MACsec Encryption

- Prerequisites for MACsec Encryption, on page 1
- Restrictions for MACsec Encryption, on page 1
- Information About MACsec Encryption, on page 2
- How to Configure MACsec Encryption, on page 10
- Configuration Examples for MACsec Encryption, on page 33
- Additional References for MACsec Encryption, on page 54
- Feature History for MACsec Encryption, on page 55

Prerequisites for MACsec Encryption

Prerequisites for Certificate-Based MACsec

- Ensure that you have a Certificate Authority (CA) server configured for your network.
- Generate a CA certificate.
- Ensure that you have configured Cisco Identity Services Engine (ISE) Release 2.0.
- Ensure that both the participating devices, the CA server, and Cisco Identity Services Engine (ISE) are synchronized using Network Time Protocol (NTP). If time is not synchronized on all your devices, certificates will not be validated.
- Ensure that 802.1x authentication and AAA are configured on your device.

Restrictions for MACsec Encryption

- MACsec configuration is not supported on EtherChannel ports. Instead, MACsec configuration can be applied on the individual member ports of an EtherChannel. To remove MACsec configuration, you must first unbundle the member ports from the EtherChannel, and then remove it from the individual member ports.
- MACsec with MKA is supported only on point-to-point links.
- GCM-AES-256 and XPN cipher suites (GCM-AES-XPN-128 and GCM-AES-XPN-256) are supported only with Network Advantage license.

- The MACsec Cipher announcement is not supported for MACsec Extended Packet Numbering (XPN) Ciphers and switch-to-switch MACsec connections.
- MACsec XPN Cipher Suites are not supported in switch-to-host MACsec connections.
- Certificate-based MACsec is supported only if the access-session is configured as closed or in multiple-host mode. None of the other configuration modes are supported.
- If the **dot1q tag vlan native** command is configured globally, the dot1x reauthentication will fail on trunk ports.
- MACsec XPN Cipher Suites do not provide confidentiality protection with a confidentiality offset, and these together are not supported in switch-to-switch MACsec connections.
- MACsec with Precision Time Protocol (PTP) is not supported.
- MACsec is not supported with Multicast VPN (mVPN).
- MACsec switch-to-host connections are not supported on Software-Defined Access deployments.
- **should-secure** access mode is supported on switch-to-switch ports only using PSK authentication.

Information About MACsec Encryption

The following sections provide information about MACsec encryption.

Recommendations for MACsec Encryption

This section list the recommendations for configuring MACsec encryption:

- Use the confidentiality (encryption) offset as 0 in switch-to-host connections.
- Use Bidirectional Forwarding and Detection (BFD) timer value as 750 milliseconds for 10Gbps ports and 1.25 seconds for any port with speed above 10Gbps.
- Execute the **shutdown** command, and then the **no shutdown** command on a port, after changing any MKA policy or MACsec configuration for active sessions, so that the changes are applied to active sessions.
- Use Extended Packet Numbering (XPN) Cipher Suite for port speeds of 40Gbps and above.
- Set the connectivity association key (CAK) rekey overlap timer to 30 seconds or more.
- Do not use Cisco TrustSec Security Association Protocol (SAP) MACsec encryption for port speeds above 10Gbps.
- Do not enable both Cisco TrustSec SAP and uplink MKA at the same time on any interface.
- We recommend that you use MACsec MKA encryption.

MACsec Encryption Overview

MACsec is the IEEE 802.1AE standard for authenticating and encrypting packets between two MACsec-capable devices. Catalyst switches support 802.1AE encryption with MACsec Key Agreement (MKA) on switch-to-host

links for encryption between the switch and host device. The switch also supports MACsec encryption for switch-to-switch (inter-network device) security using both Cisco TrustSec Network Device Admission Control (NDAC), Security Association Protocol (SAP) and MKA-based key exchange protocol.

Note When switch-to-switch MACSec is enabled, all traffic is encrypted, except the EAP-over-LAN (EAPOL) packets.

Link layer security can include both packet authentication between switches and MACsec encryption between switches (encryption is optional). Link layer security is supported on SAP-based MACsec.

Table 1: MACsec Support on Switch Ports

Connections	MACsec Support
Switch-to-Host	MACsec MKA Encryption
Switch-to-Switch	MACsec MKA encryption (recommended) Cisco TrustSec SAP

Cisco TrustSec and Cisco SAP are meant only for switch-to-switch links and are not supported on switch ports connected to end hosts, such as PCs or IP phones. MKA is supported on switch-to-host facing links as well as switch-to-switch links. Host-facing links typically use flexible authentication ordering for handling heterogeneous devices with or without IEEE 802.1x, and can optionally use MKA-based MACsec encryption. Cisco NDAC and SAP are mutually exclusive with Network Edge Access Topology (NEAT), which is used for compact switches to extend security outside the wiring closet.

Media Access Control Security and MACsec Key Agreement

MACsec, defined in 802.1AE, provides MAC-layer encryption over wired networks by using out-of-band methods for encryption keying. The MACsec Key Agreement (MKA) Protocol provides the required session keys and manages the required encryption keys. MKA and MACsec are implemented after successful authentication using the certificate-based MACsec or Pre Shared Key (PSK) framework.

A switch using MACsec accepts either MACsec or non-MACsec frames, depending on the policy associated with the MKA peer. MACsec frames are encrypted and protected with an integrity check value (ICV). When the switch receives frames from the MKA peer, it decrypts them and calculates the correct ICV by using session keys provided by MKA. The switch compares that ICV to the ICV within the frame. If they are not identical, the frame is dropped. The switch also encrypts and adds an ICV to any frames sent over the secured port (the access point used to provide the secure MAC service to a MKA peer) using the current session key.

The MKA Protocol manages the encryption keys used by the underlying MACsec protocol. The basic requirements of MKA are defined in 802.1x-REV. The MKA Protocol extends 802.1x to allow peer discovery with confirmation of mutual authentication and sharing of MACsec secret keys to protect data exchanged by the peers.

Note Starting with Cisco IOS XE 16.12.1 release, support for MKA with high availability has been introduced for Cisco Catalyst 9500 High Performance Series Switches. The high availability feature enables a pair of route processors to act as backup for each other. With high availability support for MKA if there is an active RP failure, the stand-by RP takes over existing MKA sessions in a minimally-disruptive switchover.

The EAP framework implements MKA as a newly defined EAP-over-LAN (EAPOL) packet. EAP authentication produces a master session key (MSK) shared by both partners in the data exchange. Entering the EAP session ID generates a secure connectivity association key name (CKN). The switch acts as the authenticator for both switch-to-switch and switch-to-host; and acts as the key server for switch-to-host. It generates a random secure association key (SAK), which is sent to the client partner. The client is never a key server and can only interact with a single MKA entity, the key server. After key derivation and generation, the switch sends periodic transports to the partner at a default interval of 2 seconds.

The packet body in an EAPOL Protocol Data Unit (PDU) is referred to as a MACsec Key Agreement PDU (MKPDU). MKA sessions and participants are deleted when the MKA lifetime (6 seconds) passes with no MKPDU received from a participant. For example, if a MKA peer disconnects, the participant on the switch continues to operate MKA until 6 seconds have elapsed after the last MKPDU is received from the MKA peer.

Note Integrity check value (ICV) indicator in MKPDU is optional. ICV is not optional when the traffic is encrypted.

MKA Policies

You apply a defined MKA policy to an interface to enable MKA on the interface. Removing the MKA policy disables MKA on that interface. You can configure these options:

- Policy name, not to exceed 16 ASCII characters.
- Confidentiality (encryption) offset of 0, 30, or 50 bytes for each physical interface

Definition of Policy-Map Actions

This section describes the policy-map actions and its definition:

- Activate: Applies a service template to the session.
- Authenticate: Starts authentication of the session.
- Authorize: Explicitly authorizes a session.
- Set-domain: Explicitly sets the domain of a client.
- Terminate: Terminates the method that is running, and deletes all the method details associated with the session.
- Deactivate: Removes the service-template applied to the session. If not applied, no action is taken.
- Set-timer: Starts a timer and gets associated with the session. When the timer expires, any action that needs to be started can be processed.
- Authentication-restart: Restarts authentication.

- Clear-session: Deletes a session.
- Pause: Pauses authentication.

Rest of the actions as self-explanatory and are associated with authentication.

Virtual Ports

Use virtual ports for multiple secured connectivity associations on a single physical port. Each connectivity association (pair) represents a virtual port. In switch-to-switch, you can have only one virtual port per physical port. In switch-to-host, you can have a maximum of two virtual ports per physical port, of which one virtual port can be part of a data VLAN; the other must externally tag its packets for the voice VLAN. You cannot simultaneously host secured and unsecured sessions in the same VLAN on the same port. Because of this limitation, 802.1x multiple authentication mode is not supported.

The exception to this limitation is in multiple-host mode when the first MACsec supplicant is successfully authenticated and connected to a hub that is connected to the switch. A non-MACsec host connected to the hub can send traffic without authentication because it is in multiple-host mode. We do not recommend using multi-host mode because after the first successful client, authentication is not required for other clients.

Virtual ports represent an arbitrary identifier for a connectivity association and have no meaning outside the MKA Protocol. A virtual port corresponds to a separate logical port ID. Valid port IDs for a virtual port are 0x0002 to 0xFFFF. Each virtual port receives a unique secure channel identifier (SCI) based on the MAC address of the physical interface concatenated with a 16-bit port ID.

MKA Statistics

Some MKA counters are aggregated globally, while others are updated both globally and per session. You can also obtain information about the status of MKA sessions. See Example: Displaying MKA Information, on page 48 for further information.

Key Lifetime and Hitless Key Rollover

A MACsec key chain can have multiple pre-shared keys (PSK) each configured with a key id and an optional lifetime. A key lifetime specifies at which time the key 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 key chain after the lifetime is expired. Time zone of the key can be local or UTC. Default time zone is UTC.

You can Key rolls over to the next key within the same key chain by configuring a second key in the key chain 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, key rolls over without traffic interruption.

On all participating devices, the MACsec key chain must be synchronised by using Network Time Protocol (NTP) and the same time zone must be used. If all the participating devices are not synchronized, the connectivity association key (CAK) rekey will not be initiated on all the devices at the same time.

