



Configure MACSec

This module describes how to configure Media Access Control Security (MACSec) encryption on the NCS 5500 Network Convergence System Routers. MACSec is a Layer 2 IEEE 802.1AE standard for encrypting packets between two MACSec-capable routers.

- [Understanding MACSec Encryption, on page 1](#)
- [MKA Authentication Process, on page 2](#)
- [MACsec Frame Format, on page 3](#)
- [Advantages of Using MACsec Encryption, on page 3](#)
- [Hardware Support Matrix for MacSec, on page 3](#)
- [MACsec PSK, on page 9](#)
- [Fallback PSK, on page 9](#)
- [Configuring and Verifying MACsec Encryption , on page 10](#)
- [Creating a MACsec Keychain, on page 10](#)
- [Creating a User-Defined MACsec Policy, on page 17](#)
- [Applying MACsec Configuration on an Interface, on page 21](#)
- [MACsec Policy Exceptions, on page 23](#)
- [Verifying MACsec Encryption on IOS XR, on page 24](#)
- [Verifying MACsec Encryption on NCS 5500, on page 36](#)
- [MACsec SecY Statistics, on page 39](#)
- [Secure Key Integration Protocol, on page 48](#)

Understanding MACSec Encryption

Security breaches can occur at any layer of the OSI model. At Layer 2, some of the common breaches are MAC address spoofing, ARP spoofing, Denial of Service (DoS) attacks against a DHCP server, and VLAN hopping.

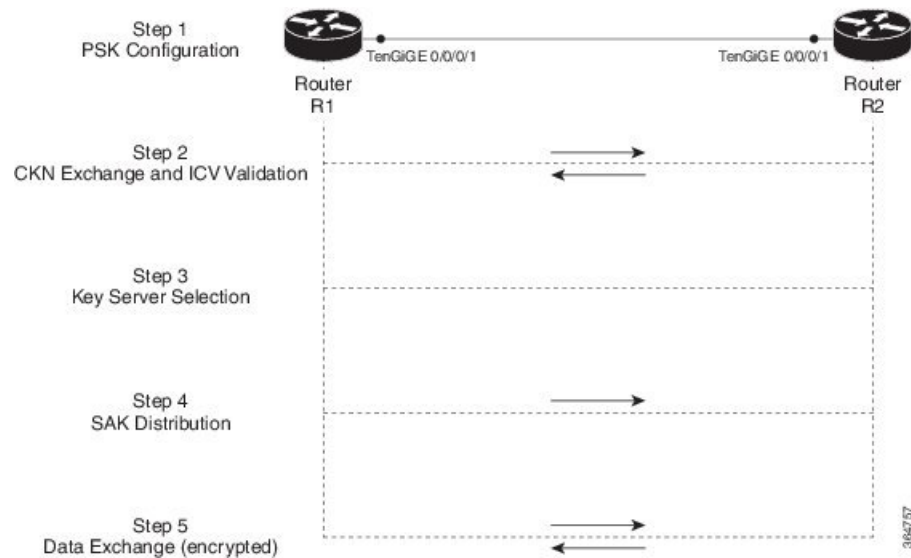
MACSec secures data on physical media, making it impossible for data to be compromised at higher layers. As a result, MACSec encryption takes priority over any other encryption method such as IPsec and SSL at higher layers. MACSec is configured on the Customer Edge (CE) router interfaces that connect to Provider Edge (PE) routers and on all the provider router interfaces.

MKA Authentication Process

MACsec provides the secure MAC Service on a frame-by-frame basis, using GCM-AES algorithm. MACsec uses the MACsec Key Agreement protocol (MKA) to exchange session keys, and manage encryption keys.

The MACsec encryption process is illustrated in the following figure and description.

Figure 1: MKA Encryption Process



Step 1: When a link is first established between two routers, they become peers. Mutual peer authentication takes place by configuring a Pre-shared Key (PSK).

Step 2: On successful peer authentication, a connectivity association is formed between the peers, and a secure Connectivity Association Key Name (CKN) is exchanged. After the exchange, the MKA ICV is validated with a Connectivity Association Key (CAK), which is effectively a secret key.

Step 3: A key server is selected between the routers, based on the configured key server priority. Lower the priority value, higher the preference for the router to become the key server. If no value is configured, the default value of 16 is taken to be the key server priority value for the router. Lowest priority value configures that router as the key server, while the other router functions as a key client. The following rules apply to key server selection:

- Numerically lower values of key server priority and SCI are accorded the highest preference.
- Each router selects a peer advertising the highest preference as its key server provided that peer has not selected another router as its key server or is not willing to function as the key server.
- In the event of a tie for highest preferred key server, the router with the highest priority SCI is chosen as key server (KS).

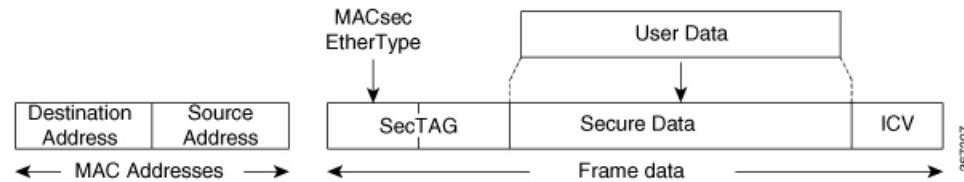
Step 4: A security association is formed between the peers. The key server generates and distributes the Secure Association Key (SAK) to the key client (peer). Each secure channel is supported by an overlapped sequence of Security Associations (SA). Each SA uses a new Secure Association Key (SAK).

Step 5: Encrypted data is exchanged between the peers.

MACsec Frame Format

The MACsec header in a frame consists of three components as illustrated in the following figure.

Figure 2: MACsec Frame Format



- **SecTAG:** The security tag is 8-16 bytes in length and identifies the SAK to be used for the frame. With Secure Channel Identifier (SCI) encoding, the security tag is 16 bytes in length, and without the encoding, 8 bytes in length (SCI encoding is optional). The security tag also provides replay protection when frames are received out of sequence.
- **Secure Data:** This is the data in the frame that is encrypted using MACsec and can be 2 or more octets in length.
- **ICV:** The ICV provides the integrity check for the frame and is usually 8-16 bytes in length, depending on the cipher suite. Frames that do not match the expected ICV are dropped at the port.

Advantages of Using MACsec Encryption

- **Data Integrity Check:** Integrity check value (ICV) is used to perform integrity check. The ICV is sent with the protected data unit and is recalculated and compared by the receiver to detect data modification.
- **Data Encryption:** Enables a port to encrypt outbound frames and decrypt MACsec-encrypted inbound frames.
- **Replay Protection:** When frames are transmitted through the network, there is a strong possibility of frames getting out of the ordered sequence. MACsec provides a configurable window that accepts a specified number of out-of-sequence frames.
- **Support for Clear Traffic:** If configured accordingly, data that is not encrypted is allowed to transit through the port.

Hardware Support Matrix for MacSec

The MACSec support on Cisco NCS 5500 Series Routers and NCS 5700 Series Routers is compatible with the following platform models, line cards (LCs), modular port adapters (MPAs), and small form-factor pluggables (SFPs).

Platform Models

The following platform models support MACSec:

Table 1: Cisco NCS 5500 Series Routers: Supported Modular Chassis for MACSec

Platform Model	Introduced Release for MACSec Support
NCS 5504	Release 6.3.1
NCS 5516	Release 6.1.3
NCS 5508	Release 6.0

Table 2: Cisco NCS 5500 Series Routers: Supported Fixed Chassis for MACSec

Platform Model	Introduced Release for MACSec Support
NCS-55A1-24Q6H-SS	Release 7.2.1
NCS-55A1-24Q6H-S	Release 6.6.2
NCS-55A1-48Q6H	Release 6.6.2
NC55A2-MOD-SE-H-S	Release 6.6.1
NCS-55A2-MOD-SE-S	Release 6.6.1
NCS-55A2-MOD-S	Release 6.6.1
NCS-55A2-MOD-HD-S	Release 6.6.1
NCS-55A2-MOD-HX-S	Release 6.6.1
NCS-55A1-36H-SE-S	Release 6.3.2
NCS-55A1-36H-S	Release 6.2.2

Table 3: Cisco NCS 5700 Series Routers: Supported Fixed Chassis for MACSec

Platform Model	Introduced Release for MACSec Support
NCS-57D2-18DD-SYS	Release 7.10.1 <i>(See, Guidelines and Limitations for MACSec Support section below)</i>
NCS-57B1-5DSE-SYS	Release 7.6.1
NCS-57B1-6D24-SYS	Release 7.6.1
NCS-57C1-48Q6-SYS	Release 7.5.2
NCS-57C3-MOD-SYS	Release 7.4.1
NCS-57C3-MODS-SYS	Release 7.4.1

Line Cards

The following line cards support MACSec:

Table 4: Cisco NCS 5500 Series Routers: Supported Line Cards for MACSec

Line Card	Introduced Release for MACSec Support
NC55-32T16Q4H-A	Release 7.7.1
NC55-MOD-A-SE-S Base	Release 6.6.1 (only for base)
NC55-MOD-A-S Base	Release 6.6.1 (only for base)
NC55-6x200-DWDM-S	Release 6.2.2 (MACSec on all ports)
NC55-36x100G-S	Release 6.1.3 (MACSec on all ports)

Table 5: Cisco NCS 5700 Series Routers: Supported Line Cards for MACSec

Line Card	Introduced Release for MACSec Support
NC57-48Q2D-SE-S	Release 7.10.1
NC57-48Q2D-S	Release 7.10.1
NC57-36H6D-S	Release 7.3.2 Release 7.4.1
NC57-MOD-S Base	Release 7.6.1 (only for base)

MPAs

The following MPAs support MACSec:

Table 6: Cisco NCS 5500 Series and 5700 Series Routers: Supported MPAs for MACSec

MPA	Hardware in Which Support is Introduced	Introduced Release for MACSec Support
NC55-MPA-12T-S	NCS-57C3-MODS-S NCS-57C3-MODS-SE-S	Release 7.4.1
	NC57-MOD-S	Release 7.6.1
	NCS-55A2-MOD-S NCS-55A2-MOD-SE-S NCS-55A2-MOD-HD-S NCS-55A2-MOD-HX-S NC55A2-MOD-SE-H-S	Release 6.6.1
	NC55-MOD-A-S NC55-MOD-A-SE-S	Release 6.6.1
	NCS-57C3-MODS-S NCS-57C3-MODS-SE-S	Release 7.4.1
NC55-MPA-2TH-S	NC57-MOD-S	Release 7.6.1
	NCS-55A2-MOD-S NCS-55A2-MOD-SE-S NCS-55A2-MOD-HD-S NCS-55A2-MOD-HX-S NC55A2-MOD-SE-H-S	Release 6.6.1
	NC55-MOD-A-S NC55-MOD-A-SE-S	Release 6.6.1
	NCS-57C3-MODS-S NCS-57C3-MODS-SE-S	Release 7.4.1
	NC57-MOD-S	Release 7.6.1

MPA	Hardware in Which Support is Introduced	Introduced Release for MACSec Support
NC55-MPA-1TH2H-S	NCS-57C3-MODS-S NCS-57C3-MODS-SE-S	Release 7.4.1
	NC57-MOD-S	Release 7.6.1
	NCS-55A2-MOD-S NCS-55A2-MOD-SE-S NCS-55A2-MOD-HD-S NCS-55A2-MOD-HX-S NC55A2-MOD-SE-H-S	Release 6.6.1
	NC55-MOD-A-S NC55-MOD-A-SE-S	Release 6.6.1
NC55-MPA-4H-S	NCS-57C3-MODS-S NCS-57C3-MODS-SE-S	Release 7.4.1
	NC57-MOD-S	Release 7.6.1
	NCS-55A2-MOD-S NCS-55A2-MOD-SE-S NCS-55A2-MOD-HD-S NCS-55A2-MOD-HX-S NC55A2-MOD-SE-H-S	Release 6.6.1
	NC55-MOD-A-S NC55-MOD-A-SE-S	Release 6.6.1
NC57-MPA-2D4H-S	NCS-55A2-MOD-S NCS-55A2-MOD-SE-S NCS-55A2-MOD-HD-S NCS-55A2-MOD-HX-S NC55A2-MOD-SE-H-S	Release 7.5.1
	NCS-57C3-MOD-S NCS-57C3-MOD-SE-S	Release 7.5.1
	NC57-MOD-S	Release 7.6.1

MPA	Hardware in Which Support is Introduced	Introduced Release for MACSec Support
NC57-MPA-12L-S	NCS-57C3-MOD-S	Release 7.6.1
	NCS-57C3-MOD-SE-S	
	NC57-MOD-S	Release 7.6.1

SFP

The following Small Form-factor Pluggables (SFPs) support MACsec:

Table 7: Cisco NCS 5500 Series and 5700 Series Routers: Supported SFPs for MACsec

SFP	Hardware in Which Support is Introduced	Introduced Release for MACsec Support
SFP-1G-SX	NCS-57C3-MOD-SYS	Release 24.1.1
SFP-1G-LH	NCS-57C3-MOD-SYS	Release 24.1.1

Guidelines and Limitations for MACSec Support

These generic guidelines and limitations apply to MACSec support on Cisco NCS 5500 Series Routers and Cisco NCS 5700 series routers.

