



Configure MACsec

This chapter describes how to configure MACsec on Cisco NX-OS devices.

- [About MACsec, on page 1](#)
- [Licensing Requirements for MACsec, on page 2](#)
- [Guidelines and Limitations for MACsec, on page 2](#)
- [Enabling MACsec, on page 5](#)
- [Disabling MACsec, on page 6](#)
- [Configuring a MACsec Keychain and Keys, on page 6](#)
- [MACsec Packet-Number Exhaustion, on page 8](#)
- [Configuring MACsec Fallback Key, on page 8](#)
- [Configuring a MACsec Policy, on page 9](#)
- [About Configurable EAPOL Destination and Ethernet Type, on page 11](#)
- [Verifying the MACsec Configuration, on page 13](#)
- [Displaying MACsec Statistics, on page 15](#)
- [Configuration Example for MACsec, on page 18](#)
- [XML Examples, on page 20](#)
- [MIBs, on page 28](#)
- [Related Documentation, on page 28](#)

About MACsec

Media Access Control Security (MACsec) an IEEE 802.1AE along with MACsec Key Agreement (MKA) protocol provide secure communications on Ethernet links. It offers the following :

- Provides line rate encryption capabilities.
- Helps to ensure data confidentiality by providing strong encryption at Layer 2.
- Provides integrity checking to help ensure that data cannot be modified in transit.
- Can be selectively enabled using a centralized policy to help ensure that it is enforced where required while allowing non-MACsec-capable components to access the network.
- Encrypts packets on a hop-by-hop basis at Layer 2, allowing the network to inspect, monitor, mark, and forward traffic according to your existing policies (unlike end-to-end Layer 3 encryption techniques that hide the contents of packets from the network devices they cross).

Key Lifetime and Hitless Key Rollover

A MACsec keychain can have multiple pre-shared keys (PSKs), each configured with a key ID and an optional lifetime. A key lifetime specifies at which time the key activates and expires. In the absence of a lifetime configuration, the default lifetime is unlimited. When a lifetime is configured, MKA rolls over to the next configured pre-shared key in the keychain after the lifetime is expired. The time zone of the key can be local or UTC. The default time zone is UTC.

To configure a MACsec keychain, see [Configuring a MACsec Keychain and Keys, on page 6](#).

A key can roll over to a second key within the same keychain by configuring the second key (in the keychain) and configuring a lifetime for the first key. When the lifetime of the first key expires, it automatically rolls over to the next key in the list. If the same key is configured on both sides of the link at the same time, then the key rollover is hitless (that is, the key rolls over without traffic interruption).

Fallback Key

A MACsec session can fail due to a key/key name (CKN) mismatch or a finite key duration between the switch and a peer. If a MACsec session does fail, a fallback session can take over if a fallback key is configured. A fallback session prevents downtime due to primary session failure and allows a user time to fix the key issue causing the failure. A fallback key also provides a backup session if the primary session fails to start. This feature is optional.

To configure a MACsec fallback key, see [Configuring MACsec Fallback Key, on page 8](#).

Licensing Requirements for MACsec

Product	License Requirement
Cisco NX-OS	MACsec requires a Security license. For a complete explanation of the Cisco NX-OS licensing scheme to obtain and apply licenses, see the Cisco NX-OS Licensing Guide .

Guidelines and Limitations for MACsec

MACsec has the following guidelines and limitations:

- MACsec is supported on the following interface types:
 - Layer 2 switch ports (access and trunk)
 - Layer 3 routed interfaces (no subinterfaces)

**Note**

Enabling MACsec on the Layer 3 routed interface also enables encryption on all the subinterfaces that are defined under that interface. However, selectively enabling MACsec on a subset of subinterfaces of the same Layer 3 routed interface is not supported.

- Layer 2 and Layer 3-port channels (no subinterfaces)

- When the Cisco Nexus ToR switches are downgraded from Cisco NX-OS Release 9.x to Cisco NX-OS Release 7.x, MACsec is not supported.
- When you disable the MACsec feature immediately after a switch-over, some sessions remain in the pending state while some peers are in a secure state.
- MKA is the only supported key exchange protocol for MACsec. The Security Association Protocol (SAP) is not supported.
- Link-level flow control (LLFC) and priority flow control (PFC) are not supported with MACsec.
- Multiple MACsec peers (different SCI values) for the same interface are not supported.
- You can retain the MACsec configuration when you disable MACsec using the **macsec shutdown** command.
- MACsec sessions are liberal in accepting packets from a key server whose latest Rx and latest Tx flags have been retired after Tx SA installation for the first time. The MACsec session then converges into a secure state.
- Beginning with Cisco NX-OS Release 9.2(1), the following configurations are allowed:
 - Allowing MACSec policy to be modified while the policy is referenced by an interface.
 - Allowing different MACsec policies across different lanes of a breakout port.
- Beginning with Cisco Nexus Release 9.2(1), MACsec is supported on Cisco Nexus 93180YC-FX switches.
- On Cisco Nexus 9300-FX switches and 9700-FX line cards:
 - MACsec is not supported on 1G ports.
 - Additionally, if a MAC block contains any 1G ports, MACsec is not supported on any port within that MAC block.
 - When using 1G optics on MACsec capable 9700-FX line cards, it is recommended to change diagnostics mode to 'minimal'. For more information on diagnostics, see the [Bootup Diagnostics](#) section in the **Cisco Nexus 9000 Series NX-OS System Management Configuration Guide**.
- On Cisco Nexus 9300-FX2 switches:
 - MACsec is supported on 1G ports, provided the ports are non-BV or non-retimer ports.
For details about retimer ports, refer to the *Supported Retimer Ports* section in the **Cisco Nexus 9000 Series NX-OS Interfaces Configuration Guide**.
- MACsec is supported on Cisco Nexus 93240YC-FX2, 9336C-FX2, 93108TC-FX, 93180YC-FX switches and the X9736C-FX, and X9732C-EXM line cards.
- Cisco Nexus 9000 Series switches do not support MACsec on any of the MACsec capable ports when QSA is being used.
- MACsec is not supported on Cisco Nexus 9500 Series switches with Nexus 9600-R line cards.
- If the MACsec feature is configured, non-disruptive ISSU is not supported.

Keychain restrictions:

- You cannot overwrite the octet string for a MACsec key. Instead, you must create a new key or a new keychain.
- A new key in the keychain is configured when you enter **end** or **exit**. The default timeout for editor mode is 6 seconds. If the key is not configured with the key octet string or/and the send lifetime within the 6-second window, incomplete information may be used to bring up the MACsec session and could result in the session being stuck in an Authorization Pending state. If the MACsec sessions are not converged after the configuration is complete, you might be advised to shut/no shut the ports.
- For a given keychain, key activation times should overlap to avoid any period of time when no key is activated. If a time period occurs during which no key is activated, session negotiation fails and traffic drops can occur. The key with the latest start time among the currently active keys takes precedence for a MACsec key rollover.

Fallback restrictions:

- If a MACsec session is secured on an old primary key, it does not go to a fallback session in case of mismatched latest active primary key. So the session remains secured on the old primary key and will show as rekeying on the old CA under status. And the MACsec session on the new key on primary PSK will be in init state.
- Use only one key with infinite lifetime in the fallback key chain. Multiple keys are not supported.
- The key ID (CKN) used in the fallback key chain must not match any of the key IDs (CKNs) used in the primary key chain.
- Once configured, fallback configuration on an interface cannot be removed, unless the complete MACsec configuration on the interface is removed.