Note The lifetime of the keys need to be overlapped in order to achieve hitless key rollover.

Replay Protection Window Size

Replay protection is a feature provided by MACsec to counter replay attacks. Each encrypted packet is assigned a unique sequence number and the sequence is verified at the remote end. Frames transmitted through a Metro Ethernet service provider network are highly susceptible to reordering due to prioritization and load balancing mechanisms used within the network.

A replay window is necessary to support the use of MACsec over provider networks that reorder frames. Frames within the window can be received out of order, but are not replay protected. The default window size is 0, which enforces strict reception ordering. The replay window size can be configured in the range of 0 to 2^{32}- 1. In case of XPN cipher suite, maximum replay window size is 2^{30}- 1, and if a higher window size is configured, the window size gets restricted to 2^{30}- 1. If the cipher suite is changed to a non-XPN cipher suite, then there is no restriction and the configured window size is used.

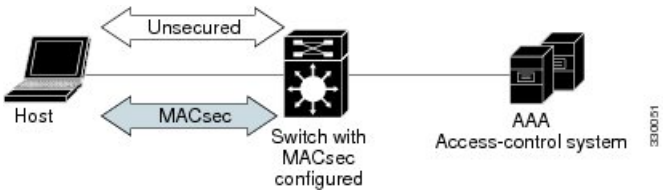
MACsec, MKA, and 802.1x Host Modes

You can use MACsec and the MKA Protocol with 802.1x single-host mode, multi-host mode, or Multi Domain Authentication (MDA) mode. Multiple authentication mode is not supported.

Single-Host Mode

The figure shows how a single EAP authenticated session is secured by MACsec by using MKA

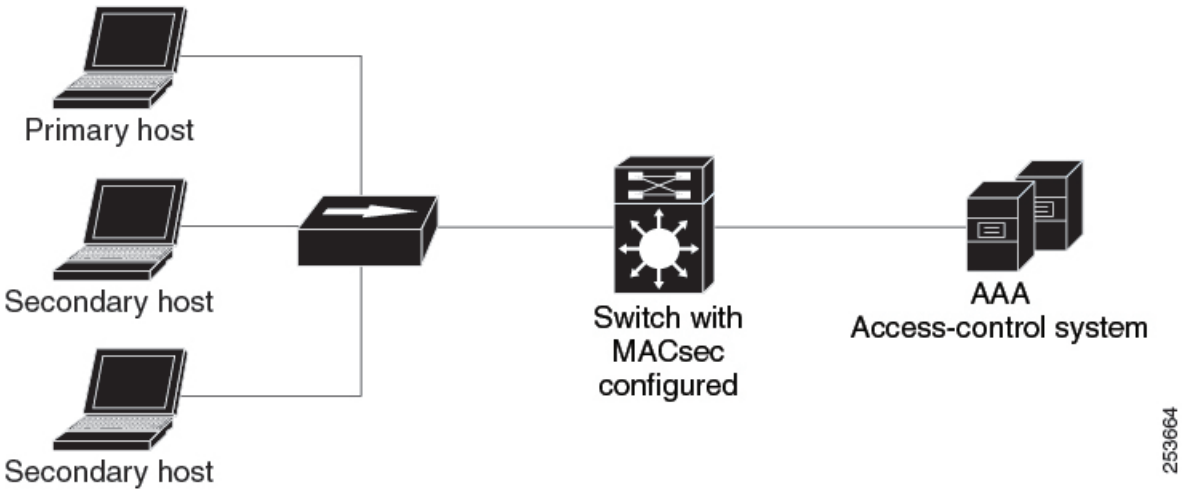
Figure 1: MACsec in Single-Host Mode with a Secured Data Session



Multiple Host Mode

In standard (not 802.1x REV) 802.1x multiple-host mode, a port is open or closed based on a single authentication. If one user, the primary secured client services client host, is authenticated, the same level of network access is provided to any host connected to the same port. If a secondary host is a MACsec supplicant, it cannot be authenticated and traffic would not flow. A secondary host that is a non-MACsec host can send traffic to the network without authentication because it is in multiple-host mode. The figure shows MACsec in Standard Multiple-Host Unsecure Mode.

Figure 2: MACsec in Multiple-Host Mode - Unsecured



Note Multi-host mode is not recommended because after the first successful client, authentication is not required for other clients, which is not secure.

Multiple-Domain Mode

In standard (not 802.1x REV) 802.1x multiple-domain mode, a port is open or closed based on a single authentication. If the primary user, a PC on data domain, is authenticated, the same level of network access is provided to any domain connected to the same port. If a secondary user is a MACsec supplicant, it cannot be authenticated and traffic would no flow. A secondary user, an IP phone on voice domain, that is a non-MACsec host, can send traffic to the network without authentication because it is in multiple-domain mode.

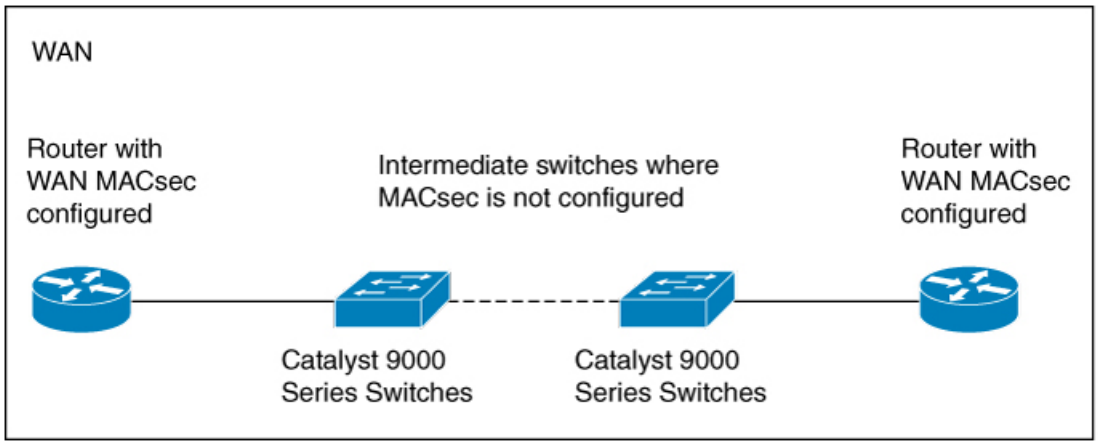
Certificate-Based MACsec Encryption

Using certificate-based MACsec encryption, you can configure MACsec MKA between device switch-to-switch ports. Certificate-based MACsec encryption allows mutual authentication and obtains an MSK (master session key) from which the connectivity association key (CAK) is derived for MKA operations. Device certificates are carried, using certificate-based MACsec encryption, for authentication to the AAA server.

MACsec Connections Across Intermediate Switches

Prior to Cisco IOS XE Gibraltar 16.10.1, MACsec connection between end devices which have WAN MACsec configured with the intermediate switches as the Cisco Catalyst 9000 Series Switches was not supported. The encrypted packets were dropped if WAN MACsec was configured on the end devices with MACsec not configured on the intermediate switches. With the ClearTag feature implemented on the ASIC, the switch forwards the encrypted packet without parsing the MACsec header. Below topology displays how the encrypted packets are forwarded through the intermediate switches with L2 switching.

Figure 3: Topology for ClearTag MACsec : MACsec Not Configured on the Intermediate Switches



Limitations for MACsec Connections Across Intermediate Switches

- Hop-by-hop MACsec encryption with Catalyst 9000 Series switches as intermediate switches where WAN MACsec is configured on the routers is not supported.
- WAN MACsec configured on the routers with intermediate switches as the Catalyst 9000 Series switches is not supported on Layer 3 VPNs.
- WAN MACsec configured on the routers with intermediate switches as the Catalyst 9000 Series switches show Cisco Discovery Protocol neighbors only in should-secure mode.

Switch-to-switch MKA MACsec Must Secure Policy

Starting with Cisco IOS XE Fuji 16.8.1a, must-secure support is enabled on both the ingress and the egress. Must-secure is supported for MKA and SAP. With must-secure enabled, only EAPoL traffic will not be encrypted. The rest of the traffic will be encrypted. Unencrypted packets are dropped.

Note Must-secure mode is enabled by default.

Prior to Cisco IOS XE Fuji 16.8.1a, should-secure was supported for MKA and SAP. With should-secure enabled, if the peer is configured for MACsec, the data traffic is encrypted, otherwise it is sent in clear text.

MACsec Extended Packet Numbering (XPN)

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^{31}- 1), SAK rekey takes place to refresh the data plane keys. For high capacity links such as 40 Gb/s, PN exhausts within a few seconds, and frequent SAK rekey to the control plane is required. When XPN is used, the PN of the MACsec frame is a 64-bit value, after reaching 75% of th of 2^{63}- 1, it will require several years to exhaust the PN; this ensures that frequent SAK rekey does not happen on high speed links. The XPN feature in MKA/MACsec eliminates the need for frequent SAK rekey that may occur in high capacity links. XPN is a mandatory requirement for FIPS/CC compliance on high speed links such as 40 Gb/s, 100 Gb/s, and so on.

Note MACsec XPN is supported only on the switch-to-switch ports.

The following rekey is possible in XPN:

- **Volume-based Rekey**—To ensure that frequent SAK rekey does not happen, you can configure XPN using the GCM-AES-XPN-128 or GCM-AES-XPN-256 cipher suites under the defined MKA policy; these cipher suites allow more than 2^{32} frames to be protected with a single SAK. XPN supports a 64-bit value for the PN. The MACsec frame contains only the lowest 32 bits and the most significant 32 bits would be maintained by the peer itself, both the sending and the receiving peers. The most significant 32 bits of the PN is incremented at the receiving end when the MSB (most significant bits) of LAPN (lowest acceptable packet number) for the respective peer is set, and the MSB of the PN value received in the MACsec frame is 0. Thus, both the sending and the receiving peer maintain the same PN value without changing the MACsec frame structure.
- **Time-based Rekey**—To set the SAK rekey manually, timer-based rekey is supported where you have the provision to start re-keying SAK at a given interval. Use the **sak rekey interval** *time-interval* command in MKA policy configuration mode to configure the SAK rekey interval for a defined MKA policy applied to the interface.