- We recommend changing the offset value of the **conf-offset** *offset_value* command and **cipher cipher-suite** *cipher* (MACsec encryption command) in the router only when the port is in admin down state (that is, when the interface is shut down). Changing the offset value or cipher otherwise may result in traffic loss.
- Although the MACSec **eapol destination-address broadcast-address** command under the interface configuration mode is present and configurable on the router, the EAPOL functionality is not yet supported.
- Data delay protection (DDP) feature is not supported on Cisco NCS 5700 Series Fixed Port Routers and on Cisco NCS 5500 Series Routers that have the Cisco NC57 line cards installed and operating in the native and compatible modes.

The following list specifies the guidelines, limitations and unsupported features applicable to specific hardware variant. This list is in addition to the generic guidelines and limitations listed above.

- **NCS-57D2-18DD-SYS:**
 - Only 100G and 400G interfaces support MACSec.
 - MACSec interfaces support only GCM-AES-XPN-128 and GCM-AES-XPN-256 ciphers for MACSec policy configuration.
 - MACSec support on bundle interfaces is not applicable.
 - You might observe interface flap or traffic loss (for approximately 6 to 7 milliseconds) while enabling or disabling MACSec on the interface.
 - Does not support these:

- Point-to-multipoint scenarios
- Fallback key chain
- Interface breakouts
- MacSec as a service
- Window size for replay protection
- MACSec using EAP-TLS (Dot1x) authentication
- **should-secure** MACSec security policy
- Virtual private wire service (VPWS)
- Manageability—configuring MACSec through data models
- SNMP counters
- Limited to 2 RxSA/TxSA (0,1) from security entity (SecY) statistics

MACsec PSK

A pre-shared key includes a connectivity association key name (CKN) and a connectivity association key (CAK). A pre-shared key is exchanged between two devices at each end of a point-to-point link to enable MACsec using static CAK security mode. The MACsec Key Agreement (MKA) protocol is enabled after the pre-shared keys are successfully verified and exchanged. The pre-shared keys, the CKN and CAK, must match on both ends of a link.

For more information on MACsec PSK configuration, see [Step 3, on page 22](#) of the [Applying MACsec Configuration on an Interface, on page 21](#) section.

Fallback PSK

Fallback is a session recovery mechanism when primary PSK fails to bring up secured MKA session. It ensures that a PSK is always available to perform MACsec encryption and decryption.

- In CAK rollover of primary keys, if latest active keys are mismatched, system performs a hitless rollover from current active key to fallback key, provided the fallback keys match.
- If a session is up with fallback, and primary latest active key configuration mismatches are rectified between peers, system performs a hitless rollover from fallback to primary latest active key.



Note A valid Fallback PSK (CKN and CAK) must be configured with infinite lifetime. If the fallback PSK is configured with CAK mismatch, the only recovery mechanism is to push a new set of PSK configurations (both on fallback PSK keychain and primary PSK chain in that order) on all the association members.

The following is a sample syslog for session secured with fallback PSK:

```
%L2-MKA-5-SESSION_SECURED_WITH_FALLBACK_PSK : (Hu0/1/0/0) MKA session secured, CKN:ABCD
```

For more information on MACsec fallback PSK configuration, see [Step 3, on page 22](#) of the [Applying MACsec Configuration on an Interface, on page 21](#) section.

Active Fallback

The Cisco IOS XR Software Release 7.1.2 introduces the support for active fallback feature that initiates a fallback MKA session on having fallback configuration under the interface.

The key benefits of active fallback feature are:

- Faster session convergence on fallback, in the event of primary key deletion, expiry or mismatch.
- Faster traffic recovery under should-secure security policy when both primary and fallback mismatch happens.

With the introduction of active fallback functionality, the output of various MACsec show commands include the fallback PSK entry as well. If the session is secured with primary key, the fallback session will be in ACTIVE state. See, [Verifying MACsec Encryption on IOS XR, on page 24](#) for details and sample outputs.



Note If the peer device is running on an older release that does not support active fallback feature, you must configure the **enable-legacy-fallback** command under the macsec-policy to ensure backward compatibility.

Configuring and Verifying MACsec Encryption

MACsec can be configured on physical ethernet interfaces or member links of the interface bundles, as explained in this section.

The following section describes procedures for configuring and verifying MACsec configuration in the described deployment modes.

Prior to configuring MACsec on a router interface the MACsec keychain must be defined. If you apply the MACsec keychain on the router without specifying a MACsec policy, the default policy is applied. A default MACsec policy is pre-configured with default values. If you need to change any of the pre-configured values, create a different MACsec policy.

Configuring MACsec involves the following steps:

1. Creating a MACsec keychain
2. Creating a user-defined MACsec policy
3. Applying MACsec configuration on physical interfaces

Creating a MACsec Keychain

A MACsec keychain is a collection of keys used to authenticate peers needing to exchange encrypted information. While creating a keychain, we define the key(s), key string with password, the cryptographic algorithm, and the key lifetime.

MACsec Keychain Keyword	Description
Key	The MACsec key or the CKN can be up to 64 characters in length. The key must be of an even number of characters. Entering an odd number of characters will exit the MACsec configuration mode.
Key-string	The MACsec key-string or the CAK can be either 32 characters or 64 characters in length (32 for AES-128, 64 for AES-256).
Lifetime	This field specifies the validity period of a key. It includes a start time, and an expiry time. We recommend you to set the value for expiry time as <i>infinite</i> .

Guidelines for Configuring MACsec Keychain

MACsec keychain management has the following configuration guidelines:

- To establish MKA session, ensure that the MACsec key (CKN) and key-string (CAK) match at both ends.
- MKA protocol uses the latest active key available in the Keychain. This key has the latest Start Time from the existing set of currently active keys. You can verify the values using the **show key chain keychain-name** command.
- Deletion or expiry of current active key brings down the MKA session resulting in traffic hit. We recommend you to configure the keys with infinite lifetime. If fallback is configured, traffic is safeguarded using fallback on expiry or deletion of primary-keychain active key.
- To achieve successful key rollover (CAK-rollover), the new key should be configured such that it is the latest active key, and kicks-in before the current key expires.
- We recommend an overlap of at least one minute for hitless CAK rollover from current key to new key.
- Start time and Expiry time can be configured with future time stamps, which allows bulk configuration for daily CAK rotation without any intervention of management agent.
- From Cisco IOS XR Software Release 7.1.2 Release 7.2.1 and later, the MACsec key IDs (configured through CLI using the **macsec key** command under the key chain configuration mode) are considered to be case insensitive. These key IDs are stored as uppercase letters. For example, a key ID of value 'FF' and of value 'ff' are considered to be the same, and both these key IDs are now stored in uppercase as 'FF'. Whereas, prior to Release 7.1.2, both these values were treated as case sensitive, and hence considered as two separate key IDs. Hence it is recommended to have unique strings as key IDs for a MACsec key chain to avoid flapping of MACsec sessions. However, the support for this case insensitive IDs is applicable only for the configurations done through CLI, and not for configurations done through Netconf protocol.

Also, it is recommended to do a prior check of the MACsec key IDs before upgrading to Release 7.1.2 Release 7.2.1 or later.

Consider a scenario where two MACsec key IDs with the same set of characters (say, ff and FF) are configured under the same key chain.

```
key chain 1
```

```
macsec
key ff
lifetime 02:01:01 may 18 2020 infinite
!
key FF
lifetime 01:01:01 may 18 2020 infinite
```

When you upgrade to Release 7.1.2 Release 7.2.1 or later, only one of these key IDs is retained. That is 'FF', the one that was applied second in this example.

- With NC55-MPA-12T-S MPA, you might experience a traffic drop in these scenarios:
 - A commit replace scenario where multiple MACsec configurations are applied across ports.
 - A process restart (say, in a SMU installation scenario) which results in MACsec rekeying on all the ports.
 - If there are multiple MACsec ports with the same rekey timeout.

SUMMARY STEPS

1. Enter the global configuration mode and provide a name for the MACsec keychain; for example, mac_chain.
2. Enter the MACsec mode.
3. Provide a name for the MACsec key.
4. Enter the key string and the cryptographic algorithm to be used for the key.
5. Enter the validity period for the MACsec key (CKN) also known as the lifetime period.
6. Commit your configuration.

DETAILED STEPS

Procedure

Step 1 Enter the global configuration mode and provide a name for the MACsec keychain; for example, mac_chain.

Example:

```
RP/0/RP0/CPU0:router(config)# key chain mac_chain
```

Step 2 Enter the MACsec mode.

Example:

```
RP/0/RP0/CPU0:router(config-mac_chain)#macsec
```

Step 3 Provide a name for the MACsec key.

The key can be up to 64 characters in length. The key must be of an even number of characters. Entering an odd number of characters will exit the MACsec configuration mode.

Example:

```
RP/0/RP0/CPU0:router(config-mac_chain-MacSec)#key 1234abcd5678
```

You can also configure a fall-back pre-shared key (PSK) to ensure that a PSK is always available to perform MACsec encryption and decryption. The fallback PSK along with the primary PSK ensures that the session remains active even if the primary PSK is mismatched or there is no active key for the primary PSK.

The configured key is the CKN that is exchanged between the peers.

See the guidelines section to know more about the need for a unique key ID for a MACsec key chain.

Note If you are configuring MACsec to interoperate with a MACsec server that is running software prior to Cisco IOS XR Release 6.1.3, then ensure that the MACsec key length is of 64 characters. You can add extra zero characters to the MACsec key so that the length of 64-characters is achieved. If the key length is lesser than 64 characters, authentication will fail.

Step 4 Enter the key string and the cryptographic algorithm to be used for the key.

Example:

The key string is the CAK that is used for ICV validation by the MKA protocol.

! For AES 128-bit encryption

```
RP/0/RP0/CPU0:router(config-mac_chain-MacSec-1234abcd5678)#
key-string 12345678123456781234567812345678 cryptographic-algorithm AES-128-CMAC
```

! For AES 256-bit encryption

```
RP/0/RP0/CPU0:router(config-mac_chain-MacSec-1234abcd5678)#
key-string 1234567812345678123456781234567812345678123456781234567812345678 cryptographic
-algorithm AES-256-CMAC
```

Note In this example, we have used the AES 256-bit encryption algorithm, and therefore, the key string is 64 hexadecimal characters in length. A 256-bit encryption algorithm uses a larger key that requires more rounds of hacking to be cracked. 256-bit algorithms provide better security against large mass security attacks, and include the security provided by 128-bit algorithms.

Step 5 Enter the validity period for the MACsec key (CKN) also known as the lifetime period.

The lifetime period can be configured, with a duration in seconds, as a validity period between two dates (for example, Jan 01 2014 to Dec 31 2014), or with infinite validity.

The key is valid from the time you configure (in HH:MM:SS format). Duration is configured in seconds.

Example:

```
RP/0/RP0/CPU0:router(config-mac_chain-MacSec-1234abcd5678)#lifetime 05:00:00 01
January 2015 duration 1800
```

An example of configuring the lifetime for a defined period:

```
RP/0/RP0/CPU0:router(config-mac_chain-MacSec-1234abcd5678)#lifetime 05:00:00 20
february 2015 12:00:00 30 september 2015
```

An example of configuring the lifetime as infinite:

```
RP/0/RP0/CPU0:router(config-mac_chain-MacSec-1234abcd5678)#lifetime
05:00:00 01 January 2015 infinite
```

Note When a key has expired, the MACsec session is torn down and running the **show macsec mka session** command does not display any information. If you run the **show macsec mka interface detail** command, the output displays ***** No Active Keys Present ***** in the PSK information.

Step 6 Commit your configuration.

Example:

```
RP/0/RP0/CPU0:router(config-mac_chain-MacSec-1234abcd5678)#commit
```

This completes the configuration of the MACsec keychain.

Securing the MACsec Pre-shared Key (PSK) Using Type 6 Password Encryption

Using the Type 6 password encryption feature, you can securely store MACsec plain text key string (CAK) in Type 6 encrypted format.

The primary key is the password or key used to encrypt all plain text MACsec key strings (CAK) in the router configuration with the use of an Advance Encryption Standard (AES) symmetric cipher. The primary key is not stored in the router configuration and cannot be seen or obtained in any way while connected to the router.

The Type 6 password encryption is effective only if a primary key is configured. The Type 6 Password Encryption is currently available on NCS-55A1-36H-SE-S Router.

Configuring a Primary Key and Enabling the Type 6 Password Encryption Feature

You can configure a primary key for Type 6 encryption and enable the Advanced Encryption Standard (AES) password encryption feature for securing the MACsec keys (key string/CAK).

