eEdge Integration with MACsec

- eEdge Integration with MACsec, on page 1

eEdge Integration with MACsec

The Media Access Control Security (MACsec) standard is the IEEE 802.1AE standard for authenticating and encrypting packets between two MACsec-capable devices. The eEdge Integration with MACsec feature allows you to integrate the MACsec standard with enterprise edge (eEdge) devices to enhance Session Aware Networking capabilities. Session Aware Networking provides a policy and identity-based framework for edge devices to deliver flexible and scalable services to subscribers.

Prerequisites for eEdge Integration with MACsec

- Layer 2 encryption protocols like the IEEE 802.1AE Media Access Control Security (MACsec) standard must register with the eEdge session manager to receive disconnect notifications and perform cleanup.
- You must provision one virtual interface per secure association.

Restrictions for eEdge Integration with MACsec

- The Media Access Control Security (MACsec) standard is supported only in single-host and multihost modes. If a link layer security policy is configured as must-secure and the host mode is not configured as a single host or a multihost, the connection is closed.
- The MACsec standard is not supported in multi-authentication mode.
- The MACsec standard supports the 802.1AE encryption with MACsec Key Agreement (MKA) only on downlink ports for encryption between a MACsec-capable device and host devices.

Information About eEdge Integration with MACsec

The following sections provide information about eEdge Integration with MACsec feature.

Overview of MACsec

Media Access Control Security (MACsec) is the IEEE 802.1AE standard for authenticating and encrypting packets between two MACsec-capable devices. Implementing the MACsec encryption standard enables support for the 802.1AE encryption with MACsec Key Agreement (MKA) on downlink ports for encryption between a MACsec-capable device and host devices. The MACsec-capable device also supports MACsec link layer device-to-device security by using Cisco TrustSec Network Device Admission Control (NDAC) and the Security Association Protocol (SAP) key exchange. Link layer security includes both packet authentication between devices and MACsec encryption between devices (encryption is optional).

MACsec Standard Encryption

The Media Access Control Security (MACsec) standard provides data link 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 encryption keys. MKA and MACsec are implemented after a successful authentication by using the 802.1X Extensible Authentication Protocol (EAP) framework. Only host-facing links (links between network access devices and endpoint devices such as a PC or an IP phone) can be secured using MACsec.

A device that uses MACsec accepts either MACsec or non-MACsec frames, depending on the policy associated with the client. MACsec frames are encrypted and protected with an integrity check value (ICV). When the device receives frames from the client, it decrypts them and calculates the correct ICV by using session keys provided by MKA. The device compares the calculated value of the ICV to the ICV within the frame. If they are not identical, the frame is dropped. The device also encrypts and adds an ICV to any frame that is sent over a secured port (the access point used to provide the secure MAC service to a client) 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-2010. 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 peers.

EAP Implementation of MKA

The Extensible Authentication Protocol (EAP) framework implements MKA as a newly defined EAP-over-LAN (EAPOL) packet. EAP authentication produces a master session key (MSK) that is shared by both partners in the data exchange. Entering the EAP session ID generates a secure connectivity association key name (CKN). Because the device is the authenticator, it is also the key server, generating a random 128-bit secure association key (SAK), which it sends it 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 device 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 and no MKPDU is received from a participant. For example, if a client disconnects, the participant on the device continues to operate MKA until 6 seconds have elapsed after the last MKPDU is received from the client.

eEdge Integration with MACsec

The Media Access Control Security (MACsec) standard is the IEEE 802.1AE standard for authenticating and encrypting packets between two MACsec-capable devices. The eEdge Integration with MACsec feature allows you to integrate the MACsec standard with enterprise edge (eEdge) devices to enhance Session Aware Networking capabilities. Session Aware Networking provides a policy and identity-based framework for edge devices to deliver flexible and scalable services to subscribers.

How to Configure eEdge Integration with MACsec

Integrating eEdge with MACsec

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **service-template** *template-name*
4. **linksec policy** {**must-not-secure** | **must-secure** | **should-secure**}
5. **exit**
6. **policy-map type control subscriber** *control-policy-name*
7. **event authentication-success** [**match-all** | **match-any**]
8. *priority-number* **class** {*control-class-name* | **always**} [**do-all** | **do-until-failure** | **do-until-success**]
9. *action-number* **activate** {**policy type control subscriber** *control-policy-name* | **service-template** *template-name* [**aaa-list** *list-name*] [**precedence** [**replace-all**]}
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**service-template** *template-name* **Example:** `Device(config)# service-template dot1x-macsec-policy`	Defines a template that contains a set of service policy attributes to apply to subscriber sessions and enters service template configuration mode.
Step 4	**linksec policy** {**must-not-secure**	**must-secure**	**should-secure**} **Example:** `Device(config-service-template)# linksec policy must-secure`	Sets the link security policy as must-secure. • Must-secure policy authorizes the eEdge device port only if a secure MACsec session is established.
Step 5	**exit** **Example:** `Device(config-service-template)# exit`	Exits service template configuration mode and returns to global configuration mode.