MKA/MACsec for Port Channel

MKA/MACsec can be configured on the port members of a port channel. MKA/MACsec is agnostic to the port channel since the MKA session is established between the port members of a port channel.

Note Etherchannel links that are formed as part of the port channel can either be congruent or disparate i.e. the links can either be MACsec-secured or non-MACsec-secured. MKA session between the port members is established even if a port member on one side of the port channel is not configured with MACsec.

It is recommended that you enable MKA/MACsec on all the member ports for better security of the port channel.

MACsec Cipher Announcement

Cipher Announcement allows the supplicant and the authenticator to announce their respective MACsec Cipher Suite capabilities to each other. Both the supplicant and the authenticator calculate the largest common supported MACsec Cipher Suite and use the same as the keying material for the MKA session.

Note Only the MACsec Cipher Suite capabilities which are configured in the MKA policy are announced from the authenticator to the supplicant.

There are two types of EAPOL Announcements:

- Unsecured Announcements (EAPOL PDUs) : Unsecured announcements are EAPOL announcements carrying MACsec Cipher Suite capabilities in an unsecured manner. These announcements are used to decide the width of the key used for MKA session prior to authentication.

- Secure Announcements (MKPDUs) : Secure announcements revalidate the MACsec Cipher Suite capabilities which were shared previously through unsecure announcements.

Once the session is authenticated, peer capabilities which were received through EAPOL announcements are revalidated with the secure announcements. If there is a mismatch in the capabilities, the MKA session tears down.

Note The MKA session between the supplicant and the authenticator does not tear down even if the MACsec Cipher Suite Capabilities configured on both do not result in a common cipher suite.

How to Configure MACsec Encryption

The following sections provide information about the various tasks that comprise MACsec encryption.

Configuring MKA and MACsec

By default, MACsec is disabled. No MKA policies are configured.

Configuring an MKA Policy

Beginning in privileged EXEC mode, follow these steps to create an MKA Protocol policy. Note that MKA also requires that you enable 802.1x.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mka policy** *policy-name* **Example:** `Device(config)# mka policy mka_policy`	Identifies an MKA policy, and enters MKA policy configuration mode. The maximum policy name length is 16 characters. **Note** The default MACsec cipher suite in the MKA policy will always be "GCM-AES-128". If the device supports both "GCM-AES-128" and "GCM-AES-256" ciphers, it is highly recommended to define and use a user defined MKA policy to include both 128 and 256 bits

	Command or Action	Purpose
		ciphers or only 256 bits cipher, as may be required.
Step 4	**key-server** *priority* **Example:** `Device(config-mka-policy)# key-server priority 200`	Configures MKA key server options and set priority (between 0-255). **Note** When value of key server priority is set to 255, the peer can not become the key server. The key server priority value is valid only for MKA PSK; and not for MKA EAPTLS.
Step 5	**include-icv-indicator** **Example:** `Device(config-mka-policy)# include-icv-indicator`	Enables the ICV indicator in MKPDU. Use the **no** form of this command to disable the ICV indicator.
Step 6	**macsec-cipher-suite** {*gcm-aes-128* | *gcm-aes-256*} **Example:** `Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128`	Configures a cipher suite for deriving SAK with 128-bit or 256-bit encryption.
Step 7	**confidentiality-offset** *offset-value* **Example:** `Device(config-mka-policy)# confidentiality-offset 0`	Set the confidentiality (encryption) offset for each physical interface. **Note** Offset Value can be 0, 30 or 50. If you are using Anyconnect on the client, it is recommended to use Offset 0.
Step 8	**ssci-based-on-sci** **Example:** `Device(config-mka-policy)# ssci-based-on-sci`	(Optional) Computes Short Secure Channel Identifier (SSCI) value based on Secure Channel Identifier (SCI) value. The higher the SCI value, the lower is the SSCI value.
Step 9	**end** **Example:** `Device(config-mka-policy)# end`	Exit enters MKA policy configuration mode and returns to privileged EXEC mode.
Step 10	**show mka policy** **Example:** `Device# show mka policy`	Displays MKA policy configuration information.

Configuring Switch-to-host MACsec Encryption

Follow these steps to configure MACsec on an interface with one MACsec session for voice and one for data:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter the password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters the global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/0/1`	Identifies the MACsec interface, and enters interface configuration mode. The interface must be a physical interface.
Step 4	**switchport access vlan***vlan-id* **Example:** `Device(config-if)# switchport access vlan 1`	Configures the access VLAN for the port.
Step 5	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	Configures the interface as an access port.
Step 6	**macsec** **Example:** `Device(config-if)# macsec`	Enables 802.1ae MACsec on the interface. The macsec command enables MKA MACsec on switch-to-host links only.
Step 7	**authentication event linksec fail action authorize vlan** *vlan-id* **Example:** `Device(config-if)# authentication event linksec fail action authorize vlan 1`	(Optional) Specifies that the switch processes authentication link-security failures resulting from unrecognized user credentials by authorizing a restricted VLAN on the port after a failed authentication attempt.
Step 8	**authentication host-mode multi-domain** **Example:** `Device(config-if)# authentication host-mode multi-domain`	Configures authentication manager mode on the port to allow both a host and a voice device to be authenticated on the 802.1x-authorized port. If not configured, the default host mode is single.
Step 9	**authentication linksec policy must-secure** **Example:** `Device(config-if)# authentication linksec policy must-secure`	Sets the LinkSec security policy to secure the session with MACsec if the peer is available. If not set, the default is *should secure*.

	Command or Action	Purpose
Step 10	**authentication port-control auto** **Example:** `Device(config-if)# authentication port-control auto`	Enables 802.1x authentication on the port. The port changes to the authorized or unauthorized state based on the authentication exchange between the switch and the client.
Step 11	**authentication periodic** **Example:** `Device(config-if)# authentication periodic`	(Optional) Enables or disables re-authentication for this port .
Step 12	**authentication timer reauthenticate** **Example:** `Device(config-if)# authentication timer reauthenticate`	(Optional) Enters a value between 1 and 65535 (in seconds). Obtains re-authentication timeout value from the server. Default re-authentication time is 3600 seconds.
Step 13	**authentication violation protect** **Example:** `Device(config-if)# authentication violation protect`	Configures the port to drop unexpected incoming MAC addresses when a new device connects to a port or when a device connects to a port after the maximum number of devices are connected to that port. If not configured, the default is to shut down the port.
Step 14	**mka policy** *policy-name* **Example:** `Device(config-if)# mka policy mka_policy`	Applies an existing MKA protocol policy to the interface, and enable MKA on the interface. If no MKA policy was configured (by entering the**mka policy** global configuration command).
Step 15	**dot1x pae authenticator** **Example:** `Device(config-if)# dot1x pae authenticator`	Configures the port as an 802.1x port access entity (PAE) authenticator.
Step 16	**spanning-tree portfast** **Example:** `Device(config-if)# spanning-tree portfast`	Enables spanning tree Port Fast on the interface in all its associated VLANs. When the Port Fast feature is enabled, the interface changes directly from a blocking state to a forwarding state without making the intermediate spanning-tree state changes
Step 17	**end** **Example:** `Device(config)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 18	**show authentication session interface** *interface-id* **details** **Example:** `Device# show authentication session interface GigabitEthernet 1/0/1`	Verifies the details of the security status of the authorized session.

	Command or Action	Purpose
Step 19	**show macsec interface** *interface-id* **Example:** `Device# show macsec interface GigabitEthernet 1/0/1`	Verifies the MACsec status on the interface.
Step 20	**show mka sessions** **Example:** `Device# show mka sessions`	Verifies the established MKA sessions.

Configuring MKA MACsec using PSK

Configuring MACsec MKA using PSK

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**key chain** *key-chain-name* **macsec** **Example:** `Device(config)# key chain keychain1 macsec`	Configures a key chain and enters the key chain configuration mode.
Step 4	**key** *hex-string* **Example:** `Device(config-key-chain)# key 1000`	Configures a unique identifier for each key in the keychain and enters the keychain's key configuration mode. **Note** For 128-bit encryption, use any value between 1 and 32 hex digit key-string. For 256-bit encryption, use 64 hex digit key-string.
Step 5	**cryptographic-algorithm** *{aes-128-cmac | aes-256-cmac}* **Example:** `Device(config-key-chain)# cryptographic-algorithm aes-128-cmac`	Set cryptographic authentication algorithm with 128-bit or 256-bit encryption.

	Command or Action	Purpose
Step 6	**key-string** *{ [0|6|7] pwd-string | pwd-string}* **Example:** `Device(config-key-chain)# key-string 12345678901234567890123456789012`	Sets the password for a key string. Only hex characters must be entered.
Step 7	**lifetime local** [*start timestamp {hh::mm::ss | day | month | year}*] [**duration** *seconds* | *end timestamp {hh::mm::ss | day | month | year}*] **Example:** `Device(config-key-chain)# lifetime local 12:12:00 July 28 2016 12:19:00 July 28 2016`	Sets the lifetime of the pre shared key.
Step 8	**end** **Example:** `Device(config-key-chain)# end`	Exits key chain configuration mode and returns to privileged EXEC mode.

Configuring MACsec MKA on an Interface using PSK

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config-if)# interface GigabitEthernet 0/0/0`	Enters interface configuration mode.
Step 4	**macsec network-link** **Example:** `Device(config-if)# macsec network-link`	Enables MACsec on the interface.
Step 5	**mka policy** *policy-name* **Example:** `Device(config-if)# mka policy mka_policy`	Configures an MKA policy.

	Command or Action	Purpose
Step 6	**mka pre-shared-key key-chain** *key-chain name* **Example:** `Device(config-if)# mka pre-shared-key key-chain key-chain-name`	Configures an MKA pre-shared-key key-chain name.
Step 7	**macsec replay-protection window-size** *frame number* **Example:** `Device(config-if)# macsec replay-protection window-size 10`	Sets the MACsec window size for replay protection.
Step 8	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

What to do next

It is not recommended to change the MKA policy on an interface with MKA PSK configured when the session is running. However, if a change is required, you must reconfigure the policy as follows:

1. Disable the existing session by removing macsec network-link configuration on each of the participating node using the **no macsec network-link** command
2. Configure the MKA policy on the interface on each of the participating node using the **mka policy policy-name** command.
3. Enable the new session on each of the participating node by using the **macsec network-link** command.

