CHAPTER 2

IPSec to Product Feature Mapping 11
- PDSN, FA and HA 11
- GGSN, FA and HA 12
- HeNBGW, HNBGW and HSGW 13
- ePDG 14
- MME, S-GW, P-GW and SAE-GW 14
- SecGW 15

CHAPTER 3

IPSec Network Applications 17
Implementing IPSec for PDN Access Applications 17
- How IPSec-based PDN Access Configuration Works 17
- Configuring IPSec Support for PDN Access 18
Implementing IPSec for Mobile IP Applications 19
- How IPSec-based Mobile IP Configuration Works 20
- Configuring IPSec Support for Mobile IP 22
  - RADIUS Attributes for IPSec-based Mobile IP Applications 24
Implementing IPSec for L2TP Applications 25
  - How IPSec is Used for Attribute-based L2TP Configurations 25
  - Configuring Support for L2TP Attribute-based Tunneling with IPSec 26
  - How IPSec is Used for PDSN Compulsory L2TP Configuration 26
  - Configuring Support for L2TP PDSN Compulsory Tunneling with IPSec 27
  - How IPSec is Used for L2TP Configurations on the GGSN 28
  - Configuring GGSN Support for L2TP Tunneling with IPSec 29
IPSec for LTE/SAE Networks 30
  - Encryption Algorithms 30
  - HMAC Functions 30
  - Diffie-Hellman Groups 31
  - Dynamic Node-to-Node IPSec Tunnels 31
  - ACL-based Node-to-Node IPSec Tunnels 31
  - Traffic Selectors 32
  - Authentication Methods 33
  - X.509 Certificate-based Peer Authentication 33
  - Certificate Revocation Lists 35
  - Child SA Rekey Support 36
IKEv2 Keep-Alive Messages (Dead Peer Detection) 36
E-UTRAN/EPC Logical Network Interfaces Supporting IPSec Tunnels 36
IPSec Tunnel Termination 37
IPSec for Femto-UMTS Networks 38
Authentication Methods 38
Crypto Map Template Configuration 38
X.509 Certificate-based Peer Authentication 39
Certificate Revocation Lists 41
Child SA Rekey Support 42
IKEv2 Keep-Alive Messages (Dead Peer Detection) 42
IPSec Tunnel Termination 42
x.509 Certificate Configuration 42

CHAPTER 4
Transform Set Configuration 45
Process Overview 45
Configuring a Transform Set 46
Verifying the Crypto Transform Set Configuration 46

CHAPTER 5
ISAKMP Policy Configuration 47
Process Overview 47
Configuring ISAKMP Policy 48
Verifying the ISAKMP Policy Configuration 48

CHAPTER 6
Crypto Maps 49
ISAKMP Crypto Map Configuration 49
Configuring ISAKMP Crypto Maps 50
Verifying the ISAKMP Crypto Map Configuration 50
Dynamic Crypto Map Configuration 51
Configuring Dynamic Crypto Maps 52
Verifying the Dynamic Crypto Map Configuration 52
Manual Crypto Map Configuration 52
Configuring Manual Crypto Maps 53
Verifying the Manual Crypto Map Configuration 54
Crypto Map and Interface Association 54
Applying a Crypto Map to an Interface 55
Verifying the Interface Configuration with Crypto Map 55

CHAPTER 7

ANSSI Enhancements for IKEv1 and IKEv2 ACL Modes 57

Feature Description 57

Auto-delete Existing IKEv1/IKEv2 ACL Tunnels 57

Remove Weak Security Algorithms 58

Configuring ANSSI Enhancements 59

Enabling Auto-deletion of Existing IKEv1/IKEv2 ACL Tunnels 59

CHAPTER 8

Crypto Templates 61

Crypto Template Parameters 61

Crypto Template IKEv2-Dynamic Payload Parameters 62

Configuring a Crypto Template 63

Verifying a Crypto Template Configuration 64

CHAPTER 9

Service Configurations 67

FA Services Configuration to Support IPSec 67

Modifying FA Service to Support IPSec 68

Verifying the FA Service Configuration with IPSec 68

HA Service Configuration to Support IPSec 68

Modifying HA Service to Support IPSec 69

Verifying the HA Service Configuration with IPSec 69

PDSN Service Configuration for L2TP Support 69

Modifying PDSN Service to Support Attribute-based L2TP Tunneling 70

RADIUS and Subscriber Attributes for L2TP Application IPSec Support 71

Modifying PDSN Service to Support Compulsory L2TP Tunneling 71

Verifying the PDSN Service Configuration for L2T 72

LAC Service Configuration to Support IPSec 72

Modifying LAC service to Support IPSec 72

Verifying the LAC Service Configuration with IPSec 73

APN Template Configuration to Support L2TP 73

Modifying an APN Template to Support L2TP 74

Verifying the APN Configuration for L2TP 74

WSG Service Configuration to Support IPSec 74

Creating a Crypto Template to Support a SecGW 74
Creating a WSG Service 75
Verifying WSG Service Creation 76