SUMMARY STEPS

1. **key config-key password-encryption [delete]**
2. **configure terminal**
3. **[no] password6 encryption aes**
4. **commit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	key config-key password-encryption [delete] Example:	Configuring a Primary Key

	Command or Action	Purpose
	<p>Configuring a Primary Key</p> <pre>Router# key config-key password-encryption New password Requirements: Min-length 6, Max-length 64 Characters restricted to [A-Z][a-z][0-9] Enter new key : Enter confirm key :</pre> <p>Example:</p> <p>Modifying the Primary Key</p> <pre>Router# key config-key password-encryption New password Requirements: Min-length 6, Max-length 64 Characters restricted to [A-Z][a-z][0-9] Enter old key : Enter new key : Enter confirm key :</pre> <p>Example:</p> <p>Deleting the Primary Key</p> <pre>Router# key config-key password-encryption delete</pre>	<p>Configures a primary key to be used with the Type 6 password encryption feature. The primary key can contain between 6 and 64 alphanumeric characters.</p> <p>Modifying the Primary Key</p> <p>If a primary key is already configured, you are prompted to enter the current primary key before entering a new primary key.</p> <p>Modifying a primary key would re-encrypt all the existing Type 6 format key strings with the new primary key. If Type 6 key strings are present, ensure that the password6 configuration aes command is present to enable re-encryption with the new primary key. Otherwise, the primary key update operation fails.</p> <p>Deleting the Primary Key</p> <p>You can use the delete form of this command to delete the primary key at any time.</p> <p>Note Before deleting the primary key, password6 encryption aes command needs to be disabled using the no password6 encryption aes command followed by configuring the commit command.</p> <p>Caution Primary key deletion would bring down MACSec traffic if MKA sessions were up with Type 6 keys. To avoid traffic disruptions, configure a new set of PSK key pairs [key (CKN) and key string (CAK)] with latest timestamps with the lifetime of infinite validity on both the peers and ensure the successful CAK rekey to the newly configured CKN and CAK.</p>
Step 2	<p>configure terminal</p> <p>Example:</p> <pre>Router# configure terminal Router(config)#</pre>	Enters global configuration mode.
Step 3	<p>[no] password6 encryption aes</p> <p>Example:</p> <pre>Router(config)# password6 encryption aes</pre>	<p>Enables or disables the Type 6 password encryption feature.</p> <p>If you enable the Type 6/AES password encryption feature before configuring a primary key, password encryption will not take place.</p>
Step 4	<p>commit</p> <p>Example:</p> <pre>Router(config)# commit</pre>	Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session.

Example

Configuring MACSec Pre-shared Key (PSK)

Before you begin

Ensure that you have configured a primary key using the **key config-key password-encryption** command and enabled the Type 6 encryption feature using the **password6 encryption aes** command.

SUMMARY STEPS

1. **configure terminal**
2. **key chain** *key chain name* **macsec**
3. **key** *hex string of even length and max 64 bytes*
4. **key-string** *hex string of length 32 bytes or 64 bytes* **cryptographic-algorithm** {**aes-128-cmac** | **aes-256-cmac**}
5. **lifetime** {*hh:mm:ss*} {*1-31*} **month year** **infinite**
6. **commit**
7. **show running-config key chain** *keychain name*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	configure terminal Example: Router# configure terminal Router(config)#	Enters global configuration mode.
Step 2	key chain <i>key chain name</i> macsec Example: Router(config)# key chain kcl macsec Router(config-kcl-MacSec)#	Configures a key chain with the MACsec submode.
Step 3	key <i>hex string of even length and max 64 bytes</i> Example: Router(config-kcl-MacSec)# key 1111 Router(config-kcl-MacSec-1111)#	Configures MACsec CKN as hex string of even length upto 64 bytes. Caution Configuring a hex string of odd number length exits from the MACsec submode. In that case, repeat from Step2 to enter the MACsec submode again.
Step 4	key-string <i>hex string of length 32 bytes or 64 bytes</i> cryptographic-algorithm { aes-128-cmac aes-256-cmac } Example:	Configures a plain text CAK of 32 byte hex string or 64 byte hex string with corresponding MKA (control plane) cryptographic algorithm (aes-128-cmac/ aes-256-cmac).

	Command or Action	Purpose
	Configuring 32 byte hex CAK <pre>Router(config-kcl-MacSec-1111)# key-string 12345678901234567890123456789022 cryptographic-algorithm aes-128-cmac</pre> Example: Configuring 64 byte hex CAK <pre>Router(config-kcl-MacSec-1111)# key-string 1234567890123456789012345678902212345678901234567890123456789022</pre> <pre>cryptographic-algorithm aes-256-cmac</pre>	
Step 5	lifetime <i>{hh:mm:ss} {1-31} month year infinite</i> Example: <pre>Router(config-kcl-MacSec-1111)# lifetime 00:00:00 1 january 2017 infinite</pre>	Configures a valid lifetime for MACsec PSK. Note Without configuring a valid lifetime, MACsec PSK will be an inactive key.
Step 6	commit Example: <pre>Router(config)# commit</pre>	Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session.
Step 7	show running-config key chain <i>keychain name</i> Example: <pre>Router# show running-config key chain kcl key chain kcl macsec key 1111 key-string password6 5d63525a58594657565e6845446842465965554862424c5 95d696554694a424c59655f504a575e6648484c484b4646 535d49675e535a60644e6045654a655f666858414142 cryptographic-algorithm aes-128-cmac lifetime 00:00:00 january 01 2017 infinite ! ! !</pre>	[Optional] Displays the Type 6 encrypted key string.

Example

Creating a User-Defined MACsec Policy

SUMMARY STEPS

1. Enter the global configuration mode, and enter a name (`mac_policy`) for the MACsec policy.

2. Configure the cipher suite to be used for MACsec encryption.
3. Configure the confidentiality offset for MACsec encryption.
4. Enter the key server priority.
5. Configure the security policy parameters, either Must-Secure or Should-Secure.
6. Configure data delay protection under MACsec policy.
7. Configure the replay protection window size.
8. Configure the ICV for the frame arriving on the port.
9. Commit your configuration and exit the global configuration mode.
10. Confirm the MACsec policy configuration.

DETAILED STEPS

Procedure

Step 1 Enter the global configuration mode, and enter a name (`mac_policy`) for the MACsec policy.

Example:

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# macsec-policy mac_policy
```

Step 2 Configure the cipher suite to be used for MACsec encryption.

Example:

```
RP/0/RP0/CPU0:router(config-mac_policy)# cipher-suite GCM-AES-XPN-256
RP/0/RP0/CPU0:router(config-mac_policy)#GCM-AES-128
GCM-AES-256
GCM-AES-XPN-128
GCM-AES-XPN-256
```

Note In this example, we have used the GCM-AES-XPN-256 encryption algorithm. A 256-bit encryption algorithm uses a larger key that requires more rounds of hacking to be cracked. 256-bit algorithms provide better security against large mass security attacks, and include the security provided by 128-bit algorithms. Extended Packet Numbering (XPN) is used to reduce the number of key rollovers while data is sent over high speed links. It is therefore highly recommended to use GCM-AES-XPN-256 encryption algorithm for higher data ports.

Step 3 Configure the confidentiality offset for MACsec encryption.

Example:

```
RP/0/RP0/CPU0:router(config-mac_policy)# conf-offset CONF-OFFSET-30
```

Note We recommend to change the offset value of the **conf-offset** *<offset_value>* command (MACsec encryption command) in Cisco NCS 5500 fixed port routers only when the port is in **admin down** state (that is, when the interface is shut down). Changing the offset value otherwise may result in traffic loss.

Step 4 Enter the key server priority.

You can enter a value between 0-255. Lower the value, higher the preference to be selected as the key server.

In this example, a value of 0 configures the router as the key server, while the other router functions as a key client. The key server generates and maintains the SAK between the two routers. The default key server priority value is 16.

Example:

```
RP/0/RP0/CPU0:router(config-mac_policy)# key-server-priority 0
```

Step 5 Configure the security policy parameters, either Must-Secure or Should-Secure.

Must-Secure: Must-Secure imposes only MACsec encrypted traffic to flow. Hence, until MKA session is not secured, traffic will be dropped.

Example:

```
RP/0/RP0/CPU0:router(config-mac_policy)# security-policy must-secure
```

Should-Secure: Should-Secure allows unencrypted traffic to flow until MKA session is secured. After the MKA session is secured, Should-Secure policy imposes only encrypted traffic to flow.

Example:

```
RP/0/RP0/CPU0:router(config-mac_policy)# security-policy should-secure
```

Table 8: MACsec Security Policies

MKA		Secured MKA Session	Unsecured MKA Session
Security Policy	Must-secure	Encrypted traffic	Traffic drop (no Tx and no Rx)
	Should-secure	Encrypted traffic	Plain text or unencrypted traffic

Step 6 Configure data delay protection under MACsec policy.

Data delay protection allows MKA participants to ensure that the data frames protected by MACsec are not delayed by more than 2 seconds. Each SecY uses MKA to communicate the lowest PN used for transmission with the SAK within two seconds. Traffic delayed longer than 2 seconds are rejected by the interfaces enabled with delay protection.

By default, the data delay protection feature is disabled. Configuring the **delay-protection** command under MACsec-policy attached to MACsec interface will enable the data delay protection feature on that interface.

Note Data delay protection is not supported on Cisco NCS 5700 series fixed port routers and the Cisco NCS 5500 series routers that have the Cisco NC57 line cards installed and operating in the native and compatible modes.

It is also not supported on ports 0 to 31 of NC55-32T16Q4H-A.

Example:

```
RP/0/RP0/CPU0:router# configure terminal
RP/0/RP0/CPU0:router(config)# macsec-policy mpl
RP/0/RP0/CPU0:router(config-macsec-policy)# delay-protection
RP/0/RP0/CPU0:router(config-macsec-policy)# commit
```

Verification:

The following show command output verifies that the data delay protection feature is enabled.

Example:

```
RP/0/RP0/CPU0:router# show macsec mka session interface GigabitEthernet 0/1/0/1 detail
MKA Policy Name      : mpl
Key Server Priority  : 16
Delay Protection     : TRUE
Replay Window Size  : 64
Confidentiality Offset : 0
Algorithm Agility    : 80C201
```

```
SAK Cipher Suite      : (NONE)
MACsec Capability     : 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired       : YES
```

Step 7 Configure the replay protection window size.

Example:

```
RP/0/RP0/CPU0:router(config-mac_policy)# window-size 64
```

This dictates the maximum out-of-sequence frames that are accepted. You can configure a value between 0 and 1024.

Step 8 Configure the ICV for the frame arriving on the port.

Example:

```
RP/0/RP0/CPU0:router(config-mac_policy)# include-icv-indicator
```

This parameter configures inclusion of the optional ICV Indicator as part of the transmitted MACsec Key Agreement PDU (MKPDU). This configuration is necessary for MACsec to interoperate with routers that run software prior to IOS XR version 6.1.3. This configuration is also important in a service provider WAN setup where MACsec interoperates with other vendor MACsec implementations that expect ICV indicator to be present in the MKPDU.

Step 9 Commit your configuration and exit the global configuration mode.

Example:

```
RP/0/RP0/CPU0:router(config-mac_policy)# exit
RP/0/RP0/CPU0:router(config)# commit
RP/0/RP0/CPU0:router(config)# exit
```

Step 10 Confirm the MACsec policy configuration.

Example:

```
RP/0/RP0/CPU0:router# show running-config macsec-policy

macsec-policy mac_policy
conf-offset CONF-OFFSET-30
security-policy must-secure
window-size 64
cipher-suite GCM-AES-XPB-256
key-server-priority 0
include-icv-indicator
```

This completes the configuration of the MACsec policy.



Note

- Small packets might be dropped when Data Delay Protection (DDP) is enabled on many MACsec enabled interfaces of a scaled setup. To avoid this, enable DDP only on the interfaces which are absolutely necessary.
 - For Cisco NCS 5500 Series Routers to interoperate with Cisco ASR9000 Series Routers that are older than Release 6.2.3, configure a user defined MACsec policy with the `policy-exception lacp-in-clear` command to bring up the MKA sessions over bundle interfaces running in LACP modes.
-

Applying MACsec Configuration on an Interface

The MACsec service configuration is applied to the host-facing interface of a CE router.

Guidelines for MACsec Interface Configuration

Following are the guidelines for configuring MACsec interface:

- Configure different keychains for primary and fallback PSKs.
- We do not recommend to update both primary and fallback PSKs simultaneously, because fallback PSK is intended to recover MACsec session on primary key mismatch.
- Although the MACsec **epol destination-address broadcast-address** command under the interface configuration mode is present and configurable on Cisco NCS 5500 Series Routers, the functionality is not yet supported.
- When using MACsec, we recommend you adjust the maximum transmission unit (MTU) of an interface to accommodate the MACsec overhead. Configuring MTU value on an interface allows protocols to do MTU negotiation including MACsec overhead. For instance, if the default MTU is 1514 bytes, configure the MTU to 1546 bytes (1514 + 32).
- The minimum MTU for IS-IS protocol on the MACsec interface is 1546 bytes.
- To enable MACsec on bundles:
 - Enable MACsec on all bundle members.
 - MACsec peers running IOS-XR version 24.1.1 or higher:
 - For routing protocols running on the bundle interface, configure **impose-overhead-on-bundle** in the MACsec policy to adjust the bundle interface MTU with MACsec overhead.
 - MACsec peers running IOS-XR versions prior to 24.1.1:
 - We recommend configuring the maximum possible MTU on the bundle interface.
 - The MTU configurations must account for the maximum packet size of the protocols running on the bundle interface and 32 bytes of MACsec overhead.
 - For IS-IS protocol running on the bundle interface, hello-padding must be disabled.

SUMMARY STEPS

1. Enter the global configuration mode.
2. Enter the interface configuration mode.
3. Apply the MACsec configuration on an interface.
4. Commit your configuration.

DETAILED STEPS

Procedure

Step 1 Enter the global configuration mode.

Example:

```
RP/0/RP0/CPU0:router# configure
```

Step 2 Enter the interface configuration mode.

Example:

```
RP/0/RP0/CPU0:router(config)# interface Te0/3/0/1/4
```

Step 3 Apply the MACsec configuration on an interface.

MACsec PSK Configuration

To apply MACsec PSK configuration on an interface, use the following command.

Example:

```
RP/0/RP0/CPU0:router(config-if)# macsec psk-keychain mac_chain policy mac_policy
RP/0/RP0/CPU0:router(config-if)# exit
```

To apply MACsec configuration on a physical interface without the MACsec policy, use the following command.

Example:

