For additional information refer to the *IPSec Certificates* chapter of this guide.

IKEv2 Keep-Alive Messages (Dead Peer Detection)

IPSec for LTE/SAE supports IKEv2 keep-alive messages, also known as Dead Peer Detection (DPD), originating from both ends of an IPSec tunnel. Per RFC 3706, DPD is used to simplify the messaging required to verify communication between peers and tunnel availability. You configure DPD on each IPSec node. You can also disable DPD, and the node will not initiate DPD exchanges with other nodes. However, the node always responds to DPD availability checks initiated by another node regardless of its DPD configuration.

For additional information refer to the *Dead Peer Detection (DPD) Configuration* section of the *Redundant IPSec Tunnel Fail-over* chapter of this guide.

E-UTRAN/EPC Logical Network Interfaces Supporting IPSec Tunnels

The figure below shows the logical network interfaces over which secure IPSec tunnels can be created in an EUTRAN/ EPC (Evolved UMTS Terrestrial Radio Access Network/Evolved Packet Core) network. The table that follows the figure provides a description of each logical network interface.

Figure 4: E-UTRAN/EPC Logical Network Interfaces Supporting IPSec Tunnels

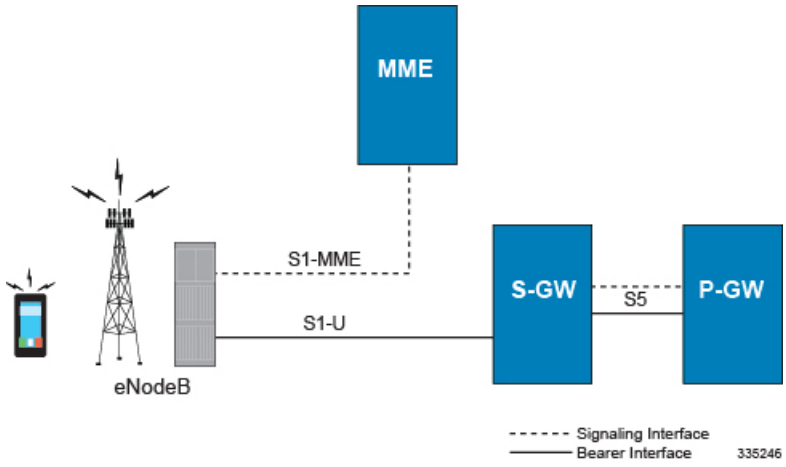


Table 10: E-UTRAN/EPC Logical Network Interfaces Supporting IPSec Tunnels 6

Interface	Description
S1-MME Interface	This interface is the reference point for the control plane protocol between the eNodeB and the MME. The S1-MME interface uses S1-AP (S1- Application Protocol) over SCTP (Stream Control Transmission Protocol) as the transport layer protocol for guaranteed delivery of signaling messages between the MME and the eNodeB (S1). When configured, the S1-AP over SCTP signaling traffic gets carried over an IPSec tunnel. When a subscriber UE initiates a connection with the eNodeB, the eNodeB initiates an IPSec tunnel with the MME, and SCTP signaling for all subsequent subscriber UEs served by this MME gets carried over the same IPSec tunnel. The MME can also initiate an IPSec tunnel with the eNodeB when the following conditions exist: • The first tunnel setup is always triggered by the eNodeB. This is the tunnel over which initial SCTP exchanges occur. • The MME initiates additional tunnels to the eNodeB after an SCTP connection is set up if the MME is multi-homed: a tunnel is initiated from MME's second address to the eNodeB. • The eNodeB is multi-homed: tunnels are initiated from the MME's primary address to each secondary address of the eNodeB. • Both of the prior two conditions: a tunnel is initiated from each of MME's addresses to each address of the eNodeB.
S1-U Interface	This interface is the reference point for bearer channel tunneling between the eNodeB and the S-GW. Typically, the eNodeB initiates an IPSec tunnel with the S-GW over this interface for subscriber data traffic. But the S-GW may also initiate an IPSec tunnel with the eNodeB, if required.

Interface	Description
S5 Interface	This interface is the reference point for tunneling between the S-GW and the P-GW. Based on the requested APN from a subscriber UE, the MME selects both the S-GW and the P-GW that the S-GW connects to. GTP-U data traffic is carried over the IPSec tunnel between the S-GW and P-GW for the current and all subsequent subscriber UEs.

IPSec Tunnel Termination

IPSec tunnel termination occurs during the following scenarios:

- **Idle Tunnel Termination.** When a session manager for a service detects that all subscriber sessions using a given IPSec tunnel have terminated, the IPSec tunnel also gets terminated after a timeout period.
- **Service Termination.** When a service running on a network node is brought down for any reason, all corresponding IPSec tunnels get terminated. This may be caused by the interface for a service going down, a service being stopped manually, or a task handling an IPSec tunnel restarting.
- **Unreachable Peer.** If a network node detects an unreachable peer via Dead Peer Detection (DPD), the IPSec tunnel between the nodes gets terminated. DPD can be enabled per P-GW, S-GW, and MME service via the system CLI during crypto template configuration.
- **E-UTRAN Handover Handling.** Any IPSec tunnel that becomes unusable due to an E-UTRAN network handover gets terminated, while the network node to which the session is handed initiates a new IPSec tunnel for the session.

IPSec for Femto-UMTS Networks

Important HNBGW is not supported from Release 20 and later, and HeNBGW is not supported in Releases 20, 21.0 and 21.1. For more information, contact your Cisco account representative.

The Cisco HNB-GW (Home-NodeB Gateway) supports IPSec and IKEv2 encryption using IPv4 addressing in Femto-UMTS IPSec and IKEv2 encryption enables network domain security for all IP packet-switched networks, providing confidentiality, integrity, authentication, and anti-replay protection via secure IPSec tunnels.

Authentication Methods

IPSec for Femto-UMTS includes the following authentication methods:

- **PSK (Pre-Shared Key) Authentication.** A pre-shared key is a shared secret that was previously shared between two network nodes. IPSec for Femto-UMTS supports PSK such that both IPSec nodes must be configured to use the same shared secret.

- **X.509 Certificate-based Peer Authentication.** IPSec for Femto-UMTS supports X.509 certificate-based peer authentication and CA (Certificate Authority) certificate authentication as described below.

Crypto Map Template Configuration

Use the following example to configure the IPSec profile and crypto template associated with an SeGW and enable IPSec tunneling.

```
configure
   context vpn_ctxt_name
      eap-profile eap_prof_name
         mode authentication-pass-through
         exit
      ip pool ipsec ip_address subnetmask
      ipsec transform-set ipsec_trans_set
      exit
      ikev2 transform-set ikev2_trans_set
      exit
      crypto template crypto_template
         authentication eap-profile eap_prof_name
         exit
         ikev2-ikesa transform set listikev2_trans_set
         payload crypto_payload_name match childsa [ match { ipv4 | ipv6 }
            ip-address-alloc dynamic
            ipsec transform-setlist ipsec_trans_set
            exit
         ikev2-ikesa keepalive-user-activity
         end
configure
   context vpn_ctxt_name
      hnbgw-service hnbgw_svc_name
         security-gateway bind address segw_ip_address crypto-template
crypto_template context segw_ctxt_name
         end
```

Notes:

- *vpn_ctxt_name* is name of the source context in which HNB-GW service is configured
- *segw_ctxt_name* is name of the context in which Se-GW service is configured. By default it takes context where HNB-GW service is configured.
- *hnbgw_svc_name* is name of the HNB-GW service which is to be configured for used for Iuh reference between HNB-GW and HNB

X.509 Certificate-based Peer Authentication

X.509 specifies standard formats for public key certificates, certificate revocation lists, attribute certificates, and a certification path validation algorithm. X.509 certificates are configured on each IPSec node so that it can send the certificate as part of its IKE_AUTH_REQ for the remote node to authenticate it. These certificates

can be in PEM (Privacy Enhanced Mail) or DER (Distinguished Encoding Rules) format, and can be fetched from a repository via HTTP or FTP.

CA certificate authentication is used to validate the certificate that the local node receives from a remote node during an IKE_AUTH exchange.

A maximum of sixteen certificates and sixteen CA certificates are supported per system. One certificate is supported per service, and a maximum of four CA certificates can be bound to one crypto template.

The figure below shows the message flow during X.509 certificate-based peer authentication. The table that follows the figure describes each step in the message flow.

Figure 5: X.509 Certificate-based Peer Authentication

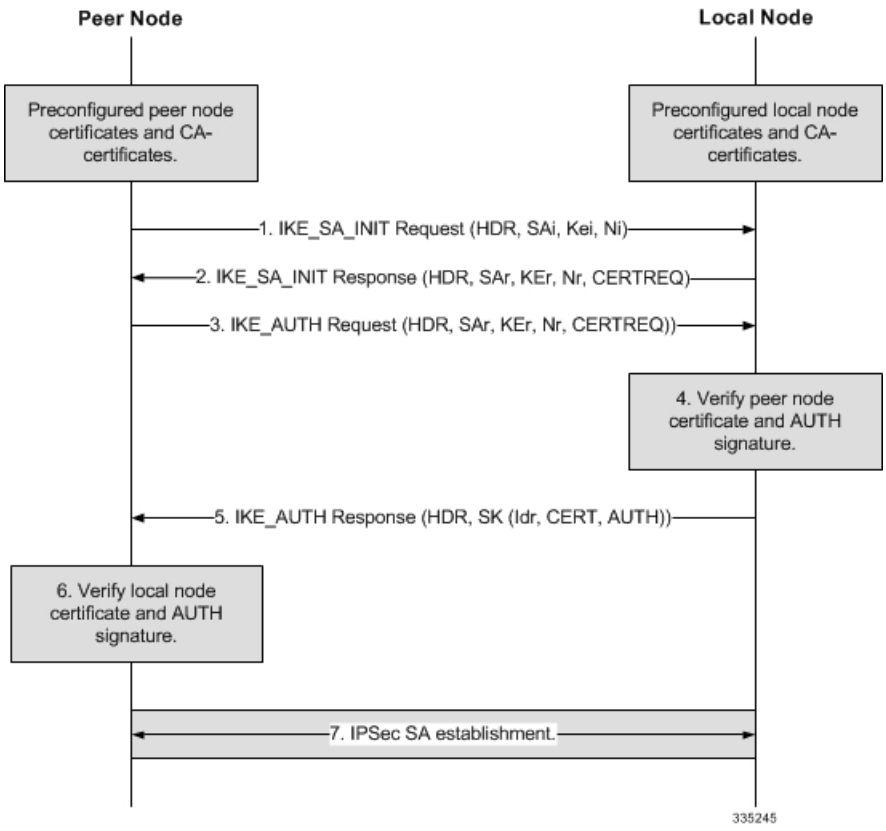


Table 11: X.509 Certificate-based Peer Authentication 9

Step	Description
1	The peer node initiates an IKEv2 exchange with the local node, known as the IKE_SA_INIT exchange, by issuing an IKE_SA_INIT Request to negotiate cryptographic algorithms, exchange nonces, and perform a Diffie-Hellman exchange with the local node.

Step	Description
2	The local node responds with an IKE_SA_INIT Response by choosing a cryptographic suite from the initiator's offered choices, completing the Diffie-Hellman and nonce exchanges with the peer node. In addition, the local node includes the list of CA certificates that it will accept in its CERTREQ payload. For successful peer authentication, the CERTREQ payload must contain at least one CA certificate that is in the trust chain of the peer certificate. At this point in the negotiation, the IKE_SA_INIT exchange is complete and all but the headers of all the messages that follow are encrypted and integrity-protected.
3	The peer node initiates an IKE_AUTH exchange with the local node by including the IDi payload, setting the CERT payload to the peer certificate, and including the AUTH payload containing the signature of the previous IKE_SA_INIT Request message (in step 1) generated using the private key of the peer certificate. The authentication algorithm used to generate the AUTH payload is also included in the AUTH payload. The peer node also includes the CERTREQ payload containing the list of SHA-1 hash algorithms for local node authentication. For successful server authentication, the CERTREQ payload must contain at least one CA certificate that is in the trust chain of the peer certificate.
4	Using the CA certificate corresponding to the peer certificate, the local node first verifies that the peer certificate in the CERT payload has not been modified and the identity included in the IDi corresponds to the identity in the peer certificate. If the verification is successful, using the public key of the peer certificate, the local node generates the expected AUTH payload and compares it with the received AUTH payload. If they match, the authentication of the peer node is successful. Otherwise, the local node sends an IKEv2 Notification message indicating authentication failure.
5	The local node responds with the IKE_AUTH Response, including the IDr payload, setting the CERT payload to the local node certificate, and including the AUTH payload containing the signature of the IKE_SA_INIT Response message (in step 2) generated using the private key of the local node certificate. The authentication algorithm used to generate the AUTH payload is also included in the AUTH payload.

Step	Description
6	Using the CA certificate corresponding to the local node certificate, the peer node first verifies that the local node certificate in the CERT payload has not been modified. If the verification is successful, using the public key of the local node certificate, the peer generates the expected AUTH payload and compares it with the received AUTH payload. If they match, the local node authentication is successful. This completes the IKE_AUTH exchange.
7	An IPSec SA gets established between the peer node and the local node. If more IPSec SAs are needed, either the peer or local node can initiate the creation of additional Child SAs using a CREATE_CHILD_SA exchange.

Certificate Revocation Lists

Certificate revocation lists track certificates that have been revoked by the CA (Certificate Authority) and are no longer valid. Per RFC 3280, during certificate validation, IPSec for LTE/SAE checks the certificate revocation list to verify that the certificate the local node receives from the remote node has not expired and hence is still valid.

During configuration via the system CLI, one certificate revocation list is bound to each crypto template and can be fetched from its repository via HTTP or FTP.

For additional information refer to the *CRL Fetching* section of the *IPSec Certificates* chapter of this guide.

Child SA Rekey Support

Rekeying of an IKEv2 Child Security Association (SA) occurs for an already established Child SA whose lifetime (either time-based or data-based) is about to exceed a maximum limit. The IPSec subsystem initiates rekeying to replace the existing Child SA. During rekeying, two Child SAs exist momentarily (500ms or less) to ensure that transient packets from the original Child SA are processed by the IPSec node and not dropped.

Child SA rekeying is disabled by default, and rekey requests are ignored. This feature gets enabled in the Crypto Configuration Payload Mode of the system's CLI.

For additional information refer to the *IPSec Certificates* chapter of this guide.

IKEv2 Keep-Alive Messages (Dead Peer Detection)

IPSec for LTE/SAE supports IKEv2 keep-alive messages, also known as Dead Peer Detection (DPD), originating from both ends of an IPSec tunnel. Per RFC 3706, DPD is used to simplify the messaging required to verify communication between peers and tunnel availability. You configure DPD on each IPSec node. You can also disable DPD, and the node will not initiate DPD exchanges with other nodes. However, the node always responds to DPD availability checks initiated by another node regardless of its DPD configuration.

For additional information refer to the *Dead Peer Detection (DPD) Configuration* section of the *Redundant IPSec Tunnel Fail-over* chapter of this guide.

IPSec Tunnel Termination

IPSec tunnel termination occurs during the following scenarios:

- **Idle Tunnel Termination.** When a session manager for a service detects that all subscriber sessions using a given IPSec tunnel have terminated, the IPSec tunnel also gets terminated after a timeout period.
- **Service Termination.** When a service running on a network node is brought down for any reason, all corresponding IPSec tunnels get terminated. This may be caused by the interface for a service going down, a service being stopped manually, or a task handling an IPSec tunnel restarting.
- **Unreachable Peer.** If a network node detects an unreachable peer via Dead Peer Detection (DPD), the IPSec tunnel between the nodes gets terminated. DPD can be enabled per P-GW, S-GW, and MME service via the system CLI during crypto template configuration.
- **Network Handover Handling.** Any IPSec tunnel that becomes unusable due to a network handover gets terminated, while the network node to which the session is handed initiates a new IPSec tunnel for the session

x.509 Certificate Configuration

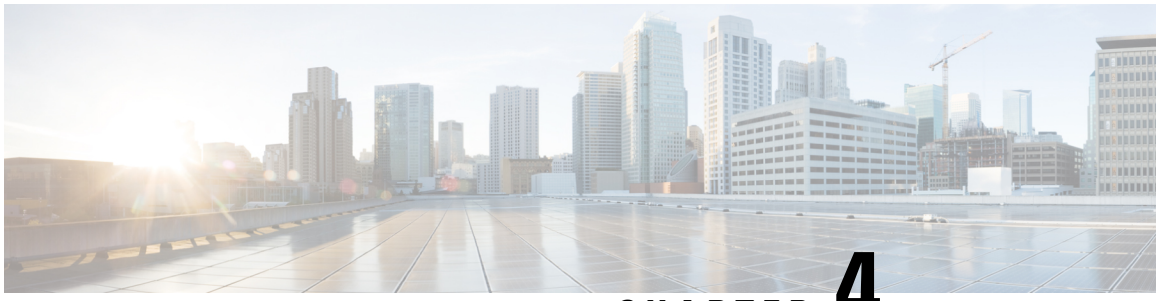
Use the following example to configure the x.509 certificates on the system to provide security certification between FAP and SeGW in Femto-UMTS network.

```
configure
  certificate name x.509_cert_name pem { data pem_data_string | url pem_data_url}
 private-key pem { [encrypted] data PKI_pem_data_string | url PKI_pem_data_url }
     ca-certificate name ca_root_cert_name pem { data pem_data_string | url
pem_data_url }
        exit
     crypto template segw_crypto_template ikev2-dynamic
        authentication local certificate
        authentication remote certificate
        keepalive interval dur timeout dur_timeout
        certificate x.509_cert_name
        ca-certificate list ca-cert-name ca_root_cert_name
        payload crypto_payload_name match childsa [match {ipv4 | ipv6}]
           ip-address-alloc dynamic
           ipsec transform-setlist ipsec_trans_set
           end
configure
  context vpn_ctxt_name
     subscriber default
        ip context-name vpn_ctxt_name
        ip address pool name ip_pool_name
        end
```

Notes:

- *vpn_ctxt_name* is name of the source context in which HNB-GW service is configured.
- *x.509_cert_name* is name of the x.509 certificate where PEM data *pem_data_string* and PKI *PKI_pem_data_string* is configured.

- *ca_root_cert_name* is name of the CA root certificate where PEM data *pem_data_string* is configured for CPE.



CHAPTER 4

Transform Set Configuration

This chapter describes how to configure IPSec transform sets.

A transform set is a combination of individual IPSec transforms designed to enact a specific security policy for traffic. During the ISAKMP IPSec security association negotiation, the peers agree to use a particular transform set for protecting a particular data flow. Transform sets combine the following IPSec factors:

- Mechanism for payload authentication—AH transform
- Mechanism for payload encryption—ESP transform
- IPSec mode (transport versus tunnel)

A transform set is a combination of an AH transform, plus an ESP transform, plus the IPSec mode (either tunnel or transport mode).

The following topics are discussed:

Process Overview

The basic sequence of actions required to configure an IPSec transform set is outlined below.

Step 1 Configure a crypto transform set by applying the example configuration in Configuring a Transform Set, on page 48.

Step 2 Verify your Crypto Transform Set configuration by following the steps in Verifying the Crypto Transform Set Configuration, on page 48.

Step 3 Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command **save configuration**. For additional information on how to verify and save configuration files, refer to the *System Administration Guide* and the *Command Line Interface Reference*.

Configuring a Transform Set

Use the following example to create the crypto transform set:

```
configure
   context ctxt_name
      crypto ipsec transform-set transform_name ah hmac { md5-96 | none
|sha1-96 } esp hmac { { md5-96 | none | sha1-96 } { cipher {des-cbc |
3des-cbc | aes-cbc } | none }
        mode { transport | tunnel }
        end
```

Notes:

- *ctxt_name* is the system context in which you wish to create and configure the crypto transform set(s).
- *transform_name* is the name of the crypto transform set in the current context that you want to configure for IPSec configuration.
- For more information on parameters, refer to the *IPSec Transform Configuration Mode Commands* chapter in the *Command Line Interface Reference*.

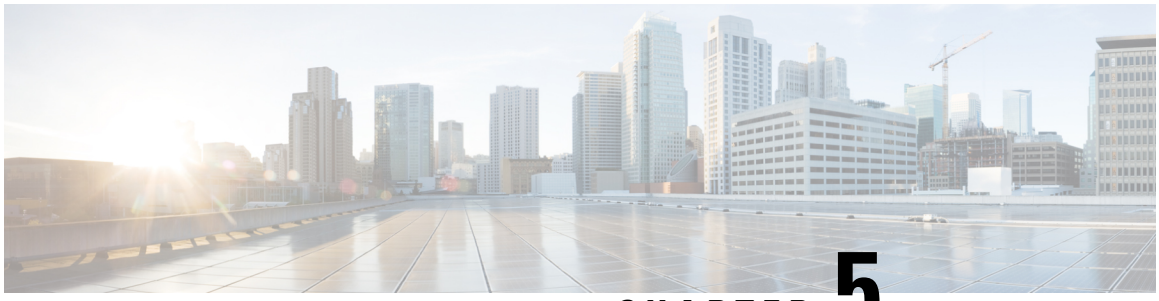
Verifying the Crypto Transform Set Configuration

Enter the following Exec mode command for the appropriate context to display and verify your crypto transform set configuration:

```
show crypto ipsec transform-set transform_name
```

This command produces an output similar to that displayed below using the configuration of a transform set named test1.

```
Transform-Set test1 :
AH  : none
ESP : hmac md5-96, 3des-cbc
Encaps Mode : TUNNEL
```



CHAPTER 5

ISAKMP Policy Configuration

This chapter describes how to create and verify ISAKMP (Internet Security Association Key Management Protocol) policies. ISAKMP is a protocol defined by RFC 2408 for establishing Security Associations (SA) and cryptographic keys in an Internet environment.

ISAKMP defines the procedures for authenticating a communicating peer, creation and management of Security Associations, key generation techniques and threat mitigation (for example, denial of service and replay attacks).

ISAKMP defines procedures and packet formats to establish, negotiate, modify and delete Security Associations. SAs contain all the information required for execution of various network security services, such as the IP layer services (header authentication and payload encapsulation), transport or application layer services or self-protection of negotiation traffic. ISAKMP defines payloads for exchanging key generation and authentication data. These formats provide a consistent framework for transferring key and authentication data which is independent of the key generation technique, encryption algorithm and authentication mechanism.

The following topics are discussed:

Process Overview

The basic sequence of actions required to configure an ISAKMP is outlined below.

Step 1 Configure a policy by applying the example configuration in Configuring ISAKMP Policy, on page 50.

Step 2 Verify your ISAKMP policy configuration by following the steps in Verifying the ISAKMP Policy Configuration, on page 50.

Step 3 Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command **save configuration**. For additional information on how to verify and save configuration files, refer to the *System Administration Guide* and the *Command Line Interface Reference*.

Configuring ISAKMP Policy

Use the following example to create the ISAKMP policy on your system:

```
configure
   context ctxt_name
      ikev1 policy priority
         encryption { 3des-cbc | des-cbc }
         hash { md5 | sha1 }
         group { 1 | 2 | 3 | 4 | 5 }
         lifetime time
         end
```

Notes:

- *ctxt_name* is the system context in which you wish to create and configure the ISAKMP policy.
- *priority* dictates the order in which the ISAKMP policies are proposed when negotiating IKE SAs.
- For more information on parameters, refer to the *ISAKMP Configuration Mode Commands* chapter in the *Command Line Interface Reference*.

Verifying the ISAKMP Policy Configuration

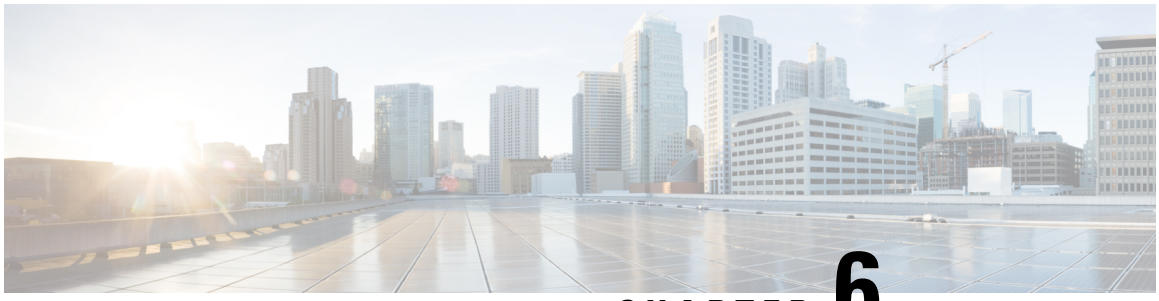
Enter the following Exec mode command for the appropriate context to display and verify your ISAKMP policy configuration:

show crypto isakmp policy *priority*

This command produces an output similar to that displayed below using the configuration of a transform set named test1.

```
1 ISAKMP Policies are configured
             Priority : 1
Authentication Method : preshared-key
             Lifetime : 120 seconds
            IKE group : 5
                 hash : md5
           encryption : 3des-cbc
```

Caution Modification(s) to an existing ISAKMP policy configuration will not take effect until the related security association has been cleared. Refer to the **clear crypto security-association** command located in the *Exec Mode Commands* chapter of the *Command Line Interface Reference* for more information.



CHAPTER 6

Crypto Maps

This chapter describes the various types of IPsec crypto maps supported under StarOS.

A crypto map is a software configuration entity that performs two primary functions:

- Selects data flows that need security processing.
- Defines the policy for these flows and the crypto peer to which that traffic needs to go.

A crypto map is applied to an interface. The concept of a crypto map was introduced in classic crypto but was expanded for IPSec.

Important A **match ip pool** command in a crypto group is not supported within crypto maps on the ASR 5500.

Guidelines are provided for configuring the following types of crypto maps:

- ISAKMP Crypto Map Configuration, on page 51
- Dynamic Crypto Map Configuration, on page 53
- Manual Crypto Map Configuration, on page 54
- Crypto Map and Interface Association, on page 56

ISAKMP Crypto Map Configuration

This section provides instructions for configuring ISAKMP crypto maps.

Important This section provides the minimum instruction set for configuring ISAKMP crypto maps on the system. For more information on commands that configure additional parameters and options, refer to the *Context Configuration Mode Commands* and *Crypto Map ISAKMP Configuration Mode* chapters in the *Command Line Interface Reference*.

To configure the ISAKMP crypto maps for IPSec:

Step 1 Configure ISAKMP crypto map by applying the example configuration in Configuring ISAKMP Crypto Maps, on page 52.

Step 2 Verify your ISAKMP crypto map configuration by following the steps in Verifying the ISAKMP Crypto Map Configuration, on page 52.

Step 3 Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command **save configuration**. For additional information on how to verify and save configuration files, refer to the *System Administration Guide* and the *Command Line Interface Reference*.

Configuring ISAKMP Crypto Maps

Use the following example to create the ISAKMP crypto map:

```
configure
  context ctxt_name
    crypto map map_name ipsec-isakmp
      set peer agw_address
      set isakmp preshared-key isakmp_key
      set mode { aggressive | main }
      set pfs { group1 | group2 | group5 }
      set transform-set transform_name
      match address acl_name [ preference ]
      match crypto-group group_name { primary | secondary }
      end
```

Notes:

- *ctxt_name* is the system context in which you wish to create and configure the ISAKMP crypto maps.
- *map_name* is name by which the ISAKMP crypto map will be recognized by the system.
- *acl_name* is name of the pre-configured Access Control List (ACL). It is used for configurations not implementing the IPSec Tunnel Failover feature and match the crypto map to a previously defined crypto ACL. This is an optional parameter.
- *group_name* is name of the Crypto group configured in the same context. It is used for configurations employing the IPSec Tunnel Failover feature. This is an optional parameter. For more information, refer to the *Redundant IPSec Tunnel Fail-Over* chapter of this guide.
- For more information on parameters, refer to the *Crypto Map ISAKMP Configuration Mode Commands* chapter in the *Command Line Interface Reference*.

Verifying the ISAKMP Crypto Map Configuration

Enter the following Exec mode command for the appropriate context to display and verify your ISAKMP crypto map:

```
show crypto map [ tag map_name | type ipsec-isakmp ]
```

This command produces an output similar to that displayed below that displays the configuration of a crypto map named test_map2.

```
Map Name : test_map2
=======================================
  Payload :
      crypto_acl2: permit tcp host 10.10.2.12 neq 35 any
```

```
Crypto map Type : ISAKMP
IKE Mode : MAIN
IKE pre-shared key : 3fd32rf09svc
Perfect Forward Secrecy : Group2
Hard Lifetime :
      28800 seconds
      4608000 kilobytes
Number of Transforms: 1
Transform : test1
      AH : none
      ESP: md5 3des-cbc
      Encaps mode: TUNNEL
Local Gateway: Not Set
Remote Gateway: 192.168.1.1
```

Caution Modification(s) to an existing ISAKMP crypto map configuration will not take effect until the related security association has been cleared. Refer to the **clear crypto security-association** command located in the *Exec Mode Commands* chapter of the *Command Line Interface Reference* for more information.

Dynamic Crypto Map Configuration

This section provides instructions for configuring dynamic crypto maps. Dynamic crypto maps should only be configured in support of L2TP or Mobile IP applications.

Important This section provides the minimum instruction set for configuring dynamic crypto maps on the system. For more information on commands that configure additional parameters and options, refer to the *Context Configuration Mode Commands* and *Crypto Map Dynamic Configuration Mode Commands* chapters in the *Command Line Interface Reference*.

To configure the dynamic crypto maps for IPSec:

Step 1 Configure dynamic crypto maps by applying the example configuration in Configuring Dynamic Crypto Maps, on page 53.

Step 2 Verify your dynamic crypto map configuration by following the steps in Verifying the Dynamic Crypto Map Configuration, on page 54.

Step 3 Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command **save configuration**. For additional information on how to verify and save configuration files, refer to the *System Administration Guide* and the *Command Line Interface Reference*.

Configuring Dynamic Crypto Maps

Use the following example to create the dynamic crypto map on your system:

```
configure
  context ctxt_name
    crypto map map_name ipsec-dynamic
```

```
set pfs { group1 | group2 | group5 }
set transform-set transform_name
end
```

Notes:

- *ctxt_name* is the system context in which you wish to create and configure the dynamic crypto maps.
- *map_name* is name by which the dynamic crypto map will be recognized by the system.
- For more information on parameters, refer to the *Crypto Map Dynamic Configuration Mode Commands* chapter in the *Command Line Interface Reference*.

Verifying the Dynamic Crypto Map Configuration

Enter the following Exec mode command for the appropriate context to display and verify your dynamic crypto map configuration:

show crypto map [tag *map_name* **| map-type ipsec-dynamic]**

This command produces an output similar to that displayed below using the configuration of a dynamic crypto map named test_map3.

```
Map Name : test_map3
=======================================
  Crypto map Type : ISAKMP (Dynamic)
  IKE Mode : MAIN
  IKE pre-shared key :
  Perfect Forward Secrecy : Group2
  Hard Lifetime :
        28800 seconds
        4608000 kilobytes
  Transform : test1
        AH : none
        ESP: md5 3des-cbc
        Encaps mode: TUNNEL
  Local Gateway: Not Set
  Remote Gateway: Not Set
```

Caution Modification(s) to an existing dynamic crypto map configuration will not take effect until the related security association has been cleared. Refer to the **clear crypto security-association** command located in the *Exec Mode Commands* chapter of the *Command Line Interface Reference* for more information.

Manual Crypto Map Configuration

This section provides the minimum instruction set for configuring manual crypto maps on the system. For more information on commands that configure additional parameters and options, refer to the *Context Configuration Mode Commands* and *Crypto Map Manual Configuration Mode Commands* chapters in the *Command Line Interface Reference*.

To configure the manual crypto maps for IPSec:

Step 1 Configure manual crypto map by applying the example configuration in Configuring Manual Crypto Maps, on page 55.

Step 2 Verify your manual crypto map configuration by following the steps in Verifying the Manual Crypto Map Configuration, on page 55.

Step 3 Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command **save configuration**. For additional information on how to verify and save configuration files, refer to the *System Administration Guide* and the *Command Line Interface Reference*.

Important Because manual crypto map configurations require the use of static security keys (associations), they are not as secure as crypto maps that rely on dynamically configured keys. Therefore, it is recommended that they only be configured and used for testing purposes.

Configuring Manual Crypto Maps

Use the following example to create the manual crypto map on your system:

```
configure
   context ctxt_name
      crypto map map_name ipsec-manual
         set peer agw_address
         match address acl_name [ preference ]
         set transform-set transform_name
         set session-key { inbound | outbound } { ah ah_spi [ encrypted ]
key ah_key | esp esp_spi [ encrypted ] cipher encryption_key [ encrypted ]
authenticator auth_key }
         end
```

Notes:

- *ctxt_name* is the system context in which you wish to create and configure the manual crypto maps.
- *map_name* is name by which the manual crypto map will be recognized by the system.
- *acl_name* is name of the pre-configured ACL. It is used for configurations not implementing the IPSec Tunnel Failover feature and match the crypto map to a previously defined crypto ACL. This is an optional parameter.
- The length of the configured key must match the configured algorithm.
- *group_name* is name of the crypto group configured in the same context. It is used for configurations using the IPSec Tunnel Failover feature. This is an optional parameter.
- For more information on parameters, refer to the *Crypto Map Manual Configuration Mode Commands* chapter in the *Command Line Interface Reference*.