CHAPTER 10

Redundant IPSec Tunnel Fail-over 77

Redundant IPSec Tunnel Fail-over (IKEv1) 77
  Overview 77
  Supported RFC Standard 78
  Redundant IPSec Tunnel Fail-over Configuration 78
  Configuring a Crypto Group 79
  Modifying a ISAKMP Crypto Map Configuration to Match a Crypto Group 80
  Verifying the Crypto Group Configuration 80

Dead Peer Detection (DPD) Configuration 80
  Configuring DPD for a Crypto Group 81
  Verifying the DPD Configuration 81

CHAPTER 11

IPSec Certificates 83

Multiple Child SA (MCSA) Support 83
  Overview 83
  Deployment Scenarios 84
  Call Flows 84
    Child SA Creation by Initiator 84
    Child SA Creation by Responder 86

Creating, Signing, and Configuring Certificates 86

CA Certificate Chaining 87
  Overview 87
  Deployment Scenarios 88
  StarOS as Responder 88
    Cert. Data in the Payload – Peer Cert. root CA1, StarOS Cert. Intm. CA1_1 88
    Cert. Data in the Payload – Peer Cert. Intm CA1_1, StarOS Certificate root CA1 88
  StarOS as Initiator 89
    Cert. Data in the Payload – Peer Cert. root CA1, StarOS Cert. Intm. CA1_1 89
    Cert. Data in the Payload – Peer Cert. Intm CA1_1, StarOS Certificate root CA1 89

External Interfaces 89

Certificate Management Protocol (CMPv2) 89
  Overview 89
Deployment Scenarios 90

Call Flows 92

  Initial Certification Request 92
  Initial Certification Request with Polling 92
  Enrollment Request 93
  Enrollment Request with Polling 93
  Certificate Update (Manual and Auto) 94
  Certificate Update (Manual and Auto) with Polling 94
  Failure Response Handling (ip/cp/kup/pollRep) 95

CLI Commands 95

  Exec Mode Commands 95
    cmp initialize modulus 95
    cmp enroll current-cert 95
    cmp update current-cert 96
    cmp fetch current-cert -name 96
    cmp poll cert-name 96
  Global Configuration Mode Commands 96
    cmp auto-fetch 96
    cmp cert-store location 96
    cmp cert-trap time 96
  show and clear Commands 97
    show cmp outstanding-req 97
    show cmp statistics 97
    show cmp history 97
    clear cmp cert-name 97
    clear cmp statistics 97

Online Certificate Status Protocol (OCSP) 97

  Overview 97
  Deployment Scenarios 98
  Call Flows 100
    Successful OCSP Response 100
    Revoked OCSP Response 101
  External Interface 101
  CLI Commands 101
    Context Configuration Mode 101
CRL Fetching 102
  Overview 102
CRL Downloads 103
  Download from CDP Extension of Self-certificate 103
  Download from CDP Extension of Peer Certificate 104
CLI Commands 104
  Global Configuration Mode 104
    ca-crl name 104
  Context Configuration Mode 105
    ca-crl list 105
    show Commands 105

CHAPTER 12
Rekeying SAs 107
  Rekey Traffic Overlap 107
    Overview 107
    Deployment Scenarios 109
      Initiator and Responder Rekeying Behavior 109
  Sequence Number-based Rekeying 110
    Overview 110
    Deployment Scenarios 110
    CLI Commands 110
      ipsec rekey 111
      Crypto Map and Crypto Template Rekey Configurations 111
        show crypto ipsec security-associations 111

CHAPTER 13
Access Control 113
  Access Control via Blacklist or Whitelist 113
    Overview 113
    Blacklisting 114
    Whitelisting 114
    Blacklist and Whitelist File Format 115
      File Format and Content 115
      Supported IKE ID Types 116
    Deployment Scenarios 116
      Blacklisting 116
Whitelisting 116
External Interfaces 116
CLI Commands 117
  Global Configuration Mode 117
dcrypto blacklist file 117
crypto whitelist file 117
  Context Configuration Mode 117
      Enable blacklist 117
      Enable whitelist 118
  Exec Mode 118
crypto blacklist file update 118
crypto whitelist file update 118
  show Commands 118
      show crypto blacklist file 118
      show crypto whitelist file 119
      show crypto statistics ikev2 119
      show crypto template 119
IKE Call Admission Control 119

CHAPTER 14
Remote Secrets 121
  PSK Support for Remote Secrets 121
      Overview 121
      Implementation 121
      Supported IKE ID Types 122
      Deployment Scenarios 122
  CLI Commands 122
      Global Configuration Mode 123
crypto remote-secret-list 123
crypto remote-secret-list 123
      remote-id id-type 123
      Context Configuration Commands 123
      Enable remote secret list 123
      show Commands 124
      show configuration 124
      show crypto map 124
      show crypto template 124
CHAPTER 15
IKEv2 RFC 5996 Compliance 125

RFC 5996 Compliance 125
Overview 125

TEMPORARY_FAILURE 125
CHILD_SA_NOT_FOUND 126
Exchange Collisions 126
Integrity with Combined Mode Ciphers 126
Negotiation Parameters in CHILDSA REKEY 126
Certificates 126
Multiple Traffic Selectors 127