Configuring Certificate-Based MACsec Encryption

To configure MACsec with MKA on point-to-point links, perform these tasks:

- Configure Certificate Enrollment
  - Generate Key Pairs
  - Configure SCEP Enrollment
  - Configure Certificates Manually
- Configure an Authentication Policy
- Configure certificate-based MACsec encryption Profiles and IEEE 802.1x Credentials
- Configure MKA MACsec using certificate-based MACsec encryption on Interfaces

Generating Key Pairs

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto key generate rsa label** *label-name* **general-keys modulus** *size* **Example:** `Device(config)# crypto key generate rsa label general-keys modulus 2048`	Generates a RSA key pair for signing and encryption. You can also assign a label to each key pair using the label keyword. The label is referenced by the trustpoint that uses the key pair. If you do not assign a label, the key pair is automatically labeled <Default-RSA-Key>. If you do not use additional keywords this command generates one general purpose RSA key pair. If the modulus is not specified, the default key modulus of 1024 is used. You can specify other modulus sizes with the modulus keyword.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.
Step 5	**show authentication session interface** *interface-id* **Example:** `Device# show authentication session interface gigabitethernet 0/1/1`	Verifies the authorized session security status.

Configuring Enrollment using SCEP

Simple Certificate Enrollment Protocol (SCEP) is a Cisco-developed enrollment protocol that uses HTTP to communicate with the certificate authority (CA) or registration authority (RA). SCEP is the most commonly used method for sending and receiving requests and certificates.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** *server name* **Example:** `Device(config)# crypto pki trustpoint ka`	Declares the trustpoint and a given name and enters ca-trustpoint configuration mode.
Step 4	**enrollment url** *url name pem* **Example:** `Device(ca-trustpoint)# enrollment url http://url:80`	Specifies the URL of the CA on which your device should send certificate requests. An IPv6 address can be added in the URL enclosed in brackets. For example: http://[2001:DB8:1:1::1]:80. The pem keyword adds privacy-enhanced mail (PEM) boundaries to the certificate request.
Step 5	**rsakeypair** *label* **Example:** `Device(ca-trustpoint)# rsakeypair exampleCAkeys`	Specifies which key pair to associate with the certificate. **Note** The **rsakeypair** name must match the trust-point name.
Step 6	**serial-number none** **Example:** `Device(ca-trustpoint)# serial-number none`	The **none** keyword specifies that a serial number will not be included in the certificate request.
Step 7	**ip-address none** **Example:** `Device(ca-trustpoint)# ip-address none`	The **none** keyword specifies that no IP address should be included in the certificate request.
Step 8	**revocation-check crl** **Example:** `Device(ca-trustpoint)# revocation-check crl`	Specifies CRL as the method to ensure that the certificate of a peer has not been revoked.

	Command or Action	Purpose
Step 9	**auto-enroll** *percent* **regenerate** **Example:** `Device(ca-trustpoint)# auto-enroll 90 regenerate`	Enables auto-enrollment, allowing the client to automatically request a rollover certificate from the CA. If auto-enrollment is not enabled, the client must be manually re-enrolled in your PKI upon certificate expiration. By default, only the Domain Name System (DNS) name of the device is included in the certificate. Use the percent argument to specify that a new certificate will be requested after the percentage of the lifetime of the current certificate is reached. Use the regenerate keyword to generate a new key for the certificate even if a named key already exists. If the key pair being rolled over is exportable, the new key pair will also be exportable. The following comment will appear in the trustpoint configuration to indicate whether the key pair is exportable: "! RSA key pair associated with trustpoint is exportable." It is recommended that a new key pair be generated for security reasons.
Step 10	**exit** **Example:** `Device(ca-trustpoint)# exit`	Exits ca-trustpoint configuration mode and returns to global configuration mode.
Step 11	**crypto pki authenticate** *name* **Example:** `Device(config)# crypto pki authenticate myca`	Retrieves the CA certificate and authenticates it.
Step 12	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.
Step 13	**show crypto pki certificate** *trustpoint name* **Example:** `Device# show crypto pki certificate ka`	Displays information about the certificate for the trust point.

Configuring Enrollment Manually

If your CA does not support SCEP or if a network connection between the router and CA is not possible. Perform the following task to set up manual certificate enrollment:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** *server name* **Example:** `Device# crypto pki trustpoint ka`	Declares the trustpoint and a given name and enters ca-trustpoint configuration mode.
Step 4	**enrollment url** *url-name* **Example:** `Device(ca-trustpoint)# enrollment url http://url:80`	Specifies the URL of the CA on which your device should send certificate requests. An IPv6 address can be added in the URL enclosed in brackets. For example: http://[2001:DB8:1:1::1]:80. The pem keyword adds privacy-enhanced mail (PEM) boundaries to the certificate request.
Step 5	**rsakeypair** *label* **Example:** `Device(ca-trustpoint)# rsakeypair exampleCAkeys`	Specifies which key pair to associate with the certificate.
Step 6	**serial-number none** **Example:** `Device(ca-trustpoint)# serial-number none`	Specifies that serial numbers will not be included in the certificate request.
Step 7	**ip-address none** **Example:** `Device(ca-trustpoint)# ip-address none`	The **none** keyword specifies that no IP address should be included in the certificate request.
Step 8	**revocation-check crl** **Example:** `Device(ca-trustpoint)# revocation-check crl`	Specifies CRL as the method to ensure that the certificate of a peer has not been revoked.

	Command or Action	Purpose
Step 9	**exit** **Example:** `Device(ca-trustpoint)# exit`	Exits ca-trustpoint configuration mode and returns to global configuration mode.
Step 10	**crypto pki authenticate** *name* **Example:** `Device(config)# crypto pki authenticate myca`	Retrieves the CA certificate and authenticates it.
Step 11	**crypto pki enroll** *name* **Example:** `Device(config)# crypto pki enroll myca`	Generates certificate request and displays the request for copying and pasting into the certificate server. Enter enrollment information when you are prompted. For example, specify whether to include the device FQDN and IP address in the certificate request. You are also given the choice about displaying the certificate request to the console terminal. The base-64 encoded certificate with or without PEM headers as requested is displayed.
Step 12	**crypto pki import** *name* **certificate** **Example:** `Device(config)# crypto pki import myca certificate`	Imports a certificate via TFTP at the console terminal, which retrieves the granted certificate. The device attempts to retrieve the granted certificate via TFTP using the same filename used to send the request, except the extension is changed from ".req" to ".crt". For usage key certificates, the extensions "-sign.crt" and "-encr.crt" are used. The device parses the received files, verifies the certificates, and inserts the certificates into the internal certificate database on the switch. **Note** Some CAs ignore the usage key information in the certificate request and issue general purpose usage certificates. If your CA ignores the usage key information in the certificate request, only import the general purpose certificate. The router will not use one of the two key pairs generated.
Step 13	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
Step 14	**show crypto pki certificate** *trustpoint name* **Example:** `Device# show crypto pki certificate ka`	Displays information about the certificate for the trust point.

Configuring Switch-to-switch MACsec Encryption

To apply MACsec MKA using certificate-based MACsec encryption to interfaces, perform the following task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/2/1`	Identifies the MACsec interface, and enters interface configuration mode. The interface must be a physical interface.
Step 4	**macsec network-link** **Example:** `Device(config-if)# macsec network-link`	Enables MACsec on the interface.
Step 5	**authentication periodic** **Example:** `Device(config-if)# authentication periodic`	Enables reauthentication for this port.
Step 6	**authentication timer reauthenticate interval** **Example:** `Device(config-if)# authentication timer reauthenticate interval`	Sets the reauthentication interval.
Step 7	**access-session host-mode multi-host** **Example:** `Device(config-if)# access-session host-mode multi-host`	Allows hosts to gain access to the interface.

	Command or Action	Purpose
Step 8	**access-session closed** **Example:** `Device(config-if)# access-session closed`	Prevents preauthentication access on the interface.
Step 9	**access-session port-control auto** **Example:** `Device(config-if)# access-session port-control auto`	Sets the authorization state of a port.
Step 10	**dot1x pae both** **Example:** `Device(config-if)# dot1x pae both`	Configures the port as an 802.1X port access entity (PAE) supplicant and authenticator.
Step 11	**dot1x credentials profile** **Example:** `Device(config-if)# dot1x credentials profile`	Assigns a 802.1x credentials profile to the interface.
Step 12	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mdoe and returns to privileged EXEC mode.
Step 13	**show macsec interface** *interface-id* **Example:** `Device# show macsec interface GigabitEthernet 1/0/1`	Displays MACsec details for the interface.

Configuring MACsec XPN

Configuring an MKA Policy for XPN

Follow these steps to configure XPN in an MKA policy:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**mka policy** *policy-name* **Example:** `Device(config)# mka policy mka_policy`	Identifies an MKA policy, and enters MKA policy configuration mode. The maximum policy name length is 16 characters. **Note** The default MACsec cipher suite in the MKA policy will always be "GCM-AES-128". If the device supports both "GCM-AES-128" and "GCM-AES-256" ciphers, it is highly recommended to define and use a user defined MKA policy to include both 128 and 256 bits ciphers or only 256 bits cipher, as may be required.
Step 4	**macsec-cipher-suite** { *gcm-aes-128* | *gcm-aes-256* | *gcm-aes-xpn-128* | *gcm-aes-xpn-256*} **Example:** `Device(config-mka-policy)# macsec-cipher-suite gcm-aes-xpn-256`	Configures cipher suite for deriving SAK with 128-bit and 256-bit encryption for XPN.
Step 5	**sak-rekey interval** *time-interval* **Example:** `Device(config-mka-policy)# sak-rekey interval 50`	(Optional) Configures the SAK rekey interval (in seconds). The range is from 30 to 65535. By default, the SAK rekey interval occurs automatically depending on the interface speed. Use the no form of this command to stop the SAK rekey timer.
Step 6	**end** **Example:** `Device(config-mka-policy)# end`	Exits MKA policy configuration mode and returns to privileged EXEC mode.