```
RP/0/RP0/CPU0:router(config-if)# macsec psk-keychain script_key_chain2
RP/0/RP0/CPU0:router(config-if)# exit
```

MACsec Fallback PSK Configuration

To apply MACsec configuration on a physical interface with a fallback PSK, use the following command.

Example:

```
RP/0/RP0/CPU0:router(config-if)# macsec psk-keychain mac_chain fallback-psk-keychain fallback_mac_chain
policy mac_policy
RP/0/RP0/CPU0:router(config-if)# exit
```

It is optional to configure a fallback PSK. If a fallback PSK is configured, the fallback PSK along with the primary PSK ensures that the session remains active even if the primary PSK is mismatched, or there is no active key for the primary PSK.

Step 4 Commit your configuration.

Example:

```
RP/0/RP0/CPU0:router(config)# commit
```

MACsec Policy Exceptions

By default, the MACsec security policy uses **must-secure** option, that mandates data encryption. Hence, the packets cannot be sent in clear-text format. To optionally bypass the MACsec encryption or decryption for Link Aggregation Control Protocol (LACP) packets, and to send the packets in clear-text format, use the **policy-exception lacp-in-clear** command in macsec-policy configuration mode. This functionality is beneficial in scenarios such as, in a network topology with three nodes, where bundles are terminated at the middle node, whereas MACsec is terminated at the end nodes.

This MACsec policy exception is also beneficial in interoperability scenarios where the node at the other end expects the data packets to be in clear text.

From Cisco IOS XR Software Release 7.3.1 and later, an alternative option, **allow**, is introduced under the macsec-policy configuration mode, that allows packets to be sent in clear-text format. You can use the **allow lacp-in-clear** command for LACP packets.

How to Create MACsec Policy Exception



Note The **policy-exception lacp-in-clear** command under macsec-policy configuration mode is deprecated. Hence, it is recommended to use the **allow lacp-in-clear** command instead, to allow LACP packets in clear-text format.

Configuration Example

Using the **policy-exception** command:

```
Router#configure
Router(config)#macsec-policy test-macsec-policy
Router(config-macsec-policy)#policy-exception lacp-in-clear
Router(config-macsec-policy)#commit
```

Using the **allow** command:

```
Router#configure
Router(config)#macsec-policy test-macsec-policy
Router(config-macsec-policy)#allow lacp-in-clear
Router(config-macsec-policy)#commit
```

Running Configuration

With the **policy-exception** command:

```
Router#show run macsec-policy test-macsec-policy
macsec-policy test-macsec-policy
  policy-exception lacp-in-clear
  security-policy should-secure
  include-icv-indicator
  sak-rekey-interval seconds 120
!
```

With the **allow** command:

```
Router#show run macsec-policy test-macsec-policy
macsec-policy test-macsec-policy
  allow lacp-in-clear
  security-policy should-secure
  include-icv-indicator
  sak-rekey-interval seconds 120
!
```

Associated Commands

- **policy-exception lacp-in-clear**
- **allow lacp-in-clear**

Verifying MACsec Encryption on IOS XR

MACsec encryption on IOS XR can be verified by running relevant commands in the Privileged Executive Mode. The verification steps are the same for MACsec encryption on L2VPN or L3VPN network.



Note With the introduction of active fallback functionality in Cisco IOS XR Software Release 7.1.2, the output of various MACsec show commands include the fallback PSK entry as well.

To verify if MACsec encryption has been correctly configured, follow these steps.

SUMMARY STEPS

1. Verify the MACsec policy configuration.
2. Verify the MACsec configuration on the respective interface.
3. Verify whether the interface of the router is peering with its neighbor after MACsec configuration. The MACsec PSK validation detects inconsistency or mismatch of primary and fallback keys (CAK) being used by MKA, allowing operators to rectify the mismatch.
4. Verify whether the MKA session is secured with MACsec on the respective interface.
5. Verify the MACsec session counter statistics.

DETAILED STEPS

Procedure

Step 1 Verify the MACsec policy configuration.

Example:

```
RP/0/RP0/CPU0:router#show macsec policy mac_policy
```

```
=====
Policy          Cipher          Key-Svr        Window  Conf
```



```

name           Suite           Priority   Size   Offset
=====

```

```

mac_policy GCM-AES-XPN-256  0         64     30

```

If the values you see are different from the ones you configured, then check your configuration by running the **show run macsec-policy** command.

Step 2 Verify the MACsec configuration on the respective interface.

You can verify the MACsec encryption on the configured interface bundle (MPLS network).

Example:

Before the introduction of active fallback functionality:

```

RP/0/RP0/CPU0:router#show macsec mka summary

```

```

NODE: node0_0_CPU0

```

```

=====
Interface      Status      Cipher Suite      KeyChain
=====

```

```

Fo0/0/0/1/0   Secured    GCM-AES-XPN-256   mac_chain

```

```

Total MACSec Sessions : 1
  Secured Sessions : 1
  Pending Sessions : 0

```

```

RP/0/RP0/CPU0:router# show macsec mka session interface Fo0/0/0/1/0

```

```

=====
Interface      Local-TxSCI      # Peers      Status      Key-Server
=====
Fo0/0/0/1/0    d46d.5023.3709/0001    1            Secured      YES

```

With the introduction of active fallback functionality:

The following is a sample output that displays active fallback PSK entry as well:

```

RP/0/RP0/CPU0:router#show macsec mka summary

```

```

NODE: node0_0_CPU0

```

```

=====
Interface-Name  Status      Cipher-Suite      KeyChain      PSK/EAP      CKN
=====
Fo0/0/0/1/0    Secured    GCM-AES-XPN-128   test2         PRIMARY      5555
Fo0/0/0/1/0    Active     GCM-AES-XPN-128   test2f        FALLBACK     5556

```

```

Total MACSec Sessions : 2
  Secured Sessions : 1
  Pending Sessions : 0
  Active Sessions : 1

```

```

RP/0/RP0/CPU0:router#show macsec mka session interface Fo0/0/0/1/0

```

```

=====
Interface-Name  Local-TxSCI      #Peers      Status      Key-Server      PSK/EAP      CKN
=====

```

```

Fo0/0/0/1/0      d46d.5023.3709/0001    1    Secured    YES    PRIMARY    5555
Fo0/0/0/1/0      d46d.5023.3709/0001    1    Active     YES    FALLBACK   5556

```

The **Status** field in the output confirms that the respective interface is **Secured**. If MACsec encryption is not successfully configured, you will see a status such as **Pending** or **Init**.

Run the **show run macsec-policy** command in the privileged executive mode to troubleshoot the configuration entered.

Step 3

Verify whether the interface of the router is peering with its neighbor after MACsec configuration. The MACsec PSK validation detects inconsistency or mismatch of primary and fallback keys (CAK) being used by MKA, allowing operators to rectify the mismatch.

Example:

The **show macsec mka session interface interface detail** command carries the Peer Validation status in the **Peer CAK** field. The values of this field can be either *Match* or *Mismatch*.

Before the introduction of active fallback functionality:

The following show command output verifies that the primary and fallback keys (CAK) are matched on both peer ends.

```

• RP/0/RP0/CPU0:router#show macsec mka session detail
Peers Status:
  Last Tx MKPDU      : 2017 Sep 02 11:24:52.369
  Peer Count         : 1
  RxSCI              : 008A960060900001
  MI                 : C2213E81C953A202C08DB999
  Peer CAK           : Match
  Latest Rx MKPDU    : 2017 Sep 02 11:24:53.360
Fallback Data:
  CKN                : ABCD
  MI                 : 84E724B4BA07CE414FEA84EF
  MN                 : 8
Peers Status:
  Last Tx MKPDU      : 2017 Sep 02 11:24:52.369
  Peer Count         : 1
  RxSCI              : 008A960060900001
  MI                 : D2B902453F90389BD3385F84
  Peer CAK           : Match
  Latest Rx MKPDU    : 2017 Sep 02 11:24:53.360

```

• Syslog

```

%L2-MKA-6-MKPDU_ICV_SUCCESS: (Hu0/5/0/1), ICV verification success for RxSCI(008a.9600.6090/0001),
CKN(1000)
%L2-MKA-6-FALLBACK_PSK_MKPDU_ICV_SUCCESS: (Hu0/5/0/1), ICV verification success for
RxSCI(008a.9600.6090/0001), CKN(FFFF)

```

The following show command output verifies that the primary and fallback keys (CAK) are mismatched on both peer ends.

```

• RP/0/RP0/CPU0:router#show macsec mka session detail
Peers Status:
  Last Tx MKPDU      : 2017 Sep 02 11:24:52.369
  Peer Count         : 1
  RxSCI              : 008A960060900001
  MI                 : C2213E81C953A202C08DB999
  Peer CAK           : Mismatch
  Latest Rx MKPDU    : 2017 Sep 02 11:24:53.360
Fallback Data:
  CKN                : ABCD
  MI                 : 84E724B4BA07CE414FEA84EF
  MN                 : 8
Peers Status:

```

```

Last Tx MKPDU      : 2017 Sep 02 11:24:52.369
Peer Count        : 1
RxSCI             : 008A960060900001
MI               : D2B902453F90389BD3385F84
Peer CAK          : Mismatch
Latest Rx MKPDU   : 2017 Sep 02 11:24:53.360

```

• Syslog

```

%L2-MKA-3-MKPDU_ICV_FAILURE: (Hu0/5/0/1), ICV verification failed for RxSCI(008a.9600.6090/0001),
CKN(1111)
%L2-MKA-3-FALLBACK_PSK_MKPDU_ICV_FAILURE: (Hu0/5/0/1), ICV verification failed for
RxSCI(008a.9600.6090/0001), CKN(9999)

```

The **#Peers** field in the following output confirms the presence of the peer you have configured on the physical interface, **Fo0/0/0/1/0**. If the number of peers is not reflected accurately in this output, run the **show run** command and verify the peer configuration on the interface.

```

RP/0/RP0/CPU0:router#show macsec mka session

NODE: node0_0_CPU0

=====
Interface      Local-TxSCI          # Peers  Status  Key-Server
=====
Fo0/0/0/1/0   001d.e5e9.aa39/0005    1        Secured  YES

```

Note If the MKA session status is shown as **Secured** with **0 (Zero)** peer count, this means that the link is locally secured (Tx). This is because of MKA peer loss caused by **No Rx Packets (MKA Packet)** from that peer.

With the introduction of active fallback functionality:

The following show command output verifies that the primary and fallback keys (CAK) are matched on both peer ends.

```

• RP/0/RP0/CPU0:router#show macsec mka session detail
Tue May 18 13:21:54.608 UTC
NODE: node0_2_CPU0

MKA Detailed Status for MKA Session
=====
Status: Secured - Secured MKA Session with MACsec

Local Tx-SCI           : 008a.96d6.194c/0001
Local Tx-SSCI         : 2
Interface MAC Address  : 008a.96d6.194c
MKA Port Identifier    : 1
Interface Name         : Hu0/2/0/11
CAK Name (CKN)        : 2111
CA Authentication Mode : PRIMARY-PSK
Keychain               : test1
Member Identifier (MI) : 69B39E87B3CBA673401E9891
Message Number (MN)   : 162
Authenticator         : NO
Key Server             : YES
MKA Cipher Suite      : AES-128-CMAC
Configured MACSec Cipher Suite : GCM-AES-XPN-128
Key Distribution Mode  : SAK

Latest SAK Status     : Rx & Tx
Latest SAK AN         : 0
Latest SAK KI (KN)   : 69B39E87B3CBA673401E989100000001 (1)

```

```

Old SAK Status           : FIRST-SAK
Old SAK AN              : 0
Old SAK KI (KN)        : FIRST-SAK (0)

SAK Transmit Wait Time  : 0s (Not waiting for any peers to respond)
SAK Retire Time         : 0s (No Old SAK to retire)
Time to SAK Rekey       : 551s
Time to exit suspension : NA

MKA Policy Name         : P12
Key Server Priority     : 20
Delay Protection        : TRUE
Replay Window Size     : 100
Include ICV Indicator   : TRUE
Confidentiality Offset  : 0
Algorithm Agility       : 80C201
SAK Cipher Suite        : 0080C20001000003 (GCM-AES-XPN-128)
MACsec Capability       : 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired          : YES

```

```

# of MACsec Capable Live Peers      : 1
# of MACsec Capable Live Peers Responded : 1

```

```

# of MACSec Suspended Peers        : 0

```

Live Peer List:

```

-----
MI              MN              Rx-SCI          SSCI  KS-Priority
-----
42A78BD6243539E917B8C6B2  101      7061.7bea.1df4/0001  1      20

```

Potential Peer List:

```

-----
MI              MN              Rx-SCI          SSCI  KS-Priority
-----

```

Suspended Peer List:

```

-----
Rx-SCI          SSCI
-----

```

Peers Status:

```

Last Tx MKPDU      : 2021 May 18 13:21:54.347
Peer Count         : 1

```

```

RxSCI              : 70617BEA1DF40001
MI                 : 42A78BD6243539E917B8C6B2
Peer CAK           : Match
Latest Rx MKPDU    : 2021 May 18 13:21:54.574

```

MKA Detailed Status for MKA Session

```

=====

```

```

Status: Active - Marked Peer as Live (Waiting for SAK generation/distribution)

```

```

Local Tx-SCI       : 008a.96d6.194c/0001
Local Tx-SSCI      : 2
Interface MAC Address : 008a.96d6.194c
MKA Port Identifier : 1
Interface Name      : Hu0/2/0/11
CAK Name (CKN)     : 2000
CA Authentication Mode : FALLBACK-PSK
Keychain           : test1f
Member Identifier (MI) : 8F59AD6021FA3E2D5F9E6231
Message Number (MN) : 160

```