CLI Commands 127
Context Configuration Mode 127
Enable Notification Payloads 127
Add Hash and URL Encoding to Certificates 127
Enable Hash and URL Certificate Encoding 128
Disable Change in Rekey Parameters in CHILDSA REKEY 128
Enable TSl Ranges 129

show commands 129

CHAPTER 16
IKEv2 Fragmentation 131

Feature Description 131
Overview 131
How IKEv2 Fragmentation Works 131
Fragmenting IKEv2 DL Packets 131
Re-assembling IKEv2 Fragmented UL Packets 132
Limitations and Restrictions 132
Standards Compliance 132
Configuring IKEv2 Fragmentation 133
Configuring IKESA Fragmentation (Tx) and Re-assembly (Rx) 133
Configuring MTU Size for the IKEv2 Payload 133
Monitoring and Troubleshooting IKEv2 Fragmentation 134
IKEv2 Fragmentation Show Command(s) and/or Outputs 134

show crypto ikev2-ikesa security-associations 134
show crypto statistics ikev2 134
show crypto template 135
IKEv2 Fragmentation Bulk Statistics 135

CHAPTER 17
Duplicate Session Detection 137
Process Overview 137
Configuring Duplicate Session Detection 140
Verifying the Duplicate Session Detection Configuration 141

CHAPTER 18
Extended Sequence Number 143
Overview 143
ESN for ikev2 143
StarOS Support for ESN 144
Configuring ESN Support 144
Verifying ESN Configuration 145
show crypto ipsec transform-set 145
show crypto template 145

CHAPTER 19
Security Gateway as Initiator 147
Overview 147
Responder-Initiator Sequence 147
Limitations 148
Configuring SecGW as Initiator 148
Create a crypto peer-list 148
Configure the Peer List in the WSG Service 149
Configure Initiator Mode and Responder Mode Durations 149
Restrictions 149
Verifying the SecGW as Initiator Configuration 150
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, page xiii
- Related Documentation, page xiv
- Contacting Customer Support, page xv

Conventions Used

The following tables describe the conventions used throughout this documentation.

<table>
<thead>
<tr>
<th>Notice Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information Note</td>
<td>Provides information about important features or instructions.</td>
</tr>
<tr>
<td>Caution</td>
<td>Alerts you of potential damage to a program, device, or system.</td>
</tr>
<tr>
<td>Notice Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Warning</td>
<td>Alerts you of potential personal injury or fatality. May also alert you of potential electrical hazards.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Typeface Conventions</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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 |

**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*
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.
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, page 1
- IPSec Terminology, page 4
- IKEv1 versus IKEv2, page 6
- Supported Algorithms, page 8
- Boost Crypto Performance, page 9
- Multiple Authentication Configuration, page 9

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](image)

- **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)**, on page 4
- **Transform Set**, on page 4
- **ISAKMP Policy**, on page 5
- **Crypto Map**, on page 5
- **Crypto Template**, on page 6

**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**

<table>
<thead>
<tr>
<th>IKEv1</th>
<th>IKEv2</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPSec Security Associations (SAs)</td>
<td>Child Security Associations (Child SAs)</td>
</tr>
<tr>
<td>IKEv1</td>
<td>IKEv2</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-------------------------------------------</td>
</tr>
<tr>
<td>Exchange modes:</td>
<td>Only one exchange mode is defined. Exchange modes were obsoleted.</td>
</tr>
<tr>
<td>• Main mode</td>
<td></td>
</tr>
<tr>
<td>• Aggressive mode</td>
<td></td>
</tr>
<tr>
<td>Number of exchanged messages required to establish a VPN:</td>
<td>Only 4 messages are required to establish a VPN.</td>
</tr>
<tr>
<td>• Main mode = 9 messages</td>
<td></td>
</tr>
<tr>
<td>• Aggressive mode = 6 messages</td>
<td></td>
</tr>
<tr>
<td>Authentication methods:</td>
<td>Authentication methods:</td>
</tr>
<tr>
<td>• Pre-Shared Key (PSK)</td>
<td>• Pre-Shared Key (PSK)</td>
</tr>
<tr>
<td>• Digital Signature (RSA-Sig)</td>
<td>• Digital Signature (RSA-Sig)</td>
</tr>
<tr>
<td>• Public Key Encryption</td>
<td>• Revised mode of public Key Encryption</td>
</tr>
<tr>
<td>• Revised mode of public Key Encryption</td>
<td></td>
</tr>
<tr>
<td>Traffic Selector:</td>
<td>Traffic Selector:</td>
</tr>
<tr>
<td>• Only a combination of a source IP range, a destination IP range, a source port and a destination port is allowed per IPSec SA.</td>
<td>• 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.</td>
</tr>
<tr>
<td>• Exact agreement of the traffic selection between peers is required.</td>
<td>• Narrowing traffic selectors between peers is allowed.</td>
</tr>
<tr>
<td>Lifetime for SAs requires negotiation between peers.</td>
<td>Lifetime for SAs is not negotiated. Each peer can delete SAs by exchanging DELETE payloads.</td>
</tr>
<tr>
<td>Multihosting is not supported</td>
<td>Multihosting is supported by using multiple IDs on a single IP address and port pair.</td>
</tr>
<tr>
<td>Rekeying is not defined.</td>
<td>Rekeying is defined and supported.</td>
</tr>
<tr>
<td>Dead peer Detection (DPD) for SAs is defined as an extension.</td>
<td>DPD is supported by default.</td>
</tr>
<tr>
<td>NAT Transversal (NATT) is defined as an extension.</td>
<td>NATT is supported by default.</td>
</tr>
</tbody>
</table>
Remote Access VPN is not defined, but is supported by vendor-specific implementations for Mode config and XAUTH.