Verifying the Manual Crypto Map Configuration

Enter the following Exec mode command for the appropriate context to display and verify your manual crypto map configuration:

show crypto map [tag *map_name* **| map-type ipsec-manual]**

This command produces an output similar to that displayed below that displays the configuration of a crypto map named test_map.

```
Map Name : test_map
=====================================
  Payload :
        crypto_acl1: permit tcp host 1.2.3.4 gt 30 any
  Crypto map Type : manual(static)
  Transform : test1
        Encaps mode: TUNNEL
  Transmit Flow
        Protocol : ESP
        SPI : 0x102 (258)
        Hmac : md5, key: 23d32d23cs89
        Cipher : 3des-cbc, key: 1234asd3c3d
  Receive Flow
        Protocol : ESP
        SPI : 0x101 (257)          Hmac : md5, key: 008j90u3rjp
        Cipher : 3des-cbc, key: sdfsdfasdf342d32
  Local Gateway: Not Set
  Remote Gateway: 192.168.1.40
```

Caution Modification(s) to an existing manual crypto map configuration will not take effect until the related security association has been cleared. Refer to the **clear crypto security-association** command located in the *Exec Mode Commands* chapter of the *Command Line Interface Reference* for more information.

Crypto Map and Interface Association

This section provides instructions for applying manual or ISAKMP crypto maps to interfaces configured under StarOS.

Important Dynamic crypto maps should not be applied to interfaces.

Important This section provides the minimum instruction set for applying manual or ISAKMP crypto maps to an interface on the system. For more information on commands that configure additional parameters and options, refer to the *Command Line Interface Reference*.

To apply the crypto maps to an interface:

Step 1 Configure a manual or ISAKMP crypto map.

Step 2 Apply the desired crypto map to a system interface by following the steps in Applying a Crypto Map to an Interface, on page 57.

Step 3 Verify your manual crypto map configuration by following the steps in Verifying the Interface Configuration with Crypto Map, on page 57.

Step 4 Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command **save configuration**. For additional information on how to verify and save configuration files, refer to the *System Administration Guide* and the *Command Line Interface Reference*.

Applying a Crypto Map to an Interface

Use the following example to apply an existing crypto map to an interface on your system:

```
configure
   context ctxt_name
      interface interface_name
      crypto-map map_name
      end
```

Notes:

- *ctxt_name* is the system context in which the interface is configured to apply crypto map.
- *interface_name* is the name of a specific interface configured in the context to which the crypto map will be applied.
- *map_name* is name of the preconfigured ISAKMP or a manual crypto map.

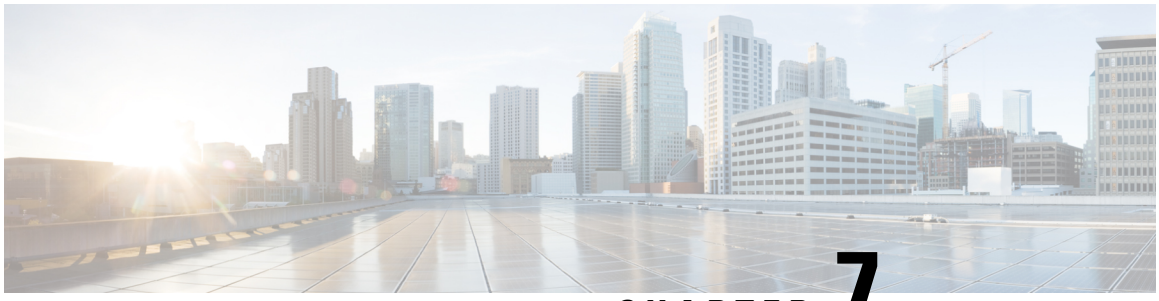
Verifying the Interface Configuration with Crypto Map

Enter the following Exec mode command for the appropriate context to display and verify that your interface is configured properly:

show configuration context *ctxt_name* **| grep interface**

The interface configuration aspect of the display should look similar to that shown below. In this example an interface named 20/6 was configured with a crypto map called isakmp_map1.

```
interface 20/6
ip address 192.168.4.10 255.255.255.0
      crypto-map isakmp_map1
```



CHAPTER 7

ANSSI Enhancements for IKEv1 and IKEv2 ACL Modes

From Release 20 onwards, the ANSSI for ACL modes have been enhanced to provide additional functionalities.

The following topics are discussed:

Feature Description

The ANSSI for ACL modes have been enhanced with the following functionalities:

Auto-delete Existing IKEv1/IKEv2 ACL Tunnels

IPSec will automatically remove existing IKEv1/IKEv2 ACL Tunnels when the following critical parameters are changed in the crypto map:

- When the IPSec or IKE algorithms change in the IPSec/IKE transform set. For example, Encryption, Integrity, PRF, or DH Group algorithms.
- When authentication methods like PSK/Cert change locally or remotely.
- When the PSK keys change.
- When the certificate, CA-Cert list or CA-CRL list changes.
- When a peer address is changed or removed.
- When the transform set in the crpyto-map is changed or removed.
- When an ACL rule that is added or deleted in the existing ACL which is attached to the map.
- When an ACL is removed from the map.
- When an ACL which is attached to the map is deleted.

- [IKEv1 only] When changes occur in the crypto group.
- [IKEv1 only] When changes occur to the IP-Pool which is associated to the crypto map.
- [IKEv1 only] When changes occur to the IKEv1 policy or policy parameters.

Important Critical parameter changes inside the IKEv1 policy will delete all the established tunnels within that context.

Remove Weak Security Algorithms

The following algorithms are considered weak and removed from the IPSec IKEv2 ACL mode:

IKE Tunnel	Encryption	DES-CBC, 3DES-CBC, NULL
	HMAC	AES-XCBC-96, MD5-96, SHA1-96
	DH Group	1, 2
	PRF	AES-XCBC-128, MD5, SHA1
IPSec Tunnel	Encryption	DES-CBC, 3DES-CBC, NULL
	HMAC	AES-XCBC-96, MD5-96, SHA1-96
	DH Group	1, 2, none

The following algorithms are considered weak and removed from the IPSec IKEv1 ACL mode:

IKE Tunnel	Encryption	DES-CBC, 3DES-CBC
	HMAC	MD5
	DH Group	1, 2
IPSec Tunnel	Encryption	DES-CBC, 3DES-CBC
	HMAC	MD5-96, none
	DH Group	1, 2

Configuring ANSSI Enhancements

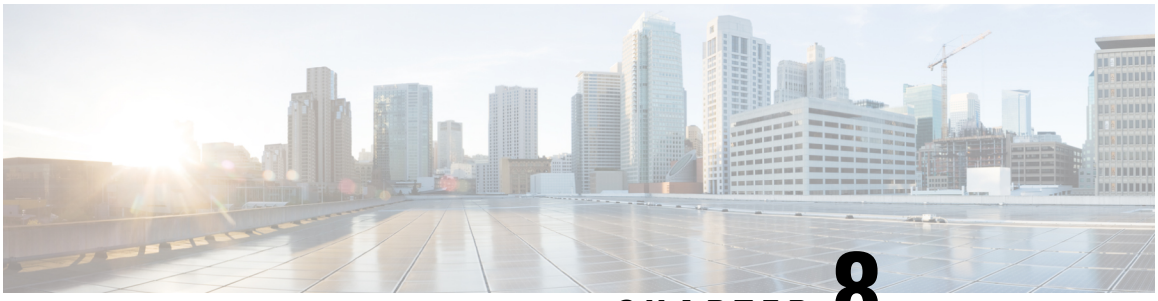
Enabling Auto-deletion of Existing IKEv1/IKEv2 ACL Tunnels

Use the **ikesa delete on-mismatch** command to enable IPSec to automatically remove existing IKEv1 and IKEv2 ACL tunnels when critical parameters are changed in the crypto map.

```
configure
   ikesa delete on-mismatch
   end
```

Notes:

- As per ANSSI standards, this configuration cannot be removed once enabled. The configuration can be removed only by rebooting.
- Use this configuration only on trusted builds.



CHAPTER 8

Crypto Templates

This chapter describes how to configure and use StarOS crypto templates.

The CLI Crypto Template Configuration Mode is used to configure an IKEv2 IPSec policy. It includes most of the IPSec parameters and IKEv2 dynamic parameters for cryptographic and authentication algorithms. A security gateway service will not function without a configured crypto template. Only one crypto template can be configured per service.

The following topics are discussed:

Crypto Template Parameters

A crypto template requires the configuration of the following parameters:

- **allow-cert-enc cert-hash-url** – Enables support for certificate enclosure type other than default.
- **allow-custom-fqdn-idr** – Allows non-standard FQDN (Fully Qualified Domain Name) strings in the IDr (Identification - Responder) payload of IKE_AUTH messages received from the UE with the payload type as FQDN.
- **authentication** – Configures the gateway and subscriber authentication methods to be used by this crypto template.
- **blacklist** – Enables use of a blacklist file
- **ca-certificate list** – Binds an X.509 Certificate Authority (CA) root certificate to a crypto template.
- **ca-crl list** – Binds one or more Certificate Authority-Certificate Revocation Lists (CA-CRLs) to this crypto template.
- **certificate** – Binds a single X.509 trusted certificate to a crypto template.
- **control-dont-fragment** – Controls the Don't Fragment (DF) bit in the outer IP header of the IPSec tunnel data packet.
- **dns-handling** – Adds a custom option to define the ways a DNS address is returned based on proscribed circumstances described below.

- **dos cookie-challenge notify-payload** – Configures the cookie challenge parameters for IKEv2 INFO Exchange notify payloads for the given crypto template.
- **identity local** – Configures the identity of the local IPSec Client (IKE ID).
- **ikev2-ikesa** – Configures parameters for the IKEv2 IKE Security Associations within this crypto template.
- **ip mtu** – Configures the MTU (Maximum Transmission Unit) of the user payload for IPv4 tunnels in bytes.
- **ipv6 mtu** – Configures the MTU of the user payload for IPv6 tunnels in bytes.
- **keepalive** – Configures keepalive or dead peer detection for security associations used within this crypto template.
- **max-childsa** – Defines a soft limit for the number of child Security Associations (SAs) per IKEv2 policy.
- **nai** – Configures the Network Access Identifier (NAI) parameters to be used for the crypto template IDr (recipient's identity).
- **natt** – Configures Network Address Translation - Traversal (NAT-T) for all security associations associated with this crypto template. This feature is disabled by default.
- **ocsp** – Enables Online Certificate Store Protocol (OCSP) requests from the crypto map/template.
- **payload** – Creates a new, or specifies an existing, crypto template payload and enters the Crypto Template Payload Configuration Mode.
- **peer network** – Configures a list of allowed peer addresses on this crypto template.
- **remote-secret-list** – Configures Remote Secret List.
- **whitelist** – Enables use of a whitelist file.

Crypto Template IKEv2-Dynamic Payload Parameters

The Crypto Template IKEv2-Dynamic Payload Configuration Mode is used to assign the correct IPSec transform-set from a list of up to four different transform-sets, and to assign Mobile IP addresses. There should be two payloads configured. The first must have a dynamic addressing scheme from which the ChildSA gets a TIA address. The second payload supplies the ChildSA with a HoA, which is the default setting for ip-address-allocation.

Crypto template payloads include the following parameters:

- **ignore-rekeying-requests** – Ignores CHILD SA rekey requests from the Packet Data Interworking Function (PDIF).
- **ip-address-allocation** – Configures IP address allocation for subscribers using this crypto template payload. Configure two payloads per crypto template. The first must have a dynamic address to assign a tunnel inner address (TIA) to the ChildSA. The second payload is configured after a successful MAnaged IP (MIP) initiation and can use the default Home Address (HoA) option.
- **ipsec transform set** -–Configures the IPSec transform set to be used for this crypto template payload.
- **lifetime** – Configures the number of seconds for IPSec Child SAs derived from this crypto template payload to exist.

- **maximum-child-sa** – Configures the maximum number of IPSec child security associations that can be derived from a single IKEv2 IKE security association.
- **rekey [disallow-param-change]** – Configures IPSec Child Security Association rekeying.
- **tsi** – Configures the IKEv2 Traffic Selector initiator (TSi) payload address options.
- **tsr** – Configures the IKEv2 Traffic Selector responder (TSr) payload address options.

Configuring a Crypto Template

The general command sequence for configuring a crypto template is as follows.

```
configure
  context ctxt_name
    crypto template template_name ikev2-dynamic
      allow-cert-enc cert-hash-url
      allow-custom-fqdn-idr
      authentication { eap-profile name [ second-phase eap-profile name ]
 | local { certificate | pre-shared-key { encrypted key value | key clear_text
 } } |  pre-shared-key { encrypted key value | key clear_text [ second-phase
 eap-profile name ] } | remote { certificate | eap-profile name [
second-phase eap-profile name ] | pre-shared-key { encrypted key value | key
 clear_text [ second-phase eap-profile name ] } } }
      blacklist
      ca-certificate list ca-cert-name name [ ca-cert-name name ]
      ca-crl list ca-crl-name name [ ca-crl-name name ]
      certificate name
      control-dont-fragment { clear-bit | copy-bit | set-bit }
      dns-handling { custom | normal }
     dos cookie-challenge notify-payload [ half-open-sess-count { start
 integer | stop integer } ]
      identity local id-type type id name
      ikev2-ikesa { allow-empty-ikesa | cert-sign { pkcs1.5 | pkcs2.0 }
 | ignore-notify-protocol-id | ignore-rekeying-requests |
keepalive-user-activity | max-retransmissions number | policy {
congestion-rejection [notify-status-value] | error-notification
[invalid-major-version] [invalid-message-id
[invalid-major-version|invalid-syntax]] | invalid-syntax
[invalid-major-version] } | rekey | retransmission-timeout msec |
setup-timer sec | transform-set list name1 name2 name3 name4 name5 name6 }
      keepalive [ interval sec ]
      max-childsa numbr [ overload action { ignore | terminate } ]
     nai { idr name [ id-type { der-asn1-dn | der-asn1-gn | fqdn | ip-addr
 | key-id | rfc822-addr } ] | use-received-idr }
      natt [ include-header ] [ send-keepalive [ idle-interval idle_secs
] [ interval interval_secs ]
      ocsp [ nonce ]
      payload payload_namee match childsa
        ignore-rekeying-requests
        ip-address-allocation { dynamic | home-address }
```

```
          ipsec transform-set list name
          lifetime { sec [ kilo-bytes kbytes ] | kilo-bytes kbytes }
          maximum-child-sa num
          rekey [ keepalive ]
          tsi start-address { any { end-address any } | endpoint {
end-address endpoint } }
       peer network ip_address {/mask | mask ip_mask } [ encrypted
pre-shared-key key | pre-shared-key key ]
       remote-secret-list list_name
       whitelist
       end
```

Notes:

- You can enable **blacklist** or **whitelist**, but not both. For additional information, refer to the *Access Control via Blacklist or Whitelist* section of the *Access Control* chapter of this guide.
- For more information on the above commands and keywords, see the *Crypto Template Configuration Mode Commands* and *Crypto Template IKEv2 Dynamic Payload Configuration Mode Commands* chapters of the *Command Line Interface Reference*.

Verifying a Crypto Template Configuration

Enter the following Exec mode command for the appropriate context to display and verify your crypto template:

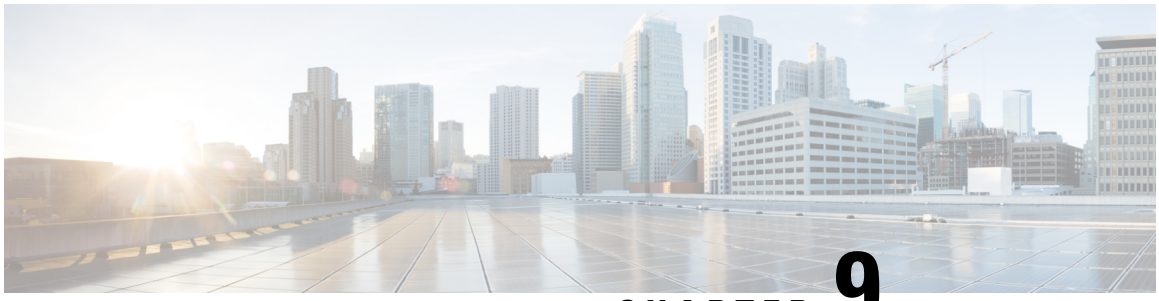
show crypto template tag *map_name*

This command outputs configuration information for the specified template.

The following is a sample output for a crypto template named *wsg-01*.

```
Map Name: wsg01
=======================================

  Map Status: Complete

  Crypto Map Type: IPSEC IKEv2 Template

  IKE SA Transform 1/1

    Transform Set: ikesa-wsg-01
      Encryption Cipher: aes-chc-128
      Pseudo Random Function: sha1
      Hashed Message Authentication Code: sha1-96
      Diffie-Hellman Group: 2
  IKE SA Rekey: Disabled
  Blacklist/Whitelist : None

  OCSP Status:        : Disabled
  OCSP Nounce Status  : Enabled

  NAI: 92.99.99.30

  Remote-secret-list: <not configured>

  Authetication Local:
         Phase 1 - Pre-Shared Key (Size = 3)
```

```
Self-certificate Validation: Diabled


IPSec SA Payload 1/1 (Generic)
  Name : wsg-sa0
  Payload Local
    Protocol 255 Port 0-0 Address Range 76.67.0.1-76.67.0.1
  Payload Remote
    Protocol 255 Port 0-0 Address Range 54.45.0.1-54.45.0.1
  IPSec SA Transform 1/1
    Transform Set: tselsa-wsg
      Protocol: esp
      Encryption Cipher: aes-cbc-128
      Hashed Message Authentication Code: sha1-96
      Diffie-Hellman Group: none
  IPSec SA Rekey: Enabled

Dead Peer Detection: Disabled

Maximum CHILD_SA: 2 Overload Action: Ignore

DOS Cookie Challenge: Disabled
Dont Fragment: Copy bit from inner header

Local Gateway: Not Set
Remote Gateway: Not Set
```



CHAPTER 9

Service Configurations

This chapter describes how to configure various StarOS services to support IPSec.

The following topics are discussed:

FA Services Configuration to Support IPSec

This section provides instructions for configuring FA (Foreign Agent) services to support IPSec. It assumes that the FA service was previously configured and system is ready to serve as an FA.

Important This section provides the minimum instruction set for configuring an FA service to support IPSec on the system. For more information on commands that configure additional parameters and options, see the *Command Line Interface Reference*.

To configure the FA service to support IPSec:

Step 1 Modify FA service configuration by following the steps in Modifying FA Service to Support IPSec, on page 70.

Step 2 Verify your FA service configuration by following the steps in Verifying the FA Service Configuration with IPSec, on page 70.

Step 3 Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command **save configuration**. For additional information on how to verify and save configuration files, refer to the *System Administration Guide* and the *Command Line Interface Reference*.

Modifying FA Service to Support IPSec

Use the following example to modify FA service to support IPSec on your system:

```
configure
   context ctxt_name
      fa-service fa_svc_name
      isakmp peer-ha ha_address crypto-map map_name [ secret preshared_secret ]
      isakmp default crypto-map map_name [ secret preshared_secret ]
      end
```

Notes:

- *ctxt_name* is the system context in which the FA service is configured to support IPSec.
- *fa_svc_name* is name of the FA service for which you are configuring IPSec.
- *ha_address* is IP address of the HA service to which FA service will communicate on IPSec.
- *map_name* is name of the preconfigured ISAKMP or a manual crypto map.
- A default crypto map for the FA service to be used in the event that the AAA server returns an HA address that is not configured as an ISAKMP peer HA.
- For maximum security, the default crypto map should be configured in addition to peer-ha crypto maps instead of being used to provide IPSec SAs to all HAs. Note that once an IPSec tunnel is established between the FA and HA for a particular subscriber, all new Mobile IP sessions using the same FA and HA are passed over the tunnel regardless of whether or not IPSec is supported for the new subscriber sessions. Data for existing Mobile IP sessions is unaffected.

Verifying the FA Service Configuration with IPSec

Enter the following Exec mode command for the appropriate context to display and verify your FA service configuration:

```
show fa-service { name service_name | all }
```

The output of this command is a concise listing of FA service parameter settings.

HA Service Configuration to Support IPSec

This section provides instructions for configuring HA (Home Agent) services to support IPSec. It assumes that the HA service was previously configured and system is ready to serve as an HA.

Important This section provides the minimum instruction set for configuring an HA service to support IPSec on the system. For more information on commands that configure additional parameters and options, see the *Command Line Interface Reference*.

To configure the HA service to support IPSec:

Step 1 Modify HA service configuration by following the steps in Modifying HA Service to Support IPSec, on page 71.

Step 2 Verify your HA service configuration by following the steps in Verifying the HA Service Configuration with IPSec, on page 71.

Step 3 Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command **save configuration**. For additional information on how to verify and save configuration files, refer to the *System Administration Guide* and the *Command Line Interface Reference*.

Modifying HA Service to Support IPSec

Use the following example to modify an existing HA service to support IPSec on your system:

```
configure
  context ctxt_name
    ha-service ha_svc_name
      isakmp aaa-context aaa_ctxt_name
      isakmp peer-fa fa_address crypto-map map_name [ secret preshared_secret ]
      end
```

Notes:

- *ctxt_name* is the system context in which the FA service is configured to support IPSec.
- *ha_svc_name* is name of the HA service for which you are configuring IPSec.
- *fa_address* is IP address of the FA service to which HA service will communicate on IPSec.
- *aaa_ctxt_name* name of the context through which the HA service accesses the HAAA server to fetch the IKE S Key and S Lifetime parameters.
- *map_name* is name of the preconfigured ISAKMP or a manual crypto map.

Verifying the HA Service Configuration with IPSec

Enter the following Exec mode command for the appropriate context to display and verify your HA service configuration:

```
show ha-service { name service_name | all }
```

The output of this command is a concise listing of HA service parameter settings.

PDSN Service Configuration for L2TP Support

PDSN service configuration is required for compulsory tunneling and optional for attribute-based tunneling.

For attribute-based tunneling, a configuration error could occur such that upon successful authentication, the system determines that the subscriber session requires L2TP but can not determine the name of the context in which the appropriate LAC (L2TP Access Concentrator) service is configured from the attributes supplied. As a precautionary, a parameter has been added to the PDSN service configuration options that will dictate the name of the context to use. It is strongly recommended that this parameter be configured.

This section contains instructions for modifying the PDSN service configuration for either compulsory or attribute-based tunneling.

These instructions assume that the PDSN service was previously configured and system is ready to serve as a PDSN.

This section provides the minimum instruction set for configuring an L2TP service on the PDSN system. For more information on commands that configure additional parameters and options, refer to the Command Line Interface Reference.

To configure the PDSN service to support L2TP:

Step 1 Modify PDSN service to configure compulsory tunneling or attribute-based tunneling by applying the example configuration in any of the following sections:

- Modifying PDSN Service to Support Attribute-based L2TP Tunneling, on page 72
- Modifying PDSN Service to Support Compulsory L2TP Tunneling, on page 73

Step 2 Verify your LAC service configuration by following the steps in Verifying the PDSN Service Configuration for L2T, on page 73.

Step 3 Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command **save configuration**. For additional information on how to verify and save configuration files, refer to the *System Administration Guide* and the *Command Line Interface Reference*.

Modifying PDSN Service to Support Attribute-based L2TP Tunneling

Use the following example to modify an existing PDSN service to support attribute-based L2TP tunneling on your system:

```
configure
   context ctxt_name
      pdsn-service pdsn_svc_name
         ppp tunnel-context lac_ctxt_name
         end
```

Notes:

- *ctxt_name* is the destination context where the PDSN service is configured.
- *pdsn_svc_name* is name of the PDSN service for which you are configuring attribute-based L2TP tunneling.
- *lac_ctxt_name* is the name of the destination context where the LAC service is located.
- Refer to for additional information on RADIUS/Subscriber attributes.

RADIUS and Subscriber Attributes for L2TP Application IPSec Support

The table below lists the RADIUS and Subscriber attributes required to support IPSec for use with attribute-based L2TP tunneling.

These attributes are contained in the following dictionaries:

- Starent
- Starent-835

Table 12: Subscriber Attributes for IPSec encrypted L2TP Support

RADIUS Attribute	Local Subscriber Attribute	Description	Variable
SN1-Tunnel-ISAKMP-Crypto-Map	tunnel l2tp crypto-map	The name of a crypto map configured on the system.	A salt-encrypted ASCII string specifying the crypto-map to use for this subscriber. It can be tagged, in which case it is treated as part of a tunnel group.
SN1 -Tunnel-ISAKMP-Secret	tunnel l2tp crypto-map isakmp-secret	The pre-shared secret that will be used as part of the D-H exchange to negotiate an IKE SA.	A salt-encrypted string specifying the IKE secret. It can be tagged, in which case it is treated as part of a tunnel group.

Modifying PDSN Service to Support Compulsory L2TP Tunneling

Use the following example to modify an existing PDSN service to support compulsory L2TP tunneling on your system:

```
configure
   context ctxt_name
      pdsn-service pdsn_svc_name
         ppp tunnel-context lac_ctxt_name
         ppp tunnel-type l2tp
         end
```

Notes:

- *ctxt_name* is the destination context where the PDSN service is configured.
- *pdsn_svc_name* is name of the PDSN service for which you are configuring attribute-based L2TP tunneling.
- *lac_ctxt_name* is the name of the destination context where the LAC service is located.

Verifying the PDSN Service Configuration for L2T

Enter the following Exec mode command for the appropriate context to display and verify your PDSN service with L2TP configuration:

```
show pdsn-service name service_name
```

The output of this command is a concise listing of PDSN service parameter settings configured on the system.

LAC Service Configuration to Support IPSec

This section provides instructions for configuring LAC (L2TP Access Concentrator) services to support IPSec.

Important These instructions are required for compulsory tunneling. They should only be performed for attribute-based tunneling if the Tunnel-Service-Endpoint, the SN1-Tunnel-ISAKMP-Crypto-Map, or the SN1 -Tunnel-ISAKMP-Secret are not configured in the subscriber profile.

These instructions assume that the LAC service was previously configured and system is ready to serve as an LAC server.

Important This section provides the minimum instruction set for configuring an LAC service to support IPSec on the system. For more information on commands that configure additional parameters and options, refer to the *Command Line Interface Reference*.

To configure the LAC service to support IPSec:

Step 1 Modify LAC service configuration by following the steps in Modifying LAC service to Support IPSec, on page 74.

Step 2 Verify your LAC service configuration by following the steps in Verifying the LAC Service Configuration with IPSec, on page 75.

Step 3 Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command **save configuration**. For additional information on how to verify and save configuration files, refer to the *System Administration Guide* and the *Command Line Interface Reference*.

Modifying LAC service to Support IPSec

Use the following example to modify an existing LAC service to support IPSec on your system:

```
configure
   context ctxt_name
      lac-service lac_svc_name
        peer-lns ip_address [encrypted] secret secret [crypto-map map_name {
[encrypted] isakmp-secret secret } ] [ description text ] [ preference integer
 ]
        isakmp aaa-context aaa_ctxt_name
        isakmp peer-fa fa_address crypto-map map_name [ secret preshared_secret ]
        end
```

Notes:

- *ctxt_name* is the destination context where the LAC service is configured to support IPSec.
- *lac_svc_name* is name of the LAC service for which you are configuring IPSec.
- *lns_address* is IP address of the LNS node to which LAC service will communicate on IPSec.

- *aaa_ctxt_name* name of the context through which the HA service accesses the HAAA server to fetch the IKE S Key and S Lifetime parameters.
- *map_name* is name of the preconfigured ISAKMP or a manual crypto map.

Verifying the LAC Service Configuration with IPSec

Enter the following Exec mode command for the appropriate context to display and verify your LAC service with IPSec configuration:

show lac-service name *service_name*

The output of this command is a concise listing of LAC service parameter settings configured on the system.

APN Template Configuration to Support L2TP

This section provides instructions for adding L2TP support for APN templates configured on the system.

These instructions assume that the APN template was previously configured on this system.

Important This section provides the minimum instruction set for configuring an APN template to support L2TP for APN. For more information on commands that configure additional parameters and options, refer to the *Command Line Interface Reference*.

To configure the APN to support L2TP:

Step 1 Modify preconfigured APN template by following the steps in Modifying an APN Template to Support L2TP, on page 75.

Step 2 Verify your APN configuration by following the steps in Verifying the APN Configuration for L2TP, on page 76.

Step 3 Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command **save configuration**. For additional information on how to verify and save configuration files, refer to the *System Administration Guide* and the *Command Line Interface Reference*.

Modifying an APN Template to Support L2TP

Use the following example to modify APN template to support L2TP:

```
configure
  context ctxt_name
    apn apn_name
      tunnel l2tp [ peer-address lns_address [ [ encrypted ] secret l2tp_secret
 ] [ preference num ] [ tunnel-context tunnel_ctxt_name ] [ local-address
agw_ip_address ] [ crypto-map map_name { [ encrypted ] isakmp-secret crypto_secret
 } ]
       end
```

Notes:

- *ctxt_name* is the system context in which the APN template is configured.
- *apn_name* is name of the preconfigured APN template in which you want to configure L2TP support.
- *lns_address* is the IP address of the LNS node with which this APN will communicate.
- *tunnel_ctxt_name* is the L2TP context in which the L2TP tunnel is configured.
- *agw_ip_address* is the local IP address of the GGSN in which this APN template is configured.
- *map_name* is the preconfigured crypto map (ISAKMP or manual) which is to use for L2TP.

Verifying the APN Configuration for L2TP

Enter the following Exec mode command for the appropriate context to display and verify your APN L2TP configuration:

show apn name *apn_name*

The output of this command contains a concise listing of LT2P settings configured for the specified APN.

WSG Service Configuration to Support IPSec

This section provides an overview of the process for enabling a WSG service with a crypto template supporting IPSec features. WSG service must be enabled to support a Security Gateway (SecGW) running on an ASR 9000 router equipped with a Virtualized Services Module (VSM).

For additional information refer to the *Security Gateway Administration Guide*.

Creating a Crypto Template to Support a SecGW

The StarOS CLI Crypto Template Configuration Mode is used to configure an IKEv2 IPSec policy. It includes most of the IPSec parameters and IKEv2 dynamic parameters for cryptographic and authentication algorithms. A security gateway service will not function without a configured crypto template. Only one crypto template can be configured per service.

A crypto template for a SecGW may require the configuration of the following parameters:

- **allow-cert-enc cert-hash-url** – Enables support for certificate enclosure type other than default.
- **allow-custom-fqdn-idr** – Allows non-standard FQDN (Fully Qualified Domain Name) strings in the IDr (Identification - Responder) payload of IKE_AUTH messages received from the UE with the payload type as FQDN.
- **authentication** – Configures the gateway and subscriber authentication methods to be used by this crypto template.
- **blacklist** – Enables use of a blacklist file
- **ca-certificate list** – Binds an X.509 Certificate Authority (CA) root certificate to a crypto template.
- **ca-crl list** – Binds one or more Certificate Authority-Certificate Revocation Lists (CA-CRLs) to this crypto template.
- **certificate** – Binds a single X.509 trusted certificate to a crypto template.

- **control-dont-fragment** – Controls the Don't Fragment (DF) bit in the outer IP header of the IPSec tunnel data packet.
- **dns-handling** – Adds a custom option to define the ways a DNS address is returned based on proscribed circumstances described below.
- **dos cookie-challenge notify-payload** – Configures the cookie challenge parameters for IKEv2 INFO Exchange notify payloads for the given crypto template.
- **identity local** – Configures the identity of the local IPSec Client (IKE ID).
- **ikev2-ikesa** – Configures parameters for the IKEv2 IKE Security Associations within this crypto template.
- **keepalive** – Configures keepalive or dead peer detection for security associations used within this crypto template.
- **max-childsa** – Defines a soft limit for the number of child Security Associations (SAs) per IKEv2 policy.
- **nai** – Configures the Network Access Identifier (NAI) parameters to be used for the crypto template IDr (recipient's identity).
- **natt** – Configures Network Address Translation - Traversal (NAT-T) for all security associations associated with this crypto template. This feature is disabled by default.
- **ocsp** – Enables Online Certificate Store Protocol (OCSP) requests from the crypto map/template.
- **payload** – Creates a new, or specifies an existing, crypto template payload and enters the Crypto Template Payload Configuration Mode.
- **peer network** – Configures a list of allowed peer addresses on this crypto template.
- **remote-secret-list** – Configures Remote Secret List.
- **whitelist** – Enables use of a whitelist file.

You must create a crypto template before creating the WSG service that enables the SecGW.

Creating a WSG Service

Execute the following command sequence to move to the Wireless Security Gateway Configuration Mode:

```
config
   context context_name
   wsg-service service_name
      bind address ip_address crypto-template template_name
      deployment-mode { remote-access | site-to-site }
      ip { access-group acl_list_name | address pool name pool_name
      ipv6 { access-group acl_list_name | address prefix-pool pool_name
      pre_fragment mtu size
```

The following command sequence sets the lookup priority:

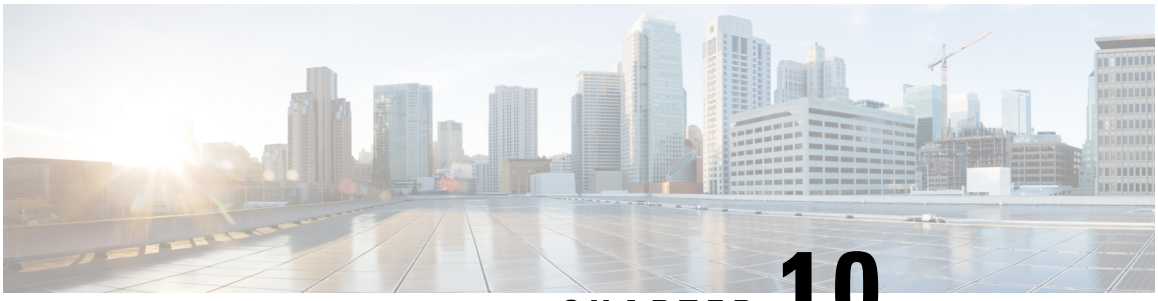
```
config
   wsg-lookup
      priority priority_level source-netmask subnet_size destination netmask
subnet_size
```

For additional information, see the *WSG-Service Configuration Mode Commands* and the *WSG Lookup Priority List Configuration Mode* chapters of the *Command Line Interface Reference*.