MACsec policy restrictions:

- BPDU packets can be transmitted before a MACsec session becomes secure.

Layer 2 Tunneling Protocol (L2TP) restrictions:

- MACsec is not supported on ports configured for dot1q tunneling (switchport mode dot1q-tunnel) or L2TP.
- L2TP does not work if STP is enabled on trunk ports for non-native VLANs.

Statistics restrictions:

- Few CRC errors should occur during the transition between MACsec and non-MACsec mode (regular port shut/no shut).
- Secy statistics are cumulative and polled every 30 seconds.
- The IEEE8021-SECY-MIB OIDs secyRxSASStatsOKPkts, secyTxSASStatsProtectedPkts, and secyTxSASStatsEncryptedPkts can carry only up to 32 bits of counter values, but the traffic may exceed 32 bits.

Interoperability restrictions:

- Interoperability of N9K-X9732C-EXM and other peer switches (other Cisco and non-Cisco switches) is supported only with the XPN cipher suite.

- MACsec peers must run the same Cisco NX-OS release in order to use the AES_128_CMAC cryptographic algorithm. For interoperability between previous releases and Cisco NX-OS Release 9.2(1), you must use keys with the AES_256_CMAC cryptographic algorithm.
- For interoperability between previous releases and Cisco NX-OS Release 9.2(1), pad the MACsec key with zeros if it is less than 32 octets.
- On any Cisco NX-OS switch, you can configure only one unique combination of an alternate MAC address and Ethernet type on all interfaces.

EAPOL has the following guidelines and limitations:

- For enabling EAPOL configuration, the range of ethernet type between 0 to 0x599 is invalid.
- While configuring EAPOL packets, the following combinations must not be used:
 - Mac address 0100.0ccd.cdd0 with any ethertype
 - Any mac address with Ether types: 0xffff0, 0x800, 0x86dd
 - The default destination MAC address, 0180.c200.0003 with the default Ethernet type, 0x888e
 - Different EAPOL DMAC addresses on both MACsec peers. The MACsec session works only if the MACsec peer is sending MKAPDUs with the DMAC configured locally.

Enabling MACsec

Before you can access the MACsec and MKA commands, you must enable the MACsec feature.

Procedure

	Command or Action	Purpose
Step 1	configure terminal Example: <pre>switch# configure terminal switch(config)#</pre>	Enters global configuration mode.
Step 2	feature macsec Example: <pre>switch(config)# feature macsec</pre>	Enables MACsec and MKA on the device.
Step 3	(Optional) copy running-config startup-config Example: <pre>switch(config)# copy running-config startup-config</pre>	Copies the running configuration to the startup configuration.

Disabling MACsec

Beginning with Cisco NX-OS Release 9.2(1), disabling the MACsec feature only deactivates this feature and does not remove the associated MACsec configurations.

Disabling MACsec has the following conditions:

- MACsec shutdown is global command and is not available at the interface level.
- The macsec shutdown, show macsec mka session/summary, show macsec mka session detail, and show macsec mka/secy statistics commands will display the 'Macsec is shutdown' message. However, the show macsec policy and show key chain commands will display the output.
- Consecutive MACsec status changes from macsec shutdown to no macsec shutdown and vice versa needs a 30 seconds time interval in between the status change.

Procedure

	Command or Action	Purpose
Step 1	configure terminal Example: <pre>switch# configure terminal switch(config)#</pre>	Enters global configuration mode.
Step 2	macsec shutdown Example: <pre>switch(config)# macsec shutdown</pre>	Disables the MACsec configuration on the device. The no option restores the MACsec feature.
Step 3	(Optional) copy running-config startup-config Example: <pre>switch(config)# copy running-config startup-config</pre>	Copies the running configuration to the startup configuration. This step is required only if you want to retain the MACsec in the shutdown state after the switch reload.

Configuring a MACsec Keychain and Keys

You can create a MACsec keychain and keys on the device.



Note Only MACsec keychains will result in converged MKA sessions.

Before you begin

Make sure that MACsec is enabled.

Procedure

	Command or Action	Purpose
Step 1	configure terminal Example: <pre>switch# configure terminal switch(config)#</pre>	Enters global configuration mode.
Step 2	(Optional) [no] key-chain macsec-psk no-show Example: <pre>switch(config)# key-chain macsec-psk no-show</pre>	Hides the encrypted key octet string in the output of the show running-config and show startup-config commands by replacing the string with a wildcard character. By default, PSK keys are displayed in encrypted format and can be easily decrypted. This command applies only to MACsec keychains. Note The key-octet-string is hidden (with asterisks) when you save the configuration to a file. As a result, if you perform an ASCII reload or config replace with this command configured, the 'key-octet-string' configuration will not be retained.
Step 3	key chain name macsec Example: <pre>switch(config)# key chain 1 macsec switch(config-macseckeychain)#</pre>	Creates a MACsec keychain to hold a set of MACsec keys and enters MACsec keychain configuration mode.
Step 4	key key-id Example: <pre>switch(config-macseckeychain)# key 1000 switch(config-macseckeychain-macseckey)#</pre>	Creates a MACsec key and enters MACsec key configuration mode. The range is from 1 to 32 octets, and the maximum size is 64. Note The key must consist of an even number of characters.
Step 5	key-octet-string octet-string cryptographic-algorithm {AES_128_CMAC AES_256_CMAC} Example: <pre>switch(config-macseckeychain-macseckey)# key-octet-string a0c0ef0123456789a0c0ef0123456789a0c0ef0123456789a0c0ef0123456789 cryptographic-algorithm AES_256_CMAC</pre>	Configures the octet string for the key. The <i>octet-string</i> argument can contain up to 64 hexadecimal characters. The octet key is encoded internally, so the key in clear text does not appear in the output of the show running-config macsec command. Note MACsec peers must run the same Cisco NX-OS release in order to use the AES_128_CMAC cryptographic algorithm. To interoperate between previous releases and Cisco NX-OS Release 7.0(3)I7(2) or a later

	Command or Action	Purpose
		release, you must use keys with the AES_256_CMAC cryptographic algorithm.
Step 6	send-lifetime <i>start-time</i> duration <i>duration</i> Example: <pre>switch(config-macseckeychain-macseckey) # send-lifetime 00:00:00 Oct 04 2016 duration 100000</pre>	Configures a send lifetime for the key. By default, the device treats the start time as UTC. The <i>start-time</i> argument is the time of day and date that the key becomes active. The <i>duration</i> argument is the length of the lifetime in seconds. The maximum length is 2147483646 seconds (approximately 68 years).
Step 7	(Optional) show key chain <i>name</i> Example: <pre>switch(config-macseckeychain-macseckey) # show key chain 1</pre>	Displays the keychain configuration.
Step 8	(Optional) copy running-config startup-config Example: <pre>switch(config-macseckeychain-macseckey) # copy running-config startup-config</pre>	Copies the running configuration to the startup configuration.

MACsec Packet-Number Exhaustion

Every MACsec frame contains a 32-bit packet number (PN), and it is unique for a given Security Association Key (SAK). Upon PN exhaustion (after reaching 75% of $2^{32} - 1$), SAK rekey takes place automatically to refresh the data plane keys and the PN will wrap around.