Multihoming is not supported.

Mobile Clients are not supported.

Denial of Service (DoS) protections are not supported.

IKEv2 Encryption Internet Key Exchange version 2

IKEv2 Pseudo Random Function

IKEv2 Integrity

IKEv2 Diffie-Hellman Group

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 supported by MOBIKE (IKEv2 Mobility and Multihoming Protocol, RFC 4555)

Mobile Clients are supported by MOBIKE.

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**

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Type</th>
<th>Supported Options</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Note</strong> AES-GCM algorithms are supported only on vPC-DI and vPC-SI Platform.</td>
</tr>
<tr>
<td>IKEv2 Pseudo Random Function</td>
<td></td>
<td>PRF-HMAC-SHA1, PRF-HMAC-MD5, AES-XCBC-PRF-128</td>
</tr>
<tr>
<td>IKEv2 Integrity</td>
<td></td>
<td>HMAC-SHA1-96, HMAC-SHA2-256-128, HMAC-SHA2-384-192, HMAC-SHA2-512-256, HMAC-MD5-96, AES-XCBC-96</td>
</tr>
<tr>
<td>IKEv2 Diffie-Hellman Group</td>
<td></td>
<td>Group 1 (768-bit), Group 2 (1024-bit), Group 5 (1536-bit), Group 14 (2048-bit)</td>
</tr>
<tr>
<td>Protocol</td>
<td>Type</td>
<td>Supported Options</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Extended Sequence Number</td>
<td>Value of 0 or off is supported (ESN itself is not supported)</td>
</tr>
<tr>
<td></td>
<td>IPSec Integrity</td>
<td>NULL, HMAC-SHA1-96, HMAC-MD5-96, AES-XCBC-96, HMAC-SHA2-256-128, HMAC-SHA2-384-192, HMAC-SHA2-512-256</td>
</tr>
<tr>
<td></td>
<td><strong>Important</strong></td>
<td>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.</td>
</tr>
</tbody>
</table>