```

Authenticator           : NO
Key Server              : YES
MKA Cipher Suite       : AES-128-CMAC
Configured MACSec Cipher Suite : GCM-AES-XPN-128
Key Distribution Mode   : SAK

Latest SAK Status      : Rx & Tx
Latest SAK AN          : 0
Latest SAK KI (KN)    : 69B39E87B3CBA673401E98910000001 (1)
Old SAK Status         : FIRST-SAK
Old SAK AN             : 0
Old SAK KI (KN)       : FIRST-SAK (0)

SAK Transmit Wait Time : 0s (Not waiting for any peers to respond)
SAK Retire Time        : 0s (No Old SAK to retire)
Time to SAK Rekey      : 551s
Time to exit suspension : NA

MKA Policy Name        : P12
Key Server Priority     : 20
Delay Protection       : TRUE
Replay Window Size    : 100
Include ICV Indicator  : TRUE
Confidentiality Offset : 0
Algorithm Agility      : 80C201
SAK Cipher Suite       : 0080C20001000003 (GCM-AES-XPN-128)
MACsec Capability      : 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired        : YES

# of MACsec Capable Live Peers      : 1
# of MACsec Capable Live Peers Responded : 0

# of MACSec Suspended Peers        : 0

Live Peer List:
-----
          MI              MN              Rx-SCI              SSCI  KS-Priority
-----
1BB9428C721F6EE3E538C942      99      7061.7bea.1df4/0001      1      20

Potential Peer List:
-----
          MI              MN              Rx-SCI              SSCI  KS-Priority
-----

Suspended Peer List:
-----
          Rx-SCI              SSCI
-----

Peers Status:
Last Tx MKPDU      : 2021 May 18 13:21:54.346
Peer Count         : 1

RxSCI              : 70617BEA1DF40001
MI                 : 1BB9428C721F6EE3E538C942
Peer CAK           : Match
Latest Rx MKPDU    : 2021 May 18 13:21:54.574
RP/0/RP0/CPU0:router#
    
```

The following show command output verifies that the primary and fallback keys (CAK) are mismatched on both peer ends.

```

• RP/0/RP0/CPU0:router#show macsec mka session detail
Tue May 18 13:37:21.473 UTC
NODE: node0_2_CPU0

MKA Detailed Status for MKA Session
=====
Status: Init - Searching for Peer (Waiting to receive first Peer MKPDU)

Local Tx-SCI           : 008a.96d6.194c/0001
Local Tx-SSCI          : 0
Interface MAC Address  : 008a.96d6.194c
MKA Port Identifier    : 1
Interface Name         : Hu0/2/0/11
CAK Name (CKN)        : 5555
CA Authentication Mode : PRIMARY-PSK
Keychain               : test2
Member Identifier (MI) : F124CAACB5D80F8976E03B9D
Message Number (MN)   : 158
Authenticator         : NO
Key Server             : YES
MKA Cipher Suite      : AES-128-CMAC
Configured MACSec Cipher Suite : GCM-AES-XPN-128
Key Distribution Mode  : NONE

Latest SAK Status      : No Rx, No Tx
Latest SAK AN          : 0
Latest SAK KI (KN)    : FIRST-SAK-INITIALIZING (0)
Old SAK Status         : FIRST-SAK
Old SAK AN             : 0
Old SAK KI (KN)       : FIRST-SAK (0)

SAK Transmit Wait Time : 0s (Not waiting for any peers to respond)
SAK Retire Time        : 0s (No Old SAK to retire)
Time to SAK Rekey     : NA
Time to exit suspension : NA

MKA Policy Name       : P12
Key Server Priority   : 20
Delay Protection      : TRUE
Replay Window Size   : 100
Include ICV Indicator : TRUE
Confidentiality Offset : 0
Algorithm Agility     : 80C201
SAK Cipher Suite      : (NONE)
MACsec Capability     : 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired       : YES

# of MACsec Capable Live Peers      : 0
# of MACsec Capable Live Peers Responded : 0

# of MACSec Suspended Peers        : 0

Live Peer List:
-----
MI              MN              Rx-SCI          SSCI  KS-Priority
-----

Potential Peer List:
-----
MI              MN              Rx-SCI          SSCI  KS-Priority
-----

Suspended Peer List:
-----

```

```

Rx-SCI          SSCI
-----
Peers Status:
Last Tx MKPDU   : 2021 May 18 13:37:21.061
Peer Count      : 1

RxSCI           : 70617BEA1DF40001
MI              : C816E45386574DF62D7D6A20
Peer CAK        : Mismatch
Latest Rx MKPDU : 2021 May 18 13:37:21.189

MKA Detailed Status for MKA Session
=====
Status: Init - Searching for Peer (Waiting to receive first Peer MKPDU)

Local Tx-SCI    : 008a.96d6.194c/0001
Local Tx-SSCI  : 0
Interface MAC Address : 008a.96d6.194c
MKA Port Identifier : 1
Interface Name   : Hu0/2/0/11
CAK Name (CKN)  : 5556
CA Authentication Mode : FALLBACK-PSK
Keychain        : test2f
Member Identifier (MI) : 2D4A9EF08A211A9525C653E4
Message Number (MN)  : 158
Authenticator    : NO
Key Server      : YES
MKA Cipher Suite  : AES-128-CMAC
Configured MACSec Cipher Suite : GCM-AES-XPN-128
Key Distribution Mode : NONE

Latest SAK Status : No Rx, No Tx
Latest SAK AN     : 0
Latest SAK KI (KN) : FIRST-SAK-INITIALIZING (0)
Old SAK Status    : FIRST-SAK
Old SAK AN        : 0
Old SAK KI (KN)  : FIRST-SAK (0)

SAK Transmit Wait Time : 0s (Not waiting for any peers to respond)
SAK Retire Time        : 0s (No Old SAK to retire)
Time to SAK Rekey     : NA
Time to exit suspension : NA

MKA Policy Name      : P12
Key Server Priority   : 20
Delay Protection     : TRUE
Replay Window Size   : 100
Include ICV Indicator : TRUE
Confidentiality Offset : 0
Algorithm Agility    : 80C201
SAK Cipher Suite     : (NONE)
MACsec Capability    : 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired       : YES

# of MACsec Capable Live Peers : 0
# of MACsec Capable Live Peers Responded : 0

# of MACSec Suspended Peers : 0

Live Peer List:
-----
MI              MN              Rx-SCI          SSCI  KS-Priority
-----

```



```
SAK Transmit Wait Time : 0s (Not waiting for any peers to respond)
SAK Retire Time       : 0s (No Old SAK to retire)
Time to SAK Rekey    : NA
MKA Policy Name      : *DEFAULT POLICY*
Key Server Priority   : 16
Replay Window Size   : 64
Confidentiality Offset : 0
Algorithm Agility     : 80C201
SAK Cipher Suite      : 0080C20001000004 (GCM-AES-XPN-256)
MACsec Capability     : 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired       : YES
```

```
# of MACsec Capable Live Peers      : 1
# of MACsec Capable Live Peers Responded : 0
```

Live Peer List:

```
MI              MN              Rx-SCI (Peer)      SSCI KS-Priority
-----
```

With the introduction of active fallback functionality:

```
RP/0/RP0/CPU0:router#show macsec mka session interface Fo0/0/0/1/0 detail Tue May 18 13:23:29.935
UTC
Tue May 18 13:23:29.935 UTC
```

MKA Detailed Status for MKA Session

=====

Status: Secured - Secured MKA Session with MACsec

```
Local Tx-SCI          : 008a.96d6.194c/0001
Local Tx-SSCI        : 2
Interface MAC Address : 008a.96d6.194c
MKA Port Identifier   : 1
Interface Name        : Hu0/2/0/11
CAK Name (CKN)       : 2111
CA Authentication Mode : PRIMARY-PSK
Keychain              : test1
Member Identifier (MI) : 69B39E87B3CBA673401E9891
Message Number (MN)   : 352
Authenticator         : NO
Key Server            : YES
MKA Cipher Suite      : AES-128-CMAC
Configured MACSec Cipher Suite : GCM-AES-XPN-128
Key Distribution Mode  : SAK

Latest SAK Status     : Rx & Tx
Latest SAK AN         : 0
Latest SAK KI (KN)    : 69B39E87B3CBA673401E989100000001 (1)
Old SAK Status        : FIRST-SAK
Old SAK AN            : 0
Old SAK KI (KN)       : FIRST-SAK (0)

SAK Transmit Wait Time : 0s (Not waiting for any peers to respond)
SAK Retire Time       : 0s (No Old SAK to retire)
Time to SAK Rekey    : 456s
Time to exit suspension : NA

MKA Policy Name      : P12
Key Server Priority   : 20
Delay Protection     : TRUE
Replay Window Size   : 100
Include ICV Indicator : TRUE
Confidentiality Offset : 0
Algorithm Agility     : 80C201
SAK Cipher Suite      : 0080C20001000003 (GCM-AES-XPN-128)
```

```
MACsec Capability           : 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired             : YES
```

```
# of MACsec Capable Live Peers      : 1
# of MACsec Capable Live Peers Responded : 1
```

```
# of MACSec Suspended Peers        : 0
```

Live Peer List:

```
-----
      MI                MN                Rx-SCI                SSCI  KS-Priority
-----
42A78BD6243539E917B8C6B2      290      7061.7bea.1df4/0001      1      20
-----
```

Potential Peer List:

```
-----
      MI                MN                Rx-SCI                SSCI  KS-Priority
-----
```

Suspended Peer List:

```
-----
      Rx-SCI                SSCI
-----
```

Peers Status:

```
Last Tx MKPDU           : 2021 May 18 13:23:29.588
Peer Count              : 1
```

```
RxSCI                  : 70617BEA1DF40001
MI                     : 42A78BD6243539E917B8C6B2
Peer CAK                : Match
Latest Rx MKPDU        : 2021 May 18 13:23:29.847
```

MKA Detailed Status for MKA Session

```
=====
Status: Active - Marked Peer as Live (Waiting for SAK generation/distribution)
```

```
Local Tx-SCI           : 008a.96d6.194c/0001
Local Tx-SSCI          : 2
Interface MAC Address   : 008a.96d6.194c
MKA Port Identifier    : 1
Interface Name         : Hu0/2/0/11
CAK Name (CKN)        : 2000
CA Authentication Mode : FALLBACK-PSK
Keychain               : test1f
Member Identifier (MI) : 8F59AD6021FA3E2D5F9E6231
Message Number (MN)   : 350
Authenticator         : NO
Key Server             : YES
MKA Cipher Suite       : AES-128-CMAC
Configured MACSec Cipher Suite : GCM-AES-XPN-128
Key Distribution Mode   : SAK
```

```
Latest SAK Status      : Rx & Tx
Latest SAK AN          : 0
Latest SAK KI (KN)    : 69B39E87B3CBA673401E989100000001 (1)
Old SAK Status         : FIRST-SAK
Old SAK AN             : 0
Old SAK KI (KN)       : FIRST-SAK (0)
```

```
SAK Transmit Wait Time : 0s (Not waiting for any peers to respond)
SAK Retire Time        : 0s (No Old SAK to retire)
Time to SAK Rekey     : 456s
Time to exit suspension : NA
```

```

MKA Policy Name           : P12
Key Server Priority       : 20
Delay Protection         : TRUE
Replay Window Size      : 100
Include ICV Indicator    : TRUE
Confidentiality Offset  : 0
Algorithm Agility       : 80C201
SAK Cipher Suite        : 0080C20001000003 (GCM-AES-XPN-128)
MACsec Capability       : 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired          : YES

```

```

# of MACsec Capable Live Peers      : 1
# of MACsec Capable Live Peers Responded : 0

```

```

# of MACSec Suspended Peers        : 0

```

Live Peer List:

```

-----
          MI              MN              Rx-SCI              SSCI  KS-Priority
-----
1BB9428C721F6EE3E538C942    288      7061.7bea.1df4/0001    1      20

```

Potential Peer List:

```

-----
          MI              MN              Rx-SCI              SSCI  KS-Priority
-----

```

Suspended Peer List:

```

-----
          Rx-SCI              SSCI
-----

```

Peers Status:

```

Last Tx MKPDU           : 2021 May 18 13:23:29.587
Peer Count              : 1

```

```

RxSCI                   : 70617BEA1DF40001
MI                       : 1BB9428C721F6EE3E538C942
Peer CAK                 : Match
Latest Rx MKPDU         : 2021 May 18 13:23:29.847

```

RP/0/RP0/CPU0:router#

The **Status** field in the output verifies if the MKA session is secured with MACsec encryption. The output also displays information about the interface and other MACsec parameters.

Step 5 Verify the MACsec session counter statistics.

Example:

```
RP/0/RP0/CPU0:router# show macsec mka statistics interface Fo0/0/0/1/0
```

```

MKA Statistics for Session on interface (Fo0/0/0/1/0)
=====
Reauthentication Attempts.. 0

CA Statistics
Pairwise CAKs Derived... 0
Pairwise CAK Rekeys..... 0
Group CAKs Generated.... 0
Group CAKs Received..... 0

```

```

SA Statistics
SAKs Generated..... 3
SAKs Rekeyed..... 2
SAKs Received..... 0
SAK Responses Received.. 3