For example, on 10G full line rate @ 64 bytes, the SAK rekey will occur every 216 seconds due to PN exhaustion.

This is applicable when using GCM-AES-PN-128 or GCM-AES-PN-256 cipher-suites.

When GCM-AES-XP-128 or GCM-AES-XP-256 cipher-suite is used, the SAK rekey happens automatically when reaching 75% of $2^{64} - 1$, which will take several years to exhaust the packet numbering. The cipher-suite is configurable under the macsec policy and the operational cipher-suite is determined by the key-server device.

It is recommended to use XPN ciphersuite on N9K-X9732C-EXM line card

Configuring MACsec Fallback Key

Beginning with Cisco NX-OS Release 9.2(1), you can configure a fallback key on the device to initiate a backup session if the primary session fails as a result of a key/key name (CKN) mismatch or a finite key duration between the switch and peer.

Before you begin

Make sure that MACsec is enabled and a primary and fallback keychain and key ID are configured. See [Configuring a MACsec Keychain and Keys](#).

Procedure

	Command or Action	Purpose
Step 1	configure terminal Example: <pre>switch# configure terminal switch(config)#</pre>	Enters the global configuration mode.
Step 2	interface name Example: <pre>switch(config)# interface ethernet 1/1 switch(config-if)#</pre>	Specifies the interface that you are configuring. You can specify the interface type and identity. For an Ethernet port, use ethernet slot/port.
Step 3	macsec keychain keychain-name policy policy-name fallback-keychain keychain-name Example: <pre>switch(config-if)# macsec keychain kc2 policy abc fallback-keychain fb_kc2</pre>	Specifies the fallback keychain to use after a MACsec session failure due to a key/key ID mismatch or a key expiration. The fallback key ID should not match any key ID from a primary keychain. Fallback keychain configuration for each interface can be changed on the corresponding interface, without removing the MACsec configuration, by reissuing the same command with the fallback keychain name changed. Note The command must be entered exactly the same as the existing configuration command for the interface, except for the fallback keychain name. See Configuring a MACsec Keychain and Keys.
Step 4	(Optional) copy running-config startup-config Example: <pre>switch(config-if)# copy running-config startup-config</pre>	Copies the running configuration to the startup configuration.

Configuring a MACsec Policy

You can create multiple MACsec policies with different parameters. However, only one policy can be active on an interface.

Before you begin

Make sure that MACsec is enabled.

Procedure

	Command or Action	Purpose
Step 1	configure terminal Example: switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	macsec policy name Example: switch(config)# macsec policy abc switch(config-macsec-policy)#	Creates a MACsec policy.
Step 3	(Optional) cipher-suite name Example: switch(config-macsec-policy)# cipher-suite GCM-AES-256	Configures one of the following ciphers: GCM-AES-128, GCM-AES-256, GCM-AES-XPB-128, or GCM-AES-XPB-256.
Step 4	(Optional) key-server-priority number Example: switch(config-macsec-policy)# key-server-priority 0	Configures the key server priority to break the tie between peers during a key exchange. The range is from 0 (highest) and 255 (lowest), and the default value is 16.
Step 5	(Optional) security-policy name Example: switch(config-macsec-policy)# security-policy should-secure	Configures one of the following security policies to define the handling of data and control packets: • must-secure—Packets not carrying MACsec headers will be dropped. • should-secure—Packets not carrying MACsec headers will be permitted. This is the default value.
Step 6	(Optional) window-size number Example: switch(config-macsec-policy)# window-size 512	Configures the replay protection window such that the secured interface will not accept any packet that is less than the configured window size. The range is from 0 to 596000000.
Step 7	(Optional) sak-expiry-time time Example: switch(config-macsec-policy)# sak-expiry-time 100	Configures the time in seconds to force an SAK rekey. This command can be used to change the session key to a predictable time interval. The default is 0.

	Command or Action	Purpose
Step 8	(Optional) conf-offset name Example: <pre>switch(config-macsec-policy)# conf-offset CONF-OFFSET-0</pre>	Configures one of the following confidentiality offsets in the Layer 2 frame, where encryption begins: CONF-OFFSET-0, CONF-OFFSET-30, or CONF-OFFSET-50. This command might be necessary for intermediate switches to use packet headers {dmac, smac, etype} like MPLS tags.
Step 9	(Optional) show macsec policy Example: <pre>switch(config-macsec-policy)# show macsec policy</pre>	Displays the MACsec policy configuration.
Step 10	(Optional) copy running-config startup-config Example: <pre>switch(config-macsec-policy)# copy running-config startup-config</pre>	Copies the running configuration to the startup configuration.

About Configurable EAPOL Destination and Ethernet Type

Beginning Cisco NX-OS Release 9.2(2), Cisco enables networks with WAN MACsec to change the Extensible Authentication Protocol (EAP) over LAN (EAPOL) protocol destination address, and the Ethernet type values to nonstandard values.

Configurable EAPOL MAC and Ethernet type provides you the ability to change the MAC address and the Ethernet type of the MKA packet, in order to allow CE device to form MKA sessions over the ethernet networks that consume the standard MKA packets.

The EAPOL destination Ethernet type can be changed from the default Ethernet type of 0x888E to an alternate value or, the EAPOL destination MAC address can be changed from the default DMAC of 01:80:C2:00:00:03 to an alternate value, to avoid being consumed by a provider bridge.

This feature is available at the interface level and the alternate EAPOL configuration can be changed on any interface at any given time as follows:

- If the MACsec is already configured on an interface, the sessions will come up with a new alternate EAPOL configuration.
- When MACsec is not configured on an interface, the EAPOL configuration is applied to the interface and is effective when MACsec is configured on that interface.

Enabling EAPOL Configuration

You can enable the EAPOL configuration on any available interface.

Before you begin

Make sure that MACsec is enabled.

Procedure

	Command or Action	Purpose
Step 1	configure terminal Example: switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	interface name Example: switch(config)# interface ethernet 1/1 switch(config-if)#	Specifies the interface that you are configuring. You can specify the interface type and identity. For an Ethernet port, use ethernet slot/port.
Step 3	eapol mac-address mac_address [ethertype eth_type]	Enables the EAPOL configuration on the specified interface type and identity. Note If the ethernet type is not specified, the default ethernet type of MKA packets, which is 0x888e, is considered.
Step 4	eapol mac-address broadcast-address [ethertype eth_type]	Enables the broadcast address as the alternate mac address.
Step 5	(Optional) copy running-config startup-config Example: switch(config-macseckeychain-macseckey)# copy running-config startup-config	Copies the running configuration to the startup configuration.
Step 6	show macsec mka session detail	Displays the EAPOL settings.

Disabling EAPOL Configuration

You can disable the EAPOL configuration on any available interface.

Procedure

	Command or Action	Purpose
Step 1	configure terminal Example: switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	interface name Example: switch(config)# interface ethernet 1/1 switch(config-if)#	Specifies the interface that you are configuring. You can specify the interface type and identity. For an Ethernet port, use ethernet slot/port.

	Command or Action	Purpose
Step 3	[no] eapol mac-address <i>mac_address</i> [ethertype <i>eth_type</i>]	Disables the EAPOL configuration on the specified interface type and identity.
Step 4	(Optional) copy running-config startup-config Example: switch(config-macseckeychain-macseckey) # copy running-config startup-config	Copies the running configuration to the startup configuration.