**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. 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>
```

IPSec Reference, StarOS Release 20

9
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, page 11
• GGSN, FA and HA, page 12
• HeNBGW, HNBGW and HSGW, page 13
• ePDG, page 14
• MME, S-GW, P-GW and SAE-GW, page 14
• SecGW, page 15

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

In Release 20.0, HNBGW and HeNBGW are not supported. 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

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 Initiator
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, page 17
- Implementing IPSec for Mobile IP Applications, page 19
- Implementing IPSec for L2TP Applications, page 25
- IPSec for LTE/SAE Networks, page 30
- IPSec for Femto-UMTS Networks, page 38

Implementing IPSec for PDN Access Applications

This section provides information on the following topics:

- How IPSec-based PDN Access Configuration Works, on page 17
- Configuring IPSec Support for PDN Access, on page 18

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

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A subscriber session or PDP context Request, in GGSN service, arrives at the system.</td>
</tr>
<tr>
<td>2</td>
<td>The system processes the subscriber session or request as it would typically.</td>
</tr>
</tbody>
</table>
Step | Description
--- | ---
3 | Prior to routing the session packets, the system compares them against configured Access Control Lists (ACLs).
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, on page 20
- Configuring IPSec Support for Mobile IP, on page 22
- RADIUS Attributes for IPSec-based Mobile IP Applications, on page 24
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>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>FA service receives a Mobile IP registration request from the mobile node.</td>
</tr>
<tr>
<td>2</td>
<td>FA sends an Access-Request to the FAAA server with the 3GPP2-IKE-Secret-Request attribute equal to yes.</td>
</tr>
<tr>
<td>3</td>
<td>The FAAA proxies the request to the HAAA.</td>
</tr>
</tbody>
</table>
### Step Description

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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. |
| 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) |
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

The HA creates a response to the D-H exchange using the "S" secret and the Key ID sent by the FA.

The HA sends IKE SA negotiation D-H exchange response to the FA.

The FA and the HA negotiate an ISAKMP (IKE) policy to use to protect further communications.

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.

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 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.
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

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Variable</th>
</tr>
</thead>
</table>
| 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, on page 25
- Configuring Support for L2TP Attribute-based Tunneling with IPSec, on page 26
- How IPSec is Used for PDSN Compulsory L2TP Configuration, on page 26
- Configuring Support for L2TP PDSN Compulsory Tunneling with IPSec, on page 27
- How IPSec is Used for L2TP Configurations on the GGSN, on page 28
- Configuring GGSN Support for L2TP Tunneling with IPSec, on page 29

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**

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A subscriber session arrives at the system.</td>
</tr>
<tr>
<td>2</td>
<td>The system attempts to authenticate the subscriber with the AAA server.</td>
</tr>
<tr>
<td>3</td>
<td>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.</td>
</tr>
<tr>
<td>4</td>
<td>The system determines that the crypto map name supplied matches a configured crypto map.</td>
</tr>
<tr>
<td>5</td>
<td>From the crypto map, the system determines the following:</td>
</tr>
<tr>
<td></td>
<td>• The map type, in this case dynamic</td>
</tr>
<tr>
<td></td>
<td>• Whether perfect forward secrecy (PFS) should be enabled for the IPSec SA and if so, what group should be used</td>
</tr>
<tr>
<td></td>
<td>• IPSec SA lifetime parameters</td>
</tr>
<tr>
<td></td>
<td>• The name of one or more configured transform set defining the IPSec SA</td>
</tr>
<tr>
<td>6</td>
<td>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.</td>
</tr>
</tbody>
</table>
Step

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.

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 attribute-based 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 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

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>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.</td>
</tr>
</tbody>
</table>
| 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.
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**

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A subscriber session/PDP Context Request arrives at the system.</td>
</tr>
<tr>
<td>2</td>
<td>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.</td>
</tr>
<tr>
<td>3</td>
<td>The system determines that the crypto map name supplied matches a configured crypto map.</td>
</tr>
</tbody>
</table>
| 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 |
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.

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>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.</td>
</tr>
<tr>
<td>6</td>
<td>The system and the LNS or security gateway negotiate an ISAKMP (IKE) policy to use to protect further communications.</td>
</tr>
<tr>
<td>7</td>
<td>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.</td>
</tr>
<tr>
<td>8</td>
<td>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.</td>
</tr>
</tbody>
</table>

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 Block Chaining)

HMAC Functions

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

- 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:

- 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 P-GW.

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): udphost2.2.2.2 eq 2123

When an IKE request arrives the S-GW from eNodeB address 1.1.1.1, the IPSec subsystem converts the payload ACL to: udphost2.2.2.2 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: udphost2.2.2.2 eq 2123 1.1.0.0 0.0.255.255
- Source port as well-known port: udphost2.2.2.2 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>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>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.</td>
</tr>
<tr>
<td>2</td>
<td>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.</td>
</tr>
</tbody>
</table>
**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.

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>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.</td>
</tr>
<tr>
<td>4</td>
<td>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.</td>
</tr>
<tr>
<td>5</td>
<td>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.</td>
</tr>
<tr>
<td>6</td>
<td>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.</td>
</tr>
<tr>
<td>7</td>
<td>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.</td>
</tr>
</tbody>
</table>
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.
Table 10: E-UTRAN/EPC Logical Network Interfaces Supporting IPSec Tunnels

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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.                                                                                                                                                          |
| 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

In Release 20.0, HNBGW and HeNBGW are not supported. 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.

```bash
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
```
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**

![Diagram of X.509 Certificate-based Peer Authentication]

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

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>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.</td>
</tr>
<tr>
<td>2</td>
<td>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.</td>
</tr>
</tbody>
</table>
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.

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.

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.

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.

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.

```plaintext
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
drop
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.
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, page 45
- Configuring a Transform Set, page 46
- Verifying the Crypto Transform Set Configuration, page 46

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 46.

Step 2 Verify your Crypto Transform Set configuration by following the steps in Verifying the Crypto Transform Set Configuration, on page 46.

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
c  context  ctxt_name
cryptopcontext 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
```
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, page 47
• Configuring ISAKMP Policy, page 48
• Verifying the ISAKMP Policy Configuration, page 48

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 48.

Step 2 Verify your ISAKMP policy configuration by following the steps in Verifying the ISAKMP Policy 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 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
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.
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.

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, page 49
• Dynamic Crypto Map Configuration, page 51
• Manual Crypto Map Configuration, page 52
• Crypto Map and Interface Association, page 54

ISAKMP Crypto Map Configuration

This section provides instructions for configuring ISAKMP crypto maps.

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 50.

**Step 2** Verify your ISAKMP crypto map configuration by following the steps in Verifying the ISAKMP Crypto Map 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 Crypto Maps

Use the following example to create the ISAKMP crypto map:

```plaintext
configure
  context ctx_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:**

- `ctx_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 ofTransforms : 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 52.