MKPDU Statistics
MKPDUs Transmitted..... 5425
"Distributed SAK".. 8
"Distributed CAK".. 0
MKPDUs Validated & Rx... 4932
"Distributed SAK".. 0
"Distributed CAK".. 0

MKA IDB Statistics
MKPDUs Tx Success..... 5425
MKPDUs Tx Fail..... 0
MKPDUS Tx Pkt build fail... 0
MKPDUs Rx CA Not found.... 0
MKPDUs Rx Error..... 0
MKPDUs Rx Success..... 4932

MKPDU Failures
MKPDU Rx Validation (ICV)..... 0
MKPDU Rx Bad Peer MN..... 0
MKPDU Rx Non-recent Peerlist MN..... 0
MKPDU Rx Drop SAKUSE, KN mismatch..... 0
MKPDU Rx Drop SAKUSE, Rx Not Set..... 0
MKPDU Rx Drop SAKUSE, Key MI mismatch.. 0
MKPDU Rx Drop SAKUSE, AN Not in Use.... 0
MKPDU Rx Drop SAKUSE, KS Rx/Tx Not Set. 0

SAK Failures
SAK Generation..... 0
Hash Key Generation..... 0
SAK Encryption/Wrap..... 0
SAK Decryption/Unwrap..... 0

```

The counters display the MACsec PDUs transmitted, validated, and received. The output also displays transmission errors, if any.

This completes the verification of MACsec encryption on the IOS-XR.

Verifying MACsec Encryption on NCS 5500

MACsec encryption on the router hardware can be verified by running relevant commands in the Privileged Executive Mode.

To verify if MACsec encryption has been correctly configured, follow these steps.

SUMMARY STEPS

1. Verify the MACsec encryption and hardware interface descriptor block (IDB) information on the interface.
2. Use the IDB handle retrieved from Step 1 to verify the platform hardware information.
3. Use the Transmitter SA retrieved from Step 2 to verify the MACsec SA information programmed in the hardware.

4. Verify the MACsec Secure Channel (SC) information programmed in the hardware.

DETAILED STEPS

Procedure

- Step 1** Verify the MACsec encryption and hardware interface descriptor block (IDB) information on the interface.

Example:

```
RP/0/RP0/CPU0:router# show macsec ea idb interface Fo0/0/0/1/0
```

```
IDB Details:
if_sname : Fo0/0/0/1/0
if_handle : 0x3480
Replay window size : 64
Local MAC : 00:1d:e5:e9:aa:39
Rx SC Option(s) : Validate-Frames Replay-Protect
Tx SC Option(s) : Protect-Frames Always-Include-SCI
Security Policy : MUST SECURE
Sectag offset : 8
Rx SC 1
Rx SCI : 001de5e9b1bf0019
Peer MAC : 00:1d:e5:e9:b1:bf
Stale : NO
SAK Data
SAK[0] : ***
SAK Len : 32
HashKey[0] : ***
HashKey Len : 16
Conf offset : 30
Cipher Suite : GCM-AES-XPN-256
CtxSalt[0] : 83 c3 7b ad 7b 6f 63 16 09 8f f3 d2
Rx SA Program Req[0]: 2015 Oct 09 15:20:53.082
Rx SA Program Rsp[0]: 2015 Oct 09 15:20:53.092

Tx SC
Tx SCI : 001de5e9aa39001a
Active AN : 0
Old AN : 255
Next PN : 1, 0, 0, 0
SAK Data
SAK[0] : ***
SAK Len : 32
HashKey[0] : ***
HashKey Len : 16
Conf offset : 30
Cipher Suite : GCM-AES-XPN-256
CtxSalt[0] : 83 c3 7b ae 7b 6f 63 16 09 8f f3 d2
Tx SA Program Req[0]: 2015 Oct 09 15:20:55.053
Tx SA Program Rsp[0]: 2015 Oct 09 15:20:55.064
```

The **if_handle** field provides the IDB instance location.

The **Replay window size** field displays the configured window size.

The **Security Policy** field displays the configured security policy.

The **Local Mac** field displays the MAC address of the router.

The **Peer Mac** field displays the MAC address of the peer. This confirms that a peer relationship has been formed between the two routers.

Step 2 Use the IDB handle retrieved from Step 1 to verify the platform hardware information.

Example:

```
RP/0/RP0/CPU0:router# show macsec platform hardware
idb location 0/0/CPU0 | b 3480

if_handle : 0x00003480
NPPort : 099 [0x063]
LdaPort : 016 [0x010] SerdesPort : 000 [0x000]
NetSoftPort : 061 [0x03d] SysSoftPort : 062 [0x03e]
Active AN : 0x00000000 Idle AN : 0x000000ff
Match-All Tx SA : 0x80010001 Match-All Rx SA : 0x00010001
Match-All Tx Flow : 0x80000003 Match-All Rx Flow : 0x00000003
Bypass Tx SA : 0x80000000 Bypass Rx SA : 0x00000000
Tx SA[0] : 0x80020002 Tx Flow[0] : 0x8000000c
Tx SA[1] : 0xffffffff Tx Flow[1] : 0xffffffff
Tx SA[2] : 0xffffffff Tx Flow[2] : 0xffffffff
Tx SA[3] : 0xffffffff Tx Flow[3] : 0xffffffff
Rx SA[0] : 0x00020002 Rx Flow[0] : 0x0000000c
Rx SA[1] : 0xffffffff Rx Flow[1] : 0xffffffff
Rx SA[2] : 0xffffffff Rx Flow[2] : 0xffffffff
Rx SA[3] : 0xffffffff Rx Flow[3] : 0xffffffff
```

Step 3 Use the Transmitter SA retrieved from Step 2 to verify the MACsec SA information programmed in the hardware.

Example:

```
RP/0/RP0/CPU0:router# show macsec platform hardware sa
0x80020002 interface Fo0/0/0/1/0 location 0/0/CPU0

MACsec HW SA Details:
Action Type : 0x00000003
Direction : Egress
Dest Port : 0x00000000
Conf Offset : 00000030
Drop Type : 0x00000002
Drop NonResvd : 0x00000000
SA In Use : YES
ConfProtect : YES
IncludeSCI : YES
ProtectFrame : YES
UseEs : NO
UseSCB : NO
SCI : 00 1d e5 e9 aa 39 00 05
Replay Window : 64 MacsecCryptoAlgo : 7
Direction : Egress AN : 0
AES Key Len : 256 X-Packet Number : 0x0000000000000000
CtxSalt : f8d88dc3e1c5e6a94ca2299
```

The output displays the details of the encryption, such as the AES key, the Auth key, and other parameters.

Step 4 Verify the MACsec Secure Channel (SC) information programmed in the hardware.

Example:

```
RP/0/RP0/CPU0:router# show macsec platform hardware msc
interface Fo0/0/0/1/0 location 0/0/CPU0
```

```
MACsec HW Cfg Details:
Mode : 0x5
Counter Clear on Read : 0x0
SA Fail Mask : 0xffff
Global SecFail Mask : 0xffffffff
Latency : 0xff
StaticBypass : 0x0
Should secure : 0x0
Global Frame Validation : 0x2
Ctrl Pkt CC Bypass : 0x1
NonCtrl Pkt CC Bypass : 0x1
Sequence Number Threshold : 0xbfffffff8
Sequence Number Threshold 64bit : 0x000002fffffffffd
Non Matching Non Control Pkts Programming
  Untagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
  Tagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
  BadTagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
  KayTagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
Non Matching Control Pkts Programming
  Untagged : Bypass: 0x1 DestPort : 0x2, DropType : 0xffffffff
  Tagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
  BadTagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
  KayTagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
```

This completes the verification of MACsec encryption on the router hardware.

This completes the configuration and verification of MACsec encryption.

MACsec SecY Statistics

The following methods are used to query MACsec SecY statistics such as, encryption, decryption, and the hardware statistics.

- CLI
- SNMP MIB

Querying SNMP Statistics Using CLI

The following example shows how to query SNMP statistics using a CLI. Use the **show macsec secy statistics interface *interface name*** command to display the MACsec SecY statistics details.

```
Router# show macsec secy statistics interface GigabitEthernet 0/1/0/0 SC
Interface Statistics
  InPktsUntagged      : 0
  InPktsNoTag        : 1
  InPktsBadTag       : 2
  InPktsUnknownSCI   : 3
  InPktsNoSCI        : 4
  InPktsOverrun      : 5
  InOctetsValidated   : 6
```

```

InOctetsDecrypted : 7
OutPktsUntagged : 8
OutPktsTooLong : 9
OutOctetsProtected : 10
OutOctetsEncrypted : 11
SC Statistics
TxSC Statistics
  OutPktsProtected : 12
  OutPktsEncrypted : 13
  OutOctetsProtected : 14
  OutOctetsEncrypted : 15
  OutPktsTooLong : 16
TxSA Statistics
  TxSA 0:
    OutPktsProtected : 17
    OutPktsEncrypted : 18
    NextPN : 19
  TxSA 1:
    OutPktsProtected : 20
    OutPktsEncrypted : 21
    NextPN : 22
  TxSA 2:
    OutPktsProtected : 23
    OutPktsEncrypted : 24
    NextPN : 25
  TxSA 3:
    OutPktsProtected : 26
    OutPktsEncrypted : 27
    NextPN : 28
RxSC Statistics
RxSC 1: 0
  InPktsUnchecked : 29
  InPktsDelayed : 30
  InPktsLate : 31
  InPktsOK : 32
  InPktsInvalid : 33
  InPktsNotValid : 34
  InPktsNotUsingSA : 35
  InPktsUnusedSA : 36
  InPktsUntaggedHit : 37
  InOctetsValidated : 38
  InOctetsDecrypted : 39
RxSA Statistics
RxSA 0:
  InPktsUnusedSA : 44
  InPktsNotUsingSA : 43
  InPktsNotValid : 42
  InPktsInvalid : 41
  InPktsOK : 40
  NextPN : 45
RxSA 1:
  InPktsUnusedSA : 50
  InPktsNotUsingSA : 49
  InPktsNotValid : 48
  InPktsInvalid : 47
  InPktsOK : 46
  NextPN : 51
RxSA 2:
  InPktsUnusedSA : 56
  InPktsNotUsingSA : 55
  InPktsNotValid : 54
  InPktsInvalid : 53
  InPktsOK : 52
  NextPN : 57

```