Verifying the MACsec Configuration

To display MACsec configuration information, perform one of the following tasks:

Command	Purpose
show key chain <i>name</i>	Displays the keychain configuration.
show macsec mka session [interface type slot/port] [detail]	Displays information about the MACsec MKA session for a specific interface or for all interfaces.
show macsec mka session details	Displays information about the MAC address and the ethernet type that is currently used by the interfaces for all EAPOL packets.
show macsec mka summary	Displays the MACsec MKA configuration.
show macsec policy [<i>policy-name</i>]	Displays the configuration for a specific MACsec policy or for all MACsec policies.
show running-config macsec	Displays the running configuration information for MACsec.

The following example displays information about the MACsec MKA session for all interfaces. .

```
switch# show macsec mka session
Interface      Local-TxSCI      #Peers      Status
Key-Server     Auth Mode
-----
Ethernet2/2    2c33.11b8.7d14/0001    1      Secured
Yes            PRIMARY-PSK
Ethernet2/3    2c33.11b8.7d18/0001    1      Secured
Yes            PRIMARY-PSK
-----
Total Number of Sessions : 2
    Secured Sessions : 2
    Pending Sessions : 0
```

The following example displays information about the MACsec MKA session for a specific interface. In addition to the common elements of the table as described in the previous example, the following also identifies the authentication mode which defines the current MACsec session type.

```
switch# show macsec mka session interface ethernet 1/1

Interface      Local-TxSCI      # Peers      Status      Key-Server      Auth Mode
```

```

-----
Ethernet1/1    70df.2fdc.baf4/0001    0    Pending    Yes    PRIMARY-PSK
Ethernet1/1    70df.2fdc.baf4/0001    1    Secured    No     FALLBACK-PSK

```

The following example displays detailed information about the MACsec MKA session for a specific Ethernet interface:

```

Interface Name      : Ethernet2/2
  Session Status      : SECURED - Secured MKA Session with MACsec
  Local Tx-SCI        : 2c33.11b8.7d14/0001
  Local Tx-SSCI       : 2
  MKA Port Identifier : 2
  CAK Name (CKN)      : 12
  CA Authentication Mode : PRIMARY-PSK
  Member Identifier (MI) : B54263EF7949A561E25CE617
  Message Number (MN)  : 523
  MKA Policy Name      : tests2
  Key Server Priority   : 16
  Key Server           : Yes
  Include ICV          : No
  SAK Cipher Suite     : GCM-AES-XPB-256
  SAK Cipher Suite (Operational) : GCM-AES-XPB-256
  Replay Window Size   : 148809600
  Confidentiality Offset : CONF-OFFSET-0
  Confidentiality Offset (Operational) : CONF-OFFSET-0
  Latest SAK Status    : Rx & TX
  Latest SAK AN        : 0
  Latest SAK KI        : B54263EF7949A561E25CE61700000001
  Latest SAK KN        : 1
  Last SAK key time    : 12:59:38 PST Tue Mar 19 2019
  CA Peer Count        : 1
  Eapol dest mac       : 0180.c200.0003
  Ether-type          : 0x888e
Peer Status:
  Peer MI              : 2C2C090E62A96F4D6E018210
  RxSCI                : 2c33.11b8.8b88/0001
  Peer CAK             : Match
  Latest Rx MKPDU      : 13:16:54 PST Tue Mar 19 2019

```

The following example displays the MACsec MKA configuration:

```

switch# show macsec mka summary
Interface      MACSEC-policy      Keychain
-----
Ethernet2/13   1                          1/10000000000000000000
Ethernet2/14   1                          1/10000000000000000000

```

The following example displays the configuration for all MACsec policies:

```

switch# show macsec policy
MACSec Policy      Cipher      Pri  Window  Offset  Security  SAK Rekey time ICV
Indicator Include-SCI
-----
KC256-Pol117b      GCM-AES-256    16   148809600  0    should-secure  pn-rollover
FALSE              True
pol1                GCM-AES-XPB-256 100  148809600  30   must-secure    60
FALSE              True
pol256-FanO         GCM-AES-XPB-256 16   148809600  0    must-secure    60
FALSE              True
pol256-MCT          GCM-AES-XPB-256 16   148809600  0    should-secure  60
FALSE              FALSE
system-default-

```

```
macsec-policy      GCM-AES-XPB-256  16   148809600   0   should-secure  pn-rollover
FALSE             FALSE
test1             GCM-AES-XPB-256  16   148809600   0   should-secure  pn-rollover
FALSE             True
```

The following example displays the key octet string in the output of the **show running-config** and **show startup-config** commands when the **key-chain macsec-psk no-show** command is not configured:

```
key chain KC256-1 macsec
  key 2000
    key-octet-string 7 075e701e1c5a4a5143475e5a527d7c7c706a6c724306170103555a5c57510b051e47080
a05000101005e0e50510f005c4b5f5d0b5b070e234e4d0a1d0112175b5e cryptographic-algorithm
AES_256_CMAC
```

The following example displays the key octet string in the output of the **show running-config** and **show startup-config** commands when the **key-chain macsec-psk no-show** command is configured:

```
key chain KC256-1 macsec
  key 2000
    key-octet-string 7 ***** cryptographic-algorithm AES_256_CMAC
```

Displaying MACsec Statistics

You can display MACsec statistics using the following commands.

Command	Description
show macsec mka statistics [<i>interface type slot/port</i>]	Displays MACsec MKA statistics.
show macsec secy statistics [<i>interface type slot/port</i>]	Displays MACsec security statistics.

The following example shows the MACsec MKA statistics for a specific Ethernet interface:

```
switch# show macsec mka statistics interface ethernet 2/2

Per-CA MKA Statistics for Session on interface (Ethernet2/2) with CKN 0x10
=====
CA Statistics
  Pairwise CAK Rekeys..... 0

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

MKPDU Statistics
  MKPDUs Transmitted..... 1096
  "Distributed SAK".. 0

  MKPDUs Validated & Rx... 0
  "Distributed SAK".. 0

MKA Statistics for Session on interface (Ethernet2/2)
=====
CA Statistics
  Pairwise CAK Rekeys..... 0

SA Statistics
```

```

SAKs Generated..... 0
SAKs Rekeyed..... 0
SAKs Received..... 0
SAK Responses Received.. 0

MKPDU Statistics
MKPDUs Transmitted..... 1096
  "Distributed SAK".. 0
MKPDUs Validated & Rx... 0
  "Distributed SAK".. 0
MKPDUs Tx Success..... 1096
MKPDUs Tx Fail..... 0
MKPDUs Tx Pkt build fail.. 0
MKPDUs No Tx on intf down.. 0
MKPDUs No Rx on intf down.. 0
MKPDUs Rx CA Not found..... 0
MKPDUs Rx Error..... 0
MKPDUs Rx Success..... 0

MKPDU Failures
MKPDU Rx Validation ..... 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
MKPDU Rx Drop Packet, Ethertype Mismatch. 0

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

CA Failures
ICK Derivation..... 0
KEK Derivation..... 0
Invalid Peer MACsec Capability... 0

MACsec Failures
Rx SA Installation..... 0
Tx SA Installation..... 0

```

The following example shows the MACsec security statistics for a specific Ethernet interface.



