# Dépannage de MACsec sur Catalyst 9000

## Table des matières



## Introduction

Ce document décrit la fonctionnalité MACsec, ses cas d'utilisation, et comment dépanner la fonctionnalité sur les commutateurs Catalyst 9000.

## Conditions préalables

### Exigences

Aucune exigence spécifique n'est associée à ce document.

### Composants utilisés

- C9300
- C9400
- C9500
- C9600

---

Remarque : consultez le guide de configuration approprié pour connaître les commandes utilisées afin d'activer ces fonctionnalités sur d'autres plates-formes Cisco.

---

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. Si votre réseau est en ligne, assurez-vous de bien comprendre l’incidence possible des commandes.

## Informations générales

Le champ d'application de ce document est le contrôle de sécurité d'accès au support (MACsec) sur le LAN, entre deux commutateurs/routeurs.

La communication de données en texte clair est sensible aux menaces de sécurité. Les failles de sécurité peuvent survenir à n’importe quelle couche du modèle OSI. Certaines des failles courantes au niveau de la couche 2 sont l’analyse, l’écoute électronique de paquets, la falsification, l’injection, l’usurpation d’adresse MAC, l’usurpation ARP, les attaques par déni de service (DoS) contre un serveur DHCP et le saut de VLAN.

MACsec est une technologie de chiffrement de couche 2 décrite dans la norme IEEE 802.1AE. MACsec sécurise les données sur les supports physiques et rend impossible toute compromission des données au niveau des couches supérieures. Par conséquent, le chiffrement MACsec est prioritaire sur toute autre méthode de chiffrement pour les couches supérieures, telles qu'IPsec et SSL.

### Avantages de MACsec

Mode orienté client : MACsec est utilisé dans les configurations où deux commutateurs qui s'appairent l'un avec l'autre peuvent alterner en tant que serveur de clés ou client de clés avant d'échanger des clés. Le serveur de clés génère et maintient le CAK entre les deux homologues.

Contrôle d'intégrité des données : MACsec utilise MKA pour générer une valeur de contrôle d'intégrité (ICV) pour la trame qui arrive sur le port. Si l’ICV généré est identique à l’ICV dans la trame, la trame est acceptée ; sinon, elle est abandonnée.

Cryptage des données : MACsec assure le cryptage au niveau du port sur les interfaces des commutateurs. Cela signifie que les trames envoyées à partir du port configuré sont chiffrées et que les trames reçues sur le port sont déchiffrées. MACsec fournit également un mécanisme qui

vous permet de configurer uniquement les trames chiffrées ou toutes les

les trames (chiffrées et en clair) sont acceptées sur l'interface.

Protection contre la relecture : lorsque des trames sont transmises sur le réseau, il est possible qu'elles sortent de la séquence ordonnée. MACsec fournit une fenêtre configurable qui accepte un nombre spécifié de trames hors séquence.

## MACsec et MTU

L'en-tête MACsec ajoute jusqu'à 32 octets de surcharge d'en-tête. Envisagez une unité de transmission maximale (MTU) système/interface plus importante sur les commutateurs du chemin pour tenir compte de la surcharge supplémentaire ajoutée par l'en-tête MACsec.  Si la MTU est trop faible, vous pouvez voir une perte/un retard de paquets inattendu pour les applications qui doivent utiliser une MTU plus élevée.

---

 Remarque : en cas de problème lié à MACsec, assurez-vous que le convertisseur d'interface gigaoctet (GBIC) aux deux extrémités est pris en charge par la matrice de compatibilité .

---

## Où MACsec est utilisé

Exemples d'utilisation sur campus

- Hôte vers commutateur
- Entre sites ou bâtiments
- Entre les étages dans une multilocation