Verifying WSG Service Creation

The following Exec mode **show** commands display information associated with WSG service parameters and operating statistics. For detailed descriptions of these commands, see the *Exec Mode show Commands* chapter of the *Command Line Interface Reference*.

- **show wsg-lookup** – Displays the priority levels, as well source an destination netmasks for all configured lookup priorities.
- **show wsg-service** – Displays information about all WSG services or a specified service. It also displays statistics for a specified WSG service or peer address.



CHAPTER 10

Redundant IPSec Tunnel Fail-over

This chapter describes the redundant IPSec tunnel fail-over feature and dead peer detection (DPD).

The following topics are discussed:

Redundant IPSec Tunnel Fail-over (IKEv1)

Overview

The Redundant IPSec Tunnel Fail-Over functionality is included with the IPSec feature license and allows the configuration of a secondary ISAKMP crypto map-based IPSec tunnel over which traffic is routed in the event that the primary ISAKMP crypto map-based tunnel cannot be used.

This feature introduces the concept of crypto (tunnel) groups when using IPSec tunnels for access to packet data networks (PDNs). A crypto group consists of two configured ISAKMP crypto maps. Each crypto map defines the IPSec policy for a tunnel. In the crypto group, one tunnel serves as the primary, the other as the secondary (redundant). Note that the method in which the system determines to encrypt user data in an IPSec tunnel remains unchanged.

Group tunnels are perpetually maintained with IPSec Dead Peer Detection (DPD) packets exchanged with the peer security gateway.

Important The peer security gateway must support RFC 3706 in order for this functionality to work properly.

When the system determines that incoming user data traffic must be routed over one of the tunnels in a group, the system automatically uses the primary tunnel until either the peer is unreachable (the IPSec DPD packets cease), or the IPSec tunnel fails to re-key. If the primary peer becomes unreachable, the system automatically begins to switch user traffic to the secondary tunnel. The system can be configured to either automatically switch user traffic back to the primary tunnel once the corresponding peer security gateway is reachable and the tunnel is configured, or require manual intervention to do so.

This functionality also supports the generation of Simple Network Management Protocol (SNMP) notifications indicating the following conditions:

- **Primary Tunnel is down.** A primary tunnel that was previously "up" is now "down" representing an error condition.
- **Primary Tunnel is up.** A primary tunnel that was previously "down" is now "up".
- **Secondary tunnel is down.** A secondary tunnel that was previously "up" is now "down" representing an error condition.
- **Secondary Tunnel is up.** A secondary tunnel that was previously "down" is now "up".
- **Fail-over successful.** The switchover of user traffic was successful. This is generated for both primary-to-secondary and secondary-to-primary switchovers.
- **Unsuccessful fail-over.** An error occurred when switching user traffic from either the primary to secondary tunnel or the secondary to primary tunnel.

Supported RFC Standard

The Redundant IPSec Tunnel Fail-over feature supports.RFC 3706, A Traffic-Based Method of Detecting Dead Internet Key Exchange (IKE) Peers, February 2004

Redundant IPSec Tunnel Fail-over Configuration

This section provides information and instructions for configuring the Redundant IPSec Tunnel Fail-over feature. These instructions assume that the system was previously configured to support subscriber data sessions either as a core service or an HA.

Important Parameters configured using this procedure must be configured in the same StarOS context.

Important StarOS supports a maximum of 32 crypto groups per context. However, configuring crypto groups to use the same loopback interface for secondary IPSec tunnels is not recommended and may compromise redundancy on the chassis.

Important This section provides the minimum instruction set for configuring crypto groups on the system. For more information on commands that configure additional parameters and options, refer *Command Line Interface Reference*.

To configure the Crypto group to support IPSec:

Step 1 Configure a crypto group by following the steps in Configuring a Crypto Group, on page 81.

Step 2 Configure one or more ISAKMP policies according to the instructions provided in the *ISAKMP Policy Configuration* chapter of this guide.

Step 3 Configure IPSec DPD settings using the instructions provided in Configuring DPD for a Crypto Group, on page 83.

Step 4 Configure an ISAKMP crypto map for the primary and secondary tunnel according to the instructions provided in the *ISAKMP Crypto Map Configuration* section of the *Crypto Maps* chapter of this guide.

Step 5 Match the existing ISAKMP crypto map to Crypto group by following the steps in Modifying a ISAKMP Crypto Map Configuration to Match a Crypto Group, on page 81.

Step 6 Verify your Crypto Group configuration by following the steps in Verifying the Crypto Group Configuration, on page 82.

Step 7 Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command **save configuration**. For additional information on how to verify and save configuration files, refer to the *System Administration Guide* and the *Command Line Interface Reference*.

Configuring a Crypto Group

Use the following example to configure a crypto group on your system for redundant IPSec tunnel fail-over support:

```
configure
  context ctxt_name
    ikev1 keepalive dpd interval dur timeout dur num-retry retries
    crypto-group group_name
      match address acl_name [ preference ]
      match ip pool pool-name pool_name
      switchover auto [ do-not-revert ]
      end
```

Important The **match ip pool** command is not supported within a crypto group on the ASR 5500 platform.

Notes:

- *ctxt_name* is the destination context where the Crypto Group is to be configured.
- *group_name* is name of the Crypto group you want to configure for IPSec tunnel failover support.
- *acl_name* is name of the pre-configured crypto ACL. It is used for configurations not implementing the IPSec Tunnel Failover feature and match the crypto map to a previously defined crypto ACL. For more information on crypto ACL, refer to the Access Control chapter of this guide.
- *pool_name* is the name of an existing IP pool that should be matched.

Modifying a ISAKMP Crypto Map Configuration to Match a Crypto Group

Use the following example to match the crypto group with ISAKMP crypto map:

```
configure
  context ctxt_name
    crypto map map_name1 ipsec-isakmp
      match crypto-group group_name primary
      end
configure
```

```
context ctxt_name
   crypto map map_name2 ipsec-isakmp
      match crypto-group group_name seondary
      end
```

Notes:

- *ctxt_name* is the system context in which you wish to create and configure the ISAKMP crypto maps.
- *group_name* is name of the Crypto group configured in the same context for IPSec Tunnel Failover feature.
- *map_name1* is name of the preconfigured ISAKMP crypto map to match with crypto group as primary.
- *map_name2* is name of the preconfigured ISAKMP crypto map to match with crypto group as secondary.

Verifying the Crypto Group Configuration

Enter the following Exec mode command for the appropriate context to display and verify your crypto group configuration:

show crypto group [summary | name *group_name* **]**

The output of this command is a concise listing of crypto group parameter settings configured on the system.

Dead Peer Detection (DPD) Configuration

This section provides instructions for configuring the Dead Peer Detection (DPD).

Defined by RFC 3706, Dead Peer Detection (DPD) is used to simplify the messaging required to verify communication between peers and tunnel availability.

DPD is configured at the context level and is used in support of the IPSec Tunnel Failover feature (refer to the Redundant IPSec Tunnel Fail-Over section) and/or to help prevent tunnel state mismatches between an FA and HA when IPSec is used for Mobile IP applications. When used with Mobile IP applications, DPD ensures the availability of tunnels between the FA and HA. (Note that the starIPSECDynTunUp and starIPSECDynTunDown SNMP traps are triggered to indicate tunnel state for the Mobile IP scenario.)

Regardless of the application, DPD must be supported/configured on both security peers. If the system is configured with DPD but it is communicating with a peer that does not have DPD configured, IPSec tunnels still come up. However, the only indication that the remote peer does not support DPD exists in the output of the **show crypto isakmp security-associations summary** command.

Important If DPD is enabled while IPSec tunnels are up, it will not take affect until all of the tunnels are cleared.

Important DPD must be configured in the same StarOS context as other IPSec Parameters.

To configure the Crypto group to support IPSec:

Step 1 Enable dead peer detection on system in support of the IPSec Tunnel Failover feature by following the steps in Configuring DPD for a Crypto Group, on page 83.

Step 2 Verify your DPD configuration by following the steps in Verifying the DPD Configuration, on page 83

Step 3 Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command **save configuration**. For additional information on how to verify and save configuration files, refer to the *System Administration Guide* and the *Command Line Interface Reference*.

Configuring DPD for a Crypto Group

Use the following example to configure a crypto group on your system for redundant IPSec tunnel fail-over support:

```
configure
  context ctxt_name
    ikev1 keepalive dpd interval dur timeout dur num-retry retries
    end
```

Notes:

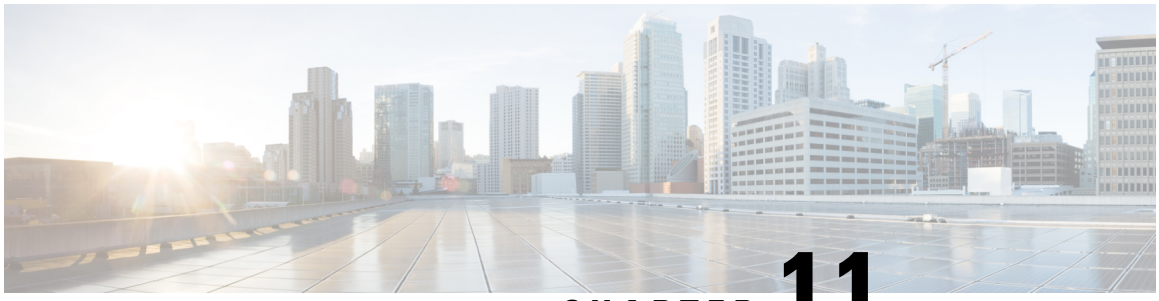
- *ctxt_name* is the destination context where the Crypto Group is to be configured.

Verifying the DPD Configuration

Enter the following Exec mode command for the appropriate context to display and verify your crypto group with DPD configuration:

```
show crypto group [ summary | name group_name ]
```

The output of this command is a concise listing of crypto group parameter settings configured on the system.



CHAPTER 11

IPSec X.509 Certificates

This chapter describes a number of StarOS features that support IPSec certificate management.

The following topics are discussed:

Multiple Child SA (MCSA) Support

Overview

A child SA is an Encapsulating Security Payload (ESP) or Authentication header (AH) security association (SA) carrying the secure user traffic. An SA is a "simplex connection"; to achieve bidirectional secure traffic a pair of SAs is required (RFC 5996). To meet this common requirement, IKE explicitly creates SA pairs. An SA pair is referred to as a "Child SA"; one child SA is a pair of IPsec SAs in each direction.

StarOS supports creation up to five child SAs under the crypto template configuration. Child SAs are supported only for IKEv2.

Each child SA should consist of mutually exclusive traffic selectors which are configured via crypto template payloads.

The following traffic selectors would match UDP packets from 198.51.100.66 to anywhere, with any of the four combinations of source/destination ports (100,300), (100,400), (200,300), and (200, 400). Thus, some types of policies may require several Child SA pairs. For instance, a policy matching only source/destination ports (100,300) and (200,400), but not the other two combinations, cannot be negotiated as a single Child SA pair.

```
TSi = ((17, 100, 198.51.100.66-198.51.100.66),(17, 200, 198.51.100.66-198.51.100.66))
TSr = ((17, 300, 0.0.0.0-255.255.255.255),(17, 400, 0.0.0.0-255.255.255.255))
```

The following triggers create Child SAs:

- The initiator of IKE_INIT can start subsequent Child SA creations after the first Child SA creation based on initiator traffic selector (TSi) configuration which calls for multiple Child SAs. StarOS receives CREATE_CHILD_SA request after IKE_AUTH.
- The responder can initiate subsequent Child SA creation after the first child SA creation based on the responder traffic selector configurations (TSr) which calls for multiple Child SAs. StarOS sends CREATE_CHILD_SA request after IKE_AUTH.

Deployment Scenarios

The creation of multiple child SAs helps an operator to segregate and limit the secure traffic into multiple flows. For example, control and data paths between two nodes can be established over two child SAs; the rest of the data between the nodes will bypass IPSec.

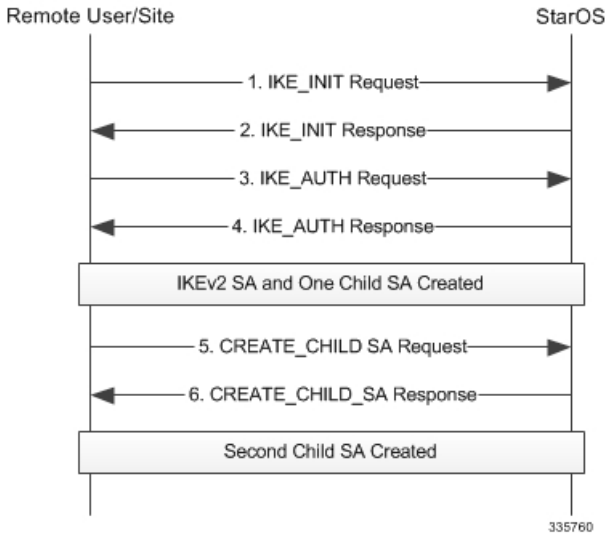
Multiple child SAs may be used for carrying traffic with different class of services (QoS). Similarly, different SAs could be used to carry different traffic with specific security properties. For example, one SA with strongest protection, another with a weaker one, and still another with a proprietary one stipulated by legal, performance or cost needs.

Call Flows

Child SA Creation by Initiator

With crypto template configuration, Child SA creation is initiated by the IKE_INIT initiator through a CREATE_CHILD_SA exchange or by StarOS acting as the responder. The first Child SA is created using the first traffic selector. After creating the first Child SA, the initiator requests the second Child SA using the second traffic selector. The responder completes the creation of the second Child SA.

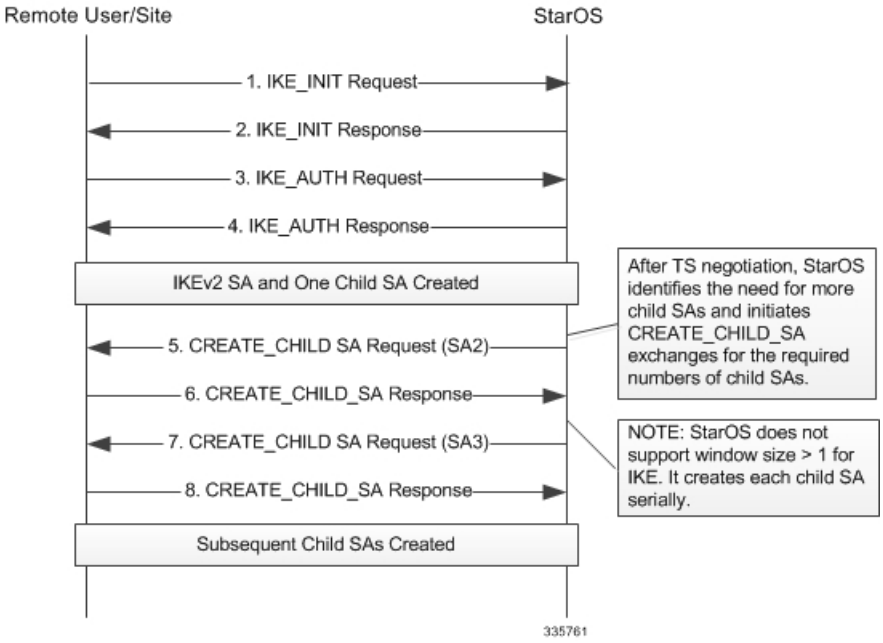
Figure 6: Child SA Creation Initiated by IKE_INIT



Child SA Creation by Responder

After negotiating a transform set (TS), the responder detects the need to create more child SAs to support configured traffic selectors. It sends CREATE_CHILD_SA to create as many child SAs as required to meet the TS configuration. The initiator completes subsequent child SA creations.

Figure 7: Child SA Creation Initiated by StarOS as Responder



Creating, Signing, and Configuring Certificates

Use the following procedure to create, sign, and configure certificates:

1. Add a file location where the certificates and private keys will be stored:

   ```
   config
      cmp cert-store location path_name
      end
   ```

2. Generate a Certificate Signing Request (CSR):

   ```
   crypto rsa-keygen modulus { 1024 | 2048 | 4096 | 512 } id-type { fqdn
    id fqdn_id | ip id IP_address | rfc822-addr id id_type } subject-name
   subject_string
   ```

   A new private key along with the certificate request will be generated in the configured file location.

3. Use the generated certificate request to apply for a digital identity certificate from the certificate authority (CA).

4. Once the certificate is received, download and configure the certificate file to an accessible path:

   ```
   certificate name name { der url pathname | pem { data pemdata | url pathname
    } private-key pem { [ encrypted ] data pemdata | url pathname [cert-enc]
    [cert-hash-url url patname ] } }
   ```

Notes:

- Use the **no cmp cert-store** command to remove the certificate storage location configuration.
- Use the **no certificate name** *name* command to remove the certificate configuration.
- *pathname* must be in one of the following formats:
  - [file:]{ /flash | /usb1 | /hd-raid }[/*directory*]/*filename*
  - tftp://*host*[:*port*][/*directory*]/*filename*
  - ftp://[*username*[:*password*]@]*host*[:*port*][/*directory*]/*filename*
  - sftp://[*username*[:*password*]@]*host*[:*port*][/*directory*]/*filename*
  - http://[*username*[:*password*]@]*host*[:*port*][/*directory*]/*filename*
- *fqdn_id*, *id_type*, *subject_string* must be an alphanumeric string from 1 to 256 characters
- *IP_address* can be an IPv4 address with dotted-decimal notation, or an IPv6 address with colon-separated hexa-decimal notation.
- *pemdata* must be an alphanumeric string of 1 through 4095 (if private key is not implemented) or 1 through 8191 (if private key is implemented) characters.

Certificate and Private Key Storage

Certificates (configured using a URL) and private keys are stored as a file in a private directory locally. The output of the **show config** command displays the local URL of the certificate (only if the bootup configuration is URL) and private key instead of the data. When the certificate is removed using the **no certificate** *certificate_name* command, the certificate and private key from the local private directory are removed.

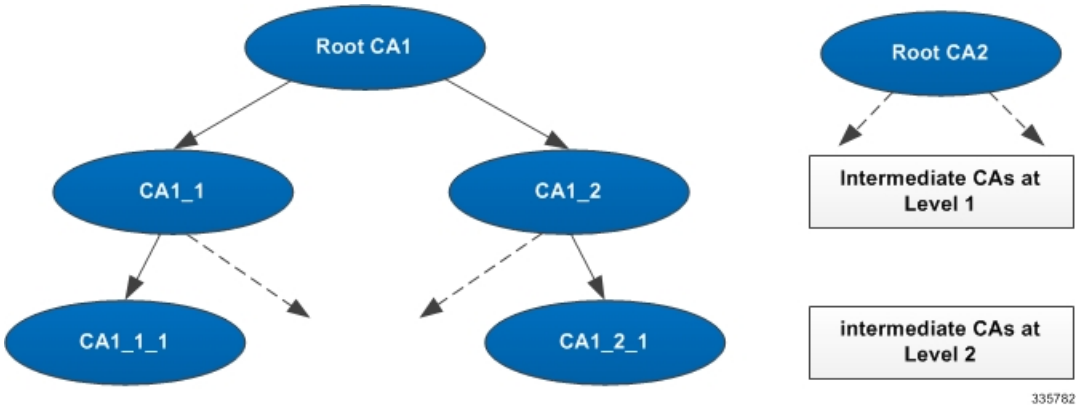
CA Certificate Chaining

Overview

Certificate chaining, also known as hierarchical CA cross certification, is a method by which an entity is authorized by walking a sequence of intermediate As up to the trust-point CA. An intermediate CA is a certification authority under a root CA, which is a self-signed authority.

The sequence of root and intermediate certificates belonging to CA is called a "chain". Each certificate in the chain is signed by the subsequent certificate. In this scheme, the web server certificate (the one that is to be installed on the web server where the user's site is hosted) is signed not by a root certificate directly but by one of the intermediates.

Figure 8: Root and Intermediate CAs in Certificate Chaining



The peer entities may obtain a certificate from any of the root CAs or intermediate CAs. A certificate may be authenticated by walking the chain up to a trust anchor, which may be either an intermediate CA or the root CA in the chain.

When an entity sends its certificate to the peer, it must also send all the certificates in the chain up to the trust anchor requested by the peer, not including the trust anchor certificate itself.

StarOS only supports X.509 Certificate encoding when sending certificates with a maximum certificate chain length of 4. The length of the certificate chain is defined as the number of all certificates in the chain, including the entity and intermediate CA certificates, but excluding the trust anchor certificate.

Deployment Scenarios

StarOS as Responder

Cert. Data in the Payload – Peer Cert. root CA1, StarOS Cert. Intm. CA1_1

1. StarOS sends IKE_SA_INIT to the peer.
2. StarOS sends IKE_SA_INIT to Peer. StarOS includes CERTREQ with Encoding = "X.509 Certificate - Signature" and Certification Authority = "Concatenated hashes of public key info of CA 1_1 and CA1 in any order".
3. Peer sends IKE_AUTH to StarOS. Peer includes CERT with requested encoding type, and an entity certificate issued by CA1. Peer includes CERTREQ with Encoding = "X.509 Certificate - Signature" and Certification Authority = "Hash of public key info of CA1". StarOS authenticates the peer certificate against CA1.
4. StarOS sends IKE_AUTH to Peer. StarOS includes two CERT payloads, with Encoding = "X.509 Certificate - Signature", and certificate data of (1) StarOS and (2) CA1_1.

Cert. Data in the Payload – Peer Cert. Intm CA1_1, StarOS Certificate root CA1

1. StarOS sends IKE_SA_INIT to the peer.
2. Peer sends IKE_SA_INIT to StarOS. This message includes CERTREQ with Encoding = "X.509 Certificate - Signature" and Certification Authority = "Hash of public key info of CA1_1 and CA1 in any order".

3. StarOS sends IKE_AUTH to peer. StarOS includes one CERT payload with requested encoding type, and the entity certificate issued by CA1. StarOS includes CERTREQ with Encoding = "X.509 Certificate - Signature" and Certification Authority = "Hash of public key info of CA1".
4. Peer sends IKE_AUTH to StarOS. Peer includes two CERT payloads, with Encoding = "X.509 Certificate - Signature", and (1) the entity certificate data, and (2) certificate data of CA1_1.

StarOS as Initiator

Cert. Data in the Payload – Peer Cert. root CA1, StarOS Cert. Intm. CA1_1

1. StarOS sends IKE_SA_INIT to the peer.
2. Peer sends IKE_SA_INIT to StarOS. This message includes CERTREQ with Encoding = "X.509 Certificate - Signature" and Certification Authority = "Hash of public key info of CA1".
3. StarOS sends IKE_AUTH to peer. StarOS includes two CERT payloads with requested encoding type, and (1) an entity certificate issued by CA1_1, and (2) a certificate of CA1_1. StarOS includes CERTREQ with Encoding = "X.509 Certificate - Signature" and Certification Authority = "Hash of public key info of CA1 and CA1_1 in any order".
4. Peer sends IKE_AUTH to StarOS. Peer includes one CERT payload, with Encoding = "X.509 Certificate - Signature", and the entity certificate data.

Cert. Data in the Payload – Peer Cert. Intm CA1_1, StarOS Certificate root CA1

1. StarOS sends IKE_SA_INIT to the peer.
2. Peer sends IKE_SA_INIT to StarOS. This message includes CERTREQ with Encoding = "X.509 Certificate - Signature" and Certification Authority = "Hash of public key info of CA1_1 and CA1 in any order".
3. StarOS sends IKE_AUTH to peer. StarOS includes one CERT payload with requested encoding type, and the entity certificate issued by CA1. StarOS includes CERTREQ with Encoding = "X.509 Certificate - Signature" and Certification Authority = "Hash of public key info of CA1".
4. Peer sends IKE_AUTH to StarOS. Peer includes two CERT payloads, with Encoding = "X.509 Certificate - Signature", and (1) the entity certificate data, and (2) certificate data of CA1_1.

External Interfaces

Support for "Hash & URL" of certificates/bundle requires HTTP or FTP interfaces to download the data which is implemented separately. OCSP verification of certificates also includes a TCP connection to the OCSP server during verification.

Certificate Management Protocol (CMPv2)

Overview

In cryptography, a public key certificate (also known as a digital certificate or identity certificate) is an electronic document which uses a digital signature to bind a public key with an identity information, such as

the name of a person or an organization, their address, and so forth. The certificate can be used to verify that a public key belongs to an individual. The Certificate Management Protocol (CMP) is an Internet protocol used for obtaining X.509 digital certificates in a public key infrastructure (PKI). It is described in RFC 4210.

StarOS implements the subset of CMPv2 functions:

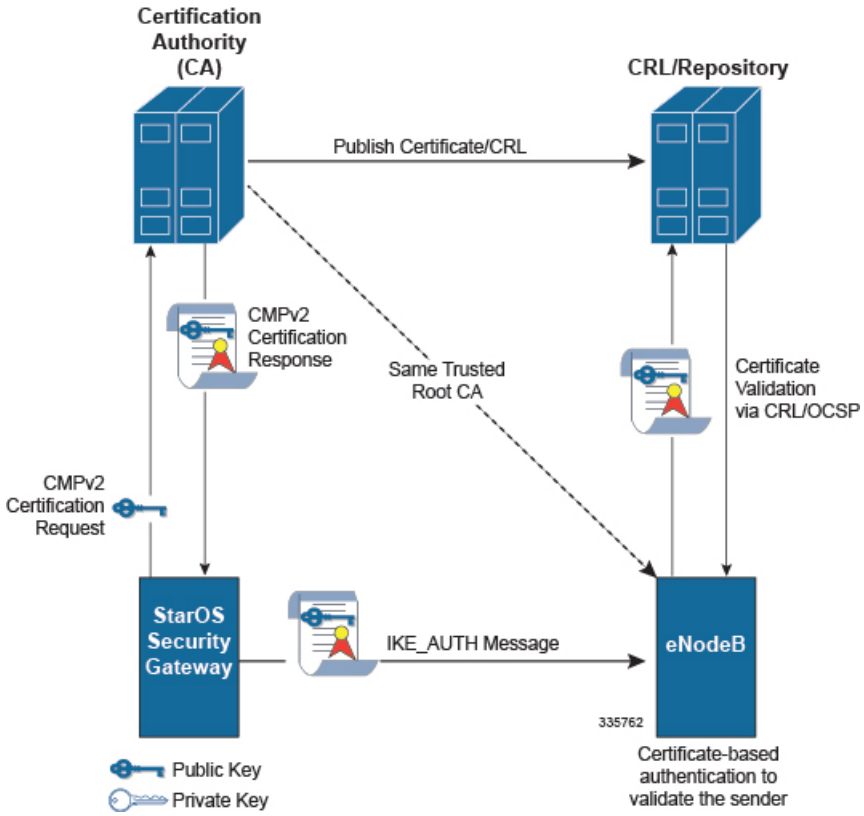
- **Key pair and X.509 certificate request generation**: The StarOS security gateway acts as an end entity as described in RFC 4210. The gateway generates the X.509 public and private key pair for authentication during IKE AUTH. It generates the public and private keys using OpenSSL libraries. The generated private key is saved locally on the management card, and the public key is embedded in the generated X.509 certificate request. The key uses RSA encryption; SHA-1 with RSA encryption is used on the Hashing function for the generated certificate. Certificate requests are sent to the Certificate Authority (CA) or Registration Authority (RA) during the certification process via CMPv2.
- **Initial certificate request transaction (ir and ip)**: A certificate request triggers the CMPv2 messaging to get the first certificate certified by the Certification Authority (CA). This CMPv2 transaction is identified by the Certification Request and Certification Response messages (ir and ip). At the end of this transaction the security gateway may receive the certificate signed by the CA in the response message. This certificate is then saved in the management card and is also propagated to the packet processing cards via internal messaging. The IKEv2 tunnel creation done at the packet processing cards requires this certificate for the IKE_AUTH transaction.
- **Certificate enrolment (cr and cp)**: This CMPv2 transaction obtains additional certificates certified by CA after the initial certification is done. The security gateway triggers additional certificate enrolment. The additional certificates are saved and used in a manner similar to the initial certificate.
- **Polling request and response (pollReq and pollRep)**: The ip, cp or the kup message received from the CA may contain a status code of "waiting". This indicates that the CA is still evaluating the certificate request and will require more time to sign the certificate. In this case the security gateway sends a pollReq message to the CA. The pollRep message from the CA may either contain the signed certificate or indicate a status of "waiting" again. If the pollRep message contains the certificate, it is treated as an ip/cp/kup message with a signed certificate and all relevant actions are taken. The security gateway also supports a CLI command to manually trigger polling for any request.
- **Certificate update transaction (kur and kup)**: This key pair update transaction re-certifies or updates a public/private key pair of the certificate after or before its validity expires. The Key Update Request (kur) message is sent to the CA with a certificate having a new public key, and the CA sends a Key Pair Update Response (kup) message with the signed certificate. The security gateway also supports two mechanisms to update an existing certificate:
  - **Manual Update**: The gateway sports a CLI command to trigger the certificate update transaction.
  - **Auto update**: The gateway can be configured to automatically trigger a certificate update a specified number of days before the certificate expires.
  - For both manual and automatic updates, the updated certificate is saved on the management card and propagated to the packet processing cards.

Deployment Scenarios

In a 4G network the data between the eNodeB and the MME/SGW is sent via a security gateway. The network between the security gateway and the MME/SGW is a trusted network of the vendor. The network between

the eNodeB and security gateway may be a public network requiring the establishment of an IPSec tunnel between eNodeB and the security gateway through which data is sent.

Figure 9: CMPv2 Deployment Scenario



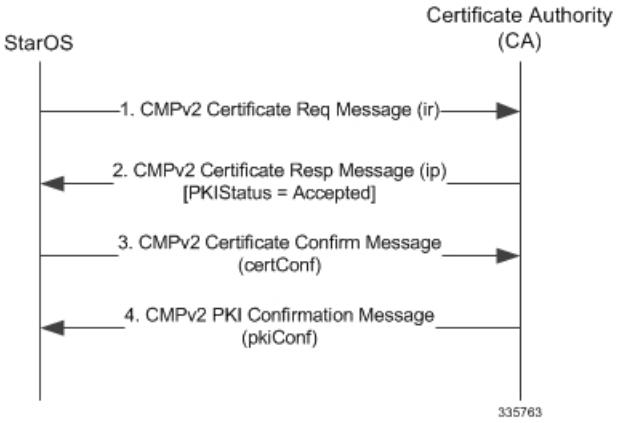
The IKEv2 protocol is used to establish the IPSec tunnel between eNodeB and the MME/SGW. Certificate-based authentication is performed during stage 2 of the IKEv2 exchange (RFC 4306). The security gateway sends its own X.509 certificate to the eNodeB in the IKE_AUTH message's CERT payload. This certificate is validated at the eNodeB and is used to decrypt the AUTH payload to authenticate the security gateway.

CMPv2 is the online mechanism for generating public and private keys and obtaining the certificate signed by a CA.