Applying the XPN MKA Policy to an Interface

To apply the XPN MKA policy to an interface, perform the following task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *interface-name* **Example:** `Device(config)# interface gigabitethernet 1/0/1`	Identifies the MACsec interface, and enters interface configuration mode. The interface must be a physical interface.
Step 4	**mka policy** *policy-name* **Example:** `Device(config-if)# mka policy mka-xpn-policy`	Applies the XPN MKA protocol policy to the interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring MKA/MACsec for Port Channel

Configuring MKA/MACsec for Port Channel using PSK

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config-if)# interface gigabitethernet 1/0/3`	Enters interface configuration mode.
Step 4	**macsec network-link** **Example:** `Device(config-if)# macsec network-link`	Enables MACsec on the interface. Supports layer 2 and layer 3 port channels.
Step 5	**mka policy** *policy-name* **Example:** `Device(config-if)# mka policy mka_policy`	Configures an MKA policy.

	Command or Action	Purpose
Step 6	**mka pre-shared-key key-chain** *key-chain-name* **Example:** `Device(config-if)# mka pre-shared-key key-chain key-chain-name`	Configures an MKA pre-shared-key key-chain name. **Note** The MKA pre-shared key can be configured on either physical interface or sub-interfaces and not on both.
Step 7	**macsec replay-protection window-size** *frame number* **Example:** `Device(config-if)# macsec replay-protection window-size 0`	Sets the MACsec window size for replay protection.
Step 8	**channel-group** *channel-group-number* **mode** {**auto** | **desirable**} | {**active** | **passive**} | {**on**} **Example:** `Device(config-if)# channel-group 3 mode auto active on`	Configures the port in a channel group and sets the mode. **Note** You cannot configure ports in a channel group without configuring MACsec on the interface. You must configure the commands in Step 3, 4, 5 and 6 before this step. The channel-number range is from 1 to 4096. The port channel associated with this channel group is automatically created if the port channel does not already exist.For mode, select one of the following keywords: • **auto**— Enables PAgP only if a PAgP device is detected. This places the port into a passive negotiating state, in which the port responds to PAgP packets it receives but does not start PAgP packet negotiation. **Note** The **auto** keyword is not supported when EtherChannel members are from different switches in the switch stack. • **desirable** — Unconditionally enables PAgP. This places the port into an active negotiating state, in which the port starts negotiations with other ports by sending PAgP packets. **Note** The **desirable** keyword is not supported when EtherChannel members are from different switches in the switch stack.

	Command or Action	Purpose
		• **on**— Forces the port to channel without PAgP or LACP. In the on mode, an EtherChannel exists only when a port group in the **on** mode is connected to another port group in the **on** mode. • **active**— Enables LACP only if a LACP device is detected. It places the port into an active negotiating state in which the port starts negotiations with other ports by sending LACP packets. • **passive**— Enables LACP on the port and places it into a passive negotiating state in which the port responds to LACP packets that it receives, but does not start LACP packet negotiation.
Step 9	**end** **Example:** `Device(config-if)# cend`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Port Channel Logical Interfaces for Layer 2 EtherChannels

To create a port channel interface for a Layer 2 EtherChannel, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *channel-group-number* **Example:** `Device(config)# interface port-channel 1`	Creates the port channel interface, and enters interface configuration mode. **Note** Use the **no** form of this command to delete the port channel interface.
Step 4	**switchport** **Example:** `Device(config-if)# switchport`	Switches an interface that is in Layer 3 mode into Layer 2 mode for Layer 2 configuration.

	Command or Action	Purpose
Step 5	**switchport mode** {**access** | **trunk**} **Example:** `Device(config-if)# switchport mode access`	Assigns all ports as static-access ports in the same VLAN, or configure them as trunks.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Port Channel Logical Interfaces for Layer 3 EtherChannels

To create a port channel interface for a Layer 3 EtherChannel, perform this task:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/0/2`	Enters interface configuration mode.
Step 4	**no switchport** **Example:** `Device(config-if)# no switchport`	Switches an interface that is in Layer 2 mode into Layer 3 mode for Layer 3 configuration.
Step 5	**ip address** *ip-address subnet-mask* **Example:** `Device(config-if)# ip address 10.2.2.3 255.255.255.254`	Assigns an IP address and subnet mask to the EtherChannel.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring MACsec Cipher Announcement

The following sections provide information about the various tasks to configure MACsec cipher announcement.

Configuring an MKA Policy for Secure Announcement

Beginning in privileged EXEC mode, follow these steps to create an MKA Protocol policy to enable secure announcement in MKPDUs. By default, secure announcements are disabled.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mka policy** *policy-name* **Example:** `Device(config)# mka policy mka_policy`	Identifies an MKA policy and enters MKA policy configuration mode. The maximum policy name length is 16 characters. **Note** The default MACsec cipher suite in the MKA policy is GCM-AES-128. If the device supports both GCM-AES-128 and GCM-AES-256 ciphers, we recommend that you define and use a user-defined MKA policy to include both 128 and 256 bits ciphers or only 256 bits cipher, as may be required.
Step 4	**key-server** *priority* **Example:** `Device(config-mka-policy)# key-server priority 200`	Configures MKA key server options and sets priority between 0-255. **Note** When value of key server priority is set to 255, the peer cannot become the key server. The key server priority value is valid only for MKA PSK. This does not apply to MKA EAP-TLS.
Step 5	**send-secure-announcements** **Example:** `Device(config-mka-policy)# send-secure-announcements`	Enables sending of secure announcements. Use the **no** form of the command to disable sending of secure announcements. By default, secure announcements are disabled.

	Command or Action	Purpose
Step 6	**macsec-cipher-suite** {*gcm-aes-128* | *gcm-aes-256*} **Example:** `Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128`	Configures cipher suite for deriving SAK with 128-bit or 256-bit encryption.
Step 7	**end** **Example:** `Device(config-mka-policy)# end`	Exits MKA policy configuration mode and returns to privileged EXEC mode.
Step 8	**show mka policy** **Example:** `Device# show mka policy`	Displays MKA policies.

Configuring Secure Announcement Globally

Beginning in privileged EXEC mode, follow these steps to enable secure announcement globally across all the MKA policies.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**mka defaults policy send-secure-announcements** **Example:** `Device(config)# mka defaults policy send-secure-announcements`	Enables sending of secure announcements in MKPDUs across MKA policies. By default, secure announcements are disabled.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring EAPOL Announcements on an Interface

Beginning in privileged EXEC mode, follow these steps to configure EAPOL Announcement on an interface.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *interface-id* **Example:** `Device(config)# interface gigabitethernet 1/0/1`	Identifies the MACsec interface, and enters interface configuration mode. The interface must be a physical interface.
Step 4	**eapol annoucement** **Example:** `Device(config-if)# eapol announcement`	Enables EAPOL announcements. Use the **no** form of the command to disable EAPOL announcements. By default, EAPOL announcements are disabled.
Step 5	**end** **Example:** `Device(config-if)# configure terminal`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Cisco TrustSec MACsec

Configuring Cisco TrustSec Switch-to-Switch Link Security in Manual Mode

Before you begin

When manually configuring Cisco TrustSec on an interface, consider these usage guidelines and restrictions:

- If no SAP parameters are defined, Cisco TrustSec encapsulation or encryption is not performed.
- If you select GCM as the SAP operating mode, you must have a MACsec Encryption software license from Cisco. If you select GCM without the required license, the interface is forced to a link-down state.
- These protection levels are supported when you configure SAP pairwise master key (sap pmk):
  - SAP is not configured: no protection.
  - **sap mode-list gcm-encrypt gmac no-encap**: protection desirable but not mandatory.
  - **sap mode-list gcm-encrypt gmac**: confidentiality preferred and integrity required. The protection is selected by the supplicant according to supplicant preference.
  - **sap mode-list gmac**: integrity only.

- **sap mode-list gcm-encrypt**: confidentiality required.
- **sap mode-list gmac gcm-encrypt**: integrity required and preferred, confidentiality optional.

- Before changing the configuration from MKA to Cisco TrustSec SAP and vice versa, we recommend that you remove the interface configuration.

Beginning in privileged EXEC mode, follow these steps to manually configure Cisco TrustSec on an interface to another Cisco TrustSec device:

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *interface-id* **Example:** `Device(config)# interface tengigabitethernet 1/1/2`	Configures an interface, and enters interface configuration mode.
Step 3	**cts manual** **Example:** `Device(config-if)# cts manual`	Enters Cisco TrustSec manual configuration mode.
Step 4	**sap pmk** *key* [**mode-list** *mode1* [*mode2* [*mode3* [*mode4*]]]] **Example:** `Device(config-if-cts-manual)# sap pmk 1234abcdef mode-list gcm-encrypt no-encap`	(Optional) Configures the SAP pairwise master key (PMK) and operation mode. SAP is disabled by default in Cisco TrustSec manual mode. • *key*: A hexadecimal value with an even number of characters and a maximum length of 32 characters. The SAP operation mode options: • **gcm-encrypt**: Authentication and encryption **Note** Select this mode for MACsec authentication and encryption if your software license supports MACsec encryption. • **gmac**: Authentication, no encryption • **no-encap**: No encapsulation

	Command or Action	Purpose
Step 5	**no propagate sgt** **Example:** `Device(config-if-cts-manual)# no propagate sgt`	Use the **no** form of this command when the peer is incapable of processing a SGT. The **no propagate sgt** command prevents the interface from transmitting the SGT to the peer.
Step 6	**exit** **Example:** `Device(config-if-cts-manual)# exit`	Exits Cisco TrustSec 802.1x interface configuration mode.
Step 7	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 8	**show cts interface** [*interface-id* | **brief** | **summary**]	(Optional) Verify the configuration by displaying TrustSec-related interface characteristics.
Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuration Examples for MACsec Encryption

The following sections provide configuration examples for MACsec encryption.