Note The following differences exist for uncontrolled and controlled packets in Rx and Tx statistics:

- Rx statistics:
 - Uncontrolled = Encrypted and unencrypted
 - Controlled = Decrypted
- Tx statistics:
 - Uncontrolled = Unencrypted
 - Controlled = Encrypted
 - Common = Encrypted and unencrypted

```
switch(config)# show macsec secy statistics interface e2/28/1
```

```
Interface Ethernet2/28/1 MACSEC SecY Statistics:
```

```
-----
```

```
Interface Rx Statistics:
```

```
Unicast Uncontrolled Pkts: 14987
Multicast Uncontrolled Pkts: 1190444
Broadcast Uncontrolled Pkts: 4
Uncontrolled Pkts - Rx Drop: 0
Uncontrolled Pkts - Rx Error: 0
Unicast Controlled Pkts: N/A (N9K-X9736C-FX not supported)
Multicast Controlled Pkts: N/A (N9K-X9736C-FX not supported)
Broadcast Controlled Pkts: N/A (N9K-X9736C-FX not supported)
Controlled Pkts: 247583
Controlled Pkts - Rx Drop: N/A (N9K-X9736C-FX not supported)
Controlled Pkts - Rx Error: N/A (N9K-X9736C-FX not supported)
In-Octets Uncontrolled: 169853963 bytes
In-Octets Controlled: 55027017 bytes
Input rate for Uncontrolled Pkts: N/A (N9K-X9736C-FX not supported)
Input rate for Uncontrolled Pkts: N/A (N9K-X9736C-FX not supported)
Input rate for Controlled Pkts: N/A (N9K-X9736C-FX not supported)
Input rate for Controlled Pkts: N/A (N9K-X9736C-FX not supported)
```

```
Interface Tx Statistics:
```

```
Unicast Uncontrolled Pkts: N/A (N9K-X9736C-FX not supported)
Multicast Uncontrolled Pkts: N/A (N9K-X9736C-FX not supported)
Broadcast Uncontrolled Pkts: N/A (N9K-X9736C-FX not supported)
Uncontrolled Pkts - Rx Drop: N/A (N9K-X9736C-FX not supported)
Uncontrolled Pkts - Rx Error: N/A (N9K-X9736C-FX not supported)
Unicast Controlled Pkts: N/A (N9K-X9736C-FX not supported)
Multicast Controlled Pkts: N/A (N9K-X9736C-FX not supported)
Broadcast Controlled Pkts: N/A (N9K-X9736C-FX not supported)
Controlled Pkts: 205429
Controlled Pkts - Rx Drop: N/A (N9K-X9736C-FX not supported)
Controlled Pkts - Rx Error: N/A (N9K-X9736C-FX not supported)
Out-Octets Uncontrolled: N/A (N9K-X9736C-FX not supported)
Out-Octets Controlled: 20612648 bytes
Out-Octets Common: 151787484 bytes
Output rate for Uncontrolled Pkts: N/A (N9K-X9736C-FX not supported)
Output rate for Uncontrolled Pkts: N/A (N9K-X9736C-FX not supported)
Output rate for Controlled Pkts: N/A (N9K-X9736C-FX not supported)
Output rate for Controlled Pkts: N/A (N9K-X9736C-FX not supported)
```

```

SECY Rx Statistics:
  Transform Error Pkts: N/A (N9K-X9736C-FX not supported)
  Control Pkts: 952284
  Untagged Pkts: N/A (N9K-X9736C-FX not supported)
  No Tag Pkts: 0
  Bad Tag Pkts: 0
  No SCI Pkts: 0
  Unknown SCI Pkts: 0
  Tagged Control Pkts: N/A (N9K-X9736C-FX not supported)

SECY Tx Statistics:
  Transform Error Pkts: N/A (N9K-X9736C-FX not supported)
  Control Pkts: 967904
  Untagged Pkts: N/A (N9K-X9736C-FX not supported)

SAK Rx Statistics for AN [3]:
  Unchecked Pkts: 0
  Delayed Pkts: 0
  Late Pkts: 0
  OK Pkts: 1
  Invalid Pkts: 0
  Not Valid Pkts: 0
  Not-Using-SA Pkts: 0
  Unused-SA Pkts: 0
  Decrypted In-Octets: 235 bytes
  Validated In-Octets: 0 bytes

SAK Tx Statistics for AN [3]:
  Encrypted Protected Pkts: 2
  Too Long Pkts: N/A (N9K-X9736C-FX not supported)
  SA-not-in-use Pkts: N/A (N9K-X9736C-FX not supported)
  Encrypted Protected Out-Octets: 334 bytes
switch(config)#

```

Configuration Example for MACsec

The following example shows how to configure a user-defined MACsec policy and then apply the policy to interfaces:

```

switch(config)# macsec policy 1
switch(config-macsec-policy)# cipher-suite GCM-AES-256
switch(config-macsec-policy)# window-size 512
switch(config-macsec-policy)# key-server-priority 0
switch(config-macsec-policy)# conf-offset CONF-OFFSET-0
switch(config-macsec-policy)# security-policy should-secure
switch(config-macsec-policy)# exit

switch(config)# int e2/13-14
switch(config-if-range)# macsec keychain 1 policy 1
switch(config-if-range)# exit
switch(config)# show macsec mka summary

```

Interface	MACSEC-policy	Keychain
Ethernet2/13	1	1/10000000000000000
Ethernet2/14	1	1/10000000000000000

```

switch(config)# show macsec mka session

```

Interface	Local-TxSCI	# Peers	Status	Key-Server
Ethernet2/13	006b.f1be.d31c/0001	1	Secured	Yes
Ethernet2/14	006b.f1be.d320/0001	1	Secured	No

```
switch(config)# show running-config macsec
!Command: show running-config macsec
!Time: Mon Dec 5 04:53:40 2016
```

```
version 9.2(1)feature macsec
macsec policy 1
  cipher-suite GCM-AES-256
  key-server-priority 0
  window-size 512
  conf-offset CONF-OFFSET-0
  security-policy should-secure
```

```
interface Ethernet2/13
  macsec keychain 1 policy 1
```

```
interface Ethernet2/14
  macsec keychain 1 policy 1
```

The following example shows how to configure a MACsec keychain and then add the system default MACsec policy to the interfaces:

```
switch(config)# key chain 1 macsec
switch(config-macseckeychain)# key 1000
switch(config-macseckeychain-macseckey)# key-octet-string
abcdef0123456789abcdef0123456789abcdef0123456789abcdef0123456789 cryptographic-algorithm
aes_256_CM4C
switch(config-macseckeychain-macseckey)# exit
```

```
switch(config)# int e2/13-14
switch(config-if-range)# macsec keychain 1
switch(config-if-range)# exit
switch(config)#
```

```
switch(config)# show running-config macsec
!Command: show running-config macsec
!Time: Mon Dec 5 04:50:16 2016
version 7.0(3)I4(5)
feature macsec
interface Ethernet2/13
  macsec keychain 1 policy system-default-macsec-policy
interface Ethernet2/14
  macsec keychain 1 policy system-default-macsec-policy
```

```
switch(config)# show macsec mka session
```

Interface	Local-TxSCI	# Peers	Status
Key-Server	Auth Mode		
Ethernet2/2	2c33.11b8.7d14/0001	1	Secured
Yes	PRIMARY-PSK		
Ethernet2/3	2c33.11b8.7d18/0001	1	Secured
Yes	PRIMARY-PSK		

```
-----
Total Number of Sessions : 2
  Secured Sessions : 2
  Pending Sessions : 0
```

```
switch(config)# show macsec mka summary
```