Exemples d'utilisation du data center

- Interconnexion de data center
- Serveur vers commutateur

Exemples d'utilisation WAN

- Interconnexion de data center
- Interconnexion de campus
- Rayon Central

## Terminologie

| | | |
|---|---|---|
| MKA | Accord de clé MACsec | défini dans la norme IEEE 802.1X REV-2010 comme protocole d'accord de clé pour la détection des homologues MACsec et la négociation des clés |
| GÂTEAU | Clé d'association de connectivité | clé primaire à vie longue utilisée pour générer toutes les autres clés utilisées pour MACsec. Les mises en oeuvre LAN dérivent ceci de MSK (généré pendant l'échange EAP) |
| PMK | Clé primaire par paire | Un des composants utilisés pour dériver les clés de session |

| | | |
|---|---|---|
| | | utilisées pour chiffrer le trafic. Configuré manuellement ou dérivé de 802.1X |
| CKN | nom de clé CAK | utilisé pour configurer la valeur de clé ou CAK. Seul le nombre pair de caractères hexadécimaux pouvant atteindre 64 caractères est autorisé. |
| SAK | Clé d'association sécurisée | est dérivée par le serveur de clés sélectionné à partir de la clé CAK et est la clé utilisée par le routeur/les périphériques finaux pour chiffrer le trafic pour une session donnée. |
| INSTRUMENT DE CONTRÔLE DE LA VITESSE | Clé de valeur de contrôle d'intégrité | est dérivée de CAK et est étiquetée dans chaque trame de données/contrôle pour prouver que la trame provient d'un homologue autorisé. 8 à 16 octets selon la suite de chiffrement |
| QUEUE | Clé de cryptage | dérivée de CAK (la clé prépartagée) et utilisée pour protéger les clés MACsec |
| SIC | Identificateur de canal sécurisé | Chaque port virtuel reçoit un identificateur de canal sécurisé (SCI) unique basé sur l'adresse MAC de l'interface physique concaténée avec un ID de port de 16 bits |

## Scénario 1 : commutateur MACsec pour commuter la sécurité de liaison avec SAP en mode clé prépartagée (PSK)

### Topologie


Étape 1. Validez la configuration des deux côtés de la liaison.

<#root>

```
9300_stack#
```

**show run interface gig 1/0/1**

```
interface GigabitEthernet1/0/1
description MACsec_manual_3850-2-gi1/0/1
switchport access vlan 10
```

switchport mode trunk

**cts manual**

**no propagate sgt**

**sap pmk**

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

**mode-list gcm-encrypt <-- use full packet encrypt mode**

3850#

**show run interface gig1/0/1**

interface GigabitEthernet1/0/1
description 9300-1gi1/0/1 MACsec manual
switchport access vlan 10
switchport mode trunk

**cts manual**

**no propagate sgt**

**sap pmk**

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

**mode-list gcm-encrypt**

**NOTE:**

cts manual

**<-- Supplies local configuration for Cisco TrustSec parameters**

no propagate sgt

**<-- disable SGT tagging on a manually-configured TrustSec-capable interface,**

**if you do not need to propage the SGT tags.**

sap pmk AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA mode-list gcm-encrypt

**<--**

**Use the sap command to manually specify the Pairwise Primary Key (PMK) and the Security Association Prot**

**authentication and encryption modes to negotiate MACsec link encryption between two interfaces.**

**The default encryption is sap modelist gcm-encrypt null**

9300_stack#(config-if-cts-manual)#

**sap pmk fa mode-list**

?
gcm-encrypt GCM authentication, GCM encryption
gmac GCM authentication, no encryption
no-encap No encapsulation
null Encapsulation present, no authentication, no encryption

**Use "gcm-encrypt" for full GCM-AES-128 encryption.**

**These protection levels are supported when you configure SAP pairwise primary 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.

## Étape 2. Vérifiez l'état MACsec et que les paramètres/compteurs sont corrects.

<#root>

**### Ping issued between endpoints to demonstrate counters ###**

Host-1#

**ping 10.10.10.12 <-- sourced from Host-1 IP 10.10.10.11**

!!!!!!!!!!!!!!!!!!!!

9300_stack#

**sh MACsec summary**

Interface

**Transmit SC Receive SC <-- Secure Channel (SC) flag is set for transmit and receive**

**GigabitEthernet1/0/1**

**1 1**

9300_stack#

**sh MACsec interface gigabitEthernet 1/0/1**

**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**

!

**Transmit Secure Channels**

SCI : 682C7B9A4D010000
SC state : notInUse(2)

**Elapsed time : 03:17:50**

Start time : 7w0d
Current AN: 0
Previous AN: 1
Next PN: 185
SA State: notInUse(2)
Confidentiality : yes
SAK Unchanged : no

**SA Create time : 03:58:39**

**SA Start time : 7w0d**

SC Statistics
Auth-only Pkts : 0
Auth-only Bytes : 0

**Encrypt Pkts : 2077**

Encrypt Bytes : 0
!