Example: Configuring MKA and MACsec

This example shows how to create an MKA policy:

```
Device> enable
Device# configure terminal
Device(config)# mka policy mka_policy
Device(config-mka-policy)# key-server priority 200
Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128
Device(config-mka-policy)# confidentiality-offset 30
Device(config-mka-policy)# ssci-based-on-sci
Device(config-mka-policy)#end
```

This example shows how to configure MACsec on an interface:

```
Device> enable
Device# configure terminal
Device(config)# interface GigabitEthernet 1/0/1
Device(config-if)# switchport access vlan 1
Device(config-if)# switchport mode access
Device(config-if)# macsec
Device(config-if)# authentication event linksec fail action authorize vlan 1
Device(config-if)# authentication host-mode multi-domain
```

```
Device(config-if)# authentication linksec policy must-secure
Device(config-if)# authentication port-control auto
Device(config-if)# authentication periodic
Device(config-if)# authentication timer reauthenticate
Device(config-if)# authentication violation protect
Device(config-if)# mka policy mka_policy
Device(config-if)# dot1x pae authenticator
Device(config-if)# spanning-tree portfast
Device(config-if)# end
```

Example: Configuring MACsec MKA using PSK

This example shows how to configure MACsec MKA using PSK.

```
Device> enable
Device# configure terminal
Device(config)# key chain keychain1 macsec
Device(config-keychain)# key 1000
Device(config-keychain-key)# cryptographic-algorithm aes-128-cmac
Device(config-keychain-key)# key-string 12345678901234567890123456789012
Device(config-keychain-key)# lifetime local 12:12:00 July 28 2016 12:19:00 July 28 2016
Device(config-keychain-key)# end
```

This example shows how to configure MACsec MKA on an interface using PSK.

```
Device> enable
Device# configure terminal
Device(config)# interface GigabitEthernet 0/0/0
Device(config-if)# mka policy mka_policy
Device(config-if)# mka pre-shared-key key-chain key-chain-name
Device(config-if)# macsec replay-protection window-size 10
Device(config-if)# end
```

MKA-PSK: CKN Behavior Change

Starting Cisco IOS XE Fuji 16.8.1 release, for MKA PSK sessions, the CKN uses exactly the same string as the CKN which is configured as the hex-string for the key, instead of the fixed 32 bytes.

```
Device> enable
Device# configure terminal
Device(config)# key chain abc macsec
Device(config-keychain)# key 11
Device(config-keychain-key)# cryptographic-algorithm aes-128-cmac
Device(config-keychain-key)# key-string 12345678901234567890123456789013
Device(config-keychain-key)# lifetime local 12:21:00 Sep 9 2015 infinite
Device(config-keychain-key)# end
```

The following is sample output of the **show mka session** command for the above configuration:

```
Device# show mka session

Total MKA Sessions....... 1
Secured Sessions... 1
Pending Sessions... 0
=============================================================================================
Interface      Local-TxSCI         Policy-Name      Inherited         Key-Server

Port-ID        Peer-RxSCI          MACsec-Peers     Status            CKN

=============================================================================================
Et0/0          aabb.cc00.6600/0002     icv            NO                NO
```

```
2              aabb.cc00.6500/0002 1                Secured             11    *Note that the
 CKN key-string is exactly the same that has been configured for the key as hex-string.*
```

In case of interoperability between two images, where one having the CKN behavior change, and one without the CKN behavior change, the hex-string for the key must be a 64-character hex-string with zero padded for it to work on a device that has an image with the CKN behavior change. See the examples below:

Configuration without CKN key-string behavior change:

```
Device# configure terminal
Device(config)# key chain abc macsec
Device(config-keychain)# key 11
Device(config-keychain-key)# cryptographic-algorithm aes-128-cmac
Device(config-keychain-key)# key-string 12345678901234567890123456789013
Device(config-keychain-key)# lifetime local 12:21:00 Sep 9 2015 infinite
Device(config-keychain-key)# end
```

Configuration with CKN key-string behavior change:

```
Device# configure terminal
Device(config)# key chain abc macsec
Device(config-keychain)# key 1100000000000000000000000000000000000000000000000000000000000000
Device(config-keychain-key)# cryptographic-algorithm aes-128-cmac
Device(config-keychain-key)# key-string 12345678901234567890123456789013
Device(config-keychain-key)# lifetime local 12:21:00 Sep 9 2015 infinite
Device(config-keychain-key)# end
```

Example: Configuring MACsec MKA Using Certificate-Based MACsec Encryption

This example shows how to configure MACsec MKA using certificate-based MACsec encryption:

```
Device> enable
Device# configure terminal
Device(config)# interface Gigabitethernet 1/0/1
Device(config-if)# macsec network-link
Device(config-if)# authentication periodic
Device(config-if)# authentication timer reauthenticate interval
Device(config-if)# access-session host-mode multi-domain
Device(config-if)# access-session closed
Device(config-if)# access-session port-control auto
Device(config-if)# dot1x pae both
Device(config-if)#dot1x credentials profile
Device(config-if)# dot1x supplicant eap profile profile_eap_tls
Device(config-if)# end
```

Example: Configuring MACsec XPN

This example shows how to configure MACsec MKA XPN policy:

```
Device> enable
Device# configure terminal
Device(config)# mka policy mka-xpn-policy
Device(config-mka-policy)# macsec-cipher-suite gcm-aes-xpn-256
Device(config-mka-policy)# end
```

This example shows how to apply MACsec MKA XPN policy to an interface:

```
Device> enable
Device# configure terminal
```

```
Device(config)#interface Fo 1/0/1
Device(config-if)# mka policy mka-xpn-policy
Device(config-if)# end
```

The following is a sample output of the **show mka sessions details** command with 128-bit XPN Cipher Suite configured:

```
Device# show mka sessions details

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

Local Tx-SCI.............. 204c.9e85.ede4/002b
Interface MAC Address.... 204c.9e85.ede4
MKA Port Identifier...... 43
Interface Name........... GigabitEthernet1/0/1
Audit Session ID.........
CAK Name (CKN)........... 01000000000000000000000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)... D46CBEC05D5D67594543CEAE
Message Number (MN)...... 89572
EAP Role................. NA
Key Server............... YES
MKA Cipher Suite......... AES-128-CMAC

Latest SAK Status........ Rx & Tx
Latest SAK AN............ 0
Latest SAK KI (KN)....... D46CBEC05D5D67594543CEAE00000001 (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)

MKA Policy Name.......... p2
Key Server Priority...... 2
Delay Protection......... NO
Replay Protection........ YES
Replay Window Size....... 0
Confidentiality Offset... 0
Algorithm Agility........ 80C201
Send Secure Announcement.. DISABLED
SAK Cipher Suite......... 0080C20001000003 (GCM-AES-XPN-128)
MACsec Capability........ 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired........... YES

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

Live Peers List:
  MI                       MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------
  38046BA37D7DA77E06D006A9  89560       c800.8459.e764/002a   10

Potential Peers List:
  MI                       MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------

Dormant Peers List:
  MI                       MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------
```

The following is sample output of the **show mka sessions details** command with 256-bit XPN Cipher Suite configured:

```
Device# show mka sessions details

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

Local Tx-SCI.............. 204c.9e85.ede4/002b
Interface MAC Address.... 204c.9e85.ede4
MKA Port Identifier...... 43
Interface Name........... GigabitEthernet1/0/1
Audit Session ID.........
CAK Name (CKN)........... 01000000000000000000000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)... D46CBEC05D5D67594543CEAE
Message Number (MN)...... 89572
EAP Role................. NA
Key Server............... YES
MKA Cipher Suite......... AES-128-CMAC

Latest SAK Status........ Rx & Tx
Latest SAK AN............ 0
Latest SAK KI (KN)....... D46CBEC05D5D67594543CEAE00000001 (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)

MKA Policy Name.......... p2
Key Server Priority...... 2
Delay Protection......... NO
Replay Protection........ YES
Replay Window Size....... 0
Confidentiality Offset... 0
Algorithm Agility........ 80C201
Send Secure Announcement.. DISABLED
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.. 1

Live Peers List:
  MI                       MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------
  38046BA37D7DA77E06D006A9  89560       c800.8459.e764/002a   10

Potential Peers List:
  MI                       MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------

Dormant Peers List:
  MI                       MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------
```

Example: Configuring MACsec MKA for Port Channel using PSK

Etherchannel Mode — Static/On

The following is sample configuration on Device 1 and Device 2 with EtherChannel Mode on:

```
Device> enable
Device# configure terminal
Device(config)# key chain KC macsec
Device(config-key-chain)# key 1000
Device(config-key-chain)# cryptographic-algorithm aes-128-cmac
Device(config-key-chain)# key-string FC8F5B10557C192F03F60198413D7D45
Device(config-key-chain)# exit
Device(config)# mka policy POLICY
Device(config-mka-policy)# key-server priority 0
Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128
Device(config-mka-policy)# confidentiality-offset 0
Device(config-mka-policy)# exit
Device(config)# interface gigabitethernet 1/0/1
Device(config-if)# channel-group 2 mode on
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# exit
Device(config)# interface gigabitethernet 1/0/2
Device(config-if)# channel-group 2 mode on
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# end
```

Layer 2 EtherChannel Configuration

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

The following is sample output from the**show etherchannel summary** command:

```
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
```

```
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


 Number of channel-groups in use: 1
 Number of aggregators:           1

 Group  Port-channel  Protocol    Ports

------+-------------+-----------+-----------------------------------------------

 2      Po2(RU)          -        Te1/0/1(P)  Te1/0/2(P)
```

Layer 3 EtherChannel Configuration

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# no switchport
Device(config-if)# ip address 10.25.25.3 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# no switchport
Device(config-if)# ip address 10.25.25.4 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# end
```

The following is sample output from the **show etherchannel summary** command:

```
 Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port
```

```
        A - formed by Auto LAG


 Number of channel-groups in use: 1
 Number of aggregators:           1

 Group  Port-channel  Protocol    Ports

------+-------------+-----------+-----------------------------------------------

 2      Po2(RU)           -        Te1/0/1(P)   Te1/0/2(P)
```

Etherchannel Mode — LACP

The following is sample configuration on Device 1 and Device 2 with EtherChannel Mode as LACP.