### Step 2
Verify your dynamic crypto map configuration by following the steps in Verifying the Dynamic 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 Dynamic Crypto Maps

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

```plaintext
configure
  context ctx_name
  crypto map map_name ipsec-dynamic
    set pfs { group1 | group2 | group5 }
    set transform-set transform_name
  end
```

Notes:

- `ctx_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:

```plaintext
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 IPSec Reference, StarOS Release 20*.
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 53.

**Step 2** Verify your manual crypto map configuration by following the steps in Verifying the Manual 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.

**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:

```plaintext
configure
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 |
        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
```

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 55.

### Step 3
Verify your manual crypto map configuration by following the steps in Verifying the Interface Configuration with Crypto Map, on page 55.

### 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:

```bash
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:

```bash
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
```
Verifying the Interface Configuration with Crypto Map
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, page 57
- Configuring ANSSI Enhancements, page 59

Feature Description

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

- Auto-delete Existing IKEv1/IKEv2 ACL Tunnels, on page 57
- Remove Weak Security Algorithms, on page 58

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 crypto-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:

<table>
<thead>
<tr>
<th>IKE Tunnel</th>
<th>Encryption</th>
<th>DES-CBC, 3DES-CBC, NULL</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMAC</td>
<td>AES-XCBC-96, MD5-96, SHA1-96</td>
<td></td>
</tr>
<tr>
<td>DH Group</td>
<td>1, 2</td>
<td></td>
</tr>
<tr>
<td>PRF</td>
<td>AES-XCBC-128, MD5, SHA1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IPSec Tunnel</th>
<th>Encryption</th>
<th>DES-CBC, 3DES-CBC, NULL</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMAC</td>
<td>AES-XCBC-96, MD5-96, SHA1-96</td>
<td></td>
</tr>
<tr>
<td>DH Group</td>
<td>1, 2, none</td>
<td></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>IKE Tunnel</th>
<th>Encryption</th>
<th>DES-CBC, 3DES-CBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMAC</td>
<td>MD5</td>
<td></td>
</tr>
<tr>
<td>DH Group</td>
<td>1, 2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IPSec Tunnel</th>
<th>Encryption</th>
<th>DES-CBC, 3DES-CBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMAC</td>
<td>MD5-96, none</td>
<td></td>
</tr>
<tr>
<td>DH Group</td>
<td>1, 2</td>
<td></td>
</tr>
</tbody>
</table>
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.
Enabling Auto-deletion of Existing IKEv1/IKEv2 ACL Tunnels
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, page 61
• Crypto Template IKEv2-Dynamic Payload Parameters, page 62
• Configuring a Crypto Template, page 63
• Verifying a Crypto Template Configuration, page 64

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.
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
custom_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 | second-phase eap-profile name } | pre-shared-key { encrypted key value | key clear_text | second-phase eap-profile name } | 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 } } | local
  identity local id-type type
  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-major-version] | invalid-syntax [invalid-major-syntax] } | 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 | ip-mask | key-id | rfc822-addr }
  | use-received-idr
  | natt { include-header | | send-keepalive | idle-interval idle_secs | | interval interval_secs | oosp | nonce
  payload payload_name 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
```
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>
Authentication 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
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, page 67
- HA Service Configuration to Support IPSec, page 68
- PDSN Service Configuration for L2TP Support, page 69
- LAC Service Configuration to Support IPSec, page 72
- APN Template Configuration to Support L2TP, page 73
- WSG Service Configuration to Support IPSec, page 74

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:

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td>Modify FA service configuration by following the steps in Modifying FA Service to Support IPSec, on page 68.</td>
</tr>
<tr>
<td>Step 2</td>
<td>Verify your FA service configuration by following the steps in Verifying the FA Service Configuration with IPSec, on page 68.</td>
</tr>
<tr>
<td>Step 3</td>
<td>Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command <code>save configuration</code>. For additional information on how to verify and save configuration files, refer to the System Administration Guide and the Command Line Interface Reference.</td>
</tr>
</tbody>
</table>
Modifying FA Service to Support IPSec

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

```
configure
  context ctx_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:

- `ctx_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 69.

Step 2 Verify your HA service configuration by following the steps in Verifying the HA Service Configuration with IPSec, on page 69.

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:

```bash
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:

```bash
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 70
- Modifying PDSN Service to Support Compulsory L2TP Tunneling, on page 71

**Step 2** Verify your LAC service configuration by following the steps in Verifying the PDSN Service Configuration for L2T, on page 72.

**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.
- `pdns_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

<table>
<thead>
<tr>
<th>RADIUS Attribute</th>
<th>Local Subscriber Attribute</th>
<th>Description</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>SN1-Tunnel-ISAKMP-Crypto-Map</td>
<td>tunnel l2tp crypto-map</td>
<td>The name of a crypto map configured on the system.</td>
<td>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.</td>
</tr>
<tr>
<td>SN1-Tunnel-ISAKMP-Secret</td>
<td>tunnel l2tp crypto-map</td>
<td>The pre-shared secret that will be used as part of the D-H exchange to negotiate an IKE SA.</td>
<td>A salt-encrypted string specifying the IKE secret. It can be tagged, in which case it is treated as part of a tunnel group.</td>
</tr>
</tbody>
</table>

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
  pdn-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.
- `pdn_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 72.

**Step 2**
Verify your LAC service configuration by following the steps in Verifying the LAC Service Configuration with IPSec, 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 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 74.

**Step 2** Verify your APN configuration by following the steps in *Verifying the APN Configuration for L2TP*, on page 74.