```

RxSA 3:
  InPktsUnusedSA      : 62
  InPktsNotUsingSA    : 61
  InPktsNotValid      : 60
  InPktsInvalid       : 59
  InPktsOK            : 58
  NextPN              : 63

```



Note Ideally, while displaying the MACsec SecY statistics, the hardware does not account the MKPDUs (MACsec control plane packets) in the *InPktsNoTag* counter. Whereas, for NC55-MPA-12T-S MPA, the MKPDU packets are considered as untagged packets, and are accounted in the *InPktsNoTag* counter. Hence, unlike for other PIDs, the *InPktsNoTag* counter increments for incoming MKPDUs in addition to untagged packets, for both Should-Secure and Must-Secure policy modes.

MACsec SNMP MIB (IEEE8021-SECY-MIB)

The IEEE8021-SECY-MIB provides Simple Network Management Protocol (SNMP) access to the MAC security entity (SecY) MIB running with IOS XR MACsec-enabled line cards. The IEEE8021-SECY-MIB is used to query on the SecY data, encryption, decryption, and the hardware statistics. The SecY MIB data is queried only on the Controlled Port.

The object ID of the IEEE8021-SECY-MIB is 1.0.8802.1.1.3. The IEEE8021-SECY-MIB contains the following tables that specifies the detailed attributes of the MACsec Controlled Port interface index.

Table 9: IEEE8021-SECY-MIB Table

Tables	OID
secyIfTable	1.0.8802.1.1.3.1.1.1
secyTxSCTable	1.0.8802.1.1.3.1.1.2
secyTxSatable	1.0.8802.1.1.3.1.1.3
secyRxSCTable	1.0.8802.1.1.3.1.1.4
secyRxSatable	1.0.8802.1.1.3.1.1.5
secyCipherSuiteTable	1.0.8802.1.1.3.1.1.6
secyTxSAStatsTable	1.0.8802.1.1.3.1.2.1
secyTxSCStatsTable	1.0.8802.1.1.3.1.2.2
secyRxSAStatsTable	1.0.8802.1.1.3.1.2.3
secyRxSCStatsTable	1.0.8802.1.1.3.1.2.4
secyStatsTable	1.0.8802.1.1.3.1.2.5

For more information, see the SecY IEEE MIB at the following URL:

<http://www.ieee802.org/1/files/public/MIBs/IEEE8021-SECY-MIB-20060110000Z.mib>

secyIfTable

The following table represents the system level information for each interface supported by the MAC security entity. The index tuple for this table is secyIfInterfaceIndex.

Table 10: secyIfTable

Object	Object identifier
secyIfInterfaceIndex	1.0.8802.1.1.3.1.1.1.1.1
secyIfMaxPeerSCs	1.0.8802.1.1.3.1.1.1.1.2
secyIfRxMaxKeys	1.0.8802.1.1.3.1.1.1.1.3
secyIfTxMaxKeys	1.0.8802.1.1.3.1.1.1.1.4
secyIfProtectFramesEnable	1.0.8802.1.1.3.1.1.1.1.5
secyIfValidateFrames	1.0.8802.1.1.3.1.1.1.1.6
secyIfReplayProtectEnable	1.0.8802.1.1.3.1.1.1.1.7
secyIfReplayProtectWindow	1.0.8802.1.1.3.1.1.1.1.8
secyIfCurrentCipherSuite	1.0.8802.1.1.3.1.1.1.1.9
secyIfAdminPt2PtMAC	1.0.8802.1.1.3.1.1.1.1.10
secyIfOperPt2PtMAC	1.0.8802.1.1.3.1.1.1.1.11
secyIfIncludeSCIEEnable	1.0.8802.1.1.3.1.1.1.1.12
secyIfUseESEEnable	1.0.8802.1.1.3.1.1.1.1.13
secyIfUseSCBEnable	1.0.8802.1.1.3.1.1.1.1.14

secyTxSCTable

The following table provides information about the status of each transmitting SC supported by the MAC security entity. The index tuple for this table is secyIfInterfaceIndex.

Table 11: secyTxSCTable

Object	Object identifier
secyTxSCI	1.0.8802.1.1.3.1.1.2.1.1
secyTxSCState	1.0.8802.1.1.3.1.1.2.1.2
secyTxSCEncodingSA	1.0.8802.1.1.3.1.1.2.1.3
secyTxSCEncipheringSA	1.0.8802.1.1.3.1.1.2.1.4
secyTxSCCreatedTime	1.0.8802.1.1.3.1.1.2.1.5

Object	Object identifier
secyTxSCStartTime	1.0.8802.1.1.3.1.1.2.1.6
secyTxSCStoppedTime	1.0.8802.1.1.3.1.1.2.1.7

secyTxSatable

The following table provides information about the status of each transmitting SA supported by the MAC security entity. The index tuple for this table is: {secyIfInterfaceIndex, secyTxSA}.

Table 12: secyTxSatable

Object	Object identifier
secyTxSA	1.0.8802.1.1.3.1.1.3.1.1
secyTxSAState	1.0.8802.1.1.3.1.1.3.1.2
secyTxSANextPN	1.0.8802.1.1.3.1.1.3.1.3
secyTxSAConfidentiality	1.0.8802.1.1.3.1.1.3.1.4
secyTxSASAKUnchanged	1.0.8802.1.1.3.1.1.3.1.5
secyTxSACreatedTime	1.0.8802.1.1.3.1.1.3.1.6
secyTxSAStartTime	1.0.8802.1.1.3.1.1.3.1.7
secyTxSAStoppedTime	1.0.8802.1.1.3.1.1.3.1.8

secyRxSCTable

The following table provides information about the status of each receiving SC supported by the MAC security entity. The index tuple for this table is: {secyIfInterfaceIndex, secyRxSCI}.

Table 13: secyRxSCTable

Object	Object identifier
secyRxSCI	1.0.8802.1.1.3.1.1.4.1.1
secyRxSCState	1.0.8802.1.1.3.1.1.4.1.2
secyRxSCCurrentSA	1.0.8802.1.1.3.1.1.4.1.3
secyRxSCCreatedTime	1.0.8802.1.1.3.1.1.4.1.4
secyRxSCStartTime	1.0.8802.1.1.3.1.1.4.1.5
secyRxSCStoppedTime	1.0.8802.1.1.3.1.1.4.1.6

secyRxSatable

The following table provides information about the status of each receiving SA supported by the MAC security entity. The index tuple for this table is: {secyIfInterfaceIndex, secyRxSCI, secyRxSA}.

Table 14: secyRxSatable

Object	Object identifier
secyRxSA	1.0.8802.1.1.3.1.1.5.1.1
secyRxSAState	1.0.8802.1.1.3.1.1.5.1.2
secyRxSANextPN	1.0.8802.1.1.3.1.1.5.1.3
secyRxSASAKUnchanged	1.0.8802.1.1.3.1.1.5.1.4
secyRxSACreatedTime	1.0.8802.1.1.3.1.1.5.1.5
secyRxSAStartedTime	1.0.8802.1.1.3.1.1.5.1.6
secyRxSAStoppedTime	1.0.8802.1.1.3.1.1.5.1.7

secyCipherSuiteTable

The following table is a list of selectable cipher suites for the MAC security entity. The index tuple for this table is: {secyCipherSuiteIndex}.

Table 15: secyCipherSuiteTable

Object	Object identifier
secyCipherSuiteIndex	1.0.8802.1.1.3.1.1.6.1.1
secyCipherSuiteId	1.0.8802.1.1.3.1.1.6.1.2
secyCipherSuiteName	1.0.8802.1.1.3.1.1.6.1.3
secyCipherSuiteCapability	1.0.8802.1.1.3.1.1.6.1.4
secyCipherSuiteProtection	1.0.8802.1.1.3.1.1.6.1.5
secyCipherSuiteProtectionOffset	1.0.8802.1.1.3.1.1.6.1.6
secyCipherSuiteDataLengthChange	1.0.8802.1.1.3.1.1.6.1.7
secyCipherSuiteICVLength	1.0.8802.1.1.3.1.1.6.1.8
secyCipherSuiteRowStatus	1.0.8802.1.1.3.1.1.6.1.9

secyTxSAStatsTable

The following table that contains the statistics objects for each transmitting SA in the MAC security entity.

Table 16: secyTxSAStatsTable

Object	Object identifier
secyTxSAStatsProtectedPkts	1.0.8802.1.1.3.1.2.1.1.1
secyTxSAStatsEncryptedPkts	1.0.8802.1.1.3.1.2.1.1.2
secyTxSCStatsProtectedPkts	1.0.8802.1.1.3.1.2.2.1.1
secyTxSCStatsEncryptedPkts	1.0.8802.1.1.3.1.2.2.1.4
secyTxSCStatsOctetsProtected	1.0.8802.1.1.3.1.2.2.1.10
secyTxSCStatsOctetsEncrypted	1.0.8802.1.1.3.1.2.2.1.11

secyTxSCStatsTable

The following table that contains the statistics objects for each transmitting SC in the MAC security entity.

Table 17: secyTxSCStatsTable

Object	Object identifier
secyTxSCStatsProtectedPkts	1.0.8802.1.1.3.1.2.2.1.1
secyTxSCStatsEncryptedPkts	1.0.8802.1.1.3.1.2.2.1.4
secyTxSCStatsOctetsProtected	1.0.8802.1.1.3.1.2.2.1.10
secyTxSCStatsOctetsEncrypted	1.0.8802.1.1.3.1.2.2.1.11

secyRxSAStatsTable

The following table that contains the statistics objects for each receiving SA in the MAC security entity.

Table 18: secyRxSAStatsTable

Object	Object identifier
secyRxSAStatsUnusedSAPkts	1.0.8802.1.1.3.1.2.3.1.1
secyRxSAStatsNoUsingSAPkts	1.0.8802.1.1.3.1.2.3.1.4
secyRxSAStatsNotValidPkts	1.0.8802.1.1.3.1.2.3.1.13
secyRxSAStatsInvalidPkts	1.0.8802.1.1.3.1.2.3.1.16
secyRxSAStatsOKPkts	1.0.8802.1.1.3.1.2.3.1.25

secyRxSCStatsTable

The following table that contains the statistics objects for each receiving SC in the MAC security entity.

Table 19: secyRxSCStatsTable

Object	Object identifier
secyRxSCStatsUnusedSAPkts	1.0.8802.1.1.3.1.2.4.1.1
secyRxSCStatsNoUsingSAPkts	1.0.8802.1.1.3.1.2.4.1.2
secyRxSCStatsLatePkts	1.0.8802.1.1.3.1.2.4.1.3
secyRxSCStatsNotValidPkts	1.0.8802.1.1.3.1.2.4.1.4
secyRxSCStatsInvalidPkts	1.0.8802.1.1.3.1.2.4.1.5
secyRxSCStatsDelayedPkts	1.0.8802.1.1.3.1.2.4.1.6
secyRxSCStatsUncheckedPkts	1.0.8802.1.1.3.1.2.4.1.7
secyRxSCStatsOKPkts	1.0.8802.1.1.3.1.2.4.1.8
secyRxSCStatsOctetsValidated	1.0.8802.1.1.3.1.2.4.1.9
secyRxSCStatsOctetsDecrypted	1.0.8802.1.1.3.1.2.4.1.10

secyStatsTable

The following table lists the objects for the statistics information of each Secy supported by the MAC security entity.

Table 20: secyStatsTable

Object	Object identifier
secyStatsTxUntaggedPkts	1.0.8802.1.1.3.1.2.5.1.1
secyStatsTxTooLongPkts	1.0.8802.1.1.3.1.2.5.1.2
secyStatsRxUntaggedPkts	1.0.8802.1.1.3.1.2.5.1.3
secyStatsRxNoTagPkts	1.0.8802.1.1.3.1.2.5.1.4
secyStatsRxBadTagPkts	1.0.8802.1.1.3.1.2.5.1.5
secyStatsRxUnknownSCIPkts	1.0.8802.1.1.3.1.2.5.1.6
secyStatsRxNoSCIPkts	1.0.8802.1.1.3.1.2.5.1.7
secyStatsRxOverrunPkts	1.0.8802.1.1.3.1.2.5.1.8

Obtaining the MACsec Controlled Port Interface Index

The ifindex of the controlled port can be obtained using the following commands:

- **snmpwalk** command on IfMib[OID: 1.3.6.1.2.1.31.1.1.1]

```
rtr1.0/1/CPU0/ $ snmpwalk -v2c -c public 10.0.0.1 1.3.6.1.2.1.31.1.1.1.1
SNMPv2-SMI::mib-2.31.1.1.1.1.3 = STRING: "GigabitEthernet0/1/0/0"
SNMPv2-SMI::mib-2.31.1.1.1.1.18 = STRING: "MACSecControlled0/1/0/0"
SNMPv2-SMI::mib-2.31.1.1.1.1.19 = STRING: "MACSecUncontrolled0/1/0/0"
```

- **show snmp interface** command

```
Router#show snmp interface
ifName : GigabitEthernet0/1/0/0 ifIndex : 3
ifName : MACSecControlled0/1/0/0 ifIndex : 18
ifName : MACSecUncontrolled0/1/0/0 ifIndex : 19
```

SNMP Query Examples

In the following examples, it is assumed that the configured SNMP community is public, and the management IP of the box is 10.0.0.1.

To perform SNMP walk on the entire SECY MIB for the router, use the following command:

```
snmpwalk -v2c -c public 10.0.0.1 1.0.8802.1.1.3
```

To query on the secyTxSCTable to get the TxSCI for interface Gi0/1/0/0, using the ifindex of MACsecControlled0/1/0/0 that is 18, use the following command:

```
snmpget -v2c -c public 10.0.0.1 iso.0.8802.1.1.3.1.1.2.1.18
```

Related Commands for MACsec

The following commands are available to verify the SNMP results.

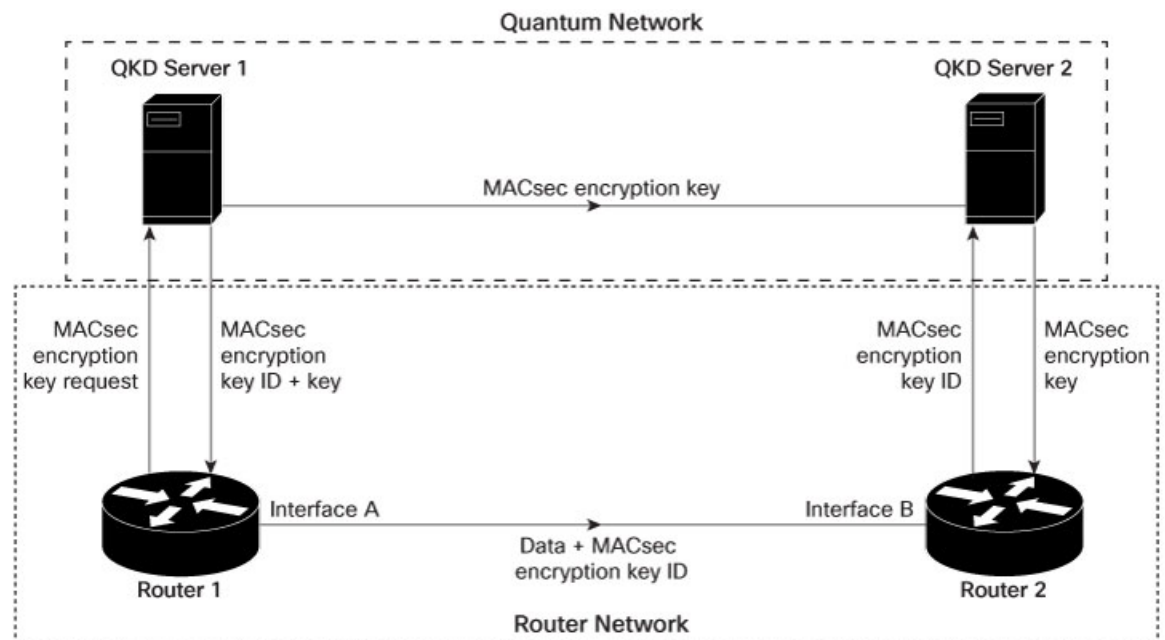
Command	Description
show macsec mka session detail	Displays the details of all MACsec Key Agreement (MKA) sessions on the device.
show macsec mka interface detail	Verifies the MACsec MKA status on the interface.
show macsec ea idb interface	Verifies the MACsec encryption and hardware interface descriptor block (IDB) information on the interface.

Secure Key Integration Protocol

Table 21: Feature History Table

Feature Name	Release Information	Feature Description
Secure Key Integration Protocol for Routers	Release 7.9.1	<p>Your routers are now capable of handling the Secure Key Integration Protocol (SKIP) protocol. The SKIP protocol enables your routers to communicate with external quantum devices. With this ability, you can use the Quantum Key Distribution (QKD) devices for exchanging MACsec encryption keys between routers. Using QKD eliminates the key distribution problem in a post quantum world where the current cryptographic systems are no longer secure due to the advent of quantum computers.</p> <p>This feature introduces the following:</p> <ul style="list-style-type: none"> • CLI: <ul style="list-style-type: none"> • crypto-sks-kme • show crypto sks profile • Yang Data Model: Cisco-IOS-XR-um-sks-server-cfg.yang (see GitHub, YANG Data Models Navigator) <p>For more information on Quantum Key Distribution, see Post Quantum Security Brief.</p>

Cisco Secure Key Integration Protocol (SKIP) enables your router that supports encryption to use keys by a quantum distribution system. SKIP implementation in Cisco IOS-XR software supports integrating external Quantum Key Distribution (QKD) devices with your routers. With integration support between the routers and QKD devices, you can use the QKD devices to exchange encryption keys for communication between the routers. And this mechanism eliminates the key distribution problem in a post quantum world.



Quantum Key Distribution (QKD) is a method for securely transmitting a secret key between two parties. QKD uses the laws of quantum mechanics to guarantee security even when eavesdroppers monitor the communication channel. In QKD, the key is encoded in the states of single photons. The QKD transmits the keys over optical fiber or free space (vacuum). The security of the key relies on the fact that measuring a quantum state introduces a change in the quantum state. The change in quantum states helps the two end parties of the communication channel to identify any interception of their key.

QKD is a secure key exchange mechanism against quantum attacks and will remain so, even with future advancements in cryptanalysis or quantum computing. Unlike other cryptographic algorithms, QKD doesn't need continual updates based on discovered vulnerabilities.

Feature Highlights

- You can use the QKD devices in the following combinations:
 - Same QKD device on the end ports of the peer routers
 - Different QKD devices on the end ports of the peer routers
 - Multiple links between the same peer routers using different QKD devices
- You can use a specific source interface for the router communication with the QKD devices. To use a specific source interface, configure the source interface in the QKD profile. Use the **source interface** command in SKS configuration mode as follows.