Call Flows

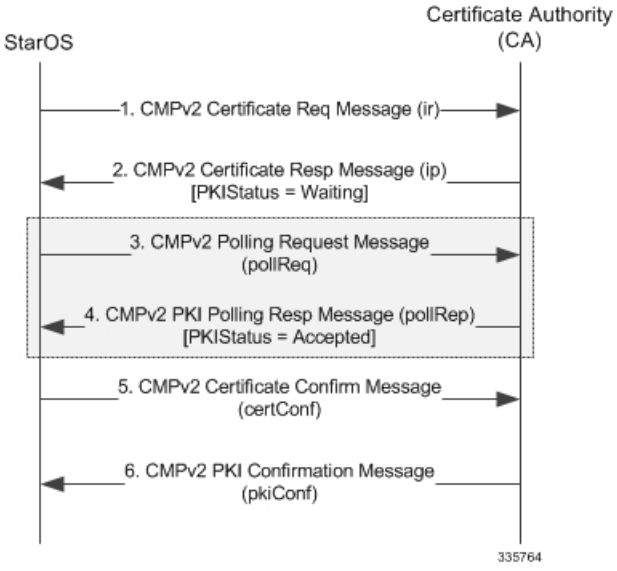
Initial Certification Request

Figure 10: Call Flow: Successful Initial Certification Request



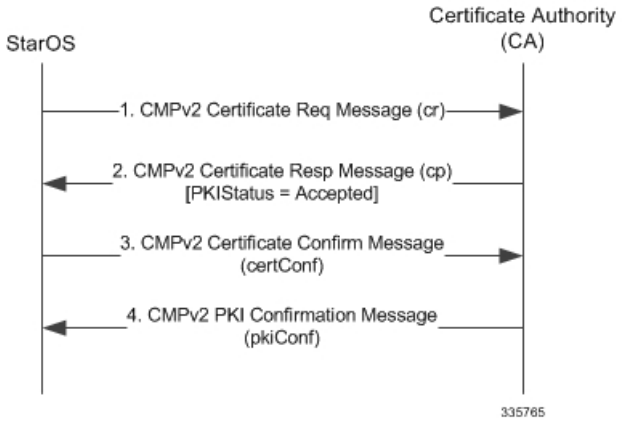
Initial Certification Request with Polling

Figure 11: Call Flow: Successful Initial Certification Request with Polling



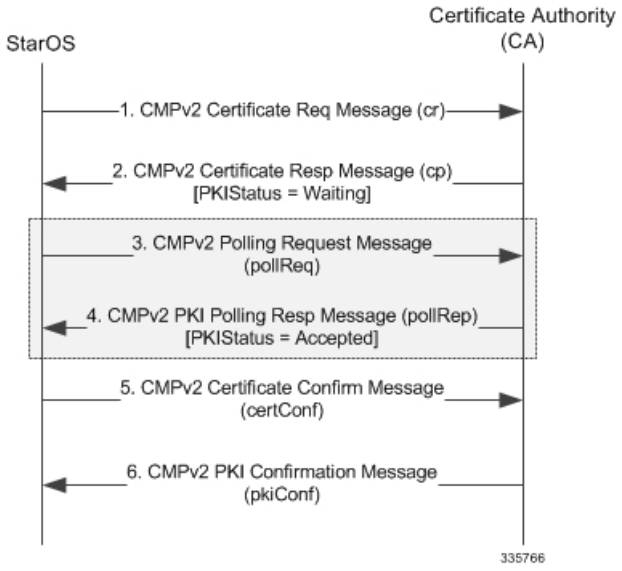
Enrollment Request

Figure 12: Call Flow: Successful Enrollment Request



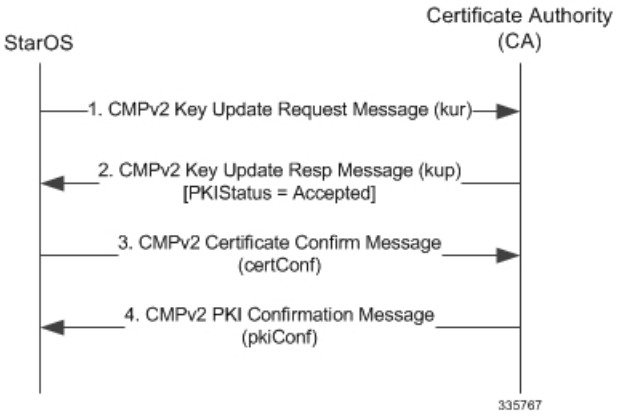
Enrollment Request with Polling

Figure 13: Call Flow: Successful Enrollment Request with Polling



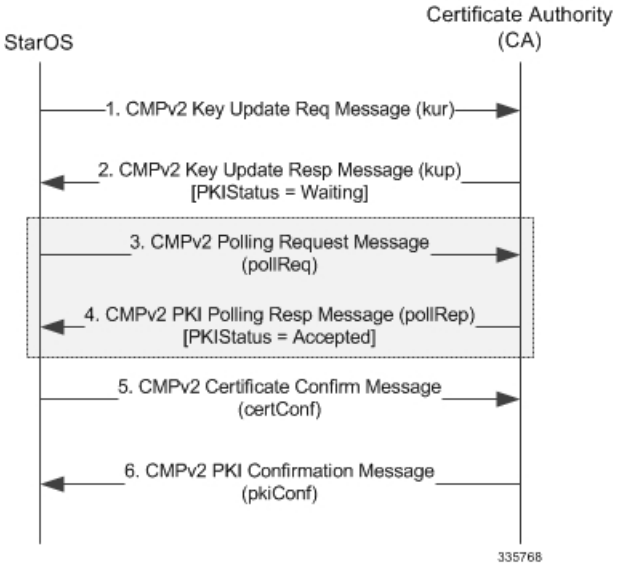
Certificate Update (Manual and Auto)

Figure 14: Call Flow: Successful Certificate Update



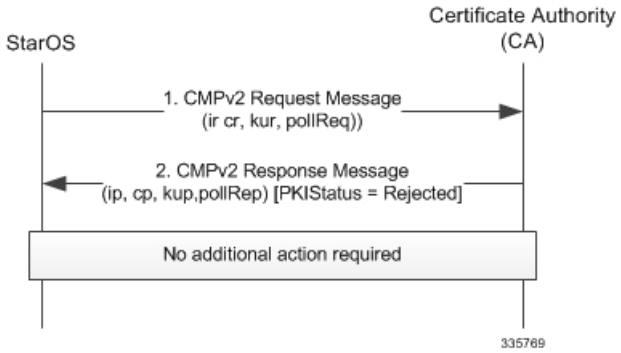
Certificate Update (Manual and Auto) with Polling

Figure 15: Call Flow: Successful Certificate Update with Polling



Failure Response Handling (ip/cp/kup/pollRep)

Figure 16: Call Flow: Failure Response Handling



CLI Commands

Important The commands described below appear in the CLI for this release. However, they have not been qualified for use with any current Cisco StarOS gateway products.

Exec Mode Commands

cmp initialize modulus

Triggers an Initial Certification Request (CR) after generating a public and private key pair, as well as an X.509 certificate to be included in the CR.

cmp intialize modulus *mod_type* **cert-name** *name* **subject-name** *"subject_string"* **ca-psk** *key* **ca-root** *ca_name* **ca-url** *url*

Refer to the *Command Line Interface Reference* for a complete description of this command and its keywords.

cmp enroll current-cert

Triggers a Certification Request (CR) after generating a public and private key pair, as well as an X.509 certificate to be included in the CR for a second certificate from the same Certificate Authority (CA).

cmp enroll current-cert *old-cert-name* **modulus** *mod_type* **subject-name** *"subject_string"* **cert-name** *name* **ca-root** *ca_name* **ca-url** *url*

Refer to the *Command Line Interface Reference* for a complete description of this command and its keywords.

cmp update current-cert

Triggers a Key Update Request after generating a public and private key pair, as well an X.509 certificate to be included in the Key Update Request for a certificate that is about to expire. This is a Certificate Management Protocol v2 command.

cmp update current-cert *old-cert-name* **modulus** *mod_type* **ca-root** *ca_name* **ca-url** *url*

Refer to the *Command Line Interface Reference* for a complete description of this command and its keywords.

cmp fetch current-cert -name

This command is only applicable for the ASR 9000 platform. CMPv2 operations are performed only on one Virtual Services Module (VSM) in the chassis. The certificates along with the private key file and the root certificate are stored on the supervisor card. When invoked on other VSMs in the chassis, this command reads the certificate, private key and the root certificate from the supervisor card.

cmp fetch current-cert *old-cert-name* **ca-root** *ca_name*

Refer to the *Command Line Interface Reference* for a complete description of this command and its keywords.

cmp poll cert-name

Triggers a pollReq for the specified certificate.

cmp poll current-cert *old-cert-name*

Global Configuration Mode Commands

cmp auto-fetch

Use this command to add a fetch configuration for each certificate for which automatic update is required. This is a Certificate Management Protocol v2 command.

cmp auto-fetch current-name *cert_name* **ca-root** *ca_name* **time** *days*

Refer to the *Command Line Interface Reference* for a complete description of this command and its keywords.

cmp cert-store location

Use this command to add a file location on /flash disk where the certificates and private keys will be stored. This is a Certificate Management Protocol v2 command.

cmp cert-store location *pathname* **[key reuse]**

Refer to the *Command Line Interface Reference* for a complete description of this command and its keywords.

cmp cert-trap time

Defines when an SNMP MIB certificate expiry trap should be sent as the number of hours before expiration.

cmp cert-trap time *hours*

Refer to the *Command Line Interface Reference* for a complete description of this command and its keywords.

show and clear Commands

show cmp outstanding-req

Displays details regarding outstanding Certificate Management Protocol v2 requests.

show cmp outstanding-req

Refer to the *Statistics and Counters Reference* for a description of the information output by this command.

show cmp statistics

Displays statistics related to Certificate Management Protocol v2 functions.

show cmp statistics

Refer to the *Statistics and Counters Reference* for a description of the information output by this command.

show cmp history

Displays historical information for the last 100 Certificate Management Protocol v2 transactions.

show cmp history

Refer to the *Statistics and Counters Reference* for a description of the information output by this command.

clear cmp cert-name

Clears information stored for the specified IPSec Certificate Management Protocol v2 (CMPv2) certificate.

clear cmp cert-name *cert_name*

Refer to the *Command Line Interface Reference* for a complete description of this command and its keywords.

clear cmp statistics

Clears statistics for IPSec Certificate Management Protocol v2 (CMPv2) certificates.

clear cmp statistics

Refer to the *Command Line Interface Reference* for a complete description of this command and its keywords.

Online Certificate Status Protocol (OCSP)

Overview

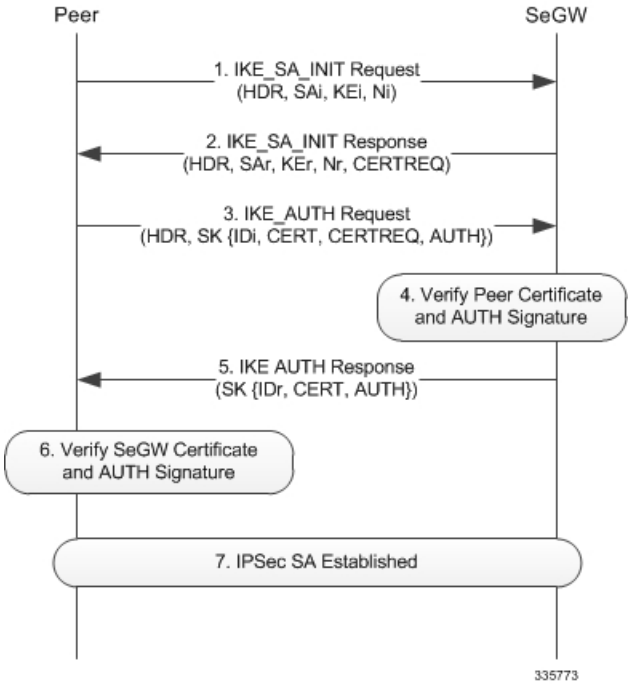
Certificates are used to establish peer identity. A certificate is issued by a trusted CA for a limited period. The validity period is an integral part of the signed certificate. Gateways exchanging certificates for establishing identity and trust check the certificate validity during the transaction. A certificate can be revoked at any instance of time (Well before the expiry of the certificate validity period). It is therefore very important to know the status of a certificate.

Online Certificate Status Protocol (OCSP) provides facility to obtain timely information on the status of a certificate (RFC 2560). OCSP messages are exchanged between a gateway and an OCSP responder during a certificate transaction. The responder immediately provides the current status of the presented certificate. The status can be good, revoked or unknown. The gateway can then proceed based on the response.

Deployment Scenarios

OCSP responders may be part of the CA/RA server or can be a separate entity authorized by the CA. The security gateway requires connectivity to this responder for status information.

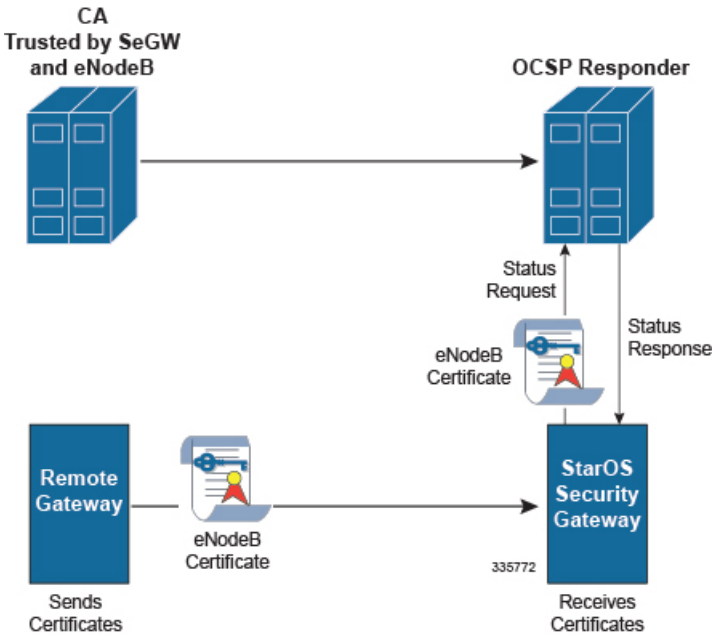
Figure 17: Call Flow: IKE Exchange



When the remote gateway presents a certificate, the security gateway forwards this certificate to the OCSP responder and queries it for the revocation status. The OCSP responder replies with the corresponding status information.

In IKE exchange (During the AUTH phase) the remote certificate is present in the CERT payload of the IKE message.

Figure 18: OCSP Status Request



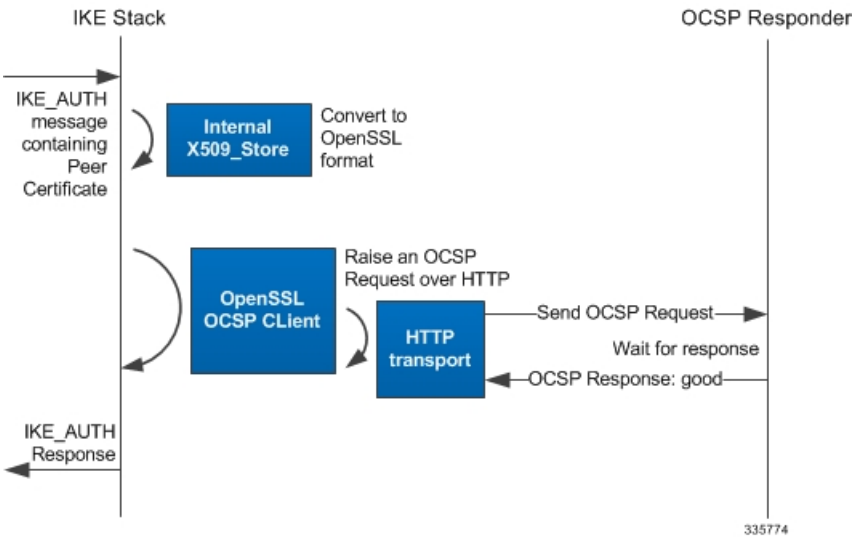
The security gateway passes this certificate along with its issuer certificate (trusted by security gateway) to the OCSP responder. IKE exchange is suspended (after step 3) until the response from the OCSP responder arrives. The OCSP request is initiated only when the presented certificate has the OCSP responder URL. If the URL is absent the OCSP request is not initiated.

If an OCSP response fails or if there is any error while contacting the responder, the certificate is validated with the CRL. Authentication is failed if an error is encountered while verifying with OCSP and or via a Certificate Revocation List (CRL).

Call Flows

Successful OCSP Response

Figure 19: Call Flow: Successful OCSP Response

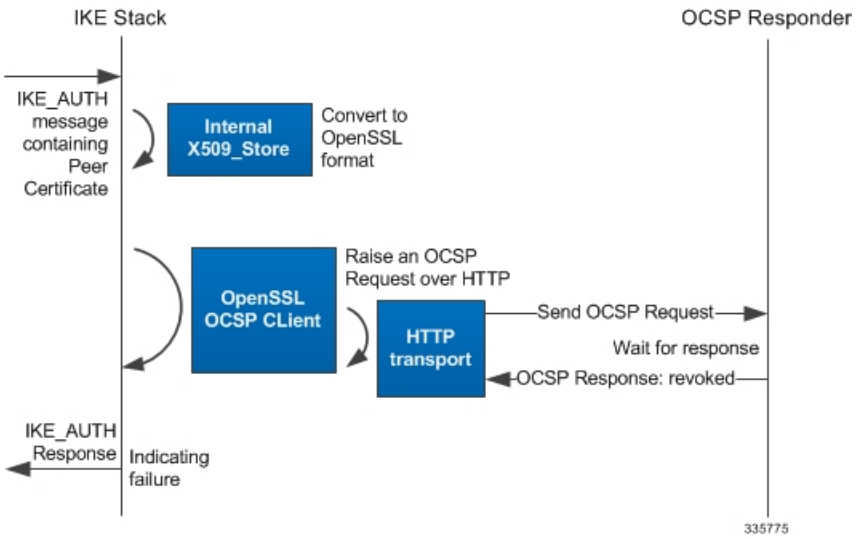


The peer certificate is obtained as CERT payload in the IKE message. The received certificate is converted to the OpenSSL format. This certificate is then passed to the OpenSSL OCSP client along with the X509_STORE to from an OCSP request. A connection to the OCSP responder is established and the request is sent.

On receipt of the response the IKE_AUTH transaction continues.

Revoked OCSP Response

Figure 20: Call Flow: Revoked OCSP Response



In this case fallback to CRL would be implemented for validating the user certificate. If this fails then the IKE_AUTH is aborted and a notification message is sent indicating authentication failure.

External Interface

The OCSP client to the OCSP responder interaction occurs over HTTP. A TCP socket connection is established to the OCSP responder. This connection is taken down once the OCSP response is received. The connection is also taken down as part of the cleanup after the setup timer expires.

CLI Commands

Important The commands described below appear in the CLI for this release. However, they have not been qualified for use with any current Cisco StarOS gateway products.

Context Configuration Mode

OCSP must be enabled in a crypto map or crypto template.

For a crypto map the configuration sequence is:

```
configure
  context ctxt_name
    crypto map template_name { ikev2-ipv4 | ikev2-ipv6 }
      ocsp [ nonce ]
```

For a crypto template the configuration sequence is:

```
configure
  context ctxt_name
    crypto template template_name ikev2-dynamic
      ocsp [ nonce ]
```

Refer to the *Command Line Interface Reference* for a complete description of these commands and their keywords.

CRL Fetching

Overview

CRLs (Certificate Revocation Lists) are issued periodically by the CA. This list contains the serial number of all the certificates that are revoked. An operator can verify the status of a certificate using a CRL. A CRL can be fetched via LDAPv3 from a CRL issuer (Trusted by CA).

When configured, this function also re-fetches the CRL once it expires in the cache. If the CRL is obtained from a CRL Distribution Point (CDP), StarOS defers the CRL fetch until the tunnel is established.

The CDP extension is read from the certificate for all protocols including HTTP, FTP, LDAPv3 and.CDP File.

StarOS initiates a CRL download in the following scenarios:

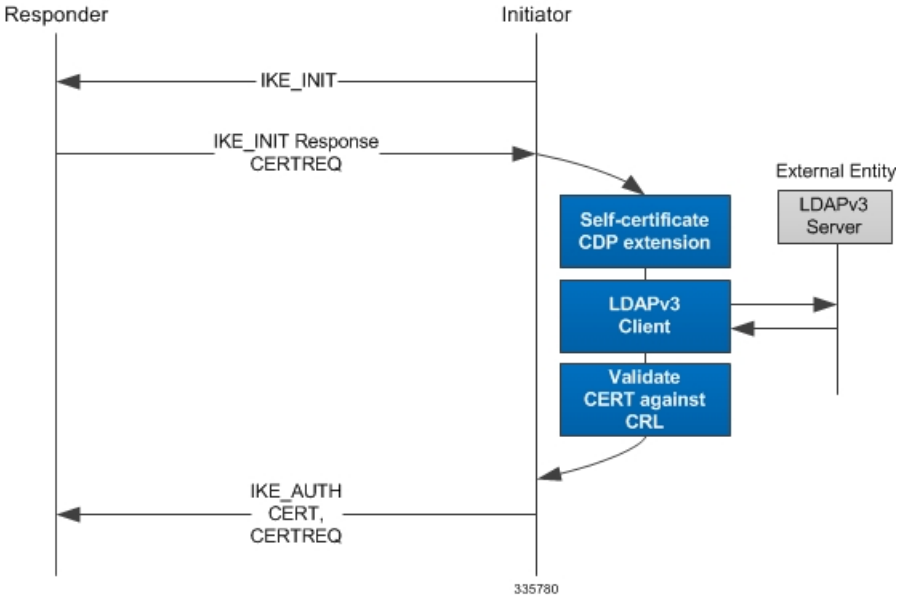
- User configuration via the CLI binds the CRL to a crypto map or template.
- During tunnel establishment:
  - The self-certificate CDP extension is used to download its latest CRL.
  - The CDP extension in the peer certificate is used to download its latest CRL.
- If the CRL (downloaded via CLI) expires during the refresh period (user configurable) a new fetch is triggered. If the CRL is obtained from the CDP extension, the fetch is deferred until tunnel establishment using the certificate.

CRL Downloads

Download from CDP Extension of Self-certificate

The following diagram illustrates the downloading of CRL from the self-certificate CDP extension (if present) at the tunnel creation.

Figure 21: Call Flow: CRL Download from CDP Extension of Self-certificate

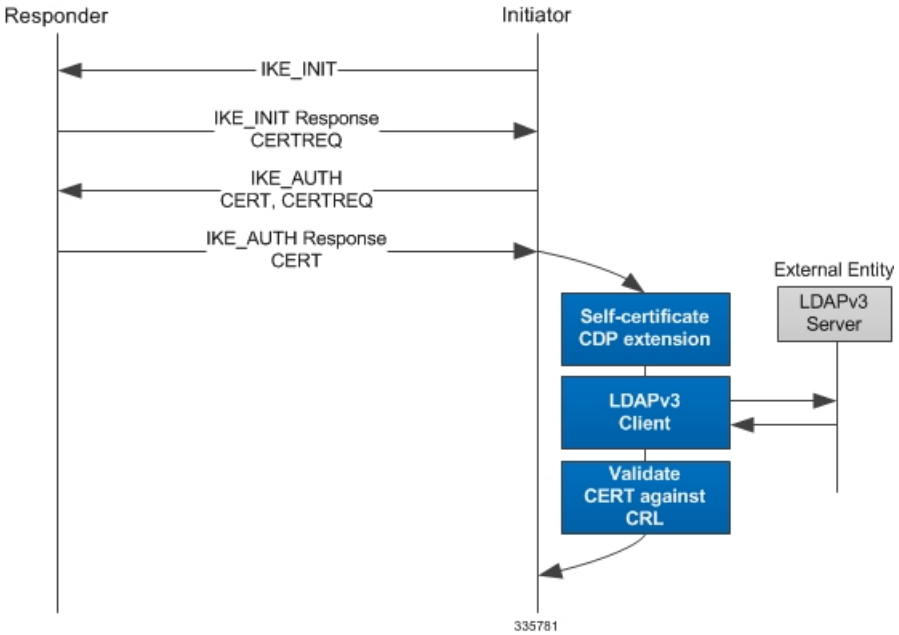


The certificate is then verified against the CRL before it is sent in the CERT payload of the IKE_AUTH message.

Download from CDP Extension of Peer Certificate

The following diagram illustrates peer certificate validations against CRLs. The CRL is fetched based on its CDP extension.

Figure 22: Call Flow: CRL Download from CDP Extension of Peer Certificate



The peer certificate is then verified against the CRL based on its status the IKE_AUTH proceeds.

CLI Commands

Important The commands described below appear in the CLI for this release. However, they have not been qualified for use with any current Cisco StarOS gateway products.

Global Configuration Mode

ca-crl name

This command configures the name and URL path of a Certificate Authority-Certificate Revocation List (CA-CRL).

The configuration sequence is as follows:

```
configure
   ca-crl name name { der | pem } { url url }
    end
```

url supports file pathname, TFTP, FTP, SFTP, HTTP and LDAP protocols.

Refer to the *Command Line Interface Reference* for a complete description of these commands and their keywords.

Context Configuration Mode

ca-crl list

This command is used to bind a CA-CRL to a crypto map or template.

For a crypto map the configuration sequence is:

```
configure
   context ctxt_name
      crypto map template_name { ikev2-ipv4 | ikev2-ipv6 }
         ca-crl list
            ca-crl-name
            end
```

For a crypto template the configuration sequence is:

```
configure
   context ctxt_name
      crypto template template_name ikev2-dynamic
         ca-crl list
            ca-crl name
            end
```

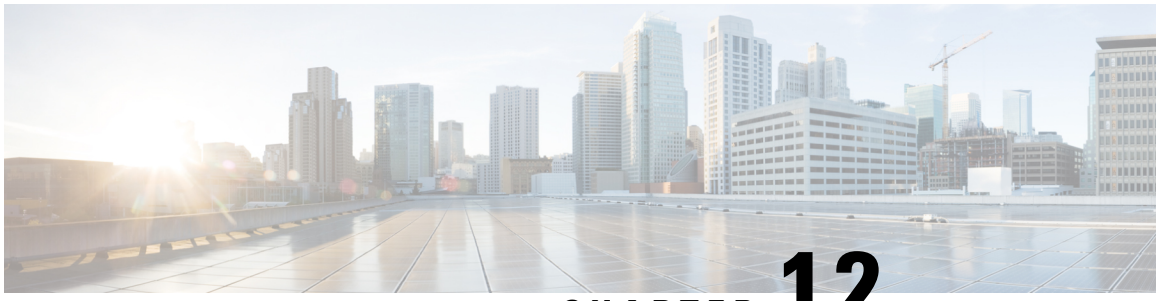
Refer to the *Command Line Interface Reference* for a complete description of these commands and their keywords.

show Commands

This command displays information for Certificate Authority (CA) Certificate Revocation Lists (CRLs) on this system.

show ca-crl { all | name *name* **}**

Refer to the *Statistics and Counters Reference* for a description of the information output by this command.



CHAPTER 12

Rekeying SAs

This chapter describes StarOS features for rekeying security Associations (SAs).

The following topics are discussed:

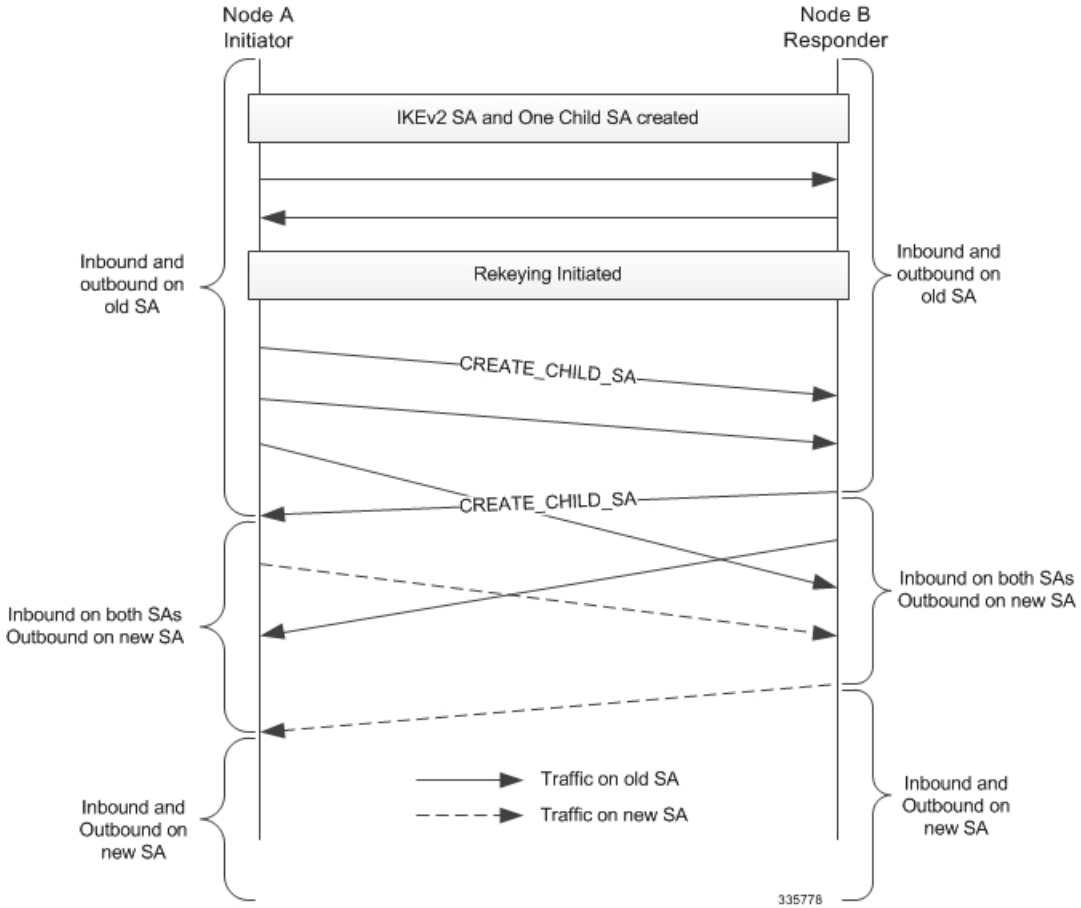
Rekey Traffic Overlap

Overview

An SA may be created with a finite lifetime, in terms of time or traffic volume. To assure interrupt-free traffic IKE SA and IPSec SAs have to be "rekeyed". By definition, rekeying is the creation of new SA to take the place of expiring SA well before the SA expires. RFC 5996 describes the procedure for IKEv2 rekeying with minimal traffic loss.

During the rekeying, both initiator and responder maintain both SAs for some duration during which they can receive (inbound) on both SAs. The inbound traffic on the old SA stops only after each node unambiguously knows that the peer is ready to start sending on the new SA (switch outbound to new SA). Switching the outbound traffic to new SA happens at the initiator and responder as depicted in following diagram.

Figure 23: Call Flow: Maintaining Old and New SAs during Child SA Rekeying



Note the following key points:

- Initiator is the first to switch outbound traffic to the new SA
- Switching outbound traffic on the responder is consequential
- Each node is ready to receive on both SAs for some duration.

If the traffic does not start flowing immediately on the new SAs, the nodes can use another mechanism to switch traffic to the new SA.

- To rekey a child SA (IPSec SA):
  - The node receives an explicit delete for the old child SA on IKE.
  - A predefined time elapses (neither of the above two events happen).

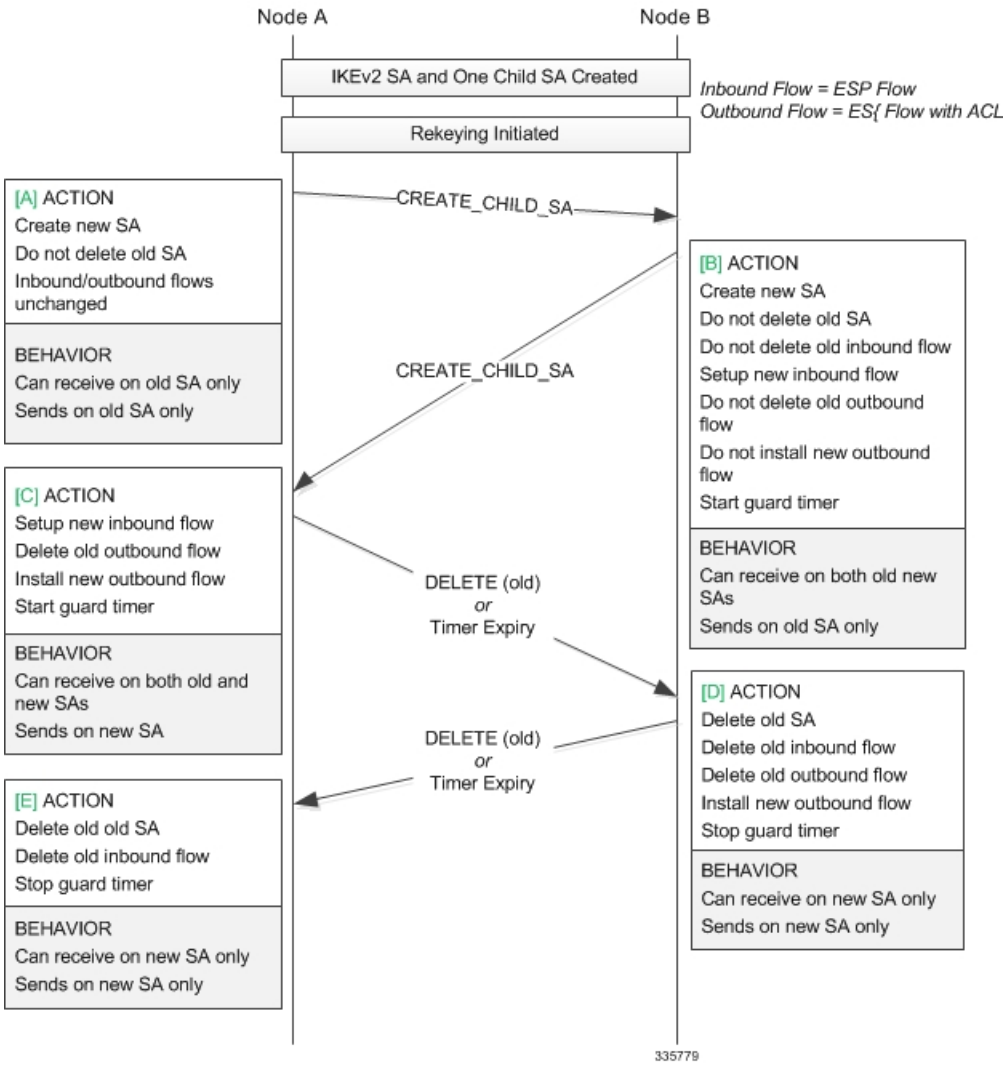
Deployment Scenarios

Network operators prefer using s finite-lifetime SA to minimize the risk of compromising the key when used indefinitely. Rekeying instead of deleting-creating an SA avoids breaks in traffic.