**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:

```plaintext
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:

```plaintext
show apn name apn_name
```

The output of this command contains a concise listing of L2TP 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.
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 and 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.
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), page 77
- Dead Peer Detection (DPD) Configuration, page 80

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 79.

**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 81.

**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 80.

**Step 6** Verify your Crypto Group configuration by following the steps in Verifying the Crypto Group Configuration, on page 80.

**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
crypto-group
text_name
ikev1 keepalive dpd interval dur timeout dur num-retry retries
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 secondary
    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.
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 81.

**Step 2**
Verify your DPD configuration by following the steps in Verifying the DPD Configuration, on page 81

**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.
Verifying the DPD Configuration
This chapter describes a number of StarOS features that support IPSec certificate management.

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:

- Multiple Child SA (MCSA) Support, page 83
- Creating, Signing, and Configuring Certificates, page 86
- CA Certificate Chaining, page 87
- Certificate Management Protocol (CMPv2), page 89
- Online Certificate Status Protocol (OCSP), page 97
- CRL Fetching, page 102

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 pathname ] ]
   ```
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.

**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 CA1_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 sendsIKE_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_1".
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_1".
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

![Call Flow: Successful Initial Certification Request]

Initial Certification Request with Polling

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

![Call Flow: Successful Initial Certification Request with Polling]
Enrollment Request

*Figure 12: Call Flow: Successful Enrollment Request*

1. CMPv2 Certificate Req Message (cr)
2. CMPv2 Certificate Resp Message (cp) [PKIStatus = Accepted]
3. CMPv2 Certificate Confirm Message (certConf)
4. CMPv2 PKI Confirmation Message (pkiConf)

Enrollment Request with Polling

*Figure 13: Call Flow: Successful Enrollment Request with Polling*

1. CMPv2 Certificate Req Message (cr)
2. CMPv2 Certificate Resp Message (cp) [PKIStatus = Waiting]
3. CMPv2 Polling Request Message (pollReq)
4. CMPv2 PKI Polling Resp Message (pollRep) [PKIStatus = Accepted]
5. CMPv2 Certificate Confirm Message (certConf)
6. CMPv2 PKI Confirmation Message (pkiConf)
Certificate Update (Manual and Auto)

Figure 14: Call Flow: Successful Certificate Update

Certificate Authority (CA)

1. CMPv2 Key Update Request Message (kur)
2. CMPv2 Key Update Resp Message (kup) [PKIStatus = Accepted]
3. CMPv2 Certificate Confirm Message (certConf)
4. CMPv2 PKI Confirmation Message (pkConf)

Certificate Update (Manual and Auto) with Polling

Figure 15: Call Flow: Successful Certificate Update with Polling

Certificate Authority (CA)

1. CMPv2 Key Update Req Message (kur)
2. CMPv2 Key Update Resp Message (kup) [PKIStatus = Waiting]
3. CMPv2 Polling Request Message (pollReq)
4. CMPv2 PKI Polling Resp Message (pollRes) [PKIStatus = Accepted]
5. CMPv2 Certificate Confirm Message (certConf)
6. CMPv2 PKI Confirmation Message (pkConf)
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 initialize 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 as 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

show cmp outstanding-req
Refer to the Statistics and Counters Reference for a description of the information output by this command.

show cmp statistics

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

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

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

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

<table>
<thead>
<tr>
<th>Important</th>
</tr>
</thead>
<tbody>
<tr>
<td>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.</td>
</tr>
</tbody>
</table>

Context Configuration Mode

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

For a crypto map the configuration sequence is:

```bash
configure
context ctxt_name
  crypto map template_name { ikev2-ipv4 | ikev2-ipv6 }
  ocsp [ nonce ]
```
For a crypto template the configuration sequence is:
```
configure
crypt
context ctx_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*

![](image)

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

**CLI Commands**

<table>
<thead>
<tr>
<th>Important</th>
</tr>
</thead>
<tbody>
<tr>
<td>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.</td>
</tr>
</tbody>
</table>

**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.
Rekeying SAs

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

The following topics are discussed:

- Rekey Traffic Overlap, page 107
- Sequence Number-based Rekeying, page 110

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 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.
Access Control

This chapter describes enhancements to IPSec Access Control.
The following topics are discussed:

- Access Control via Blacklist or Whitelist, page 113
- IKE Call Admission Control, page 119

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](image)

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.

```plaintext
# 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"
```
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
A blacklist or whitelist must be available to StarOS or blacklisting/whitelisting will not be performed even if enabled.

**CLI Commands**

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:

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

For a crypto template the configuration sequence is:

```
cfg 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.
Remote Secrets

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

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.