Interface	Status	Cipher (Operational)	Key-Server	MACSEC-policy	Keychain
Fallback-keychain					
Ethernet2/1	down	-	-	tests1	keych1

no keychain						
Ethernet2/2	Secured	GCM-AES-XPB-256	Yes	tests2	keych2	
no keychain						
Ethernet2/3	Secured	GCM-AES-256	Yes	tests3	keyc3	
no keychain						

XML Examples

MACsec supports XML output for the following **show** commands for scripting purposes using **| xml**:

- **show key chain name | xml**
- **show macsec mka session interface interface slot/port details | xml**
- **show macsec mka statistics interface interface slot/port | xml**
- **show macsec mka summary | xml**
- **show macsec policy name | xml**
- **show macsec secy statistics interface interface slot/port | xml**
- **show running-config macsec | xml**

The following are example outputs for each of the preceding **show** commands:

Example 1: Displays the keychain configuration.

```
switch# show key chain "Kc2" | xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<nf:rpc-reply xmlns:nf="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns="http://www.cisco.com/nxos:1.0:rpm">
  <nf:data>
    <show>
      <key>
        <chain>
          <__XML__OPT_Cmd_rpm_show_keychain_cmd_keychain>
            <keychain>Kc2</keychain>
          </__XML__OPT_Cmd_rpm_show_keychain_cmd_keychain>
        </chain>
      </key>
    </show>
  </nf:data>
</nf:rpc-reply>
]]>]]>
```

Example 2: Displays information about the MACsec MKA session for a specific interface.

```
switch# show macsec mka session interface ethernet 4/31 details | xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<nf:rpc-reply xmlns:nf="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns="http://www.cisco.com/nxos:1.0">
  <nf:data>
    <show>
      <macsec>
        <mka>
          <session>
            <__XML__OPT_Cmd_show_macsec_mka_session_interface>
              <interface>
                <__XML__INTF_ifname>
```

Example 3: Displays MACsec MKA statistics.

Configure MACsec

```

    </__XML__INTF_ifname>
  </interface>
  <__XML__OPT_Cmd_some_macsec_mka_statistics__readonly__>
    <__readonly__>
      <TABLE_mka_intf_stats>
        <ROW_mka_intf_stats>
          <TABLE_ca_stats>
            <ROW_ca_stats>
              <ca_stat_ckn>0x2</ca_stat_ckn>
              <ca_stat_pairwise_cak_rekey>0</ca_stat_pairwise_cak_rekey>
              <sa_stat_sak_generated>0</sa_stat_sak_generated>
              <sa_stat_sak_rekey>0</sa_stat_sak_rekey>
              <sa_stat_sak_received>91</sa_stat_sak_received>
              <sa_stat_sak_response_rx>0</sa_stat_sak_response_rx>
              <mkpdu_stat_mkpdu_tx>2808</mkpdu_stat_mkpdu_tx>
              <mkpdu_stat_mkpdu_tx_distsak>0</mkpdu_stat_mkpdu_tx_distsak>
              <mkpdu_stat_mkpdu_rx>2714</mkpdu_stat_mkpdu_rx>
              <mkpdu_stat_mkpdu_rx_distsak>91</mkpdu_stat_mkpdu_rx_distsak>
            </ROW_ca_stats>
          </TABLE_ca_stats>
        </ROW_mka_intf_stats>
      </TABLE_mka_intf_stats>
    </__readonly__>
  </__XML__OPT_Cmd_some_macsec_mka_statistics__readonly__>
</interface>
  <__XML__INTF_ifname>
  <__XML__PARAM_value>
    <__XML__INTF_output>Ethernet4/31</__XML__INTF_output>
  </__XML__PARAM_value>
</__XML__INTF_ifname>
</interface>
  <__XML__OPT_Cmd_some_macsec_mka_statistics__readonly__>
    <__readonly__>
      <TABLE_mka_intf_stats>
        <ROW_mka_intf_stats>
          <TABLE_idb_stats>
            <ROW_idb_stats>
              <ca_stat_pairwise_cak_rekey>0</ca_stat_pairwise_cak_rekey>
              <sa_stat_sak_generated>0</sa_stat_sak_generated>
              <sa_stat_sak_rekey>0</sa_stat_sak_rekey>
              <sa_stat_sak_received>91</sa_stat_sak_received>
              <sa_stat_sak_response_rx>0</sa_stat_sak_response_rx>
              <mkpdu_stat_mkpdu_tx>2808</mkpdu_stat_mkpdu_tx>
              <mkpdu_stat_mkpdu_tx_distsak>0</mkpdu_stat_mkpdu_tx_distsak>
              <mkpdu_stat_mkpdu_rx>2714</mkpdu_stat_mkpdu_rx>
              <mkpdu_stat_mkpdu_rx_distsak>91</mkpdu_stat_mkpdu_rx_distsak>
              <idb_stat_mkpdu_tx_success>2808</idb_stat_mkpdu_tx_success>
              <idb_stat_mkpdu_tx_fail>0</idb_stat_mkpdu_tx_fail>
              <idb_stat_mkpdu_tx_pkt_build_fail>0</idb_stat_mkpdu_tx_pkt_build_fail>
              <idb_stat_mkpdu_no_tx_on_intf_down>0</idb_stat_mkpdu_no_tx_on_intf_down>
              <idb_stat_mkpdu_no_rx_on_intf_down>0</idb_stat_mkpdu_no_rx_on_intf_down>
              <idb_stat_mkpdu_rx_ca_notfound>0</idb_stat_mkpdu_rx_ca_notfound>
              <idb_stat_mkpdu_rx_error>0</idb_stat_mkpdu_rx_error>
              <idb_stat_mkpdu_rx_success>2714</idb_stat_mkpdu_rx_success>
              <idb_stat_mkpdu_failure_rx_integrity_check_error>0</idb_stat_mkpdu_
failure_rx_integrity_check_error>
              <idb_stat_mkpdu_failure_invalid_peer_mn_error>0</idb_stat_mkpdu_fai
lure_invalid_peer_mn_error>
              <idb_stat_mkpdu_failure_nonrecent_peerlist_mn_error>1</idb_stat_mkp
du_failure_nonrecent_peerlist_mn_error>
              <idb_stat_mkpdu_failure_sakuse_kn_mismatch_error>0</idb_stat_mkpdu_
failure_sakuse_kn_mismatch_error>
              <idb_stat_mkpdu_failure_sakuse_rx_not_set_error>0</idb_stat_mkpdu_f
ailure_sakuse_rx_not_set_error>
            </ROW_idb_stats>
          </TABLE_idb_stats>
        </ROW_mka_intf_stats>
      </TABLE_mka_intf_stats>
    </__readonly__>
  </__XML__OPT_Cmd_some_macsec_mka_statistics__readonly__>