Initiator and Responder Rekeying Behavior

During rekeying, the old SA must not be deleted when the new SA is created. Traffic transmission on the new SA and deletion of the old child SA occurs as depicted in the following diagram.

Figure 24: Initiator and Responder Behavior During Rekeying



Notes:

1. If Node-A does not send DELETE at [C], guard timer expiry in Node-B replaces event [D]; guard timer expiry in Node-A replaces event [E}.
2. If Node-B does not send DELETE at [D], guard timer expiry in Node-A replaces event [E].
3. Guard timer expiry is fixed at 120 seconds.

Sequence Number-based Rekeying

Overview

IKE, ESP, and AH security associations use secret keys to encrypt the data traffic for a limited amount of time and for limited amount of data. This limits the lifetime of the entire security association.

If the life time of a security association expires, new security association needs to be established to replace the expired security association. This reestablishment of security associations to take the place of ones that expire is referred to as "rekeying".

The rekeying can be done for the IKE SA and also for the child (ESP or AH) SA. This feature triggers rekeying only for the Child SA.

This feature supports sequence number based rekeying where the lifetime for the child SA is processed in terms of sequence number of the child SA data flow.

Sequence number-based rekeying is applicable only for the 32-bit based sequence number, so as to protect against the wrapping of sequence number before it reach its maximum limit of 4,293,918,720. The soft limit threshold for sequence number-based rekey trigger is fixed to 90% of the maximum sequence number limit.

Important This feature is not applicable on the configuration that supports Extended Sequence Number (ESN).

This feature can be activated only when the anti-replay functionality is enabled in the configuration. In StarOS the anti-replay is enabled by default.

Deployment Scenarios

This feature can be used to rekey a child SA when the sequence number of the packet passed through the SA exceeds the predefined sequence number threshold.

CLI Commands

Sequence number-based rekeying is enabled when the Context Configuration Mode **ipsec replay** command is enabled along with crypto map and crypto template rekeying configurations.

ipsec rekey

This Context Configuration Mode command configures IKEv2 IPSec specific anti-replay.

```
configure
  context ctxt_name
    ipsec replay [ window-size window_size ]
    end
```

Crypto Map and Crypto Template Rekey Configurations

There are a number of Context Configuration mode commands with rekey keywords.

For crypto maps refer to the following commands:

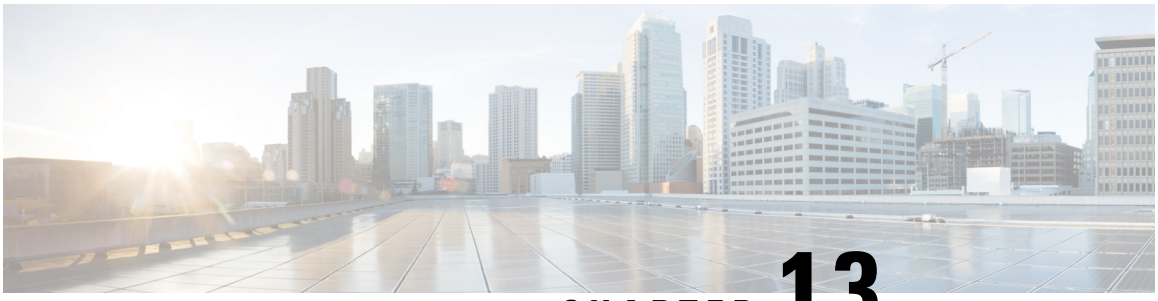
- **crypto map** *map_name* **ikev2-ikesa replay**
- **crypto map** *map_name* **ikev2-ipv4 rekey**
- **crypto map** *map_name* **ikev2-ipv6 rekey**

For crypto template refer to the following commands:

- **crypto template** *template_name* **ikev2-dynamic payload rekey**
- **crypto template** *template_name* **ikev2-ikesa rekey**

show crypto ipsec security-associations

This Exec mode **show** command displays the childSA lifetime based on sequence number.



CHAPTER 13

Access Control

This chapter describes enhancements to IPSec Access Control.

The following topics are discussed:

Access Control via Blacklist or Whitelist

Important The commands described in this section appear in the CLI for this release. However, they have not been qualified for use with any current Cisco StarOS gateway products.

Overview

A blacklist or block list is a basic access control mechanism that allows everyone access, except for the members of the black list. The opposite is a whitelist, which denies access to everybody except for members of the white list.

A blacklist is a list or register of entities that, for one reason or another, are being denied a particular privilege, service, mobility, access or recognition.

A whitelist is a list or register of entities that, for one reason or another, are being provided a particular privilege, service, mobility, access or recognition.

With blacklisting, any peer is allowed to connect as long as it does not appear in the list. With whitelisting, no peer is allowed to connect unless it appears in the list. An operator may choose to implement one or the other. You can implement either a blacklist or whitelist; both listing techniques cannot be implemented simultaneously on a security gateway.

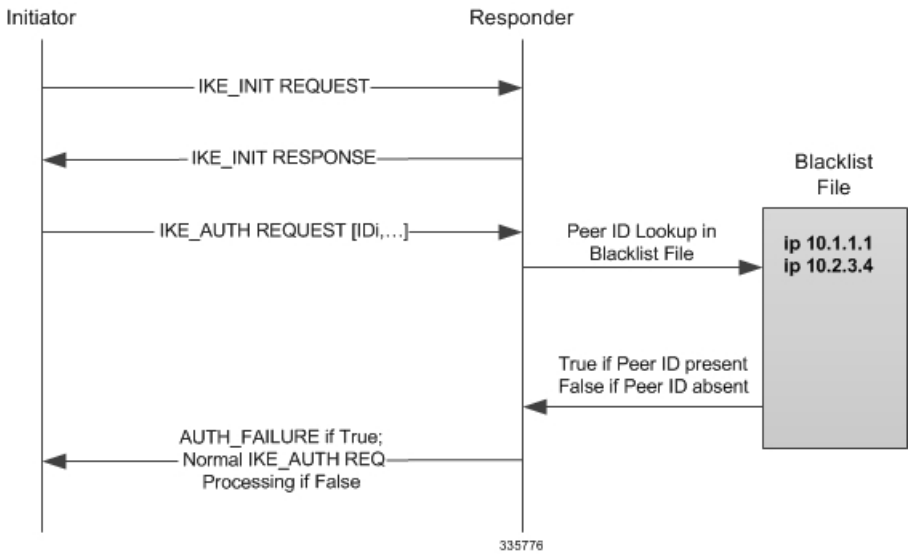
Blacklisting

The sequence of events when implementing blacklisting is briefly described below:

- The initiator sends IKE_INIT_REQUEST to the responder.
- The responder replies with IKE_INIT_RESPONSE.

- Once the IKE_INIT_RESPONSE is done, the Initiator sends IKE_AUTH_REQUEST to the responder along with its ID.
- Upon receipt of the IKE_AUTH_REQUEST, the responder checks for the presence of a matching peer ID in the blacklist.
- If the peer ID is present in the blacklist, the responder sends an IKE_AUTH_FAILURE to the initiator. Otherwise, the processing of IKE_AUTH_REQUEST follows the normal procedure for tunnel setup.

Figure 25: Blacklisting Implementation

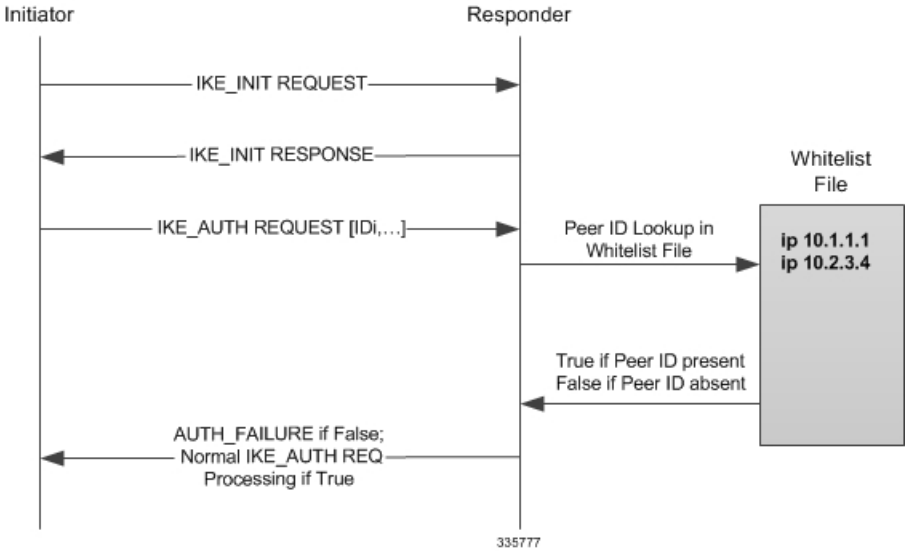


Whitelisting

The sequence of events when implementing whitelisting is briefly described below:

- The initiator sends IKE_INIT_REQUEST to the responder.
- The responder replies with IKE_INIT_RESPONSE.
- Once the IKE_INIT_RESPONSE is done, the Initiator sends IKE_AUTH_REQUEST to the responder along with its ID.
- Upon receipt of the IKE_AUTH_REQUEST, the responder checks for the presence of a matching peer ID in the whitelist.
- If the peer ID is present in the whitelist, the IKE_AUTH_REQUEST is processed normally. Otherwise, the gateway sends an IKE_AUTH_FAILURE to the initiator.

Figure 26: Whitelist Implementation



Blacklist and Whitelist File Format

File Format and Content

The blacklist/whitelist file can be in DOS or Unix format. DOS files will be internally converted to Unix format before being read.

The file contents should follow the standard format described below. Each entry in the blacklist/whitelist file should contain the ID type so that the validation is performed for that ID type. The ID type and ID value in each entry should be separated by a **space**.

Important No other file types or formats are supported.

The sample file content is shown below.

```
# IP address IDS
ipv4 "33.33.33.1"
ipv4 "66.66.66.1"
ipv6 "11::1"
# FQDN IDs
fqdn "LS1-0.cisco.com"

# Email ID
email "user@sample.com"

# Distinguished Name ID
dn "C=US,ST=CA,L=SanJose,O=Cisco,OU=SMBU,CN=ixia.organization.bu.org"
```

Supported IKE ID Types

The following IKE ID types are supported in a blacklist or whitelist:

- ID_IPV4_ADDR (IPv4 address in dotted-decimal notation)
- ID_FQDN (Fully Qualified Domain Name
- ID_RFC822_ADDR (Email address)
- ID_IPV6_ADDR (IPv6 address in colon-separated notation)
- ID_DER_ASN1_DN (Abstract Syntax Notation One – Distinguished Name)
- ID_DER_ASN1_GN (Abstract Syntax Notation One – General Name)
- ID_KEY_ID (Opaque byte stream)

Deployment Scenarios

Blacklisting

Blacklisting can be used when requests from a particular identity must be blocked for a short period of time, such as if the subscriber has not paid his/her bill.

Whitelisting

Whitelisting can be used when requests from particular identities must be allowed to set up tunnels for a short period of time, such as when certain services are allowed only for subscribers who have subscribed for the service.

External Interfaces

The blacklist/whitelist file will be read from locations accessible by StarOS. Locations and protocols include:

- [file:]{/flash | /pcmcia1 | /hd-raid}[/directory]/filename
- [file:]{/flash | /usb1 | /hd-raid}[/directory]/filename
- tftp://host[:port][/directory]/filename
- ftp://[username[:password]@]host[:port][/directory]/filename
- sftp://[username[:password]@]host[:port][/directory]/filename

Important A black list or whitelist must be available to StarOS or blacklisting/whitelisting will not be performed even if enabled.

CLI Commands

Important The commands described below appear in the CLI for this release. However, they have not been qualified for use with any current Cisco StarOS gateway products.

Global Configuration Mode

crypto blacklist file

Configures a blacklist (access denied) file to be used by a security gateway (SeGW).

crypto blacklist file *pathname*

pathname specifies the location and protocol from which StarOS will retrieve the blacklist file.

Refer to the *Command Line Interface Reference* for a complete description of this command and its keywords.

crypto whitelist file

Configures a whitelist (access permitted) file to be used by a security gateway (SeGW).

crypto whitelist file *pathname*

pathname specifies the location and protocol from which StarOS will retrieve the whitelist file.

Refer to the *Command Line Interface Reference* for a complete description of this command and its keywords.

Context Configuration Mode

Enable blacklist

The blacklist must be enabled in a crypto map or crypto template.

For a crypto map the configuration sequence is:

```
configure
  context ctxt_name
    crypto map template_name { ikev2-ipv4 | ikev2-ipv6 }
      blacklist
```

For a crypto template the configuration sequence is:

```
configure
  context ctxt_name
    crypto template template_name ikev2-dynamic
      blacklist
```

Refer to the *Command Line Interface Reference* for a complete description of these commands and their keywords.

Enable whitelist

A whitelist must be enabled in a crypto map or crypto template.

For a crypto map the configuration sequence is:

```
configure
  context ctxt_name
    crypto map template_name { ikev2-ipv4 | ikev2-ipv6 }
      whitelist
```

For a crypto template the configuration sequence is:

```
configure
  context ctxt_name
```

```
crypto template template_name ikev2-dynamic
  whitelist
```

Refer to the *Command Line Interface Reference* for a complete description of these commands and their keywords.

Exec Mode

crypto blacklist file update

Updates the blacklist (access denied) file using the path specified when the blacklist was enabled.

```
crypto blacklist file update
```

Refer to the *Command Line Interface Reference* for a complete description of these commands and their keywords. For additional information on blacklisting, refer to the *System Administration Guide*

crypto whitelist file update

Updates the whitelist (access granted) file using the path specified when the whitelist was enabled.

```
crypto whitelist file update
```

Refer to the *Command Line Interface Reference* for a complete description of these commands and their keywords. For additional information on blacklisting, refer to the *System Administration Guide*

show Commands

show crypto blacklist file

Displays the contents of the blacklist (access denied) file.

```
show crypto blacklist file
```

Refer to the *Statistics and Counters Reference* for a description of the information output by this command.

show crypto whitelist file

Displays the contents of the whitelist (access granted) file.

```
show crypto blacklist file
```

Refer to the *Statistics and Counters Reference* for a description of the information output by this command.

show crypto statistics ikev2

The output of this command displays statistics for blacklist or whitelist activities, including Child SA exchanges and SA rekeys.

show crypto template

The output of this command indicates whether blacklisting or whitelisting has been enabled.

IKE Call Admission Control

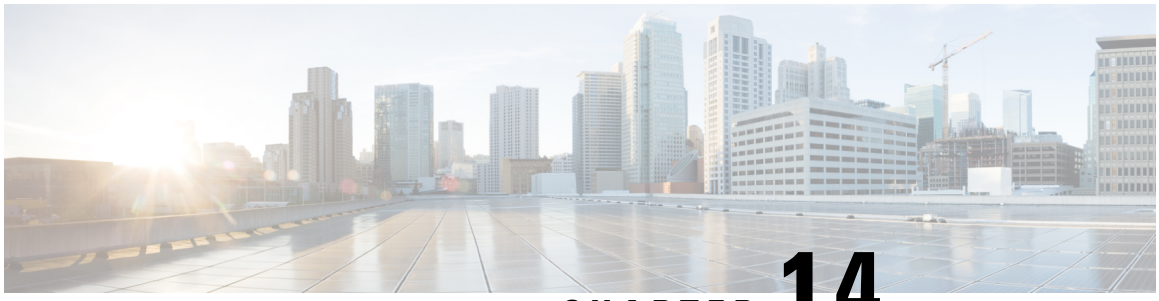
Call Admission Control (CAC) rate limits new IKE calls whenever a security gateway (SeGW) is experiencing an overload. If the SeGW receives more IKE_SA_INIT requests than it can handle, already established tunnels

could be affected as system resources, such as CPU, Message Queue etc., would be utilized to handle the new calls. The SeGW may be unable to process the Dead Peer Detections (DPDs) of existing tunnels on time, leading to their tear-off. Rate limiting preserves enough system resources to maintain existing calls.

In StarOS, this functionality is achieved through **congestion-control threshold** Global Configuration mode CLI commands. These commands monitor a variety of parameters that indicate whether the system has gone into overload. Parameters that can be monitored for congestion include (but are not limited to):

- **congestion-control threshold license-utilization** percent – percentage of maximum number of licensed sessions
- **congestion-control threshold max-sessions-per-service-utilization** percent – percentage of maximum subscriber sessions (**congestion-control threshold per-service-**service percent command)
- **congestion-control threshold message-queue-utilization** percent – percentage of message queue utilization (**congestion-control threshold message-queue-utilization** percent command)
- **congestion-control threshold message-queue-wait-time** time – wait time in seconds
- **congestion-control threshold port-rx-utilization** percent – average percentage of receive port utilization
- **congestion-control threshold port-specific {** slot/port **|** **all }** – percentage of utilization for a specific port
- **congestion-control threshold port-specific-rx-utilization** percent – percentage of receive utilization for a specific port
- **congestion-control threshold port-specific-tx-utilization** percent – transmit utilization for a specific port
- **congestion-control threshold port-tx-utilization** percent – average percentage of port transmit utilization
- **congestion-control threshold service-control-cpu-utilization** percent – average percentage of CPU utilization for service control
- **congestion-control threshold system-cpu-utilization** percent – average percentage of system CPU utilization
- **congestion-control threshold system-memory-utilization** percent – average percentage of CPU memory utilization

Refer to the *Command Line Interface Reference* for a complete description of these commands and their keywords.



CHAPTER 14

Remote Secrets

This chapter describes how StarOS supports the use of remote secrets.

Important The commands described in this chapter appear in the CLI for this release. However, they have not been qualified for use with any current Cisco StarOS gateway products.

The following topics are discussed:

PSK Support for Remote Secrets

Overview

StarOS CLI commands support the creation of local and remote pre-shared keys (PSKs) associated with crypto maps and crypto templates. Refer to the descriptions of the **crypto map** and **crypto template** commands in the *Context Configuration Mode Commands* chapter of the *Command Line Interface Reference*.

StarOS also allows the operator to configure a remote secret list that contains PSKs based on remote ID types. The remote secret list can contain up to 1000 entries; only one remote secret list is supported per system. The remote secret list bound to a crypto map and/or crypto template.

Each entry in the remote secret list consists of either an alphanumerical string of 1 through 255 characters, or a hexadecimal string of 16 to 444 bytes.

Implementation

The general sequence for implementing the use of a remote PSK is as follows:

- The initiator sends an IKE_INIT_REQUEST to the responder.
- The responder replies with an IKE_INIT_RESPONSE.
- When the IKE_INIT_RESPONSE is received, the Initiator sends an IKE_AUTH_REQUEST to the responder along with its peer ID.

- When the responder receives the IKE_AUTH_REQUEST, it derives the peer ID from the IKE_AUTH_REQUEST to search the remote secret list for the PSK. If the remote secret list is bound to the respective map/template, it takes the PSK from the list. Otherwise, it will take the remote PSK from the respective map or template.

Supported IKE ID Types

The following IKE ID types are support supported in a remote secret list entry:

- ID_IP_ADDR (supports IPv4 and IPv6 address notations)
- ID_IPV4_ADDR (IPv4 address in dotted-decimal notation)
- ID_FQDN (Fully Qualified Domain Name
- ID_RFC822_ADDR (Email address)
- ID_IPV6_ADDR (IPv6 address in colon-separated notation)
- ID_DER_ASN1_DN (Abstract Syntax Notation One – Distinguished Name)
- ID_DER_ASN1_GN (Abstract Syntax Notation One – General Name)
- ID_KEY_ID (Opaque byte stream)

Deployment Scenarios

A group of remote clients can be configured to use a separate pre-shared key, even if they are using the same crypto map or crypto template.

CLI Commands

Important The commands described below appear in the CLI for this release. However, they have not been qualified for use with any current Cisco StarOS gateway products.

Global Configuration Mode

crypto remote-secret-list

Specifies the name of the remote secret list for storing remote secrets based on the ID type. This command sends you to the Remote Secret List Configuration mode and the **remote-id-id-type** command. Only one active remote-secret-list is supported per system.

crypto remote-secret-list*listname*

Important You must unbind the remote-secret-list from any crypto maps or templates before it can be deleted.

For additional information, refer to the *Remote Secret List Configuration Commands* chapter of the *Command Line Interface Reference* and the *System Administration Guide*.

remote-id id-type

Configures the remote pre-shared key based on the ID type.

```
remote-id id-type { der-asn1-dn | fqdn | ip-addr | key-id | rfc822-addr
} id id_value secret [ encrypted ]  key key_value
```

Context Configuration Commands

Enable remote secret list

The remote secret list must be enabled in a crypto map or crypto template.

For a crypto map the configuration sequence is:

```
configure
  context ctxt_name
    crypto map template_name { ikev2-ipv4 | ikev2-ipv6 }
      remote-secret-list
```

For a crypto template the configuration sequence is:

```
configure
  context ctxt_name
    crypto template template_name ikev2-dynamic
      remote-secret-list
```

Refer to the *Command Line Interface Reference* for a complete description of these commands and their keywords.

show Commands

show configuration

Configured remote secrets are displayed in the output of the **show configuration** command

show crypto map

Configured remote secrets are also displayed in the following **show crypto map** commands:

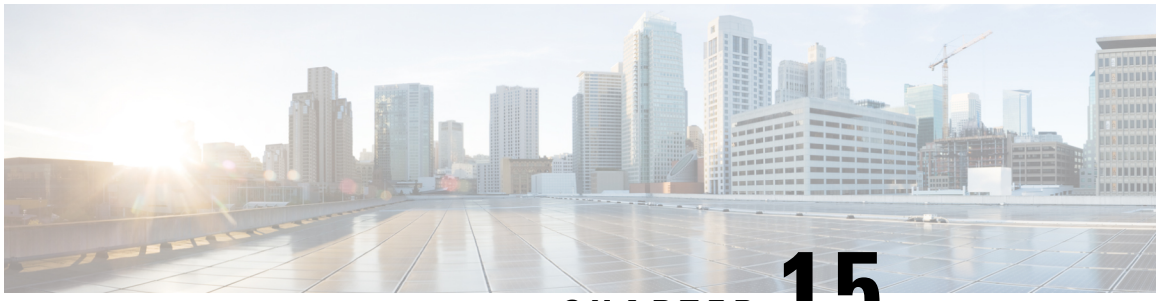
- **show crypto map**
- **show crypto map map-type ikev2-ipv4-cfg**
- **show crypto map map-type ikev2-ipv6-cfg**
- **show crypto map tag** *map-name*

show crypto template

Configured remote secrets are also displayed in the following **show crypto template** commands:

- **show crypto template**
- **show crypto template map-type ikev2-dynamic**
- **show crypto template tag** *map-name*

Refer to the *Command Line Interface Reference* for a complete description of these commands and their keywords.



CHAPTER 15

IKEv2 RFC 5996 Compliance

This chapter describes how StarOS complies with RFC 5996 – Internet Key Exchange Protocol Version 2 (IKEv2).

The following topics are discussed:

RFC 5996 Compliance

Overview

StarOS currently complies with RFC 4306 – *Internet Key Exchange (IKEv2) Protocol*. StarOS IKEv2 has been enhanced to comply with RFC 5996 – *Internet Key Exchange Protocol Version 2 (IKEv2)*.

RFC 5996 introduces two new notification payloads using which certain conditions of the sender can be notified to the receiver. The IANA assigned numbers for these payloads are as follows:

- TEMPORARY_FAILURE – IANA Assigned Number = 43
- CHILD_SA_NOT_FOUND – IANA Assigned Number = 44

StarOS sends the above payloads only in collision scenarios as mentioned in RFC 5996 Section 2.25.

TEMPORARY_FAILURE

A TEMPORARY_FAILURE notification should be sent when a peer receives a request that cannot be completed due to a temporary condition. When StarOS receives this notification type, it waits (50% of the remaining time of the IKESA/Child SA) and then retries a maximum of eight times until the hard lifetimer expires. A retry is initiated only if 50% of the remaining time is greater than or equal to two minutes. If it continues to receive TEMPORARY_FAILURE for all the retries initiated, no further retry is done and the IKESA/Child SA is deleted after its hard lifetime expiry.

When TEMPORARY_FAILURE is received, retry is done only for an exchange corresponding to REKEYS. If temporary failure is received for a non-rekey exchange, the temporary failure is considered as failed for the exchange.

CHILD_SA_NOT_FOUND

A CHILD_SA_NOT_FOUND notification should be sent when a peer receives a request to rekey a Child SA that does not exist. If StarOS receives this notification, it silently deletes the Child SA.

On receipt of CHILD_SA_NOT_FOUND, the CHILDSA for which REKEY was initiated is terminated. If the CHILDSA is the only CHILDSA under the IKESA, the IKESA is terminated and a DELETE request is sent to the peer for the same.

Exchange Collisions

In IKEv2 exchange collisions may happen when both peers start an exchange for an IKE SA at the same time. For example UE starts CHILDSA REKEY using CREATE_CHILD_SA and a security gateway also starts CHILDSA REKEY when SA soft lifetime has expired in both at the same time.

RFC 5996 defines a framework to resolve this collision so that only one of the exchanges succeeds. The collision handling mechanism supported in StarOS complies with the mechanism defined in RFC 5996.

Integrity with Combined Mode Ciphers

RFC 5996 makes changes in specifications to allow negotiation of combined mode ciphers. Combined mode ciphers are algorithms that support integrity and encryption in a single encryption algorithm. RFC 5996 makes negotiation for the integrity algorithm optional if combined mode cipher is used. In RFC 4306 the integrity algorithm was mandatory in the SA payload.

StarOS does not support the combined mode cipher. Staros IKEv2 has been enhanced to identify a currently defined combined cipher. If a proposal for combined mode cipher is received, StarOS responds with NO_PROPOSAL_CHOSEN if no other proposal matches.

Negotiation Parameters in CHILDSA REKEY

On rekeying of a CHILD SA the traffic selectors and algorithms match the ones negotiated during the set up of the child SA. StarOS IKEv2 does not send any new parameters in CREATE_CHILD_SA for a child SA being rekeyed.

Certificates

StarOS supports a CLI command to enable sending and receiving HTTP method for hash-and-URL lookup with CERT/CERTREQ payloads.

If configured and if a peer requests CERT using encoding type as "Hash and URL of X.509 certificate" and send HTTP_CERT_LOOKUP_SUPPORTED using notify payload in the first IKE_AUTH, StarOS sends the URL in the CERT payload instead of sending the entire certificate in the payload.

If not configured and CERTREQ is received with encoding type as "Hash and URL for X.509 certificate", StarOS responds with entire certificate as it in release 14.1, even if peer had sent HTTP_CERT_LOOKUP_SUPPORTED.

If configured for Hash and URL while sending the CERTREQ request, StarOS sends the request with encoding type as "Hash and URL of X.509 certificate" and sends notify payload HTTP_CERT_LOOKUP_SUPPORTED. However, also sends another CERTREQ with encoding type as X.509 certificate (as in release 14.1) and accepts the entire certificate coming in the CERT payload. If CERT payload is received with encoding type as hash and URL, StarOS fetches the certificate using the URL.

Multiple Traffic Selectors

During traffic selector negotiation, the gateway should be able to narrow down the UE's request for a range of traffic selectors in accordance with RFC 5996.

CLI Commands

Important The commands and new keywords described below appear in the CLI for this release. However, they have not been qualified for use with any current Cisco StarOS gateway products.

Context Configuration Mode

Enable Notification Payloads

To enable the sending and receiving of TEMPORARY_FAILURE and CHILD_SA_NOT_FOUND notifications, use one of the following configuration sequences.

For a crypto map the configuration sequence is:

```
configure
  context ctxt_name
    crypto map template_name { ikev2-ipv4 | ikev2-ipv6 }
      ikev2-ikesa
        policy use-rfc5996-notification
```

For a crypto template the configuration sequence is:

```
configure
  context ctxt_name
    crypto template template_name ikev2-dynamic
      ikev2-ikesa
        policy use-rfc5996-notification
```

Refer to the *Command Line Interface Reference* for a complete description of these commands and their keywords.

Add Hash and URL Encoding to Certificates

Use the following configuration to add Hash and URl encoding of certificates.

For a crypto map the configuration sequence is:

```
configure
  context ctxt_name
    crypto map template_name { ikev2-ipv4 | ikev2-ipv6 }
      certificate name
        pem url url
        cert-enc cert-hash-url url url url
```

For a crypto template the configuration sequence is:

```
configure
  context ctxt_name
    crypto template template_name ikev2-dynamic
         pem url url
         cert-enc cert-hash-url url url url
```

Refer to the *Command Line Interface Reference* for a complete description of these commands and their keywords.

Enable Hash and URL Certificate Encoding

Hash and URL encoding must be enabled in a crypto map or crypto template.

For a crypto map the configuration sequence is:

```
configure
  context ctxt_name
    crypto map template_name { ikev2-ipv4 | ikev2-ipv6 }
      allow-cert-enc cert-hash-url
```

For a crypto template the configuration sequence is:

```
configure
  context ctxt_name
    crypto template template_name ikev2-dynamic
      allow-cert-enc cert-hash-url
```

Refer to the *Command Line Interface Reference* for a complete description of these commands and their keywords.

Disable Change in Rekey Parameters in CHILDSA REKEY

Disabling of rekey parameters must be enabled in a crypto map or crypto template.

For a crypto map the configuration sequence is:

```
configure
  context ctxt_name
    crypto map template_name { ikev2-ipv4 | ikev2-ipv6 }
      ikev2-ikesa
        rekey disallow-param-change
```

For a crypto template the configuration sequence is:

```
configure
  context ctxt_name
    crypto template template_name ikev2-dynamic
      ikev2-ikesa
        rekey disallow-param-change
```

Refer to the *Command Line Interface Reference* for a complete description of these commands and their keywords.

Enable TSr Ranges

To support multiple traffic selectors, the **tsr start-address** command has been modified to process both IPv4 and IPv6 addresses.

```
configure
   context context_name
      crypto templatetnplt_name ikev2-dynamic
         payload payload_name match childsa match any
            tsr start-address ipv4v6_address end-address ipv4v6_address
          end
```

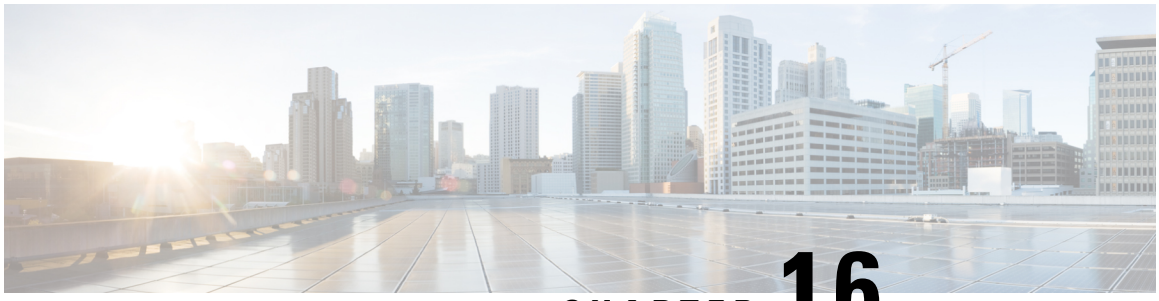
Notes:

- The configuration is restricted to a maximum of four TSrs per payload and per childsa.
- Overlapping TSrs are not allowed either inside the same payload or across different payloads.
- When a TSr is configured via this command, only the configured TSr will be considered for narrowing-down. For example, if one IPv4 TSr is configured, and the gateway receives an IPv6 TSr, the gateway will reject the call with a TS_UNACCEPTABLE notification.
- The UE must send both INTERNAL_IP4_ADDRESS and INTERNAL_IP6_ADDRESS in the Configuration Payload, whenever it needs both IPv4 and IPv6 addresses in TSrs. Otherwise, the gateway will respond back with only one type depending upon the type of address received in the Configuration Payload. For example,.if the gateway receives only INTERNAL_IP4_ADDRESS in the Configuration Payload but both IPv4 and IPv6 addresses are in the TSrs, the gateway will narrow down only the IPv4 address, and ignore the IPv6 TSrs.
- IPv4 TSrs are not allowed inside IPv6 payloads.
- IPv6 TSrs are not allowed inside IPv4 payloads.

show commands

The following **show** commands display configuration parameters associated with support of RFC 5996:

- Statistics for notification payloads
  - **show crypto statistics ikev2**
  - **show crypto ikev2-ikesa security-associations**
  - **show crypto ipsec security-associations**
  - **show crypto statistics ikev2**
- Send and receive statistics for hash-url encrypted certificates
  - **show crypto statistics ikev2**
- RFC 5996 configuration options
  - **show configuration**



CHAPTER 16

IKEv2 DSCP Marking

This feature enables DSCP marking for all IKEv2 messages to the peer.

The following topics are discussed:

Feature Description

This feature enables DSCP values to be included in all IKEv2 packets sent to the peer on the SWu interface. These IKEv2 packets are sent from the gateway to the UE (peer) in the IKEv2 SA_INIT Response message, and subsequently in all IKEV2 AUTH_RESP messages. The DSCP value is either marked in the TOS octet of the IPv4 header or the Traffic Class octet of the IPv6 header as defined in *RFC 2474*. The DSCP-marked IKEv2 packets are given higher priority on the network, and enables sessions to be established faster while minimizing packet drops.