```
Router# config
Router(config)# sks profile ProfileR1toR2 type remote
Router(config-sks-profile)# kme server ipv4 192.0.2.34 port 10001
Router(config-sks-profile)# source interface hundredGigE 0/1/0/17
Router(config-sks-profile)# commit
```

- You can use an HTTP Proxy for the router communication with the QKD devices. Use the following configuration for the router to use an HTTP proxy server to communicate to the QKD devices.

```
Router# config
Router(config)# sks profile ProfileR1toR2 type remote
Router(config-sks-profile)# kme server ipv4 192.0.2.34 port 10001
Router(config-sks-profile)# http proxy ipv4 192.0.2.68 port 804
Router(config-sks-profile)# commit
```



Note The **http proxy server** command supports configuration using IPv4 address, IPv6 address, and hostname of the HTTP proxy.

Restrictions

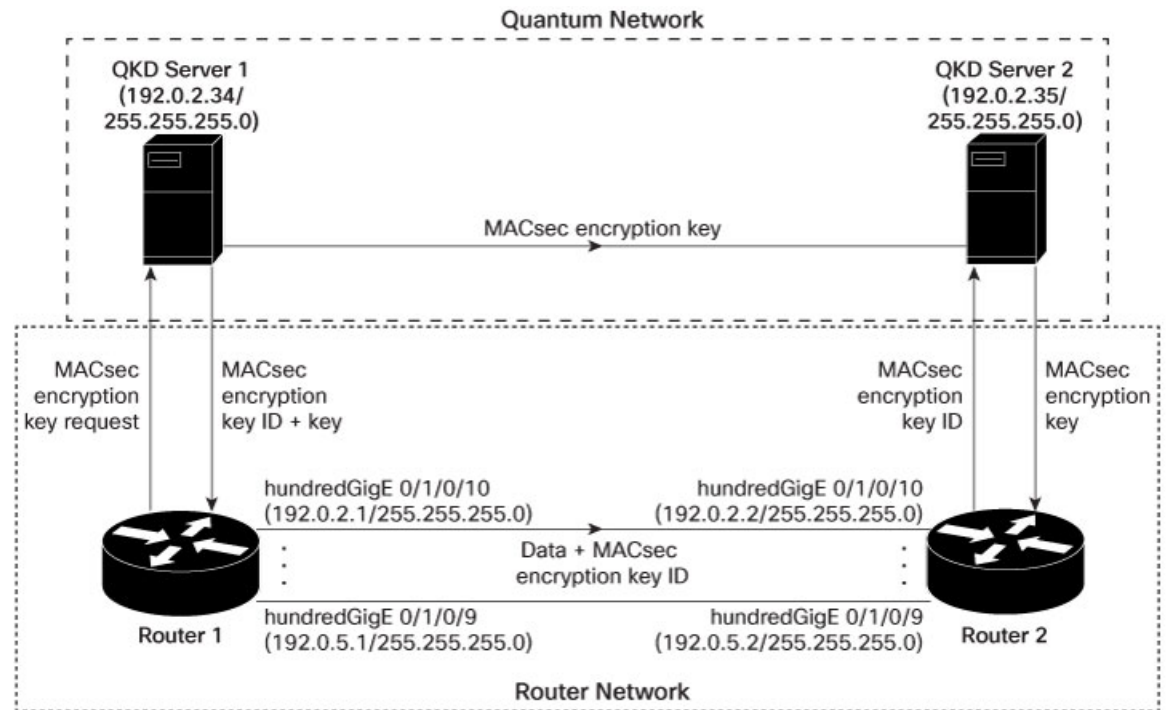
The following section lists the restriction to consider prior implementing SKIP:

- The SKIP protocol is supported only on the following line cards or chassis:
 - NCS-57C3-MOD
 - NCS-55A2-MOD-S, NCS-55A2-MOD-HD-S, NCS-55A2-MOD-HX-S, NCS-55A2-MOD-SE-S, and NC55A2-MOD-SE-H-S
- You can use the SKIP protocol only in a Point to Point MACsec link encryption scenario.
- The SKIP protocol is available only on the interfaces that support MACsec encryption.

Configuring Point to Point MACsec Link Encryption using SKIP

In Point-to-Point MACsec Link Encryption, the router uses SKIP to establish secure encryption. This encryption is set up between two interfaces in peer routers and requires the assistance of an external QKD device network. The QKD network shares the MACsec encryption key instead of the router network. Thus, when the router needs to create a MACsec link between peer router interfaces, it contacts the external QKD device and requests the key. The external QKD device generates a Key pair comprising the Key ID and the Key. The Key ID serves as the unique identification string for the Key (Shared Secret). The QKD then shares both the Key ID and Key with the router and the router shares only the Key ID with its peer. The Peer router uses this Key ID to retrieve encryption keys from its QKD device. Therefore, Quantum networks securely communicate encryption keys always.

Figure 3: Point to Point MACsec Link Encryption using SKIP



Prerequisites

- Configure MACsec Pre-Shared Key (PSK). For more information, see [MACsec PSK, on page 9](#).
- Configure MACsec in the PPK mode.
- An external QKD devices network.
- Add the QKD server CA to the trustpoint in the router. For more information, see [Configure Trustpoint](#).
- Import the QKD server root CA certificate in the router. For more information, see [Configure Certificate Enrollment Using Cut-and-Paste](#).

Configuration

The following example details how to establish Point to Point MACsec Link Encryption using SKIP:

Router 1:

1. Configure the QKD profile.

```
Router# config
Router(config)# sks profile ProfileR1toR2 type remote
Router(config-sks-profile)# kme server ipv4 192.0.2.34 port 10001
Router(config-sks-profile)# commit
```

2. Map the QKD profile to the MACsec policy.

```
Router# config
Router(config)# macsec-policy R1toR2
Router(config-macsec-policy)# ppk sks-profile ProfileR1toR2
Router(config-macsec-policy)# commit
```



Note For more information on MACsec Policy, see [Creating a User-Defined MACsec Policy, on page 17](#).

3. Apply MACsec policy to the interfaces.

```
Router# config
Router(config)#interface hundredGigE 0/1/0/10
Router(config-if)# ipv4 address 192.0.2.1 255.255.255.0
Router(config-if)# macsec psk-keychain mac_chain policy R1toR2
Router(config)# commit
Router(config)#interface hundredGigE 0/1/0/11
Router(config-if)# ipv4 address 192.0.3.1 255.255.255.0
Router(config-if)# macsec psk-keychain mac_chain policy R1toR2
Router(config)# commit
Router(config)#interface hundredGigE 0/1/0/12
Router(config-if)# ipv4 address 192.0.4.1 255.255.255.0
Router(config-if)# macsec psk-keychain mac_chain policy R1toR2
Router(config)# commit
Router(config)#interface hundredGigE 0/1/0/9
Router(config-if)# ipv4 address 192.0.5.1 255.255.255.0
Router(config-if)# macsec psk-keychain mac_chain policy R1toR2
Router(config)# commit
```

Router 2:

1. Configure the QKD profile.

```
Router# config
Router(config)# sks profile ProfileR2toR1 type remote
Router(config-sks-profile)# kme server ipv4 192.0.2.35 port 10001
Router(config-sks-profile)# commit
```

2. Map the QKD profile to the MACsec policy.

```
Router# config
Router(config)# macsec-policy R2toR1
Router(config-macsec-policy)# ppk sks-profile ProfileR2toR1
Router(config-macsec-policy)# commit
```



Note For more information on MACsec Policy, see [Creating a User-Defined MACsec Policy, on page 17](#).

3. Apply MACsec policy to the interfaces.

```
Router# config
Router(config)#interface hundredGigE 0/1/0/10
Router(config-if)# ipv4 address 192.0.2.2 255.255.255.0
Router(config-if)# macsec psk-keychain mac_chain policy R2toR1
Router(config-if)# commit
Router(config)#interface hundredGigE 0/1/0/11
Router(config-if)# ipv4 address 192.0.3.2 255.255.255.0
Router(config-if)# macsec psk-keychain mac_chain policy R2toR1
Router(config-if)# commit
Router(config)#interface hundredGigE 0/1/0/12
Router(config-if)# ipv4 address 192.0.4.2 255.255.255.0
Router(config-if)# macsec psk-keychain mac_chain policy R2toR1
Router(config-if)# commit
Router(config)#interface hundredGigE 0/1/0/9
Router(config-if)# ipv4 address 192.0.5.2 255.255.255.0
Router(config-if)# macsec psk-keychain mac_chain policy R2toR1
```

```
Router(config-if)# commit
```

Running Configuration

Router 1:

```
sks profile ProfileR1toR2 type remote
 kme server ipv4 192.0.2.34 port 10001
!
macsec-policy R1toR2
 ppk
  sks-profile ProfileR1toR2
!
!
interface hundredGigE 0/1/0/10
 ipv4 address 192.0.2.1 255.255.255.0
 macsec psk-keychain mac_chain policy R1toR2
!
interface hundredGigE 0/1/0/11
 ipv4 address 192.0.3.1 255.255.255.0
 macsec psk-keychain mac_chain policy R1toR2
!
interface hundredGigE 0/1/0/12
 ipv4 address 192.0.4.1 255.255.255.0
 macsec psk-keychain mac_chain policy R1toR2
!
interface hundredGigE 0/1/0/9
 ipv4 address 192.0.5.1 255.255.255.0
 macsec psk-keychain mac_chain policy R1toR2
!
```

Router 2:

```
sks profile ProfileR2toR1 type remote
 kme server ipv4 192.0.2.35 port 10001
!
macsec-policy R2toR1
 ppk
  sks-profile ProfileR2toR1
!
!
interface hundredGigE 0/1/0/10
 ipv4 address 192.0.2.2 255.255.255.0
 macsec psk-keychain mac_chain policy R2toR1
!t
interface hundredGigE 0/1/0/11
 ipv4 address 192.0.3.2 255.255.255.0
 macsec psk-keychain mac_chain policy R2toR1
!
interface hundredGigE 0/1/0/12
 ipv4 address 192.0.4.2 255.255.255.0
 macsec psk-keychain mac_chain policy R2toR1
!
interface hundredGigE 0/1/0/9
 ipv4 address 192.0.5.2 255.255.255.0
 macsec psk-keychain mac_chain policy R2toR1
!
```

Verification

```

Router(config)# show crypto sks profile all
Profile Name :ProfileR1toR2
Myidentifier :Router1
Type :Remote
Reg Client Count :1

Server
IP :192.0.2.34
Port :10001
Vrf :Notconfigured
Source Interface :Notconfigured
Status :Connected
Entropy :true
Key :true
Algorithm :QKD
Local identifier :Alice
Remote identifier :Alice

Peerlist
QKD ID :Bob
State :Connected

Peerlist
QKD ID :Alice
State :Connected

Router# show crypto sks profile all stats
Profile Name : ProfileR1toR2
My identifier : Router1
Server
IP : 192.0.2.34
Port : 10001
Status : connected
Counters
Capability request : 1
Key request : 3
Key-id request : 0
Entropy request : 0
Capability response : 1
Key response : 3
Key-id response : 0
Entropy response : 0
Total request : 4
Request failed : 0
Request success : 4
Total response : 4
Response failed : 0
Response success : 4
Retry count : 0
Response Ignored : 0
Cancelled count : 0
Response time
Max Time : 100 ms
Avg Time : 10 ms
Min Time : 50 ms
Last transaction
Transaction Id : 9
Transaction type : Get key
Transaction status : Response data received, successfully
Http code : 200 OK (200)

```