



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.

Command History	Release	Modification
	Release 6.1.3	This feature is introduced for physical interfaces and bundle member interfaces.

- [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 for MACsec, on page 3](#)
- [MACsec PSK, on page 3](#)
- [Configuring and Verifying MACsec Encryption , on page 4](#)
- [Creating a MACsec Keychain, on page 4](#)
- [Creating a User-Defined MACsec Policy, on page 10](#)
- [Applying MACsec Configuration on an Interface, on page 13](#)
- [Verifying MACsec Encryption on IOS XR, on page 14](#)
- [Verifying MACsec Encryption on NCS 5500, on page 17](#)
- [MACsec SecY Statistics, on page 20](#)

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.

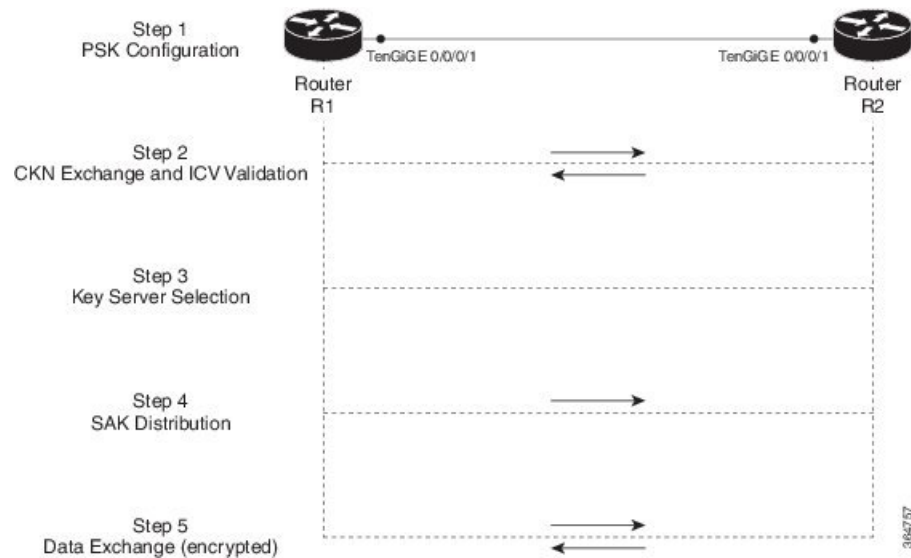
MACsec secures data on physical media, making it impossible for data to be compromised at higher layers. As a result, MACsec encryption takes priority over any other encryption method such as IPsec and SSL at higher layers. MACsec is configured on the Customer Edge (CE) router interfaces that connect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

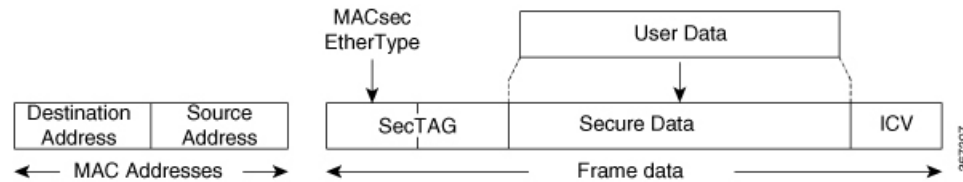
Step 4: A security association is formed between the peers. The key server generates and distributes the Secure Association Key (SAK) to the key client (peer). Each secure channel is supported by an overlapped sequence of Security Associations (SA). Each SA uses a new Secure Association Key (SAK).

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 for MACsec

The MACsec technology is supported only on 36-port 100 GE line cards.

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 13](#) of the [Applying MACsec Configuration on an Interface, on page 13](#) section.

Configuring and Verifying MACsec Encryption

MACsec can be configured on physical ethernet interfaces or interface bundles (link 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:

- 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.

SUMMARY STEPS

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

DETAILED STEPS

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)# keychain mac_chain
```

Step 2 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.

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.

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

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

	Command or Action	Purpose
	<p>Example: Deleting the Primary Key</p> <pre>Router# key config-key password-encryption delete</pre>	<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

	Command or Action	Purpose
Step 1	configure terminal Example: <pre>Router# configure terminal Router(config)#</pre>	Enters global configuration mode.
Step 2	key chain <i>key chain name</i> macsec Example: <pre>Router(config)# key chain kcl macsec Router(config-kcl-MacSec)#</pre>	Configures a key chain with the MACsec submodule.
Step 3	key <i>hex string of even length and max 64 bytes</i> Example: <pre>Router(config-kcl-MacSec)# key 1111 Router(config-kcl-MacSec-1111)#</pre>	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 submodule. In that case, repeat from Step2 to enter the MACsec submodule 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: 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 cryptographic-algorithm aes-256-cmac</pre>	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).
Step 5	lifetime {hh:mm:ss} {1-31} month year infinite 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.

	Command or Action	Purpose
Step 6	commit Example: Router(config)# commit	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 kc1 key chain kc1 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 the replay protection window size.
7. Configure the ICV for the frame arriving on the port.
8. Commit your configuration and exit the global configuration mode.
9. Confirm the MACsec policy configuration.

DETAILED STEPS

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 1: 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 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 7 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 8 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 9 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-XPN-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.

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

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
```

Step 4 Commit your configuration.

Example:

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

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.

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.
4. Verify whether the MKA session is secured with MACsec on the respective interface.
5. Verify the MACsec session counter statistics.

DETAILED STEPS

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-XP  0             64      30
            N-256
```

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:

```
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-XP  mac_chain
            N-256
```



```

Latest SAK Status      : Rx & Tx
Latest SAK AN         : 0
Latest SAK KI (KN)    : E3D39135831AFBCDA7AA9DBB00000001 (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      : 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
-----

```

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

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 : f8d88dc3elc5e6a94ca2299

```

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 : 0xbffffb8
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

```

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 2: 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-200601100000Z.txt>

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 3: secyIfTable

Object	Object identifier
secyIfInterfaceIndex	1.0.8802.1.1.3.1.1.1.1
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

Object	Object identifier
secyIfAdminPt2PtMAC	1.0.8802.1.1.3.1.1.1.1.10
secyIfOperPt2PtMAC	1.0.8802.1.1.3.1.1.1.1.11
secyIfIncludeSCIEnable	1.0.8802.1.1.3.1.1.1.1.12
secyIfUseESEnable	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 4: 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
secyTxSCStartedTime	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 5: 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

Object	Object identifier
secyTxSACreatedTime	1.0.8802.1.1.3.1.1.3.1.6
secyTxSAStartedTime	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 6: 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
secyRxSCStartedTime	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 7: 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 8: 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 9: 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 10: 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

secyRxSASStatsTable

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

Table 11: secyRxSASStatsTable

Object	Object identifier
secyRxSASStatsUnusedSAPkts	1.0.8802.1.1.3.1.2.3.1.1
secyRxSASStatsNoUsingSAPkts	1.0.8802.1.1.3.1.2.3.1.4
secyRxSASStatsNotValidPkts	1.0.8802.1.1.3.1.2.3.1.13
secyRxSASStatsInvalidPkts	1.0.8802.1.1.3.1.2.3.1.16
secyRxSASStatsOKPkts	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 12: 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

Object	Object identifier
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 13: 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.0.8802.1.1.3.1.1.2.1.1.18
```

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.