The DSCP value can be configured using the **ikev2-ikesa dscp** command under the Crypto Template Configuration Mode. A default DSCP value of zero (0x00) is set when no configuration exists. For more information on configuring the DSCP value, refer Configuring IKEv2 DSCP Marking, on page 131 of this chapter.

Standards Compliance

This feature complies with the following standard(s):

- **RFC 2474** - Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers

Configuring IKEv2 DSCP Marking

Setting DSCP Value

Use the following configuration to set the DSCP value for the IKEv2 packets sent to the peer:

```
config
   context context_name
      crypto template template_name ikev2-dynamic
         ikev2-ikesa dscp dscp_hex_value
         end
```

Notes:

- *dscp_hex_value* must be an hexa-decimal value between 0x00 and 0x3F.
- Use the **default ikev2-ikesa dscp** command to restore the configuration to its default value.
- **Default:** 0x00

Monitoring and Troubleshooting IKEv2 DSCP Marking

IKEv2 DSCP Marking Show Command(s) and/or Outputs

show crypto ikev2-ikesa security-associations

The following field is available to the output of the **show crypto ikev2-ikesa security-associations** command in support of this feature:

```
IKEv2 SA: 1
        DSCP           : 0x23
```

Table 13: show crypto ikev2-ikesa security-associations Command Output Descriptions

Field	Description
DSCP	Configured DSCP value to be included in the IKEv2 SA packets for the crypto template.

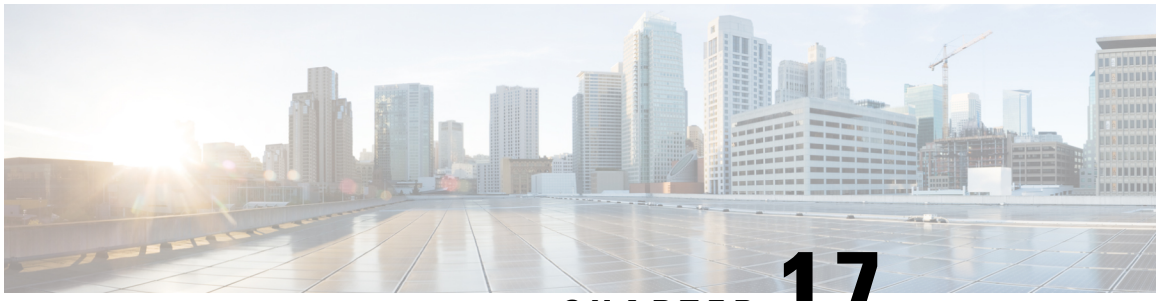
show crypto template

The following field is available to the output of the **show crypto template** command in support of this feature:

```
IKE SA DSCP: 0x23
```

Table 14: show crypto template Command Output Descriptions

Field	Description
IKE SA DSCP	Configured DSCP value to be included in the IKEv2 packets for the crypto template.



CHAPTER 17

IKEv2 Fragmentation

This feature enables IPSec to fragment large messages at IKEv2 as defined in *RFC 7383*.

The following topics are discussed:

Feature Description

Overview

Most Internet Key Exchange (IKEv2) messages are usually small. IP defines a mechanism for fragmentation of oversized UDP messages, but implementations vary in the maximum message size supported. Some NAT and/or Firewall implementations and intermittent routers do not handle IP fragments and these fragmented packets might be dropped. This can cause issues when large IKEv2 messages like digital certificates are transferred over the network.

With this feature, IPSec can fragment large messages at IKEv2 itself and replace them with a series of smaller messages as defined in RFC 7383. The IP datagrams are small enough that fragmentation does not occur at the IP level.

How IKEv2 Fragmentation Works

Fragmenting IKEv2 DL Packets

Only messages that contain an Encrypted payload are subject to IKE fragmentation. For the purpose of construction of IKE Fragment messages, the original (unencrypted) content of the Encrypted payload is split into chunks. The content is treated as a binary blob and is split regardless of the boundaries of inner payloads. Each of the resulting chunks is treated as an original content of the Encrypted Fragment payload, and is encrypted and authenticated. The Encrypted Fragment payload thus contains a chunk of the original content of the Encrypted payload in encrypted form. The Encrypted Fragment payload, if present in a message, will be the last payload in the message.

The maximum fragmentation size is selected based on the value configured through the CLI under the Crypto Template Configuration Mode. For more information, refer *Configuring MTU Size for the IKEv2 Payload*.

Re-assembling IKEv2 Fragmented UL Packets

IPSec receives the encrypted IKEv2 packets and decrypts the encrypted payload. If all fragments are not received, IPSec maintains a timer of 10 seconds, after which the fragments are discarded.

IPSec drops the received fragmented packets during the following scenarios:

- When the received fragment is already present in the buffer.
- When the received fragment exceeds the maximum IKEv2 fragments (255) allowed.
- When each fragment's size exceeds 1932 bytes for IPv4 and 1912 bytes for IPv6.
- When the packet size exceeds 10,000 bytes after re-assembly.

Limitations and Restrictions

This section identifies limitations and restrictions for IKEv2 fragmentation:

- Since IKE Fragment messages are cryptographically protected, SK_a and SK_e must already be calculated. In general, the original message can be fragmented if and only if it contains an encrypted payload. Unprotected payloads cannot be fragmented.
- Among existing IKEv2 extensions, messages of an IKE_SESSION_RESUME exchange, as defined in RFC 5723 cannot be fragmented.
- The Gateway allows fragmenting of IKEv2 packet in downlink, and re-assembling of received IKEv2 packet only if "fragmentation_supported" is negotiated by both peers.
- If the received IKEv2 fragments are greater than 255, the fragments are dropped.

Standards Compliance

This feature complies with the following standards:

- RFC 7383 - Internet Key Exchange Protocol Version 2 (IKEv2) Message Fragmentation
- RFC 7296 - Internet Key Exchange Protocol Version 2 (IKEv2) (for cryptographic processing)

Configuring IKEv2 Fragmentation

Configuring IKESA Fragmentation (Tx) and Re-assembly (Rx)

Use the following configuration to enable or disable IKESA fragmentation (Tx) and re-assembly (Rx):

```
configure
   context context_name
      crypto template template_name ikev2-dynamic
```

```
[ no | default ] ikev2-ikesa fragmentation
end
```

Notes:

- If previously configured, use the **no** keyword to disable IKESA fragmentation and re-assembly support.
- Use the **default** keyword to set the configuration to its default value. By default, IKESA fragmentation and re-assembly is allowed.

Configuring MTU Size for the IKEv2 Payload

Use the following configuration to set the Maximum Transmission Unit (MTU) size for the IKEv2 payload over the IPv4 and/or IPv6 tunnels:

```
configure
   context context_name
      crypto template template_name ikev2-dynamic
         { ip | ipv6 } ikev2-mtu ikev2_mtu_size
         end
```

Notes:

- Use the **default ip ikev2-mtu** or **default ipv6 ikev2-mtu** commands to set the IKEv2 payload to its default value.
- **Default (IPv4):** 1384 bytes
- **Default (IPv6):** 1364 bytes
- *ikev2_mtu_size* (IPv4)must be an integer from 460 through 1932.
- *ikev2_mtu_size* (IPv6)must be an integer from 1144 through 1912.

Monitoring and Troubleshooting IKEv2 Fragmentation

IKEv2 Fragmentation Show Command(s) and/or Outputs

show crypto ikev2-ikesa security-associations

The following field is available to the output of the **show crypto ikev2-ikesa security-associations** command in support of this feature:

```
Fragmentation Supported : Yes
```

*Table 15: **show crypto ikev2-ikesa security-associations** Command Output Descriptions*

Field	Description
Fragmentation Supported	Indicates if IKEv2-IKESA fragmentation is supported.

show crypto statistics ikev2

The following fields are available to the output of the **show crypto statistics ikev2 | more** command in support of this feature:

```
Control Statistics for Context: <context_name>
  Detailed IKE Statistics:
    Total Fragments In:                   0  Total Fragments Out:                  0
    Total IKE Fragmented Packets In:      0  Total IKE Fragmented Packets out:     0
    Total IKE Fragments Dropped:          0
...
  Total IKEv2 Notify Payload Sent Statistics
    Fragmentation Supported Notify Sent:       0
...
  Total IKEv2 Notify Payload Received Statistics:
    Fragmentation Supported Notify Rcvd:       0
```

Table 16: show crypto statistics ikev2 Command Output Descriptions

Field	Description
Detailed IKE Statistics:	
Total Fragments In	Total fragments received.
Total Fragments Out	Total fragments sent.
Total IKE Fragmented Packets In	Total number of IKE fragmented packets received.
Total IKE Fragmented Packets out	Total number of IKE fragmented packets sent.
Total IKE Fragments Dropped	Total number of IKE fragments dropped.
Total IKEv2 Notify Payload Sent Statistics:	
Fragmentation Supported Notify Sent	Total number of IKEv2 Fragmentation Supported notify message sent.
Total IKEv2 Notify Payload Received Statistics:	
Fragmentation Supported Notify Rcvd	Total number of IKEv2 Fragmentation Supported notify message received.

show crypto template

The following fields are available to the output of the **show crypto template** command in support of this feature:

```
Map Name: <map_name>
=======================================
```

```
IKEv2 Fragmentation                    :  Enabled
IKEv2 MTU Size IPv4/IPv6               :  1438/1422
```

Table 17: show crypto template Command Output Descriptions

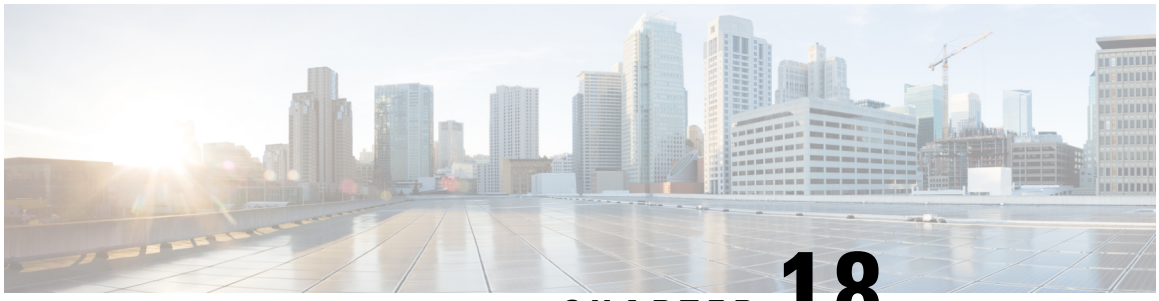
Field	Description
IKEv2 Fragmentation	Indicates if the configuration for IKEv2 fragmentation is enabled or disabled.
IKEv2 MTU Size IPv4/IPv6	Configured IKEv2 payload MTU size over the IPv4/IPv6 tunnels.

IKEv2 Fragmentation Bulk Statistics

The following bulks statistics included in the System schema support this feature:

Variable	Description	Data Type
ikev2-tx-fragments	**Description:** The total number of fragments transmitted to UE with Internet Key Exchange v2 (IKEv2). **Triggers:** Increments for each fragmented packet transmitted to UE with Internet Key Exchange v2 (IKEv2). **Availability:** System **Type:** Counter	Int32
ikev2-rx-fragments	**Description:** The total number of fragments received from UE with Internet Key Exchange v2 (IKEv2). **Triggers:** Increments for each fragmented packet received from UE with Internet Key Exchange v2 (IKEv2). **Availability:** System **Type:** Counter	Int32
ikev2-tx-fragmented-packet	**Description:** The total number of fragmented packets transmitted to UE with Internet Key Exchange v2 (IKEv2). **Triggers:** Increments if packets are fragmented and transmitted to UE with Internet Key Exchange v2 (IKEv2). **Availability:** System **Type:** Counter	Int32
ikev2-rx-fragmented-packet	**Description:** The total number of fragmented packets received from UE with Internet Key Exchange v2 (IKEv2). **Triggers:** Increments if fragmented packets are received from UE and reassembled with Internet Key Exchange v2 (IKEv2). **Availability:** System **Type:** Counter	Int32

Variable	Description	Data Type
ikev2-rx-fragments-dropped	**Description:** The total number of fragments received from UE dropped with Internet Key Exchange v2 (IKEv2). **Triggers:** Increments for each fragment received from UE dropped with Internet Key Exchange v2 (IKEv2). **Availability:** System **Type:** Counter	Int32



CHAPTER 18

IKEv2 Mobility and Multi-homing Protocol

This chapter provides information on the IKEv2 Mobility and Multi-homing Protocol feature.

The following topics are discussed:

Feature Description

Overview

IPSec can support IKEv2 Mobility and Multi-homing protocol (MOBIKE) as defined in *RFC 4555*. IKEv2 Mobility and Multi-homing Protocol (MOBIKE) allows the IP addresses associated with IKEv2 and tunnel mode IPSec Security Associations (SA) to change. A mobile Virtual Private Network (VPN) client could use MOBIKE to keep the connection with the VPN gateway active while moving from one address to another. Similarly, a multi-homed host could use MOBIKE to move the traffic to a different interface if, for instance, the one currently being used stops working. This enables peer hosts to change its point of network attachment and use different interfaces without removing the existing IPSec tunnel.

The MOBIKE feature is suited when the address of at least one peer is stable, and can be discovered using mechanisms such as DNS. While both parties can be mobile, one party must be rooted at any given time. Additionally, the Gateway is neither multi-homed nor possess mobility capabilities.

Supported Platforms

Currently, IPSec supports the MOBIKE feature on Cisco ASR 5500 and Ultra Services platforms.

How IKEv2 Mobility and Multi-homing Protocol Works

Signaling

MOBIKE is initiated when the feature is enabled, and the Gateway receives an IKE_AUTH containing the SA payload with MOBIKE_SUPPORTED from the peer. The Gateway responds with an IKE_AUTH containing the SA payload with MOBIKE_SUPPORTED.

The feature can be enabled using the **ikesa mobike** command under the Crypto Template Configuration Mode. For more information on the **ikesa mobike** command, refer Enabling IKEv2 Mobility and Multi-homing Protocol, on page 141.

Return Routability Check

A return routability check ensures that the other party can receive packets at the claimed address. When the Gateway receives an UPDATE_SA_ADDRESS, IKE SA/IPSec SA is updated and return routability check is triggered. This function can be enabled using the **ikesa mobike cookie-challenge** command under the Crypto Template Configuration Mode. By default, the Gateway does not perform the return routability check. When enabled, the Gateway sends a cookie challenge to the peer. The session is deleted when a response is not received, or when an invalid response is received. It is assumed that when a valid response is received for any informational exchange, the peer can receive packets at the claimed address.

For more information on the **ikesa mobike cookie-challenge** command, refer Enabling IKEv2 Mobility and Multi-homing Protocol, on page 141.

To ensure that the peer cannot generate the correct INFORMATIONAL response without seeing the request, a new payload is added to INFORMATIONAL messages. The sender of an INFORMATIONAL request can include a COOKIE2 notification, and if included, the recipient of an INFORMATIONAL request copies the notification as-is to the response. When processing the response, the original sender verifies that the value is the same as sent. If the values do not match, the IKE_SA is closed.

Limitations and Restrictions

This section identifies limitations and restrictions for the MOBIKE feature:

- Mobility is supported for peers only. The Gateway has a fixed interface/IP and does not move across the network.
- IPSec supports MOBIKE for Subscriber mode only.
- When a change of address occurs, the peers must always notify the change to the Gateway.
- As the Gateway has a single IP address, an ADDITION_*_ADDRESS message will not be sent to the peers.
- The Gateway does not store the ADDITION_*_ADDRESS information. Any payload with the information from the peer is ignored.
- Retransmission/Dead Peer Detection (DPD) timeout must be configured with a larger value when the MOBIKE feature is enabled.

Standards Compliance

This feature complies with the following standard(s):

- RFC 4555 - IKEv2 Mobility and Multi-homing Protocol (MOBIKE)

Configuring IKEv2 Mobility and Multi-homing Protocol

Enabling IKEv2 Mobility and Multi-homing Protocol

Use the following configuration under the Crypto Template Configuration Mode to enable the MOBIKE feature:

```
config
   context context_name
      crypto template template_name ikev2-dynamic
         ikev2-ikesa mobike [ cookie-challenge ]
         end
```

- Use the **default ikev2-ikesa mobike** command to restore the configuration to its default setting. By default, mobike is disabled.
- If previously configured, use the **no ikev2-kesa mobike** command to remove the configuration.
- Use the **cookie-challenge** keyword to enable the return routability check. The Gateway performs a return routability check when MOBIKE is enabled along with this keyword. This configuration is disabled by default.

Monitoring and Troubleshooting IKEv2 Mobility and Multi-homing Protocol

IKEv2 Mobility and Multi-homing Protocol Show Command(s) and/or Outputs

show crypto ikev2-ikesa security-associations

The following field is available to the output of the **show crypto ikev2-ikesa security-associations** command to indicate if the MOBIKE feature is supported:

```
Mobike Supported         :  Enabled
```

Table 18: show crypto ikev2-ikesa security-associations Command Output Descriptions

Field	Description
Mobike Supported	Indicates if the MOBIKE feature is supported for this IKEv2 security association.

show crypto statistics ikev2

The following fields are available to the output of the **show crypto statistics ikev2** command in support of this feature:

```
Total IKEv2 Notify Message Receive Statistics:
    MOBIKE Supported:         2   Additional IPv4 Address:     0
    Additional IPv6 Address:  0   No Additional Address:       0
    Update SA Addresses:      2   COOKIE2:                     0
    No NAT Allowed:           0

 Total IKEv2 MOBIKE Statistics:
    MOBIKE Notify Sent:       2   MOBIKE Notify Rcvd:          2
    MOBIKE Ignored:           0   MOBIKE Unexpected NAT Sent:  0
    MOBIKE Unacceptable Address Sent: 0  MOBIKE Cookie2 Rcvd:  0
    MOBIKE Cookie2 Sent:      0  MOBIKE COOKIE2 Mismatch:      0
```

Table 19: show crypto statistics ikev2 Command Output Descriptions

Field	Description
Total IKEv2 Notify Message Receive Statistics:	
MOBIKE Supported	Total number of MOBIKE_SUPPORTED notify payload received.
Additional IPv4 Address	Total number of additional IPv4 addresses received from the peers.
Additional IPv6 Address	Total number of additional IPv6 addresses received from the peers.
No Additional Address	Total number of NO_ADDITIONAL_ADDRESSES notify payload received.
Update SA Addresses	Total number of UPDATE_SA_ADDRESSES notify payload received.
COOKIE2	Total number of COOKIE2 notify payload received.
No NAT Allowed	Total number of NO_NATS_ALLOWED notify payload received.
Total IKEv2 MOBIKE Statistics:	
MOBIKE Notify Sent	Total number of MOBIKE_SUPPORTED notify payload sent from Gateway.
MOBIKE Notify Rcvd	Total number of MOBIKE_SUPPORTED notify payload received and processed successfully.
MOBIKE Ignored	Total number of MOBIKE_SUPPORTED notify payload received, processed and ignored.
MOBIKE Unexpected NAT Sent	Total number of UNEXPECTED_NAT_DETECTED notify payload sent.
MOBIKE Unacceptable Address Sent	Total number of UNACCEPTABLE_ADDRESSES notify payload sent by the Gateway.
MOBIKE Cookie2 Rcvd	Total number of COOKIE2 notify payload received and decoded successfully.
MOBIKE Cookie2 Sent	Total number of COOKIE2 notify payload sent.

Field	Description
MOBIKE COOKIE2 Mismatch	Total number of Cookie2 mismatch occurrences at the Gateway

show crypto template

The following field is available to the output of the **show crypto template** command to indicate if the MOBIKE feature is enabled:

```
IKEv2 Mobike          :  Enabled
```

Table 20: show crypto template Command Output Descriptions

Field	Description
IKEv2 Mobike	Indicates if the MOBIKE feature is enabled or disabled for this crypto template.

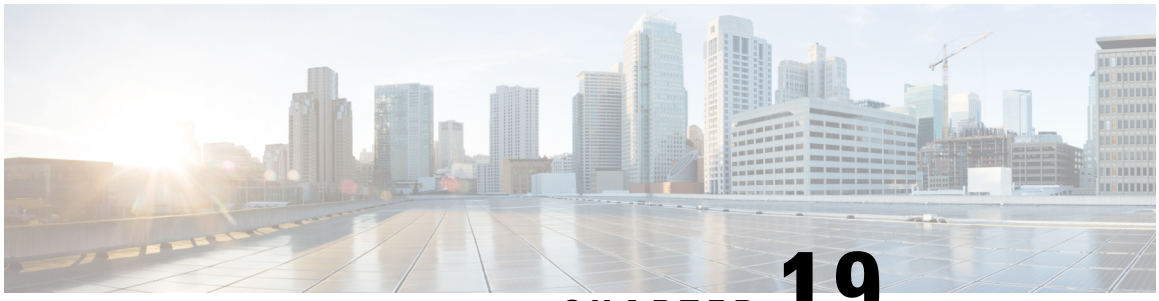
IKEv2 Mobility and Multi-homing Protocol Bulk Statistics

The following bulks statistics included in the System schema support this feature:

Variable	Description	Data Type
ikev2-mobike-sent	**Description**: Total number of MOBIKE_SUPPORTED notify payload sent from Gateway. **Triggers**: Increments when MOBIKE_SUPPORTED notify payload is sent from the Gateway in IKE_AUTH response. **Availability**: Service independent. IPSec Subscriber mode. **Type**: Counter	Int32
ikev2-mobike-recv	**Description**: Total number of MOBIKE_SUPPORTED notify payload received and processed successfully. **Triggers**: Increments when the MOBIKE_SUPPORTED payload received in the IKE_AUTH request is processed. **Availability**: Service independent. IPSec Subscriber mode. **Type**: Counter	Int32
ikev2-mobike-ignored	**Description**: Total number of MOBIKE_SUPPORTED notify payload received, processed and ignored. **Triggers**: Increments when the received MOBIKE_SUPPORTED notify payload is processed and ignored, when it is received in the IKE_AUTH request as MOBIKE is not configured on the Gateway. **Availability**: Service independent. IPSec Subscriber mode. **Type**: Counter	Int32

Variable	Description	Data Type
ikev2-mobike-unexpected-natt-detected-sent	**Description**: Total number of UNEXPECTED_NAT_DETECTED notify payload sent. **Triggers**: **Availability**: Service independent. IPSec Subscriber mode. **Type**: Counter	Int32
ikev2-mobike-cookie2-rcvd	**Description**: Total number of COOKIE2 notify payload received and decoded successfully. **Triggers**: Increments when UNEXPECTED_NAT_DETECTED is sent in IKEv2 exchange response because NO_NATS_ALLOWED notify payload was received in IKEv2 exchange request and natt is detected. **Availability**: Service independent. IPSec Subscriber mode. **Type**: Counter	Int32
ikev2-mobike-cookie2-sent	**Description**: Total number of COOKIE2 notify payload sent. **Triggers**: Increments when COOKIE2 notify payload is received and decoded successfully. **Availability**: Service independent. IPSec Subscriber mode. **Type**: Counter	Int32
ikev2-mobike-cookie2-mismatch	**Description**: Total number of Cookie2 mismatch occurrences at the Gateway. **Triggers**: Increments when COOKIE2 notify payload is sent from the Gateway. **Availability**: Service independent. IPSec Subscriber mode. **Type**: Counter	Int32
ikev2-mobike-cookie2-match	**Description**: Total number of Cookie2 matched. **Triggers**: Increments when COOKIE2 notify payload is received in IKEv2 exchange response and is different from the Cookie2 sent in IKEv2 exchange request. **Availability**: Service independent. IPSec Subscriber mode. **Type**: Counter	Int32

Variable	Description	Data Type
ikev2-notifrecv-mobikesupp	**Description**: Total number of MOBIKE_SUPPORTED notify payload received. **Triggers**: Increments when COOKIE2 notify payload is received in IKEv2 exchange response and is same as the Cookie2 sent in IKEv2 exchange request. **Availability**: Service independent. IPSec Subscriber mode. **Type**: Counter	Int32
ikev2-notifrecv-updsaaddr	**Description**: Total number of UPDATE_SA_ADDRESSES notify payload received. **Triggers**: Increments when MOBIKE_SUPPORTED notify payload is received in any IKEv2 exchange. **Availability**: Service independent. IPSec Subscriber mode. **Type**: Counter	Int32
ikev2-notifrecv-cookie2	**Description**: Total number of COOKIE2 notify payload received. **Triggers**: Increments when COOKIE2 notify payload is received in any IKEv2 exchange. **Availability**: Service independent. IPSec Subscriber mode. **Type**: Counter	Int32
ikev2-notifrecv-nonatallow	**Description**: Total number of NO_NATS_ALLOWED notify payload received. **Triggers**: Increments when NO_NATS_ALLOWED notify is received in any IKEv2 exchange. **Availability**: Service independent. IPSec Subscriber mode. **Type**: Counter	Int32



CHAPTER 19

IKEv2 - Protection Against Distributed Denial of Service

This feature provides security mechanisms for IKEv2 to defend against Distributed Denial-of-Service (DDoS) attacks.

The following topics are discussed:

Feature Description

Overview

A distributed denial-of-service (DDoS) attack is caused when multiple malicious systems flood the targeted system with messages in the intention of exhausting the memory of the targeted system. This causes the affected system to run out of sufficient resources to service requests from legitimate peers. Attackers targeting a system can employ any of the following methods:

- Send large amounts of IKE_SA_INIT messages (but no IKE_Auth) for which half-open IKE SA structures are created. This causes the system to utilize resources and run out of memory.
- Send a large amount of junk IKE_Auth packets with correct SPI_i and SPI_r. This causes the system to run out of memory while trying to decrypt the packets.
- Provide an illegitimate URL with a certificate of large size.
- Send continuous SA_INIT packets. This causes the system to run out of memory while trying to generate keys for encrypted packets.
- Send large amounts of rekey requests per second.
- Send large amounts of messages with distinct message IDs. This causes the system to queue all incoming IKE messages, and run out of memory.

This feature provides mechanisms to defend against the DDoS attacks outlined above.

How IKEv2 Protection Against DDoS Works

Architecture

The following prevention mechanisms are available on IPSec against DDoS:

- **Half-open IKE SA timeout** – When an IKE_SA_INIT request is received, a half-open IKE SA timer starts. If an IKE_AUTH message is not received before the timer expires, the half-open IKEv2 IKE SA is cleared. The timer value can be configured using the **ikev2-ikesa ddos half-open-sa-timer** command under the Crypto Template Configuration Mode.
- **Consecutive IKE_AUTH decryption failure detection** – During session creation, if IKE_AUTH decryption fails consecutively for a specified number of times, the IKEv2 IKE SA is cleared. The number of consecutive failure count can be configured using the **ikev2-ikesa ddos decrypt-fail-count** command under the Crypto Template Configuration Mode.

  Alarms will be triggered if decryption fails during post message creation.
- **Certificate validation while downloading** – Downloading large certificates from illegitimate URLs can be avoided by defining the maximum certificate size for the IKE SA. The maximum certificate size can be configured using the **ikev2-ikesa ddos max-cert-size** command under the Crypto Template Configuration Mode.
- **Limit on Child SA re-key per second** – When the specified number of Child SA rekey requests per second is exceeded, TEMPORARY_FAILURE notifications will be sent to the peer to indicate that the peer must slow down their requests. The maximum Child SA re-key request per second can be configured using the **ikev2-ikesa ddos rekey-rate** command under the Crypto Template Configuration Mode.
- **Limit on IKE messages per IKE SA** – When the incoming queued IKE messages per IKE SA exceeds the specified limit, the IKE messages exceeding the limit are dropped. The limit for the IKE messages per IKE SA can be configured using the **ikev2-ikesa ddos message-queue-size** under the Crypto Template Configuration Mode.

For more information on the **ikev2-ikesa ddos** command, refer Configuring IKEv2 Protection Against DDoS, on page 149 of this chapter.

Standards Compliance

This feature complies with the following standards:

- **RFC 7296** – Internet Key Exchange Protocol Version 2 (IKEv2) (for handling DoS attacks)
- IETF reference – "Protecting Internet Key Exchange Protocol version 2 (IKEv2), Implementations from Distributed Denial of Service Attacks"

Configuring IKEv2 Protection Against DDoS

Configuring Half-open SA Timer

Use the following configuration to set the half-open IKE SA timeout duration:

```
config
   context context_name
      crypto template template_name ikev2-dynamic
         ikev2-ikesa ddos half-open-sa-timer half_open_timer_duration
         end
```

Notes:

- *half_open_timer_duration* must be an integer between 1 and 1800.
- **Default**: 60 seconds
- Use the **default ikev2-ikesa ddos half-open-sa-timer** command to restore the configuration to its default value.
- Use the **no ikev2-ikesa ddos half-open-sa-timer** command to disable a previously enabled configuration.

Configuring Decryption Failure Count

Use the following configuration to set the maximum tolerable consecutive IKE_AUTH decryption failure count:

```
config
   context context_name
      crypto template template_name ikev2-dynamic
         ikev2-ikesa ddos decrypt-fail-count failure_count
         end
```

Notes:

- *failure_count* must be an integer between 1 and 100.
- **Default**: 30 times
- Use the **default ikev2-ikesa ddos decrypt-fail-count** keyword to restore the configuration to its default value.
- Use the **no ikev2-ikesa ddos decrypt-fail-count** keyword to disable a previously enabled configuration.

Configuring Re-key Rate

Use the following configuration to set the maximum number of Child SA rekey requests per second:

```
config
   context context_name
      crypto template template_name ikev2-dynamic
```

```
ikev2-ikesa ddos rekey-rate rekey_rate_value
end
```

Notes:

- *rekey_rate_value* must be an integer between 1 and 50.
- **Default**: 5
- Use the **default ikev2-ikesa ddos rekey-rate** keyword to restore the configuration to its default value.
- Use the **no ikev2-ikesa ddos rekey-rate** keyword to disable a previously enabled configuration.

Configuring Message Queue Size

Use the following configuration to set the queue size for incoming IKE messages per IKE SA:

```
config
   context context_name
      crypto template template_name ikev2-dynamic
         ikev2-ikesa ddos message-queue-size queue_size
         end
```

Notes:

- *queue_size* must be an integer between 1 and 50.
- **Default**: 20
- Use the **default ikev2-ikesa ddos message-queue-size** keyword to restore the configuration to its default value.
- Use the **no ikev2-ikesa ddos message-queue-size** keyword to disable a previously enabled configuration.

Configuring Maximum Certificate Size

Use the following configuration to set the maximum certificate size for IKE SA:

```
config
   context context_name
      crypto template template_name ikev2-dynamic
         ikev2-ikesa ddos max-cert-size cert_size
         end
```

Notes:

- *cert_size* must be an integer between 512 and 8192.
- **Default**: 2048 bytes
- Use the **default ikev2-ikesa ddos max-cert-size** keyword to restore the configuration to its default value.
- Use the **no ikev2-ikesa ddos max-cert-size** keyword to disable a previously enabled configuration.

Monitoring and Troubleshooting

Show Command(s) and/or Outputs

show crypto ikev2-ikesa security-associations

The following fields are available in the output of the **show crypto ikev2-ikesa security-associations** command in support of this feature:

```
Detailed IKEv2 stats
       0 Request Dropped - Message Queue Size Exceeded
       0 Total IKEv2 IKESA Rekey Requests Temporary Failure Rekey Rate
       0 Total IKEv2 ChildSA Rekey Requests Temporary Failure Rekey Rate
       0 Decryption Fail Count Exceeded
```

*Table 21: **show crypto ikev2-ikesa security-associations** Command Output Descriptions*

Field	Description
Detailed IKEv2 stats	
Request Dropped - Message Queue Size Exceeded	Total number of messages dropped due to exceeding the specified queue size.
Total IKEv2 IKESA Rekey Requests Temporary Failure Rekey Rate	Total number of temporary failure messages sent for IKE SA rekey requests due to exceeding the specified rekey rate.
Total IKEv2 ChildSA Rekey Requests Temporary Failure Rekey Rate	Total number of temporary failure messages sent for CHILD SA rekey requests due to exceeding the specified rekey rate.
Decryption Fail Count Exceeded	Total number of messages dropped due to exceeding the specified decryption failure rate.

show crypto statistics ikev2

The following fields are available in the output of the **show crypto statistics ikev2** command in support of this feature:

```
Certificate Authentication Statistics:
    Large Certificate Length:             0
Total IKEv2 Timer Expiration Statistics:
    IKE_SA Half Open:                  7 IKE_SA Half Open (No XCHG):           0
Total IKEv2 Child_SA Rekey Statistics:
    IKE_SA Temp Failure Rekey Rate:    0 CHILD_SA Temp Failure Rekey Rate:     0
Total IKEv2 Exchanges Dropped:
    Message Queue Size Exceeded           0
Total IKEv2 Decrypt Failure Statistics:
    Decrypt Fail Count Exceeded           0

IKEv2 DEBUG Statistics:
------------------------------------------------------------------------
       11 tot_ikev2_ikesa_half_open_sa_timer_start
```

```
        0 tot_ikev2_ikesa_half_open_sa_timer_start_ignored
        0 tot_ikev2_ikesa_half_open_sa_timer_stop
-------------------------------------------------------------------------
```

*Table 22: **show crypto statistics ikev2** Command Output Descriptions*

Field	Description
Certificate Authentication Statistics	
Large Certificate Length	Total IKEv2 certification authentication failures due to large certificate length.
Total IKEv2 Timer Expiration Statistics	
IKE_SA Half Open	The total number of IKESA half open SA timer expirations (valid exchange).
IKE_SA Half Open (No XCHG)	The total number of IKE SA Half Open SA timer expirations (no exchange).
Total IKEv2 Child_SA Rekey Statistics	
IKE_SA Temp Failure Rekey Rate	Total number of temporary failures sent for IKE SA rekey requests due to rekey rate exceeded.
CHILD_SA Temp Failure Rekey Rate	Total number of temporary failures sent for CHILD SA rekey requests due to rekey rate exceeded.
Total IKEv2 Exchanges Dropped	
Message Queue Size Exceeded	The total number of IKEv2 exchanges dropped (message queue size exceeded).
Total IKEv2 Decrypt Failure Statistics	
Decrypt Fail Count Exceeded	Total IKEv2 decryption failures (Fail Count Exceeded).

show crypto template

The following fields are available in the output of the **show crypto template** command in support of this feature:

```
IKEv2 IKESA DDOS Mitigation Params:
  IKE SA Half Open Timer: 30
  IKE SA Decrypt Fail Count: 30 [Default]
  IKE SA Message Queue Size: 20 [Default]
  IKE SA Rekey Rate: 5 [Default]
  IKE SA Max Certificate Size: Disabled
```

*Table 23: **show crypto template** Command Output Descriptions*

Field	Description
IKEv2 IKESA DDOS Mitigation Params:	

Field	Description
IKE SA Half Open Timer	Configured IKE SA half-open timer duration (in seconds) for the crypto template.
IKESA Decrypt Fail Count	Configured IKE SA decryption failure count for the crypto template.
IKE SA Message Queue Size	Configured IKE SA maximum message queue size for the crypto template.
IKE SA Rekey Rate	Configured IKE SA rekey rate for the crypto template.
IKE SA Max Certificate Size	Configured IKE SA maximum certificate download size (in bytes) for the crypto template.