```

```

        <idb_stat_mkpdu_failure_sakuse_key_mi_mismatch_error>0</idb_stat_mk
pdu_failure_sakuse_key_mi_mismatch_error>
        <idb_stat_mkpdu_failure_sakuse_an_not_in_use_error>0</idb_stat_mkpdu
failure_sakuse_an_not_in_use_error>
        <idb_stat_mkpdu_failure_sakuse_ks_rx_tx_not_set_error>0</idb_stat_m
kpdu_failure_sakuse_ks_rx_tx_not_set_error>
        <idb_stat_mkpdu_failure_sakuse_eapol_ethertype_mismatch_error>0</id
b_stat_mkpdu_failure_sakuse_eapol_ethertype_mismatch_error>
        <idb_stat_sak_failure_sak_generate_error>0</idb_stat_sak_failure_s
ak_generate_error>
        <idb_stat_sak_failure_hash_generate_error>0</idb_stat_sak_failure_h
ash_generate_error>
        <idb_stat_sak_failure_sak_encryption_error>0</idb_stat_sak_failure_
sak_encryption_error>
        <idb_stat_sak_failure_sak_decryption_error>0</idb_stat_sak_failure_
sak_decryption_error>
        <idb_stat_sak_failure_ick_derivation_error>0</idb_stat_sak_failure_
ick_derivation_error>
        <idb_stat_sak_failure_kek_derivation_error>0</idb_stat_sak_failure_
kek_derivation_error>
        <idb_stat_sak_failure_invalid_macsec_capability_error>0</idb_stat_s
ak_failure_invalid_macsec_capability_error>
        <idb_stat_macsec_failure_rx_sa_create_error>0</idb_stat_macsec_fail
ure_rx_sa_create_error>
        <idb_stat_macsec_failure_tx_sa_create_error>0</idb_stat_macsec_fail
ure_tx_sa_create_error>
    </ROW_idb_stats>
</TABLE_idb_stats>
</ROW_mka_intf_stats>
</TABLE_mka_intf_stats>
</__readonly__>
</__XML__OPT_Cmd_some_macsec_mka_statistics__readonly__>
</__XML__OPT_Cmd_some_macsec_mka_statistics_interface>
</statistics>
</mka>
</macsec>
</show>
</nf:data>
</nf:rpc-reply>
]]>]]>

```

Example 4: Displays the MACsec MKA configuration.

```

switch# show macsec mka summary | xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<nf:rpc-reply xmlns:nf="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns="http://w
ww.cisco.com/nxos:1.0">
  <nf:data>
    <show>
      <macsec>
        <mka>
          <__XML__OPT_Cmd_some_macsec_summary>
            <__XML__OPT_Cmd_some_macsec__readonly__>
              <__readonly__>
                <TABLE_mka_summary>
                  <ROW_mka_summary>
                    <ifname>Ethernet2/1</ifname>
                    <policy>am2</policy>
          <keychain>kc2/0200000000000000000000000000000000000000000000000000000000000000
00000000</keychain>
                </ROW_mka_summary>
                <ROW_mka_summary>
                  <ifname>Ethernet3/1</ifname>
                  <policy>am2</policy>

```

```

        <keychain>kc2/0200000000000000000000000000000000000000000000000000000000000000
00000000</keychain>
        </ROW_mka_summary>

[TRUNCATED FOR READABILITY]

<ROW_mka_summary>
    <ifname>Ethernet3/32</ifname>
    <policy>am2</policy>
    <keychain>kc2/0200000000000000000000000000000000000000000000000000000000000000
00000000</keychain>
    </ROW_mka_summary>
</TABLE_mka_summary>
</__readonly__>
</__XML_OPT_Cmd_some_macsec__readonly__>
</__XML_OPT_Cmd_some_macsec_summary>
</mka>
</macsec>
</show>
</nf:data>
</nf:rpc-reply>
]]>]]>

```

Example 5: Displays the configuration for a specific MACsec policy.

```

switch# show macsec policy am2 | xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<nf:rpc-reply xmlns:nf="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns="http://w
ww.cisco.com/nxos:1.0">
  <nf:data>
    <show>
      <macsec>
        <policy>
          <__XML_OPT_Cmd_some_macsec_policy_name>
            <policy_name>am2</policy_name>
            <__XML_OPT_Cmd_some_macsec__readonly__>
              <__readonly__>
                <TABLE_macsec_policy>
                  <ROW_macsec_policy>
                    <name>am2</name>
                    <cipher_suite>GCM-AES-XPB-256</cipher_suite>
                    <keyserver_priority>0</keyserver_priority>
                    <window_size>512</window_size>
                    <conf_offset>0</conf_offset>
                    <security_policy>must-secure</security_policy>
                    <sak-expiry-time>60</sak-expiry-time>
                  </ROW_macsec_policy>
                </TABLE_macsec_policy>
              </__readonly__>
            </__XML_OPT_Cmd_some_macsec__readonly__>
          </__XML_OPT_Cmd_some_macsec_policy_name>
        </policy>
      </macsec>
    </show>
  </nf:data>
</nf:rpc-reply>
]]>]]>

```

Example 6: Displays MACsec security statistics.

```

switch# show macsec secy statistics interface ethernet 4/31 | xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<nf:rpc-reply xmlns:nf="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns="http://w

```

```

ww.cisco.com/nxos:1.0">
<nf:data>
  <show>
    <macsec>
      <secy>
        <statistics>
          <interface>
            <__XML__INTF_ifname>
              <__XML__PARAM_value>
                <__XML__INTF_output>Ethernet4/31</__XML__INTF_output>
              </__XML__PARAM_value>
            <__XML__OPT_Cmd_some_macsec_secy_statistics__readonly__>
              <__readonly__>
                <TABLE_statistics>
                  <ROW_statistics>
                    <in_pkts_unicast_uncontrolled>0</in_pkts_unicast_uncontrolled>
                    <in_pkts_multicast_uncontrolled>42</in_pkts_multicast_uncontrolled>
                    <in_pkts_broadcast_uncontrolled>0</in_pkts_broadcast_uncontrolled>
                    <in_rx_drop_pkts_uncontrolled>0</in_rx_drop_pkts_uncontrolled>
                    <in_rx_err_pkts_uncontrolled>0</in_rx_err_pkts_uncontrolled>
                    <in_pkts_unicast_controlled>0</in_pkts_unicast_controlled>
                    <in_pkts_multicast_controlled>2</in_pkts_multicast_controlled>
                    <in_pkts_broadcast_controlled>0</in_pkts_broadcast_controlled>
                    <in_rx_drop_pkts_controlled>0</in_rx_drop_pkts_controlled>
                    <in_rx_err_pkts_controlled>0</in_rx_err_pkts_controlled>
                    <in_octets_uncontrolled>7230</in_octets_uncontrolled>
                    <in_octets_controlled>470</in_octets_controlled>
                    <input_rate_uncontrolled_pps>0</input_rate_uncontrolled_pps>
                    <input_rate_uncontrolled_bps>9</input_rate_uncontrolled_bps>
                    <input_rate_controlled_pps>0</input_rate_controlled_pps>
                    <input_rate_controlled_bps>23</input_rate_controlled_bps>
                    <out_pkts_unicast_uncontrolled>0</out_pkts_unicast_uncontrolled>
                    <out_pkts_multicast_uncontrolled>41</out_pkts_multicast_uncontrolled>
                    <out_pkts_broadcast_uncontrolled>0</out_pkts_broadcast_uncontrolled>
                    <out_rx_drop_pkts_uncontrolled>0</out_rx_drop_pkts_uncontrolled>
                    <out_rx_err_pkts_uncontrolled>0</out_rx_err_pkts_uncontrolled>
                    <out_pkts_unicast_controlled>0</out_pkts_unicast_controlled>
                    <out_pkts_multicast_controlled>2</out_pkts_multicast_controlled>
                    <out_pkts_broadcast_controlled>0</out_pkts_broadcast_controlled>
                    <out_rx_drop_pkts_controlled>0</out_rx_drop_pkts_controlled>
                    <out_rx_err_pkts_controlled>0</out_rx_err_pkts_controlled>
                    <out_octets_uncontrolled>6806</out_octets_uncontrolled>
                    <out_octets_controlled>470</out_octets_controlled>
                    <out_octets_common>7340</out_octets_common>
                    <output_rate_uncontrolled_pps>2598190092</output_rate_uncontrolled_pps>
                    <output_rate_uncontrolled_bps>2598190076</output_rate_uncontrolled_bps>
                    <output_rate_controlled_pps>0</output_rate_controlled_pps>
                    <output_rate_controlled_bps>23</output_rate_controlled_bps>
                    <in_pkts_transform_error>0</in_pkts_transform_error>
                    <in_pkts_control>40</in_pkts_control>
                    <in_pkts_untagged>0</in_pkts_untagged>
                    <in_pkts_no_tag>0</in_pkts_no_tag>
                    <in_pkts_badtag>0</in_pkts_badtag>
                    <in_pkts_no_sci>0</in_pkts_no_sci>
                    <in_pkts_unknown_sci>0</in_pkts_unknown_sci>
                    <in_pkts_tagged_ctrl>0</in_pkts_tagged_ctrl>
                    <out_pkts_transform_error>0</out_pkts_transform_error>
                    <out_pkts_control>41</out_pkts_control>
                    <out_pkts_untagged>0</out_pkts_untagged>
                    <rx_sa_an>1</rx_sa_an>
                    <in_pkts_unchecked>0</in_pkts_unchecked>
                    <in_pkts_delayed>0</in_pkts_delayed>
                    <in_pkts_late>0</in_pkts_late>
                    <in_pkts_ok>1</in_pkts_ok>