	Command or Action	Purpose
Step 6	**policy-map type control subscriber** *control-policy-name* **Example:** `Device(config)# policy-map type control subscriber cisco-subscriber`	Defines a control policy for subscriber sessions and enters control policy-map event configuration mode.
Step 7	**event authentication-success** [**match-all** | **match-any**] **Example:** `Device(config-event-control-policymap)# event authentication-success match-all`	Specifies the type of event that triggers actions in a control policy if all authentication events are a match and enters control policy-map class configuration mode.
Step 8	*priority-number* **class** {*control-class-name* | **always**} [**do-all** | **do-until-failure** | **do-until-success**] **Example:** `Device(config-class-control-policymap)# 10 class always do-until-failure`	Specifies that the control class should execute the actions in a control policy, in the specified order, until one of the actions fails, and enters control policy-map action configuration mode.
Step 9	*action-number* **activate** {**policy type control subscriber** *control-policy-name* | **service-template** *template-name* [**aaa-list** *list-name*] [**precedence** [**replace-all**]} **Example:** `Device(config-action-control-policymap)# 10 activate service-template dot1x-macsec-policy`	Activates a control policy on a subscriber session.
Step 10	**end** **Example:** `Device(config-action-control-policymap)# end`	Exits control policy-map action configuration mode and enters privileged EXEC mode.

Identifying Link Layer Security Failures

SUMMARY STEPS

1. **configure terminal**
2. **class-map type control subscriber** {**match-all** | **match-any** | **match-none**} *control-class-name*
3. **match authorization-failure** {**domain-change-failed** | **linksec-failed**}
4. **exit**
5. **policy-map type control subscriber** *control-policy-name*
6. **event authentication-failure** [**match-all** | **match-any**]
7. *priority-number* **class** {*control-class-name* | **always**} [**do-all** | **do-until-failure** | **do-until-success**]
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**class-map type control subscriber** {**match-all** | **match-any** | **match-none**} *control-class-name* **Example:** `Device(config)# class-map type control subscriber match-all linksec-failed`	Creates a control class, which defines the conditions under which the actions of a control policy are executed and enters control class-map filter configuration mode.
Step 3	**match authorization-failure** {**domain-change-failed** | **linksec-failed**} **Example:** `Device(config-filter-control-classmap)# match authorization-failure linksec-failed`	Configures a match condition in a control class based on the type of authorization failure received from an authorization failed event of a link layer security failure.
Step 4	**exit** **Example:** `Device(config-class-control-policymap)# exit`	Exits control class-map filter configuration mode and enters global configuration mode.
Step 5	**policy-map type control subscriber** *control-policy-name* **Example:** `Device(config)# policy-map type control subscriber cisco-subscriber`	Defines a control policy for subscriber sessions and enters control policy-map event configuration mode.
Step 6	**event authentication-failure** [**match-all** | **match-any**] **Example:** `Device(config-event-control-policymap)# event authentication-failure match-all`	Specifies the type of event that triggers actions in a control policy if session authentication fails and enters control policy-map class configuration mode.
Step 7	*priority-number* **class** {*control-class-name* | **always**} [**do-all** | **do-until-failure** | **do-until-success**] **Example:** `Device(config-class-control-policymap)# 10 class linksec-failed do-until-failure`	Specifies that the control class must execute the actions in a control policy, in the specified order, until one of the actions fails and enters control policy-map action configuration mode.
Step 8	**end** **Example:** `Device(config-action-control-policymap)# end`	Exits control policy-map action configuration mode and enters privileged EXEC mode.

Configuration Examples for eEdge Integration with MACsec

Example: Integrating eEdge with MACsec

```
Device> enable
Device# configure terminal
Device(config)# service-template dot1x-macsec-policy
Device(config-service-template)# linksec policy must-secure
Device(config-service-template)# exit
Device(config)# policy-map type control subscriber cisco-subscriber
Device(config-event-control-policymap)# event authentication-success match-all
Device(config-class-control-policymap)# 10 class always do-until-failure
Device(config-action-control-policymap)# 10 activate service-template dot1x-macsec-policy
Device(config-action-control-policymap)# end
```

Example: Identifying Linksec Failures

```
Device# configure terminal
Device(config)# class-map type control subscriber match-all linksec-failure
Device(config-filter-control-classmap)# match authorization-failure linksec-failed
Device(config-class-control-classmap)# exit
Device(config)# policy-map type control subscriber cisco-subscriber
Device(config-event-control-policymap)# event authentication-failure match-all
Device(config-class-control-policymap)# 10 class linksec-failed do-until-failure
Device(config-action-control-policymap)# end
```

Feature Information for eEdge Integration with MACsec

Table 1: Feature Information for eEdge Integration with MACsec

Release	Feature Name	Feature Information
Cisco IOS XE Everest 16.6.1	eEdge Integration with MACsec	The eEdge Integration with MACsec feature allows you to integrate the MACsec standard with enterprise edge (eEdge) devices to enhance Session Aware Networking capabilities.
Cisco IOS XE Cupertino 17.7.1	eEdge Integration with MACsec	This feature was implemented on supervisor modules C9400X-SUP-2 and C9400X-SUP-2XL, which were introduced in this release.