Bulk Statistics

The following bulks statistics included in the System schema support this feature:

Variable	Description	Data Type
ikev2-exp-half-open-sa-noxchg	**Description:** The total number of IKE SA Half Open SA timer expirations (no exchange). **Triggers:** Increments when a Half Open SA timer expires without any valid exchange. **Availability:** ePDG Service **Type:** Counter	Int32
ikev2-exp-half-open-sa	**Description:** The total number of IKESA half open SA timer expirations (valid exchange). **Triggers:** Increments when a Half Open SA timer expires with a valid exchange. **Availability:** ePDG Service **Type:** Counter	Int32
ikev2-xchg-drop_msg-queuesize-exceeded	**Description:** The total number of IKEv2 exchanges dropped (message queue size exceeded). **Triggers:** Increments when IKEv2 messages get dropped due to message queue size exceeded. **Availability:** ePDG Service **Type:** Counter	Int32

Variable	Description	Data Type
ikev2-decryptfail-count-exceeded	**Description:** Total IKEv2 decryption failures (Fail Count Exceeded). **Triggers:** Increments when IKEv2 decryption failure count exceeds the configured value. **Availability:** ePDG Service **Type:** Counter	Int32
ikev2-ikesa-rekey-rate-temp-failure	**Description:** Total number of temporary failures sent for IKE SA rekey requests due to rekey rate exceeded. **Triggers:** Increments when a temporary failure to IKESA request is sent due to rekey rate exceeded. **Availability:** ePDG Service **Type:** Counter	Int32
ikev2-childsa-rekey-rate-temp-failure	**Description:** Total number of temporary failures sent for CHILD SA rekey requests due to rekey rate exceeded. **Triggers:** Increments when a temporary failure to CHILDSA request is sent due to rekey rate exceeded. **Availability:** ePDG Service **Type:** Counter	Int32
ikev2-cert-auth-fail-large-length	**Description:** Total IKEv2 certification authentication failures due to large certificate length. **Triggers:** Increments when a certificate with a length larger than the configured value is downloaded. **Availability:** ePDG Service **Type:** Counter	Int32

Thresholds

DoS Cookie Challenge

An **IPSecMgr IKEv2 DOS Attack** alarm is generated when the high or low threshold is reached for DOS cookie challenge. The alarm is triggered when the configured DOS cookie challenge reaches the high threshold limit (system is under attack). This alarm is an indication of a security threat. The alarm is cleared when the configured DOS cookie challenge reaches the low threshold limit (system is out of attack).

IKE_Auth Decryption Failure

An **IPSecMgr Decryption Failure** alarm is generated when the high or low threshold is reached for IKE_Auth decryption failure. The alarm is triggered when the configured decryption failure count is reached. The alarm is cleared when the IKEv2 message from the peer is decrypted successfully, or if the session is cleared.

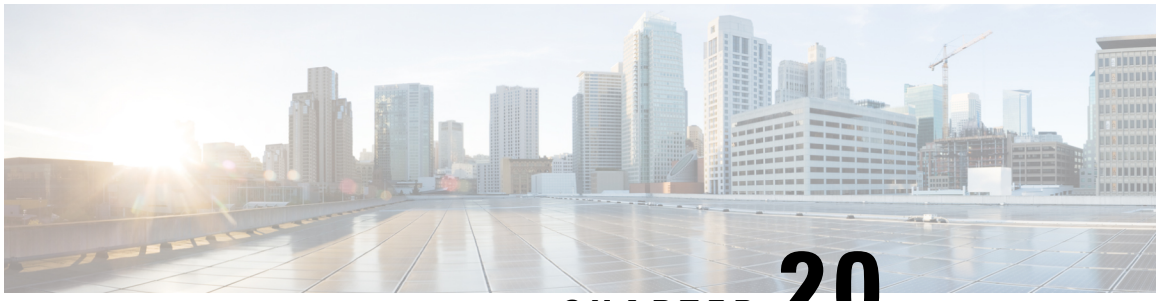
SNMP Traps

The following traps are available to track status and conditions relating to DDoS attack:

- **starIKEv2DOSAttack**: An ipsecmgr facility is under DDOS attack.
- **starIKEv2ClearDOSAttack**: An ipsecmgr facility is out of DDOS attack.

The following traps are available to track status and conditions relating to decryption failure:

- **starIKEv2DecryptionFailThreshold**: The decryption fail count for subsequent IKEV2 messages from a UE exceeds the configured value.
- **starIKEv2ClearDecryptionFailThreshold**: The UE sends a valid IKEv2 packet for which decryption passes.



CHAPTER 20

IKEv2 and IPSec Parameter Setting Per Device Type

Feature Information

Summary Data

Status	New Feature
Introduced-In Release	21.2
Modified-In Release(s)	Not Applicable
Applicable Product(s)	ePDG
Applicable Platform(s)	ASR 5500 VPC-SI VPC-DI
Default Setting	Disabled
Related CDETS ID(s)	CSCvc38683
Related Changes in This Release	Not Applicable
Related Documentation	IPSec Reference Guide Command Line Interface Reference Guide

Revision History

Revision Details	Release	Release Date
New in this release.	21.2	April 27, 2017

Feature Description

Overview

This feature provides an IKEv2 and IPSec framework for operators to configure the system where different peer devices can establish an IKEv2/IPSec tunnel with different set of capabilities and features. Vendor-specific crypto templates which are a subset of the IKEv2 dynamic template can be configured for each device separately. Multiple vendor templates can be grouped under a vendor policy and associated to the parent IKEv2 dynamic template. A single Gateway service (supporting IPSec) can serve different peers with different IKEv2/IPSec (pre-defined) requirements.

How IKEv2/IPSec Parameter Setting Per Device Type Works

Feature Components

Vendor Template

A vendor template is a subset of the IKEv2 dynamic crypto template. The available subset of features and configurations for vendor template remain the same as that of the IKEv2 dynamic crypto template.

The following functions/configurations are currently available for the vendor template:

- DPD timer
- IKESA transform-set list
- IPSec transform-set list
- IKE features like Mobike and Fragmentation
- Private numbers for PCSCF/IMEI payloads
- IPSec rekey related configurations
- IKE rekey related configurations

Important It is recommended to use one vendor template to configure each IKEv2 or IPSec functionality as required for the device.

For configuration information, refer the configuration section of this chapter.

Vendor Policy

A vendor policy is used to group multiple vendor templates to a single policy using a vendor ID and the vendor template name. This vendor ID will be used to make updated configuration while comparing vendor ID value received in the IKE SA packet. The vendor policy is associated to the main crypto (service) template. When similar configurations exist under the vendor template and the main crypto (service) template, the configuration under the vendor template takes priority if the vendor ID matches the vendor ID payload from IKE_SA_INIT during tunnel exchange (based on precedence).

A maximum of 32 vendor policies can be configured. Each vendor policy can be associated with a maximum of 64 vendor templates.

For configuration information, refer the configuration section of this chapter.

Associating Vendor Policy to a Crypto (Service) Template

Only one vendor policy can be attached to a crypto (service) template.

Architecture

During IKE_SA_INIT exchange, on receiving the vendor ID payload, allocated IKE SA is updated with the configuration from both the main crypto (service) template and the vendor templates. Configuration values are then taken from IKE SA during tunnel negotiation. The priority of the configuration in the vendor template over the main crypto (service) template is decided by matching the vendor ID string associated to the vendor template (in the vendor policy) with the received vendor ID in the IKE_SA_INIT message.

If the IKE_SA_INIT message contains more than one vendor ID, all the vendor IDs will be matched with the configuration under the vendor policy. For all successful matches, the configuration parameters are taken from the vendor template and the rest of the configuration is taken from the main crypto (service) template. Matching is performed based on the precedence associated with the vendor template. When a configuration is available in more than one vendor template, the configuration is chosen based on the highest precedence in the vendor policy. A precedence value of 1 indicates the highest priority with the vendor policy for a set of associated vendor templates.

Limitations

Architectural Limitations

- In a downgrade scenario for Inter-chassis session recovery, as this feature will not be available in the stand-by chassis, the IKE SA updated with the vendor template configuration will not be synced to the stand-by chassis.

Configuration Limitations

- **IKE SA rekey configuration** – If some of the **ikev2-ikesa rekey**, **ikev2-ikesa rekey disallow-param-change**, or **ikev2-ikesa ignore rekeying requests** commands exist in the main crypto (service) template configuration, and if any of these commands also exist in the vendor template, all configuration will be taken from the vendor template if the vendor ID matches.
- **IPSec rekey and lifetime configuration** – If any of the **rekey keepalive**, **ignore rekeying requests**, or **lifetime** command exists in the vendor template, all IPSec rekey configurations will be taken from the vendor template.

- Currently, only one payload configuration is effective.

Configuring IKEv2 and IPSec Parameter Per Device Type

Configuring Vendor Template for Vendor-specific Information

Use the following configuration to create a vendor template, and get into the Crypto Template IKEv2 Vendor Configuration Mode:

```
config
   context context_name
      crypto template template_name ikev2-vendor
```

Notes:

- The following commands are available under the Crypto Template IKEv2 Vendor configuration mode to configure IPSec-related parameters. Their functionalities are similar to the commands under the Crypto Template Configuration Mode.
  - configuration-payload
  - ikev2-ikesa
  - keepalive
  - payload

Configuring Vendor Policy and Associating With Vendor Template

Use the following configuration to create a vendor policy, and associate it with the vendor template:

```
config
   context context_name
      crypto vendor-policy vendor_policy_name
         precedence value vendor-id vendor_id vendor-template template_name
         end
```

Notes:

- A maximum of 32 vendor policies can be configured.
- A maximum of 64 vendor templates can be associated with a vendor policy.
- *vendor_id* must be an integer from 1 through 64.
- **template_name** must be an alphanumeric string from 1 to 127 characters.

Associating Vendor Policy to Crypto Template

Use the following configuration to associate the vendor policy to the crypto (services) template:

```
config
   context context_name
      crypto template template_name ikev2-dynamic
         vendor-policy policy_name
         end
```

Notes:

- *policy_name* must be an alphanumeric string from 1 to 127 characters.

Monitoring and Troubleshooting IKEv2 and IPSec Parameter Setting Per Device Type

Show Command(s) and/or Outputs

show crypto statistics ikev2

The following fields are available in the output of the **show crypto statistics ikev2** command in support of this feature:

```
IKEv2 SA_INIT Vendor-ID Matching Statistics:
Total packet rcvd with Vendor ID:  1  Total Vendor-ID's rcvd in IKE_SA_INIT:   4
Rcvd Vendor-ID successful Match:   3  Rcvd Vendor-ID no Match              :   1
```

Table 24: show crypto statistics ikev2 Command Output Descriptions

Field	Description
IKEv2 SA_INIT Vendor-ID Matching Statistics:	
Total packet rcvd with Vendor ID	Total number of packets received with vendor ID.
Total Vendor-ID's rcvd in IKE_SA_INIT	Total number of vendor IDs received in the IKE_SA_INIT message.
Rcvd Vendor-ID successful Match	Total number of matches for the vendor IDs received in the IKE_SA_INIT message.
Rcvd Vendor-ID no Match	Total number of vendor IDs that did not match.

show crypto template

The following fields are available in the output of the **show crypto template** command in support of this feature:

Main (services) template values

```
Attached vendor policy: vp1
```

*Table 25: **show crypto template** Command Output Descriptions*

Field	Description
Attached vendor policy	Specifies the vendor policy associated with the crypto template.

Vendor template configured values

```
Crypto Map Type: IPSEC IKEv2 Vendor Template
```

*Table 26: **show crypto template** Command Output Descriptions*

Field	Description
Crypto Map Type	Specifies that the crypto map type used is from the vendor template.

Important The output also displays those configured parameters applicable to the vendor template. The fields are similar to the configuration available for the IKEv2 dynamic crypto template.

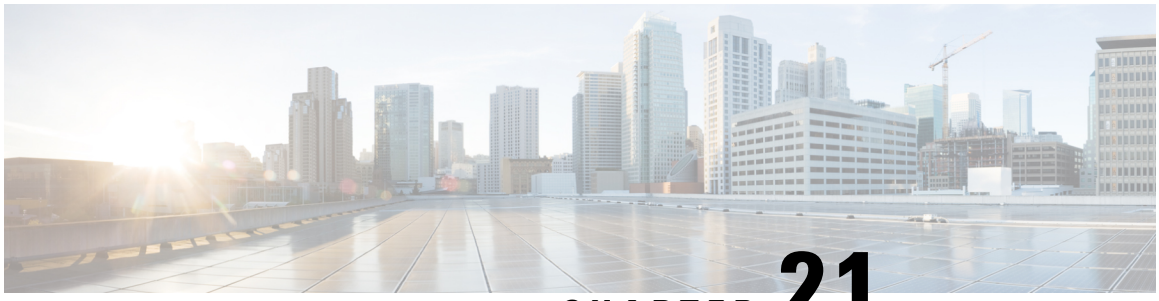
show crypto vendor-policy

The following fields are available in the output of the **show crypto vendor-policy** command in support of this feature:

```
Crypto Vendor Policy Name vp1
  VID IKESA1 vendor template v1 precedence 1
  VID IKESA2 vendor template v2 precedence 2
1 Crypto vendor policy are configured
```

*Table 27: **show crypto vendor-policy** Command Output Descriptions*

Field	Description
Crypto Vendor Policy Name	Specifies the name of the vendor policy.
Example: VID IKESA2 vendor template v2 precedence 2	Specifies the vendor policy, associated vendor template, vendor ID, and Precedence.



CHAPTER 21

IPSec Manager Support on Demux DPC2 cards

Feature Summary and Revision History

Summary Data

Applicable Product(s) or Functional Area	IPSec (IKEv1/IKEv2 ACL Mode)
Applicable Platform(s)	ASR 5500 (DPC2)
Default Setting	Disabled - Configuration Required
Related Changes in This Release	Not Applicable
Related Documentation	*Command Line Interface Reference*

Revision History

Revision Details	Release
First introduced.	21.13

Feature Description

With 21.13 release, Crypto processing for Crypto map has moved to demux card. As there is no session manager spawned in it, the under-utilized cores are used for IPSec traffic processing without affecting user data processing on non-Demux DPC2 cards. A CLI command is available to control the spawning of IPSec manager in the Demux card.

How it Works

Limitations

This section describes the known limitations for IPSec Manager Support on Demux DPC2 cards.

- This feature is supported only for DPC2.
- This feature is applicable for ACL mode for IKEv1, IKEv2-v4, and IKEv2-v6.
- **ipsec-on-demux** CLI command does not work when Demux on MIO is enabled.
- Each IPSec manager will serve only eight Crypto maps.
- Maximum IPSec managers supported per CPU is one, and maximum of three at card level for IKEv1 and IKEv2 separately.
- If more than 24 active Crypto maps are configured, then the fourth IPSec manager and subsequent IPSec managers are spawned on the non-Demux DPC2 card. New IPSec manager handles original number of Crypto maps (that is, 150).
- If a new Crypto map without the CLI is configured, then it spawns or reuse IPSec managers present on a non-Demux DPC2 card. New IPSec manager does not reuse or spawn IPSec managers already running on the demux card.
- If a new Crypto map with the CLI is configured, then it will spawn or re-use IPSec managers present on the Demux DPC2 card. It does not reuse or spawn IPSec managers already running on the non Demux card.
- When the limit of 24 Crypto maps that is configured is exceeded, then the subsequent new Crypto map (whether configured or not) is served by IPSec managers present in the Demux DPC2 cards.
- If any of the Crypto maps with CLI that are served by IPSec managers in Demux is removed and then any new or same map is added with CLI again, it will serve the IPSec manager in Demux.
- When the context is removed, IPSec managers are also removed. This creates room for new crypto maps with CLI and IPSec managers with the limit of 24 maps and 3 IPSec managers on demux card.
- Only one IPSec manager is spawned in each core of the Demux Card, due to this maximum of three IPSec manager are spawned in Demux DPC2 card.
- Each IPSec manager running on Demux card serve as a maximum of eight active Crypto maps.
- Demux on MIO card is not supported.
- To spawn IPSec managers on Demux, **ipsec-on-demux** must be configured before associating it with the interface.
- Every new context spawns new IPSec manager if a new Crypto Map is added under it. If there are 3 contexts, then individual contexts must not have more than 8 Crypto maps to utilize optimum resources. If an individual context have more than 8 Crypto maps then not all the 24 Crypto maps will serve by IPSec managers running on Demux card.
- IKEv1 and IKEv2 spawn IPSec managers independently, IPSec managers share the same resources if it is used in combination. Therefore it is recommended to use either IKEv1 or IKEv2 for Demux card.

- The CLI is visible in DPC1 platform, but it is not be supported.
- Because each Crypto group spawns two IPSec managers as peers are different in primary and secondary IKEv1 maps, only 8 sets of Crypto groups are allowed.
- Not more than eight Crypto maps can be used with same SRC and DST IP address, as they are served by the same IPSec manager and each IPSec manager on demux has limitation of 8 Crypto maps.

Configuring IPSec Manager Support on Demux DPC2 cards

This section provides information on the CLI commands to configure IPSec Manager on Demux of DPC2.

Enabling IPSec Manager Spawning

Use the following configuration to enable spawning of IPSec manager for a Crypto map on the Demux Card.

Important It is mandatory to configure **require demux processing-card** and **require session recovery** commands before configuring **ipsec-on-demux**command.

```
configure
   context context_name
      crypto map  policy_name ipsec-ikev1
         ipsec-on-demux
         end
```

NOTES:

- **no**: Disables the spawning of IPSec manager for Crypto map on Demux Card.

  Important If the configuration is removed using **no ipsec-on-demux** option, then this Crypto map must be removed and added again for this configuration to work.

- **ipsec-on-demux**: Enables the spawning of IPSec manager for a Crypto map on Demux Card.

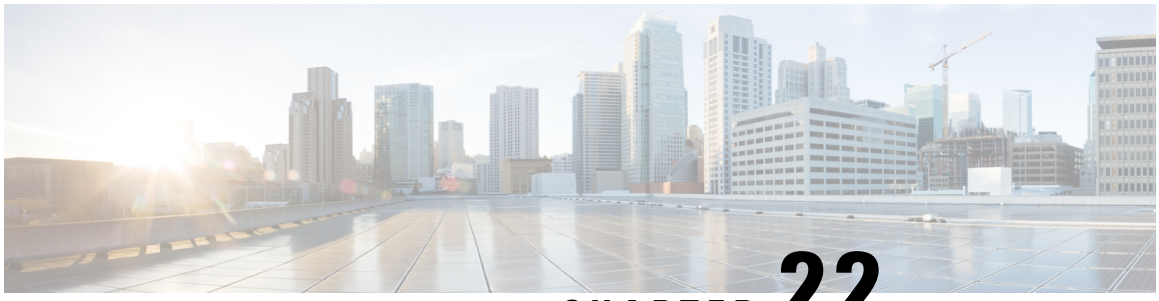
Monitoring and Troubleshooting

This section provides information regarding show commands available to monitor and troubleshoot the IPSec Manager Support on Demux DPC2 cards.

Show Commands and Outputs

show crypto managers *ipsec_manager_instance*

The output of this command includes the "Demux Card" field.



CHAPTER 22

IPSec Packet Capture (PCAP) Trace Support

Feature Information

Summary Data

Status	New Feature
Introduced-In Release	21.2
Modified-In Release(s)	Not Applicable
Applicable Product(s)	ePDG, IPSec
Applicable Platform(s)	ASR 5500 vPC-SI vPC-DI
Default Setting	Disabled
Related CDETS ID(s)	CSCvc75540
Related Changes in This Release	Not Applicable
Related Documentation	IPSec Reference Guide Command Line Interface Reference Guide

Revision History

Important Revision history details are not provided for features introduced before release 21.2.

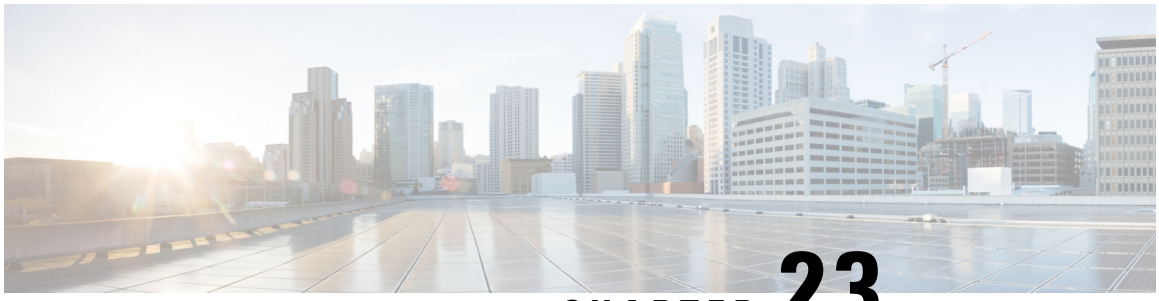
Revision Details	Release	Release Date
New in this release.	21.2	April 27, 2017

Feature Description

The PCAP trace output of the **monitor subscriber** and **monitor protocol** commands can be exported as a hexdump file for IPSec (ePDG) sessions. The hexdump capture can be stored in a text file in a hard disk, and later transferred to an external server through SFTP using a PUSH or PULL method.

PCAP trace and hexdump file collection for **IPSec IKEv2 Subscriber** can be enabled or disabled under the **monitor protocol** and **monitor subscriber** commands.

For more information on PCAP Trace, Refer the *Packet Capture (PCAP) Trace* chapter in the *ASR 5500 System Administration Guide*.



CHAPTER 23

IPSec Slow Path Data Plane

This chapter describes the following topics:

Feature Summary and Revision History

Summary Data

Applicable Product(s) or Functional Area	IPSec (ACL Mode)
Applicable Platform(s)	VPC-SI
Feature Default	Disabled - Configuration Required
Related Changes in This Release	Not applicable
Related Documentation	• *Command Line Interface Reference* • *IPSec Administration Guide*

Revision History

Revision Details	**Release**
First introduced.	21.9

Feature Description

Once this feature is enabled which is CLI controlled, IPSec data plane operations are handled in slow path IPSec Manager. For each IKEv1/IKEv2 Crypto Map one IPsec Manager is spawned. Once maximum limit

is reached, the new Crypto Map starts reusing existing IPSec Manager. The CLI command controlling this feature must be configured during the boot time.

Limitations

This section describes the known limitations for IPSec Software Data Path feature

- Transport mode IPsec is not supported.
- Associating IPSec Software Data Path to Virtual Routing and Forwarding (VRF) is not supported.

Configuring IPSec Software Data Path

This section provides information on CLI commands available in support of this feature.

Configuring IPSec Software Data Path

Use the following configuration to enable IPSec Software Data Path for IKEv1/IKEv2 Maps.

```
configure
 [ no ] require crypto { ikev1-acl software | ikev2-acl software }
 end
```

NOTES:

- **require crypto**: Enables Crypto related parameters.
- **ikev1-acl**: Configures IKEv1-ACL IPSec sessions.
- **ikev2-acl**: Configures IKEv2-ACL IPSec sessions.
- **software**: IPSec Manager performs encryption, decryption and DH calculations.
- **no**: Disables IPSEC Manager from encryption, decryption and DH calculations.

Monitoring and Troubleshooting

This section provides information on the show commands available to support IPSec Software Data Path for IKEv1/IKEv2 Maps.

Show Commands and Outputs

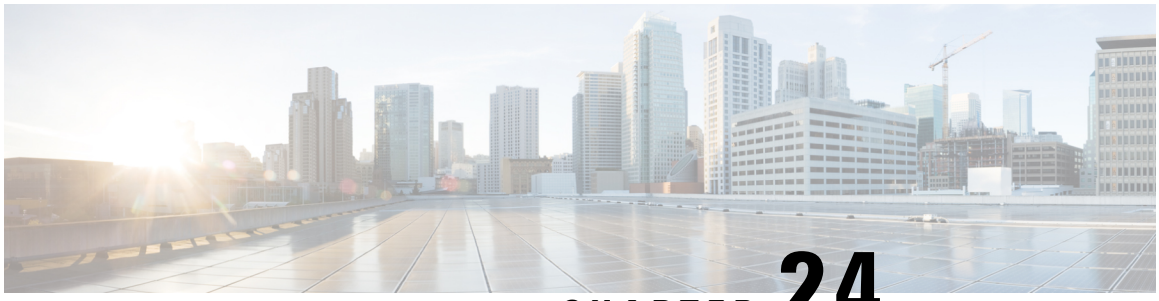
This section provides information on show commands and their corresponding outputs for the IPSec Software Data Path for IKEv1/IKEv2 Maps:

show configuration

The output of this command includes the following fields:

- require ikev1-acl software

- require ikev2-acl software



CHAPTER 24

Limit Max Number of IKEv1 IPSEC Managers within a Context

Feature Summary and Revision History

Summary Data

Applicable Product(s) or Functional Area	IPSec
Applicable Platform(s)	ASR 5500
Feature Default	Disabled – Configuration Required
Related Changes in This Release	Not applicable
Related Documentation	• *Command Line Interface Reference* • *IPSec Administration Guide*

Revision History

Revision Details	Release
First introduced.	21.9.4

Feature Changes

If the maximum number of IKEV1 IPSec managers are already spawned in the system, then a new context with crypto map cannot be added as crypto map in a new context requires a new IPSec manager. A new CLI is introduced to limit the spawning of IKEv1 IPSec managers within a context so that the customer can limit

the number of IPSec managers in each existing context and manage the spawning of new IPSec managers in new context if required later.

Previous Behavior: Any number of number of IPSec managers can be spawned within a context (depends on how many crypto maps are configured).

New Behavior: New CLI **limit ipsecmgr ikev1 max** enables to limit number of IPSec managers within a context.

Customer Impact: Flexibility to manage resources under multiple contexts and avoid the situation where no more crypto maps are available to add in the system when new context is added.

Important This feature works only when **require crypto ikev1-acl software** (Software Datapath feature) is enabled at Global Configuration Mode.

Command Changes

This section describes the CLI configuration required to enable limiting of IPSec managers spawning within a context.

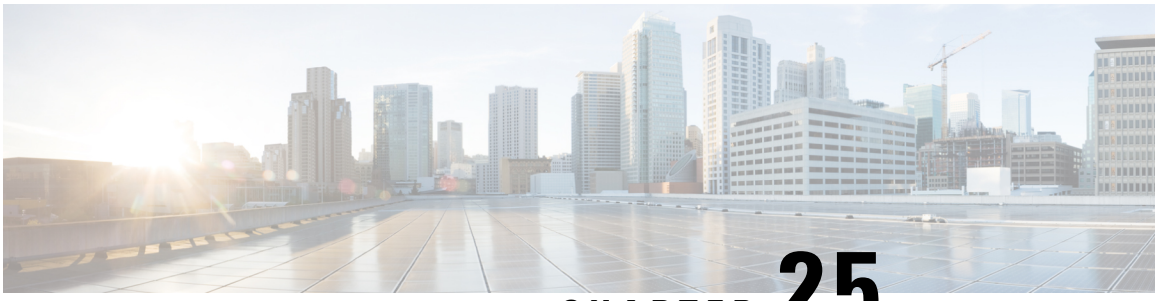
limit ipsecmgr ikev1 max

Use the following configuration to limit the parameter for this context.

```
configure
  context context_name
    limit ipsecmgr ikev1 maxmax_value
    default limit ipsecmgr ikev1 max
    end
```

NOTES:

- **default** : Sets/Restores default value assigned for specified parameter.
- **limit** : Limits the parameter for this context.
- **ipsecmgr** : To limit ipsecmgr manager settings.
- **ikev1** : Specifies IKEv1 tasks.
- **max** *max_value* : Specifies maximum ipsecmgr IKEv1 tasks. *max_value* must be an integer from 1 to 176.



CHAPTER 25

Duplicate Session Detection

This chapter describes how to configure IPSec to maintain only one IKE-SA per remote ID (peer IKE_ID). This feature is only support for the Wireless Security Gateway (WSG) service.

The following topics are discussed:

Process Overview

RFC 5996 does not restrict the creation of multiple IKE SAs having the same remote IKE_ID (not necessarily from the same peer). The remote IKE_ID specifies the remote peer ID: IDi when the gateway is the responder, and IDr when the gateway is the initiator. In such implementations, a new IKE_SA is created for every IKE_SA_INIT/IKE_AUTH exchanges, unless INITIAL_CONTACT is indicated. If an IKE_AUTH is received with INITIAL_CONTACT, the node is expected to delete all IKE_SAs having the same authenticated identity.

Important The StarOS IPSec stack does not currently support INITIAL_CONTACT.

When enabled via the StarOS **duplicate-session-detection** command in a WSG service, only one IKE_SA is allowed per remote IKE_ID. This feature is supported for WSG service, both RAS (Remote Access Service) and S2S (Site-to-Site) tunnel types.

The following sequence of figures indicates how StarOS IPSec managers handle duplicate IKE_SA scenarios when this feature is enabled.