```
Device> enable
Device# configure terminal
Device(config)# key chain KC macsec
Device(config-key-chain)# key 1000
Device(config-key-chain)# cryptographic-algorithm aes-128-cmac
Device(config-key-chain)# key-string FC8F5B10557C192F03F60198413D7D45
Device(config-key-chain)# exit
Device(config)# mka policy POLICY
Device(config-mka-policy)# key-server priority 0
Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128
Device(config-mka-policy)# confidentiality-offset 0
Device(config-mka-policy)# exit
Device(config)# interface gigabitethernet 1/0/1
Device(config-if)# channel-group 2 mode active
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# exit
Device(config)# interface gigabitethernet 1/0/2
Device(config-if)# channel-group 2 mode active
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# end
```

Layer 2 EtherChannel Configuration

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
```

```
Device(config-if)# no shutdown
Device(config-if)# end
```

The following is sample output from the **show etherchannel summary** command:

```
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


Number of channel-groups in use: 1
Number of aggregators:           1


------+-------------+-----------+-----------------------------------------------

2      Po2(SU)          LACP      Te1/1/1(P)   Te1/1/2(P)
```

Layer 3 EtherChannel Configuration

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# no switchport
Device(config-if)# ip address 10.25.25.3 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# no switchport
Device(config-if)# ip address 10.25.25.4 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# end
```

The following is sample output from the **show etherchannel summary** command:

```
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
```

```
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


 Number of channel-groups in use: 1
 Number of aggregators:           1

 Group  Port-channel  Protocol    Ports

------+-------------+-----------+-----------------------------------------------

 2      Po2(RU)          LACP      Te1/1/1(P)   Te1/1/2(P)
```

Etherchannel Mode — PAgP

The following is sample configuration on Device 1 and Device 2 with EtherChannel Mode as PAgP:

```
Device> enable
Device# configure terminal
Device(config)# key chain KC macsec
Device(config-key-chain)# key 1000
Device(config-key-chain)# cryptographic-algorithm aes-128-cmac
Device(config-key-chain)# key-string FC8F5B10557C192F03F60198413D7D45
Device(config-key-chain)# exit
Device(config)# mka policy POLICY
Device(config-mka-policy)# key-server priority 0
Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128
Device(config-mka-policy)# confidentiality-offset 0
Device(config-mka-policy)# exit
Device(config)# interface gigabitethernet 1/0/1
Device(config-if)# channel-group 2 mode desirable
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# exit
Device(config)# interface gigabitethernet 1/0/2
Device(config-if)# channel-group 2 mode desirable
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# end
```

Layer 2 EtherChannel Configuration

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

The following shows a sample output from the **show etherchannel summary** command.

```
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


Number of channel-groups in use: 1
Number of aggregators:           1


------+-------------+-----------+-----------------------------------------------

2      Po2(SU)          PAgP      Te1/1/1(P)  Te1/1/2(P)
```

Layer 3 EtherChannel Configuration

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# no switchport
Device(config-if)# ip address 10.25.25.3 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# no switchport
Device(config-if)# ip address 10.25.25.4 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# end
```

The following is sample output from the **show etherchannel summary** command:

```
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
```

```
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


 Number of channel-groups in use: 1
 Number of aggregators:           1

 Group  Port-channel  Protocol    Ports

------+-------------+-----------+-----------------------------------------------

 2      Po2(RU)          PAgP      Te1/1/1(P)  Te1/1/2(P)
```

Displaying Active MKA Sessions

The following shows all the active MKA sessions.

```
Device# show mka sessions interface Te1/0/1

=======================================================================================
Interface      Local-TxSCI          Policy-Name      Inherited
Key-Server
Port-ID        Peer-RxSCI           MACsec-Peers     Status            CKN
=======================================================================================
Te1/0/1        00a3.d144.3364/0025 POLICY           NO                NO

37             701f.539b.b0c6/0032 1                Secured
1000
```

Example: Configuring MACsec Cipher Announcement

This example shows how to configure MKA policy for Secure Announcement:

```
Device> enable
Device# configure terminal
Device(config)# mka policy mka_policy
Device(config-mka-policy)# key-server 2
Device(config-mka-policy)# send-secure-announcements
Device(config-mka-policy)#macsec-cipher-suite gcm-aes-128confidentiality-offset 0
Device(config-mka-policy)# end
```

This example shows how to configure Secure Announcement globally:

```
Device> enable
Device# configure terminal
Device(config)# mka defaults policy send-secure-announcements
Device(config)# end
```

This example shows how to configure EAPoL Announcements on an interface:

```
Device> enable
Device# configure terminal
Device(config)# interface GigabitEthernet 1/0/1
Device(config-if)# eapol announcement
Device(config-if)# end
```

The following is a sample output for **show running-config interface** *interface-name* command with EAPoL announcement enabled.

```
Device# show running-config interface GigabitEthernet 1/0/1

switchport mode access
 macsec
 access-session host-mode multi-host
 access-session closed
 access-session port-control auto
 dot1x pae authenticator
 dot1x timeout quiet-period 10
 dot1x timeout tx-period 5
 dot1x timeout supp-timeout 10
 dot1x supplicant eap profile peap
 eapol announcement
 spanning-tree portfast
 service-policy type control subscriber Dot1X
```

The following is a sample output of the **show mka sessions interface** *interface-name* **detail** command with secure announcement disabled.

```
Device# show mka sessions interface GigabitEthernet 1/0/1 detail


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

Local Tx-SCI.............. 204c.9e85.ede4/002b
Interface MAC Address.... 204c.9e85.ede4
MKA Port Identifier...... 43
Interface Name........... GigabitEthernet1/0/1
Audit Session ID.........
CAK Name (CKN)...........
0100000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)... D46CBEC05D5D67594543CEAE
Message Number (MN)...... 89567
EAP Role................. NA
Key Server............... YES
MKA Cipher Suite......... AES-128-CMAC

Latest SAK Status........ Rx & Tx
Latest SAK AN............ 0
Latest SAK KI (KN)....... D46CBEC05D5D67594543CEAE00000001 (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)

MKA Policy Name.......... p2
Key Server Priority...... 2
Delay Protection......... NO
Replay Protection........ YES
Replay Window Size....... 0
Confidentiality Offset... 0
Algorithm Agility........ 80C201
Send Secure Announcement.. DISABLED
SAK Cipher Suite......... 0080C20001000001 (GCM-AES-128)
MACsec Capability........ 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired........... YES

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

Live Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------
  38046BA37D7DA77E06D006A9  89555       c800.8459.e764/002a   10

Potential Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------

Dormant Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------
```

The following is sample output of the **show mka sessions details** command with secure announcement disabled.

```
Device# show mka sessions details

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

Local Tx-SCI............. 204c.9e85.ede4/002b
Interface MAC Address.... 204c.9e85.ede4
MKA Port Identifier...... 43
Interface Name........... GigabitEthernet1/0/1
Audit Session ID.........
CAK Name (CKN)...........
0100000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)... D46CBEC05D5D67594543CEAE
Message Number (MN)...... 89572
EAP Role................. NA
Key Server............... YES
```

```
MKA Cipher Suite......... AES-128-CMAC

Latest SAK Status........ Rx & Tx
Latest SAK AN............ 0
Latest SAK KI (KN)....... D46CBEC05D5D67594543CEAE00000001 (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)

MKA Policy Name.......... p2
Key Server Priority...... 2
Delay Protection......... NO
Replay Protection........ YES
Replay Window Size....... 0
Confidentiality Offset... 0
Algorithm Agility........ 80C201
Send Secure Announcement.. DISABLED
SAK Cipher Suite......... 0080C20001000001 (GCM-AES-128)
MACsec Capability........ 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired........... YES

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

Live Peers List:
  MI                       MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------
  38046BA37D7DA77E06D006A9  89560       c800.8459.e764/002a   10

Potential Peers List:
  MI                       MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------

Dormant Peers List:
  MI                       MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------
```

The following is sample output of the **show mka policy** *policy-name* **detail** command with secure announcement disabled.

```
Device# show mka policy p2 detail

MKA Policy Configuration ("p2")
========================
MKA Policy Name........ p2
Key Server Priority.... 2
Confidentiality Offset. 0
Send Secure Announcement..DISABLED
Cipher Suite(s)........ GCM-AES-128
```

```
Applied Interfaces...
  GigabitEthernet1/0/1
```

Example: Displaying MKA Information

The following is a sample output from the **show mka sessions** command:

```
Device# show mka sessions

Total MKA Sessions....... 1
      Secured Sessions... 1
      Pending Sessions... 0

=============================================================================================
Interface      Local-TxSCI         Policy-Name      Inherited         Key-Server
Port-ID        Peer-RxSCI          MACsec-Peers     Status            CKN
=============================================================================================
Gi1/0/1        204c.9e85.ede4/002b p2               NO                YES
43             c800.8459.e764/002a 1                Secured
0100000000000000000000000000000000000000000000000000000000000000
```

The following is a sample output from the **show mka sessions interface** *interface-name* command:

```
Device# show mka sessions interface GigabitEthernet 1/0/1

Summary of All Currently Active MKA Sessions on Interface GigabitEthernet1/0/1...

=============================================================================================
Interface      Local-TxSCI         Policy-Name      Inherited         Key-Server
Port-ID        Peer-RxSCI          MACsec-Peers     Status            CKN
=============================================================================================
Gi1/0/1        204c.9e85.ede4/002b p2               NO                YES
43             c800.8459.e764/002a 1                Secured
0100000000000000000000000000000000000000000000000000000000000000
```