Important

The following topics are discussed:

• PSK Support for Remote Secrets, page 121
• CLI Commands, page 122

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 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**

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 ]
```

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.
IEEEv2 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, page 125
- CLI Commands, page 127

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
cryptomap template_name { ikev2-ipv4 | ikev2-ipv6 }
ikev2-ikesa
policy use-rfc5996-notification
```

For a crypto template the configuration sequence is:

```
configure
cryptotemplate 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
cryptomapctxt_name
cryptomaptemplate_name { ikev2-ipv4 | ikev2-ipv6 }
certificate_name
pemurlurl
cert-enc cert-hash-url url url

For a crypto template the configuration sequence is:
configure
cryptotemplatectxt_name
cryptotemplatetemplate_name ikev2-dynamic
pemurlurl
cert-enc cert-hash-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
cryptomapctxt_name
cryptomaptemplate_name { ikev2-ipv4 | ikev2-ipv6 }
allow-cert-enc cert-hash-url

For a crypto template the configuration sequence is:
configure
cryptotemplatectxt_name
cryptotemplatetemplate_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
cryptomapctxt_name
cryptomaptemplate_name { ikev2-ipv4 | ikev2-ipv6 }
ikev2-ikesa
rekey disallow-param-change

For a crypto template the configuration sequence is:
configure
cryptotemplatectxt_name
cryptotemplatetemplate_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.

```plaintext
configure
cryptotemplate template_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
IKEv2 Fragmentation

This feature enables IPSec to fragment large messages at IKEv2 as defined in RFC 7383. The following topics are discussed:

- Feature Description, page 131
- How IKEv2 Fragmentation Works, page 131
- Configuring IKEv2 Fragmentation, page 133
- Monitoring and Troubleshooting IKEv2 Fragmentation, page 134

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):

```plaintext
configure
crypto template template_name
  [ 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 not 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:

```plaintext
configure
crypto template template_name
  { 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 13: show crypto ikev2-ikesa security-associations Command Output Descriptions*

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fragmentation Supported</td>
<td>Indicates if IKEv2-IKESA fragmentation is supported.</td>
</tr>
</tbody>
</table>

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:

<table>
<thead>
<tr>
<th>Control Statistics for Context: &lt;context_name&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detailed IKE Statistics:</td>
</tr>
<tr>
<td>Total Fragments In: 0</td>
</tr>
<tr>
<td>Total Fragments Out: 0</td>
</tr>
<tr>
<td>Total IKE Fragmented Packets In: 0</td>
</tr>
<tr>
<td>Total IKE Fragmented Packets out: 0</td>
</tr>
<tr>
<td>Total IKE Fragments Dropped: 0</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>Total IKEv2 Notify Payload Sent Statistics</td>
</tr>
<tr>
<td>Fragmentation Supported Notify Sent: 0</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>Total IKEv2 Notify Payload Received Statistics</td>
</tr>
<tr>
<td>Fragmentation Supported Notify Rcvd: 0</td>
</tr>
</tbody>
</table>

*Table 14: show crypto statistics ikev2 Command Output Descriptions*

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detailed IKE Statistics:</td>
<td></td>
</tr>
<tr>
<td>Total Fragments In</td>
<td>Total fragments received.</td>
</tr>
<tr>
<td>Total Fragments Out</td>
<td>Total fragments sent.</td>
</tr>
<tr>
<td>Total IKE Fragmented Packets In</td>
<td>Total number of IKE fragmented packets received.</td>
</tr>
<tr>
<td>Total IKE Fragmented Packets out</td>
<td>Total number of IKE fragmented packets sent.</td>
</tr>
</tbody>
</table>
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>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IKEv2 Fragmentation</td>
<td>Indicates if the configuration for IKEv2 fragmentation is enabled or disabled.</td>
</tr>
<tr>
<td>IKEv2 MTU Size IPv4/IPv6</td>
<td>Configured IKEv2 payload MTU size over the IPv4/IPv6 tunnels.</td>
</tr>
</tbody>
</table>

IKEv2 Fragmentation Bulk Statistics

The following bulks statistics included in the System schema support this feature:
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Data Type</th>
</tr>
</thead>
</table>
| 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     |
| 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     |
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, page 137
- Configuring Duplicate Session Detection, page 140
- Verifying the Duplicate Session Detection Configuration, page 141

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**

![Diagram showing the process of no duplicate session found](image)

**Figure 28: Duplicate Session Found in Same StarOS IPSec Manager**

![Diagram showing the process of duplicate session found in the same manager](image)

**Figure 29: Duplicate Session Found in Different StarOS IPSec Manager**

![Diagram showing the process of duplicate session found in a different manager](image)
Duplicate Session Detection

**Process Overview**

**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:

```bash
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
```
Verifying the Duplicate Session Detection Configuration
CHAPTER 18

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, page 143
• Configuring ESN Support, page 144
• Verifying ESN Configuration, page 145

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 ASR 5000 or 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
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 tsela:
  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: tsela
      Protocol: esp
      Encryption Cipher: aes-cbc-128
      Hashed Message Authentication Code: sha1-96
      Diffie-Hellman Group: none
      ESN     : Enabled
```
show crypto template
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, page 147
- Configuring SecGW as Initiator, page 148
- Verifying the SecGW as Initiator Configuration, page 150

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 148.

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 148.
- Configure the Peer List in the WSG Service, on page 149.
- Configure Initiator Mode and Responder Mode Durations, on page 149.

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
cannotex context context_name
cryptopeer-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
current-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 148.
- 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
current-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
```