Figure 27: No Duplicate Session Found

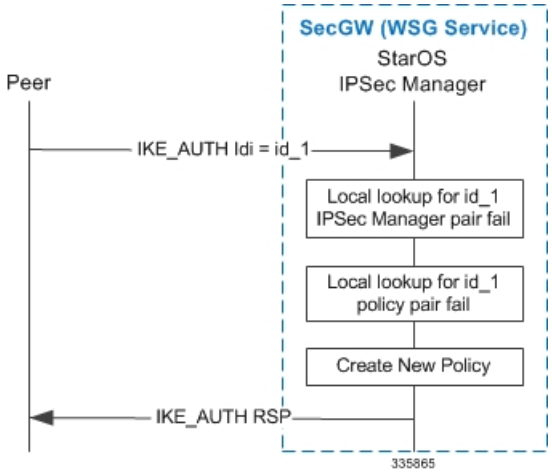


Figure 28: Duplicate Session Found in Same StarOS IPSec Manager

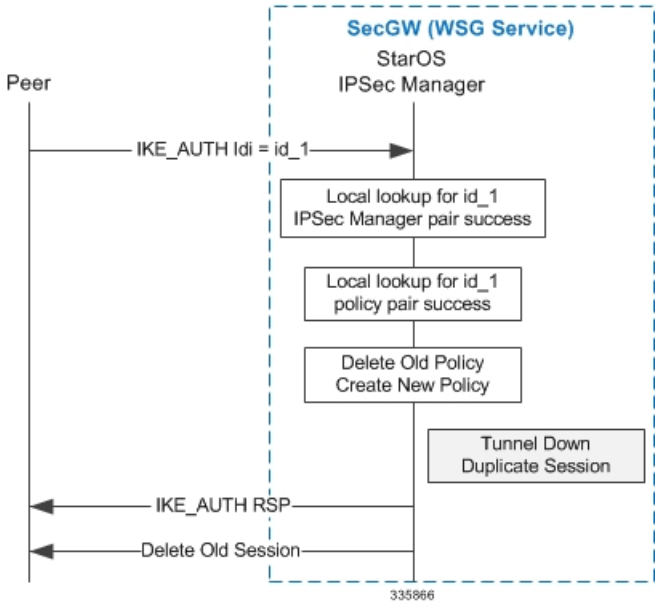


Figure 29: Duplicate Session Found in Different StarOS IPSec Manager

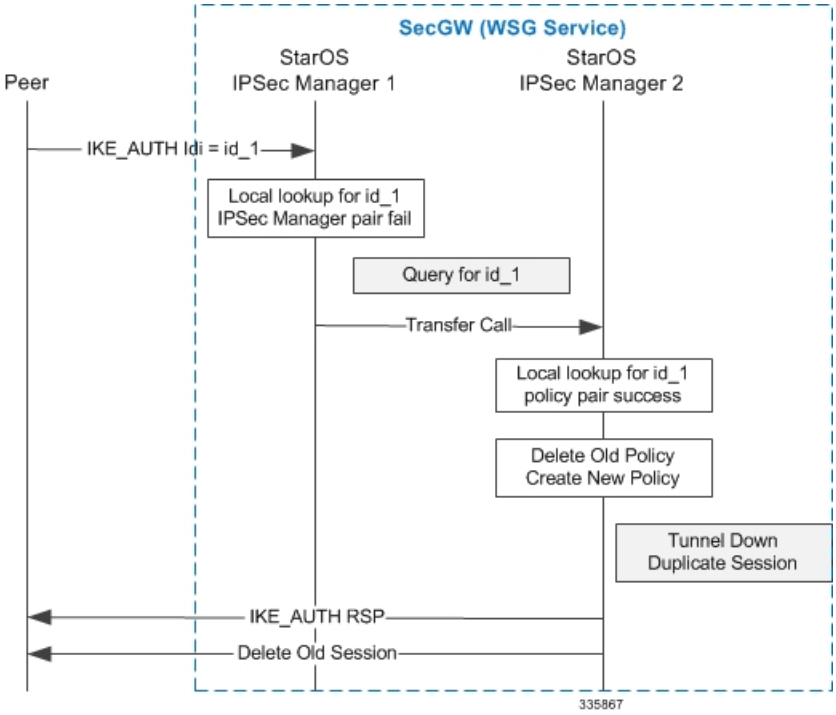


Figure 30: Duplicate Session Found When SecGW (WSG Service) is the Initiator

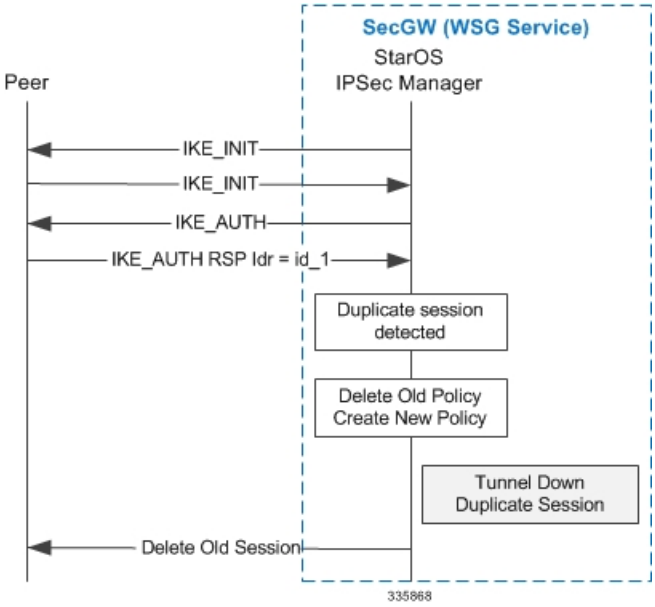
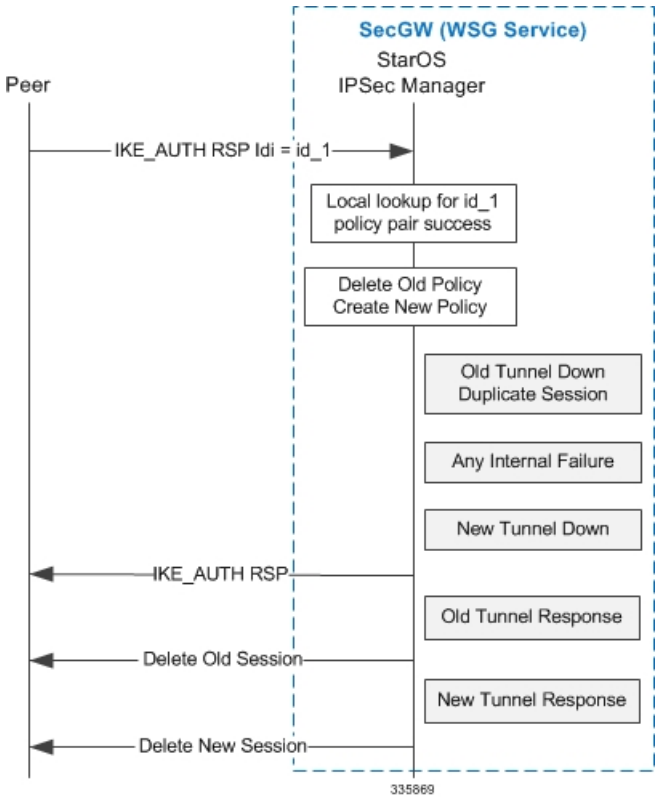


Figure 31: Internal Failures Encountered After Duplicate Session is Detected<



Configuring Duplicate Session Detection

Use the following example to enable duplicate session detection:

```
configure
   context wsg_ctx_name
      wsg-service wsg_srvc_name
         duplicate-session-detection
         end
```

Notes:

- *wsg_ctx_name* is the StarOS context associated with a WSG service.
- *wsg_srvc_name* is the name of the WSG service in the current context that you want to configure for duplicate session detection.
- Any changes made to a WSG service require that the service must be restarted to apply any changed parameters. You restart the service by unbinding and binding the IP address to the service context.
- For more information on parameters, see the *WSG Service Configuration Mode Commands* chapter in the *Command Line Interface Reference*.
- By default duplicate session detection is disabled.

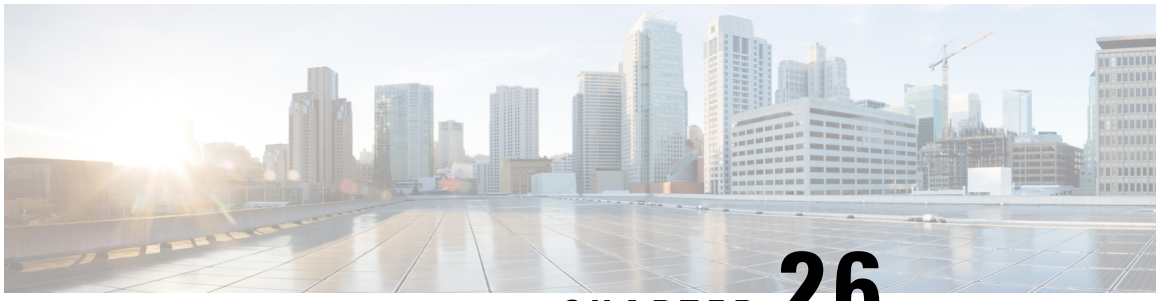
Verifying the Duplicate Session Detection Configuration

Enter the following Exec mode command for the WSG context to display and verify your duplicate session detection configuration:

show wsg-service all *wsg_srvc_name*

The output of this command will include the following parameter:

```
Duplicate-session-detection : Enabled/Disabled
```



CHAPTER 26

Extended Sequence Number

This chapter describes support of 64-bit Extended Sequence Numbers (ESNs) for Encapsulating Security Payload (ESP) and Authentication Header (AH) packets. ESN is defined in RFC 4304.

This chapter includes the following sections:

Overview

ESN for ikev2

Every IKE message contains a Message ID (sequence number) as part of its fixed header. This sequence number is a monotonically increasing integer (incremented by 1 for every packet sent) used to match up requests and responses, and to identify retransmissions of messages. The sequence is a 32-bit integer which is zero for the first IKE request in each direction.

Sequence numbers are cryptographically protected to protect against message replays. In the unlikely event that Message IDs grow too large to fit in 32 bits (0xFFFFFFFF = 4294967295 packets), the IKE_SA must be closed. Rekeying an IKE_SA resets the sequence numbers.

RFC 4304 outlines support for a 64-bit ESN implemented for ikev2. The ESN transform is included in an ikev2 proposal used in the negotiation of IKE SAs as part of the IKE_SA_INIT exchange.

The ESN transform has the following meaning:

- A proposal containing one ESN transform with value 0 means "do not use extended sequence numbers".
- A proposal containing one ESN transform with value 1 means "use extended sequence numbers".
- A proposal containing two ESN transforms with values 0 and 1 means "I support both normal and extended sequence numbers, you choose". This case is only allowed in requests; the response will contain only one ESN transform.

In most cases, the exchange initiator will include either the first or third alternative in its SA payload. The second alternative is rarely useful for the initiator: it means that using normal sequence numbers is not acceptable (so if the responder does not support ESNs, the exchange will fail with NO_PROPOSAL_CHOSEN.

Including the ESN transform is mandatory when creating ESP/AH SAs.

StarOS Support for ESN

StarOS supports ESN for ESP packets using ikev2 negotiation; ESN is not supported for ikev1. The configuration and processing sequence is as follows:

- Enable ESN in an IPSec transform set via a StarOS CLI command.
- Negotiate ESN (IPSec Domain of Interpretation (DOI) for Ikev2.
  - Send ESN in the proposal based on configuration.
  - Accept and process ESN in the proposal based on configuration.
- Configure data-path to use ESN.
- Read and checkpoint ESN.

Important ESN is only supported on ASR 5500 and ASR 9000 Virtualized Services Modules (VSMs). It is not supported on the VPC-SI.

Configuring ESN Support

The IPSec Transform Set Configuration mode includes an **esn** command that enables ESN support.

```
configure
   context ipsec_ctx_name
      ipsec transform-set tset_name
         esn
         end
```

Notes:

- *ipsec_ctx_name* is the StarOS context associated with IPSec.
- *tset_name* is the name of the transform set in the current context that you want to configure for ESN.
- For more information on parameters, see the *IPSec Transform Set Configuration Mode Commands* chapter in the *Command Line Interface Reference*.
- By default ESN support is disabled.
- Enabling the **esn** command is the equivalent of sending ESN Transform = 0 and 1; supports both 32-bit and 64-bit sequence numbers. If the **esn** command is not enabled, support is only 32-bit sequence numbers (default behavior).

Verifying ESN Configuration

The following Exec mode **show** commands display ESN configuration parameters.

show crypto ipsec transform-set

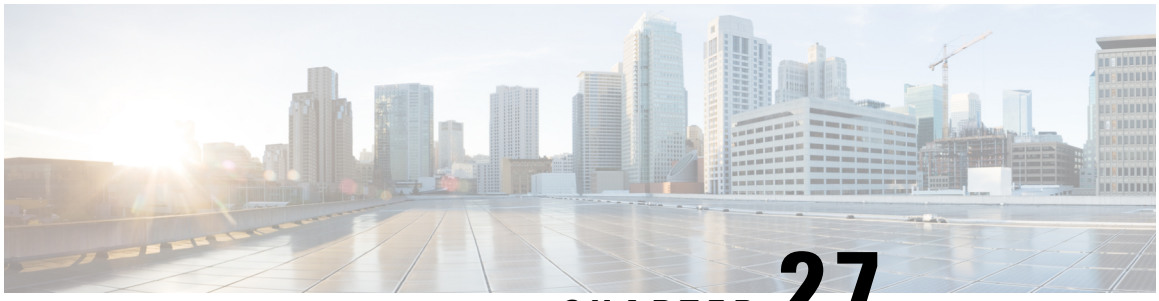
This command displays the IPSec transform set parameters as configured in a specific context and includes ESN status. A sample output appears below:

```
show crypto ipsec transform-set tsela
IKEv2 IPSec Transform-Set tselsa :
    Cipher      : aes-cbc-128
    HMAC        : sha1-96
    DH Group    : none
    Encaps Mode : TUNNEL
    ESN         : Enabled/Disabled
```

show crypto template

This command displays ESN status under IPSec SA Payload. A sample output appears below.

```
show crypto template tag foo-sa0
  IPSec SA Payload 1/2 (SIP Address - Static Pool)
    Name : foo-sa0
    IPSec SA Transform 1/1
      Transform Set: tselsa
        Protocol: esp
        Encryption Cipher: aes-cbc-128
        Hashed Message Authentication Code: sha1-96
        Diffie-Hellman Group: none
        ESN : Enabled
```



CHAPTER 27

Security Gateway as Initiator

This chapter describes how to configure a Security Gateway (SecGW) running on an ASR 9000 Virtualized Services Module (VSM) as an initiator of an IKEv2 session.

This chapter includes the following sections:

Overview

By default SecGW (WSG service) only responds to a setup request for an IKEv2 session. However, an SecGW can also be configured to initiate an IKEv2 session setup request when the peer does not initiate a setup request within a specified time interval.

Important This functionality is only applicable for site-to-site (S2S) based tunnels within a WSG service. For remote access tunnels the peer is always the initiator.

Responder-Initiator Sequence

The following is the general event sequence for an SecGW acting as an initiator.

1. The SecGW waits for the peer to initiate a tunnel within a configurable time interval during which it is in responder mode. The default responder mode interval is 10 seconds.
2. Upon expiry of the responder mode timer, the SecGW switches to initiator mode for a configurable time interval. The default initiator mode interval is 10 seconds.
3. The SecGW retries the call if there is no response from the peer during the initiator mode interval.
4. When the SecGW is in initiator mode and the peer does not respond to the IKE messages or fails to establish the call, SecGW reverts to responder mode and waits for the peer to initiate the IKEv2 session.
5. If call creation is successful, the SecGW stops initiating any further calls to that peer.
6. If the SecGW and peer initiate a session call simultaneously (possible collision), the SecGW defers to the peer initiated call and drops any incoming packets.

When the SecGW as initiator feature is enabled, the SecGW only supports up to 1,000 peer addresses. This restriction is applied when configuring a crypto peer list. See Create a crypto peer-list, on page 186.

Limitations

The following limitations apply when the SecGW as initiator feature is enabled:

- The SecGW will only support up to 1,000 peers. This restriction is applied when configuring a crypto peer list.
- SecGW will not support the modification of an IPv4/IPv6 peer list on the fly (call sessions in progress). The modification will be allowed only after all the calls are removed.

The SecGW does support wild card peer address provisioning along with subnets.

Configuring SecGW as Initiator

The following is the general sequence for configuring this feature:

- Create a crypto peer-list, on page 186.
- Configure the Peer List in the WSG Service, on page 186.
- Configure Initiator Mode and Responder Mode Durations, on page 187.

See the *Command Line Interface Reference* for complete information about the commands described below.

Create a crypto peer-list

The CLI command sequence for creating a crypto peer list is shown below.

```
configure
  context context_name
    crypto peer-list { ipv4 | ipv6 } peer_list_name
      address peer_address
      exit
```

Notes:

- *peer_list_name* is specified as an alphanumeric string of 1 through 32 characters.
- Running the **crypto peer-list** command moves you to the Peer List Configuration mode where you have access to the **address** command.
- Repeat the **address** *peer_address* command to add up to 1,000 peer IP addresses. The IP addresses in the list can only be entered in either IPv4 or IPv6 notation, depending on the address type specified when the list was created.
- Use the **no address** *peer_address* command to remove a peer address from the peer list.

Configure the Peer List in the WSG Service

The following CLI command sequence configures the previously created peer list for use in the WSG service.

```
configure
  context wsg_ctxt_name
    wsg-service wsg_service_name
      peer-list peer_list_name
      exit
```

Notes:

- *peer_list_name* must have been previously configured as described in Create a crypto peer-list, on page 186.
- Use the **no peer-list** command to remove the peer-list and disable the SecGW as initiator feature.
- Any changes made to a WSG service require that the service must be restarted to apply any changed parameters. You restart the service by unbinding and binding the IP address to the service context.

Configure Initiator Mode and Responder Mode Durations

When a peer list has been configured in the WSG service, the initiator and responder mode timer intervals each default to 10 seconds. The SecGW will wait for 10 seconds in the responder mode for a peer session initiation request before switching to the initiator mode and waiting 10 seconds for a peer response.

You can change the default settings for the initiator and/or responder mode intervals using the following CLI command sequence.

```
configure
  context wsg_ctxt_name
    wsg-service wsg_service_name
      initiator-mode-duration seconds
      responder-mode-duration seconds
      exit
```

Notes:

- *seconds* is an integer from 5 through 250.

Restrictions

The following restrictions apply when configuring an SecGW as an Initiator:

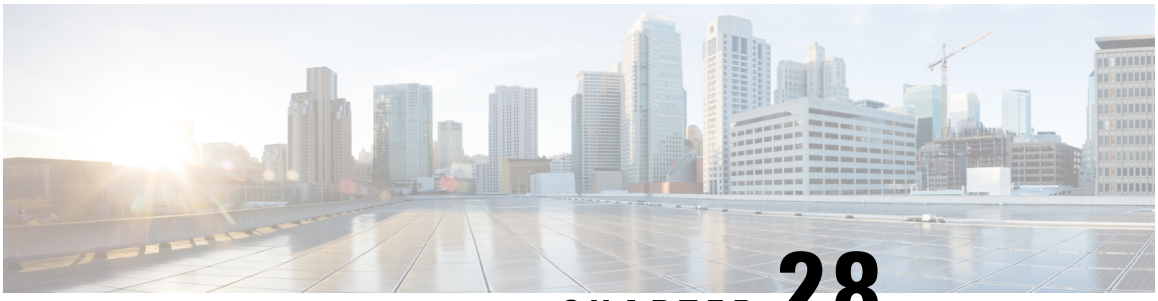
- The **peer-list** *peer_list_name* command is only executed if the deployment mode for WSG service is **site-to-site**, and the bind address matches with the peer list address type (IPv4 or IPv6).
- You cannot change the WSG service deployment-mode if **peer-list** *peer_list_name* is enabled under the service. You will be prompted to remove the peer list before changing the mode.
- A maximum of 1,000 peer IP addresses can be added to the peer list via the Peer List Configuration mode **address** command.
- WSG service address binding is not allowed if a peer list is configured and both address types do not match. An error message is generated if they do not match.
- An IPv4 or IPv6 peer list cannot be modified if **peer-list** *peer_list_name* is enabled under the WSG service.

Verifying the SecGW as Initiator Configuration

Run the **show wsg-service** CLI command to display the current crypto peer list configuration. A sample output of this command appears below.

```
show wsg-service all
Service name: wsg1
  Context: wsg
  ...
  peer list : peer01
```

```
Initiator mode duration : 10 seconds
Responder mode duration : 10 seconds
```



CHAPTER 28

User Equipment Identity in IKE_AUTH Message

The following topics are discussed:

Feature Description

Overview

On untrusted WLAN networks that support Mobile Equipment Identity signalling, ePDG can request the subscriber's User Equipment (UE) for the International Mobile Equipment Identity (IMEI) or IMEI SV (Software Version) information, when the UE does not share this information in the first IKE_AUTH_REQ message in the configuration attributes. On receiving the IMEI or IMEI SV information from the UE, ePDG can share this information with the AAA server in the Diameter EAP Request (DER) message over the SWm interface, and in the ME Identity (MEI) IE with P-GW in the second Create Session Request (CSR) message over the S2b interface.

How UE Identity in IKE_AUTH Message Works

Architecture

During IKEv2 authentication and security association (SA) establishment for UICC devices, when the UE does not share the IMEI or IMEI SV information in the first IKE_AUTH_REQ message, ePDG can request the UE for this information. ePDG includes a DEVICE_IDENTITY notify payload in the IKE_AUTH_RESP message to UE. Based on the availability of IMEI or IMEI SV information, the UE includes the value in the DEVICE_IDENTITY attribute with the Identity Type field set to IMEI or IMEI SV. The UE then shares this information with ePDG in the second IKE_AUTH_REQ message. The structure of the DEVICE_IDENTITY notify payload is as defined in *3GPP TS 24.302*.

ePDG can be configured to request the UE for the IMEI or IMEISV information using the **notify-payload device-id** command under the Crypto Template Configuration Mode. For more configuration information, refer the configuration section of this chapter.

For non-UICC devices, ePDG will not request for the IMEI or IMEI SV information from the UE for single exchange authentication methods like certificate-based authentication. For other authentication methods that uses multiple IKE_AUTH exchanges, the behaviour to request for the IMEI or IMEI SV information is the same as that of UICC devices.

Standards Compliance

This feature complies with the following standards:

- **3GPP TS 24.302**: "3rd Generation Partnership Project; Technical Specification Group Core Network and Terminals; Access to the 3GPP Evolved Packet Core (EPC) via non-3GPP access networks; Stage 3"

Configuring UE Identity in IKE_AUTH Message

Use the following configuration to enable ePDG to request the UE for the IMEI or IMEI SV information using the DEVICE_IDENTITY notify payload:

```
config
    context context_name
        crypto template template_name ikev2-dynamic
           notify-payload device-id
           end
```

Notes:

- Use the **no notify-payload device-id** command to disable the configuration.
- Use the **default notify-payload device-id** command to restore the configuration to its default value.
- **Default:** Enabled

Monitoring and Troubleshooting

Show Command(s) and/or Outputs

show crypto statistics ikev2

The following fields are available in the output of the **show crypto statistics ikev2** command in support of this feature:

```
Total IKEv2 Notify Statistics:
    Device ID Req Sent:   0
    Device ID Rsp Rcvd:   0
```

Table 28: show crypto statistics ikev2 Command Output Descriptions

Field	Description
Total IKEv2 Notify Statistics:	
Device ID Req Sent	Total number of IKEv2 Notify payloads sent (device id).
Device Identity Rsp Rcvd	Total IKEv2 Notify payloads received (device id).

show crypto template

The following field is available in the output of the **show crypto template** command in support of this feature:

```
IKEv2 Notify Payload:
    Device Identity: Enabled [Default]
```

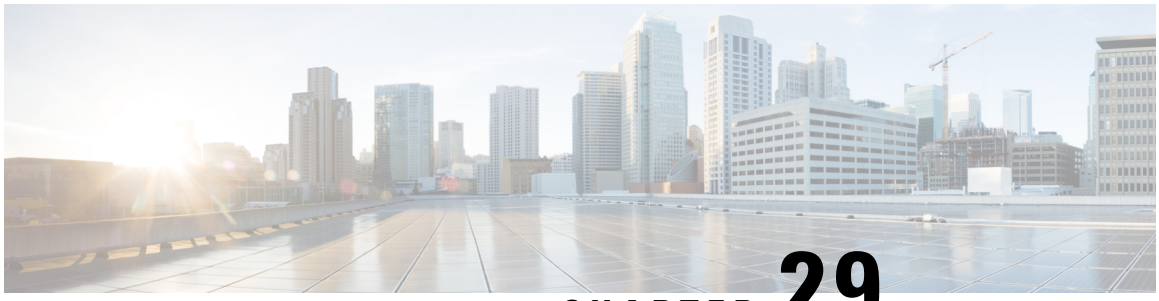
Table 29: show crypto template Command Output Descriptions

Field	Description
IKEv2 Notify Payload:	
Device Identity	Indicates if ePDG is configured to request for device identity in the IKEv2 Notify payload message.

Bulk Statistics

The following bulks statistics included in the system schema support this feature:

Variable	Description	Data Type
ikev2-notifpaysent-deviceid	**Description:** Total number of IKEv2 Notify payloads sent (device id). **Triggers:** Increments when ePDG sends a Device Identity Notify Payload. **Availability:** ePDG Service **Type:** Counter	Int32
ikev2-notifpayrecv-deviceid	**Description:** Total IKEv2 Notify payloads received (device id). **Triggers:** Increments when ePDG receives a Device Identity Notify Payload. **Availability:** ePDG Service **Type:** Counter	Int32



CHAPTER 29

Monitor CPU Crypto Core Utilization

Feature Information

Summary Data

Status	New Feature
Introduced-In Release	21.2
Modified-In Release(s)	Not Applicable
Applicable Product(s)	ePDG
Applicable Platform(s)	ASR 5500 VPC-SI VPC-DI
Default Setting	Disabled
Related CDETS ID(s)	CSCvc38683
Related Changes in This Release	Not Applicable
Related Documentation	IPSec Reference Guide Command Line Interface Reference Guide

Revision History

Revision Details	Release	Release Date
New in this release.	21.2	April 27, 2017

Feature Description

This feature provides mechanisms to monitor the crypto-specific CPU cores using the StarOS threshold framework on the ASR 5500 (DPC, DPC2). Alarms and bulk statistics enable the crypto core utilization to be monitored. Packet drops can thus be prevented by taking preventive actions when the safe limit is exceeded. The high and low thresholds for the alarm can be configured using the **threshold cpu-crypto-cores-utilization** command.

For more information, refer the configuring and monitoring sections of this chapter.

Configuring Crypto Core Utilization Thresholds

Use the following configuration to set the threshold upper and lower limits, and the polling interval for crypto core utilization:

```
config
   threshold cpu-crypto-cores-utilization high_thresh [ clear low_thresh ]
   threshold poll cpu-crypto-cores-utilization interval duration
   end
```

Notes:

- Use the **threshold cpu-crypto-cores-utilization** *high_thresh* **[clear** *low_thresh* **]** command to specify the alarm or alert thresholds for crypto core utilization.
  - The measured value is the sum of the most recent system and IRQ core usage.
  - *high_thresh* and *low_thresh* must be an integer from 0 through 100.
- Use the **threshold poll cpu-crypto-cores-utilization interval** *duration* command to specify the polling interval after which the crypto core utilization is measured.
  - *duration* must be an integer from 30 through 60000.
  - Use the **default threshold poll crypto-cores-utilization interval** command to set the threshold polling interval to its default value.
  - Default polling interval: 300 seconds

Monitoring and Troubleshooting Crypto Core Utilization

Show Command(s) and/or Outputs

The show command(s) in this section are available in support of this feature:

show cpu

The output of the **show cpu info card** *card_num* **[cpu** *cpu_num* **] crypto-cores** will display statistics about the CPU crypto core usage:

The following is a sample output of this command:

```
Card 10, CPU 0:
  Status                : Active, Kernel Running, Tasks Running
  Load Average          : 0.14, 0.17, 0.13 (0.86 max)
  Total Memory          : 65536M (32768M node-0, 32768M node-1)
  Kernel Uptime         : 0D 0H 8M
  Last Reading:
    CPU Usage All       : 0.2% user, 0.1% sys, 0.0% io, 0.0% irq, 99.7% idle
      Node 0            : 0.1% user, 0.0% sys, 0.0% io, 0.0% irq, 99.8% idle
        Core 26         : 0.0% user, 0.0% sys, 0.0% io, 0.0% irq, 100.0% idle
        Core 27         : 0.0% user, 0.0% sys, 0.0% io, 0.0% irq, 100.0% idle
        Core 28         : 0.0% user, 0.0% sys, 0.0% io, 0.0% irq, 100.0% idle
        Core 29         : 0.0% user, 0.0% sys, 0.0% io, 0.0% irq, 100.0% idle
        Core 30         : 0.0% user, 0.1% sys, 0.0% io, 0.0% irq, 99.9% idle
        Core 31         : 0.0% user, 0.0% sys, 0.0% io, 0.0% irq, 100.0% idle
        Core 32         : 0.0% user, 0.0% sys, 0.0% io, 0.0% irq, 100.0% idle
        Core 33         : 0.0% user, 0.0% sys, 0.0% io, 0.0% irq, 100.0% idle
      Node 1            : 0.2% user, 0.1% sys, 0.0% io, 0.0% irq, 99.6% idle
        Core 38         : 0.4% user, 0.2% sys, 0.0% io, 0.0% irq, 99.4% idle
        Core 39         : 0.0% user, 0.0% sys, 0.0% io, 0.1% irq, 99.9% idle
        Core 40         : 0.0% user, 0.1% sys, 0.0% io, 0.0% irq, 99.9% idle
        Core 41         : 0.0% user, 0.0% sys, 0.0% io, 0.0% irq, 100.0% idle
        Core 42         : 0.0% user, 0.0% sys, 0.0% io, 0.0% irq, 100.0% idle
        Core 43         : 0.1% user, 0.1% sys, 0.0% io, 0.0% irq, 99.8% idle
        Core 44         : 0.0% user, 0.0% sys, 0.0% io, 0.0% irq, 100.0% idle
        Core 45         : 0.0% user, 0.0% sys, 0.0% io, 0.0% irq, 100.0% idle
  5-Minute Average:
    CPU Usage All       : 0.2% user, 0.1% sys, 0.0% io, 0.0% irq, 99.7% idle
      Node 0            : 0.2% user, 0.1% sys, 0.0% io, 0.0% irq, 99.7% idle
        Core 26         : 0.0% user, 0.1% sys, 0.0% io, 0.0% irq, 99.9% idle
        Core 27         : 0.0% user, 0.0% sys, 0.0% io, 0.0% irq, 100.0% idle
        Core 28         : 0.0% user, 0.1% sys, 0.0% io, 0.0% irq, 99.9% idle
        Core 29         : 0.0% user, 0.1% sys, 0.0% io, 0.0% irq, 99.9% idle
        Core 30         : 0.0% user, 0.1% sys, 0.0% io, 0.0% irq, 99.9% idle
        Core 31         : 0.0% user, 0.0% sys, 0.0% io, 0.0% irq, 100.0% idle
        Core 32         : 0.0% user, 0.1% sys, 0.0% io, 0.0% irq, 99.9% idle
        Core 33         : 0.0% user, 0.0% sys, 0.0% io, 0.0% irq, 100.0% idle
      Node 1            : 0.2% user, 0.1% sys, 0.0% io, 0.0% irq, 99.7% idle
        Core 38         : 0.1% user, 0.1% sys, 0.0% io, 0.0% irq, 99.7% idle
        Core 39         : 0.0% user, 0.0% sys, 0.0% io, 0.1% irq, 99.9% idle
        Core 40         : 0.2% user, 0.1% sys, 0.0% io, 0.0% irq, 99.7% idle
        Core 41         : 0.0% user, 0.1% sys, 0.0% io, 0.0% irq, 99.9% idle
        Core 42         : 0.0% user, 0.1% sys, 0.0% io, 0.0% irq, 99.9% idle
        Core 43         : 0.0% user, 0.1% sys, 0.0% io, 0.0% irq, 99.9% idle
        Core 44         : 0.0% user, 0.0% sys, 0.0% io, 0.0% irq, 100.0% idle
        Core 45         : 0.0% user, 0.0% sys, 0.0% io, 0.0% irq, 100.0% idle
  Maximum/Minimum:
    CPU Usage All       : 0.4% user, 0.6% sys, 0.0% io, 0.0% irq, 99.0% idle
      Node 0            : 0.5% user, 0.8% sys, 0.0% io, 0.0% irq, 98.7% idle
        Core 26         : 0.3% user, 1.1% sys, 0.0% io, 0.0% irq, 98.6% idle
        Core 27         : 0.1% user, 0.3% sys, 0.0% io, 0.0% irq, 99.7% idle
        Core 28         : 0.1% user, 1.1% sys, 0.0% io, 0.0% irq, 98.9% idle
        Core 29         : 0.3% user, 2.6% sys, 0.0% io, 0.0% irq, 97.1% idle
        Core 30         : 0.2% user, 0.6% sys, 0.0% io, 0.0% irq, 99.2% idle
        Core 31         : 0.4% user, 0.2% sys, 0.0% io, 0.0% irq, 99.6% idle
        Core 32         : 0.3% user, 0.6% sys, 0.0% io, 0.0% irq, 99.2% idle
        Core 33         : 0.4% user, 0.1% sys, 0.0% io, 0.0% irq, 99.6% idle
      Node 1            : 0.5% user, 0.5% sys, 0.0% io, 0.0% irq, 98.9% idle
        Core 38         : 0.4% user, 1.4% sys, 0.0% io, 0.1% irq, 98.4% idle
        Core 39         : 0.7% user, 0.8% sys, 0.0% io, 0.2% irq, 98.5% idle
        Core 40         : 1.0% user, 1.6% sys, 0.0% io, 0.0% irq, 98.0% idle
        Core 41         : 0.2% user, 0.5% sys, 0.0% io, 0.0% irq, 99.3% idle
        Core 42         : 0.3% user, 0.7% sys, 0.0% io, 0.0% irq, 99.0% idle
```

```
Core 43         : 0.2% user, 0.5% sys, 0.0% io, 0.0% irq, 99.3% idle
Core 44         : 0.1% user, 0.1% sys, 0.0% io, 0.0% irq, 99.9% idle
Core 45         : 0.1% user, 0.5% sys, 0.0% io, 0.0% irq, 99.4% idle
```

show threshold

The following fields are available in the output of the **show threshold** command in support of this feature:

```
Threshold operation model: ALARM

Configured thresholds:
        Name:             crypto-cores-utilization
        Config Scope:     SYSTEM
        Threshold:        80%
        Clear Threshold:  10%
```

Table 30: show threshold Command Output Descriptions

Field	Description
Threshold operation model	Indicates that the threshold operation model is alarm.
Configured thresholds:	
Name	Statistics for the crypto core utilization threshold.
Config Scope	Indicates that the scope of configuration is across the system.
Threshold	Indicates the high threshold value of the crypto cores utilized, after which the alarm is generated.
Clear Threshold	Indicates the low threshold value of the crypto cores utilized, after which the alarm is cleared.

Bulk Statistics

The following bulk statistic included in the card schema support this feature.

Variable	Description	Data Type
cpu*cpu_no*-core*core_no*-coreused-crypto	**Description:** The percentage of resources on CPU <*cpu_no*> CORE <*core_no*> that are used for crypto operations. *cpu_no* must be an integer between 0 and 2. *core_no* must be an integer between 0 and 47. **Triggers:** N/A **Availability:** All **Type:** Gauge	Float

Thresholds

The following alarms are available in support of this feature:

- A **ThreshCPUCryptoCoresUtilization** alarm is generated when the crypto core utilization exceeds the configured high threshold.
- A **ThreshClearCPUCryptoCoresUtilization** alarm is generated when the crypto core utilization drops below the configured low threshold.