```

```

<in_pkts_invalid>0</in_pkts_invalid>
<in_pkts_not_valid>0</in_pkts_not_valid>
<in_pkts_not_using_sa>0</in_pkts_not_using_sa>
<in_pkts_unused_sa>0</in_pkts_unused_sa>
<in_octets_decrypted>223</in_octets_decrypted>
<in_octets_validated>0</in_octets_validated>
<tx_sa_an>1</tx_sa_an>
<out_pkts_encrypted_protected>1</out_pkts_encrypted_protected>
<out_pkts_too_long>0</out_pkts_too_long>
<out_pkts_sa_not_inuse>0</out_pkts_sa_not_inuse>
<out_octets_encrypted_protected>223</out_octets_encrypted_protected>
</ROW_statistics>
</TABLE_statistics>
</__readonly__>
</__XML__OPT_Cmd_some_macsec_secy_statistics__readonly__>
</__XML__INTF_ifname>
</interface>
</statistics>
</secy>
</macsec>
</show>
</nf:data>
</nf:rpc-reply>
]]>]]>

```

Example 7: Displays the running configuration information for MACsec.

```
switch# show running-config macsec | xml
```

```

!Command: show running-config macsec
!Time: Fri Jan 20 07:12:34 2017

version 7.0(3)I4(6)
*****
This may take time. Please be patient.
*****
<?xml version="1.0"?>
<nf:rpc xmlns:nf="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns="http://www.cis
co.com/nxos:7.0.3.I4.6.:configure_" xmlns:m="http://www.cisco.com/nxos:7.0.3.I4.
6.:_exec" xmlns:m1="http://www.cisco.com/nxos:7.0.3.I4.6.:configure__macsec-poli
cy" xmlns:m2="http://www.cisco.com/nxos:7.0.3.I4.6.:configure__if-eth-non-member
" message-id="1">
  <nf:get-config>
    <nf:source>
      <nf:running/>
    </nf:source>
    <nf:filter>
      <m:configure>
        <m:terminal>
          <feature>
            <macsec/>
          </feature>
          <macsec>
            <policy>
              <__XML__PARAM_policy_name>
                <__XML__value>am2</__XML__value>
              <m1:cipher-suite>
                <m1:__XML__PARAM_suite>
                  <m1:__XML__value>GCM-AES-XPB-256</m1:__XML__value>
                </m1:__XML__PARAM_suite>
              </m1:cipher-suite>
              <m1:key-server-priority>
                <m1:__XML__PARAM_pri>
                  <m1:__XML__value>0</m1:__XML__value>

```

```

        </m1:__XML__PARAM__pri>
    </m1:key-server-priority>
<m1>window-size>
<m1:__XML__PARAM__size>
    <m1:__XML__value>512</m1:__XML__value>
</m1:__XML__PARAM__size>
</m1>window-size>
<m1:conf-offset>
    <m1:__XML__PARAM__offset>
    <m1:__XML__value>CONF-OFFSET-0</m1:__XML__value>
</m1:__XML__PARAM__offset>
</m1:conf-offset>
<m1:security-policy>
    <m1:__XML__PARAM__policy>
    <m1:__XML__value>must-secure</m1:__XML__value>
</m1:__XML__PARAM__policy>
</m1:security-policy>
<m1:sak-expiry-time>
    <m1:__XML__PARAM__ts>
    <m1:__XML__value>60</m1:__XML__value>
</m1:__XML__PARAM__ts>
</m1:sak-expiry-time>
</__XML__PARAM__policy_name>
</policy>
</macsec>
<interface>
    <__XML__PARAM__interface>
    <__XML__value>Ethernet2/1</__XML__value>
    <m2:macsec>
        <m2:keychain>
            <m2:__XML__PARAM__keychain_name>
            <m2:__XML__value>kc2</m2:__XML__value>
            <m2:policy>
                <m2:__XML__PARAM__policy_name>
                <m2:__XML__value>am2</m2:__XML__value>
                </m2:__XML__PARAM__policy_name>
            </m2:policy>
            </m2:__XML__PARAM__keychain_name>
        </m2:keychain>
    </m2:macsec>
    </__XML__PARAM__interface>
</interface>

```

[TRUNCATED FOR READABILITY]

```

<interface>
    <__XML__PARAM__interface>
    <__XML__value>Ethernet4/31</__XML__value>
    <m2:macsec>
        <m2:keychain>
            <m2:__XML__PARAM__keychain_name>
            <m2:__XML__value>kc2</m2:__XML__value>
            <m2:policy>
                <m2:__XML__PARAM__policy_name>
                <m2:__XML__value>am2</m2:__XML__value>
                </m2:__XML__PARAM__policy_name>
            </m2:policy>
            </m2:__XML__PARAM__keychain_name>
        </m2:keychain>
    </m2:macsec>
    </__XML__PARAM__interface>
</interface>
</m:terminal>
</m:configure>

```

```
</nf:filter>
</nf:get-config>
</nf:rpc>
]]>]]>
```

MIBs

MACsec supports the following MIBs:

- IEEE8021-SECY-MIB
- CISCO-SECY-EXT-MIB

To locate and download supported MIBs, go to the following URL: <https://cisco.github.io/cisco-mibs/supportlists/nexus9000/Nexus9000MIBSupportList.html>.

Related Documentation

Related Topic	Document Title
Keychain management	Cisco Nexus 9000 Series NX-OS Security Configuration Guide
System messages	Cisco Nexus 9000 Series NX-OS System Messages References