The following is sample output from the **show mka sessions interface** *interface-name* **detail** command.

```
Device# show mka sessions interface GigabitEthernet 1/0/1 detail

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

Local Tx-SCI.............. 204c.9e85.ede4/002b
Interface MAC Address.... 204c.9e85.ede4
MKA Port Identifier...... 43
Interface Name........... GigabitEthernet1/0/1
Audit Session ID.........
CAK Name (CKN)........... 0100000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)... D46CBEC05D5D67594543CEAE
Message Number (MN)...... 89567
EAP Role................. NA
Key Server............... YES
MKA Cipher Suite......... AES-128-CMAC

Latest SAK Status........ Rx & Tx
Latest SAK AN............ 0
Latest SAK KI (KN)....... D46CBEC05D5D67594543CEAE00000001 (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)

MKA Policy Name..... p2
Key Server Priority..... 2
Delay Protection..... NO
Replay Protection..... YES
Replay Window Size..... 0
Confidentiality Offset... 0
Algorithm Agility..... 80C201
Send Secure Announcement.. DISABLED
SAK Cipher Suite..... 0080C20001000001 (GCM-AES-128)
MACsec Capability..... 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired..... YES

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

Live Peers List:
  MI                       MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------
  38046BA37D7DA77E06D006A9  89555       c800.8459.e764/002a   10

Potential Peers List:
  MI                       MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------

Dormant Peers List:
  MI                       MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------
```

The following is a sample output from the **show mka sessions details** command:

```
Device# show mka sessions details

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

Local Tx-SCI..... 204c.9e85.ede4/002b
Interface MAC Address.... 204c.9e85.ede4
MKA Port Identifier..... 43
Interface Name..... GigabitEthernet1/0/1
Audit Session ID.....
CAK Name (CKN)..... 01000000000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)... D46CBEC05D5D67594543CEAE
Message Number (MN)..... 89572
EAP Role..... NA
Key Server..... YES
MKA Cipher Suite..... AES-128-CMAC

Latest SAK Status..... Rx & Tx
Latest SAK AN..... 0
Latest SAK KI (KN)..... D46CBEC05D5D67594543CEAE00000001 (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)

MKA Policy Name..... p2
Key Server Priority..... 2
Delay Protection..... NO
Replay Protection..... YES
Replay Window Size..... 0
```

```
Confidentiality Offset... 0
Algorithm Agility........ 80C201
Send Secure Announcement.. DISABLED
SAK Cipher Suite......... 0080C20001000001 (GCM-AES-128)
MACsec Capability........ 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired........... YES

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

Live Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------
  38046BA37D7DA77E06D006A9  89560       c800.8459.e764/002a   10

Potential Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------

Dormant Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ------------------------------------------------------------------------
```

The following is a sample output from the **show mka policy** command:

```
Device# show mka policy

MKA Policy Summary...

Policy            KS       Delay   Replay  Window Conf   Cipher          Interfaces
Name              Priority Protect Protect Size   Offset Suite(s)        Applied
=============================================================================================================
*DEFAULT POLICY*  0        FALSE   TRUE    0      0      GCM-AES-128

p1                1        FALSE   TRUE    0      0      GCM-AES-128

p2                2        FALSE   TRUE    0      0      GCM-AES-128     Gi1/0/1
```

The following is a sample output from the **show mka policy** *policy-name* command:

```
Device# show mka policy p2

MKA Policy Summary...

Policy            KS       Delay   Replay  Window Conf   Cipher          Interfaces
Name              Priority Protect Protect Size   Offset Suite(s)        Applied
=============================================================================================================
p2                2        FALSE   TRUE    0      0      GCM-AES-128     Gi1/0/1
```

The following is a sample output from the **show mka policy** *policy-name* **detail** command:

```
Device# show mka policy p2 detail

MKA Policy Configuration ("p2")
=======================
MKA Policy Name........ p2
Key Server Priority.... 2
Confidentiality Offset. 0
Send Secure Announcement..DISABLED
Cipher Suite(s)........ GCM-AES-128

Applied Interfaces...
  GigabitEthernet1/0/1
```

The following is a sample output from the **show mka statistics interface** *interface-name* command:

```
Device# show mka statistics interface GigabitEthernet 1/0/1

MKA Statistics for Session
==========================
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.......... 1
   SAKs Rekeyed............ 0
   SAKs Received........... 0
   SAK Responses Received.. 1

MKPDU Statistics
   MKPDUs Validated & Rx... 89585
      "Distributed SAK".. 0
      "Distributed CAK".. 0
   MKPDUs Transmitted...... 89596
      "Distributed SAK".. 1
      "Distributed CAK".. 0
```

The following is a sample output from the **show mka summary** command:

```
Device# show mka summary

Total MKA Sessions....... 1
      Secured Sessions... 1
      Pending Sessions... 0

=============================================================================================================
Interface      Local-TxSCI         Policy-Name      Inherited         Key-Server
Port-ID        Peer-RxSCI          MACsec-Peers     Status            CKN
=============================================================================================================
Gi1/0/1        204c.9e85.ede4/002b p2               NO                YES
43             c800.8459.e764/002a 1                Secured
0100000000000000000000000000000000000000000000000000000000000000



MKA Global Statistics
=====================
MKA Session Totals
   Secured..................... 1
   Reauthentication Attempts.. 0

   Deleted (Secured)........... 0
   Keepalive Timeouts......... 0

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

SA Statistics
   SAKs Generated............. 1
   SAKs Rekeyed............... 0
   SAKs Received.............. 0
   SAK Responses Received..... 1
```

```
MKPDU Statistics
   MKPDUs Validated & Rx...... 89589
      "Distributed SAK"..... 0
      "Distributed CAK"..... 0
   MKPDUs Transmitted......... 89600
      "Distributed SAK"..... 1
      "Distributed CAK"..... 0

MKA Error Counter Totals
========================
Session Failures
   Bring-up Failures................ 0
   Reauthentication Failures........ 0
   Duplicate Auth-Mgr Handle........ 0

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

CA Failures
   Group CAK Generation............. 0
   Group CAK Encryption/Wrap........ 0
   Group CAK Decryption/Unwrap...... 0
   Pairwise CAK Derivation.......... 0
   CKN Derivation................... 0
   ICK Derivation................... 0
   KEK Derivation................... 0
   Invalid Peer MACsec Capability... 0
MACsec Failures
   Rx SC Creation................... 0
   Tx SC Creation................... 0
   Rx SA Installation............... 0
   Tx SA Installation............... 0

MKPDU Failures
   MKPDU Tx......................... 0
   MKPDU Rx Validation.............. 0
   MKPDU Rx Bad Peer MN............. 0
   MKPDU Rx Non-recent Peerlist MN.. 0
```

The following is a sample output from the **show macsec interface** command:

```
Device# show macsec interface HundredGigE 2/0/4

 MACsec is enabled
  Replay protect : enabled
  Replay window : 0
  Include SCI : yes
  Use ES Enable : no
  Use SCB Enable : no
  Admin Pt2Pt MAC : forceTrue(1)
  Pt2Pt MAC Operational : no
  Cipher : GCM-AES-128
  Confidentiality Offset : 0

 Capabilities
  ICV length : 16
  Data length change supported: yes
  Max. Rx SA : 16
  Max. Tx SA : 16
  Max. Rx SC : 8
```

```
 Max. Tx SC : 8
 Validate Frames : strict
 PN threshold notification support : Yes
 Ciphers supported : GCM-AES-128
                     GCM-AES-256
                     GCM-AES-XPN-128
                     GCM-AES-XPN-256

Access control : must secure

Transmit Secure Channels
 SCI : 3C5731BBB5850475
 SC state : inUse(1)
  Elapsed time : 7w0d
  Start time : 7w0d
  Current AN: 0
  Previous AN: -
  Next PN: 149757
  SA State: inUse(1)
  Confidentiality : yes
  SAK Unchanged : yes
  SA Create time : 00:04:41
  SA Start time : 7w0d
  SC Statistics
   Auth-only Pkts : 0
   Auth-only Bytes : 0
   Encrypted Pkts : 0
   Encrypted Bytes : 0
  SA Statistics
   Auth-only Pkts : 0
   Auth-only Bytes : 0
   Encrypted Pkts : 149756
   Encrypted Bytes : 16595088

 Port Statistics
  Egress untag pkts  0
  Egress long pkts  0

Receive Secure Channels
 SCI : 3C5731BBB5C504DF
 SC state : inUse(1)
  Elapsed time : 7w0d
  Start time : 7w0d
  Current AN: 0
  Previous AN: -
  Next PN: 149786
  RX SA Count: 0
  SA State: inUse(1)
  SAK Unchanged : yes
  SA Create time : 00:04:39
  SA Start time : 7w0d
  SC Statistics
   Notvalid pkts 0
   Invalid pkts 0
   Valid pkts 0
   Late pkts 0
   Uncheck pkts 0
   Delay pkts 0
   UnusedSA pkts 0
   NousingSA pkts 0
   Validated Bytes 0
   Decrypted Bytes 0
  SA Statistics
   Notvalid pkts 0
```

```
    Invalid pkts 0
    Valid pkts 149784
    Late pkts 0
    Uncheck pkts 0
    Delay pkts 0
    UnusedSA pkts 0
    NousingSA pkts 0
    Validated Bytes 0
    Decrypted Bytes 16654544

  Port Statistics
   Ingress untag pkts  0
   Ingress notag pkts  631726
   Ingress badtag pkts  0
   Ingress unknownSCI pkts  0
   Ingress noSCI pkts  0
   Ingress overrun pkts  0
```

Additional References for MACsec Encryption

Standards and RFCs

Standard/RFC	Title
IEEE 802.1AE-2006	*Media Access Control (MAC) Security*
IEEE 802.1X-2010	*Port-Based Network Access Control*
IEEE 802.1AEbw-2013	*Media Access Control (MAC) Security (Amendment to IEEE 802.1AE-2006)—Extended Packet Numbering (XPN)*
IEEE 802.1Xbx-2014	*Port-Based Network Access Control (Amendment to IEEE 802.1X-2010)*
RFC 4493	*The AES-CMAC Algorithm*

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature History for MACsec Encryption

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE Everest 16.5.1a	MACsec Encryption	MACsec is the IEEE 802.1AE standard for authenticating and encrypting packets between two MACsec-capable devices. Catalyst switches support 802.1AE encryption with MACsec Key Agreement (MKA) encryption between the switch and host device. Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, and C9500-40X models of the Cisco Catalyst 9500 Series Switches.
Cisco IOS XE Fuji 16.8.1a	MACsec Encryption	Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches.
Cisco IOS XE Gibraltar 16.12.1	MKA with High Availability	MKA with high availability is supported. Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches.

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.