**SA Statistics**

Auth-only Pkts : 0

**Encrypt Pkts : 184**

**<-- packets are being encrypted and transmitted on this link**

!
Port Statistics
Egress untag pkts 0
Egress long pkts 0
!

**Receive Secure Channels**

SCI : D0C78970C3810000
SC state : notInUse(2)
Elapsed time : 03:17:50
Start time : 7w0d
Current AN: 0
Previous AN: 1

```
   Next PN: 2503
   RX SA Count: 0
   SA State: notInUse(2)
   SAK Unchanged : no

SA Create time : 03:58:39


   SA Start time : 7w0d


   SC Statistics
   Notvalid pkts 0
   Invalid pkts 0
   Valid pkts 28312
   Valid bytes 0
   Late pkts 0
   Uncheck pkts 0
   Delay pkts 0
   UnusedSA pkts 0
   NousingSA pkts 0
   Decrypt bytes 0
!
   SA Statistics


      Notvalid pkts 0
      Invalid pkts 0

Valid pkts 2502



<-- number of valid packets received on this link


      UnusedSA pkts 0
      NousingSA pkts 0
!
Port Statistics
Ingress untag pkts 0
Ingress notag pkts 36
Ingress badtag pkts 0
Ingress unknownSCI pkts 0
Ingress noSCI pkts 0
Ingress overrun pkts 0
!

9300_stack#
sh cts interface summary


Global Dot1x feature is Disabled
CTS Layer2 Interfaces
---------------------
Interface  Mode   IFC-state  dot1x-role   peer-id  IFC-cache  Critical-Authentication
-------------------------------------------------------------------------------
Gi1/0/1
MANUAL    OPEN
```

```
      unknown    unknown    invalid     Invalid

CTS Layer3 Interfaces
---------------------
Interface IPv4 encap IPv6 encap IPv4 policy IPv6 policy
-------------------------------------------------------------------------------
!

9300_stack#
```

**sh cts interface gigabitEthernet 1/0/1**

```
Global Dot1x feature is Disabled
Interface GigabitEthernet1/0/1:
```

**CTS is enabled, mode: MANUAL**

```
IFC state: OPEN
```

**Interface Active for 04:10:15.723 <--- Uptime of MACsec port**

```
Authentication Status: NOT APPLICABLE
Peer identity: "unknown"
Peer's advertised capabilities: "sap"
Authorization Status: NOT APPLICABLE
!
```

**SAP Status: SUCCEEDED <-- SAP is successful**

```
   Version: 2
   Configured pairwise ciphers:
   gcm-encrypt
!
```

**Replay protection: enabled**

**Replay protection mode: STRICT**

```
!
Selected cipher: gcm-encrypt
!
Propagate SGT: Disabled
Cache Info:
Expiration : N/A
Cache applied to link : NONE
!
Statistics:
   authc success: 0
   authc reject: 0
   authc failure: 0
   authc no response: 0
   authc logoff: 0
```

**sap success: 1  <-- Negotiated once**

**sap fail: 0     <-- No failures**

**authz success: 0**

**authz fail: 0**

**port auth fail: 0**

L3 IPM: disabled

Étape 3. Vérifiez les débogages logiciels lorsque le lien est activé.

<#root>

**### Verify CTS and SAP events ###**

debug cts sap events
debug cts sap packets

**### Troubleshoot MKA session bring up issues ###**

debug mka event
debug mka errors
debug mka packets

**### Troubleshoot MKA keep-alive issues ###**

debug mka linksec-interface
debug mka MACsec
debug MACsec

*May 8 00:48:04.843: %LINK-3-UPDOWN: Interface GigabitEthernet1/0/1, changed state to down
*May 8 00:48:05.324:  interface GigabitEthernet1/0/1 is UP

***May 8 00:48:05.324: CTS SAP ev (Gi1/0/1): Session started (new).**

***May 8 00:48:05.324: cts_sap_session_start CTS SAP ev (Gi1/0/1) peer:0000.0000.0000**
**AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA**

CTS SAP ev (Gi1/0/1): Old state: [waiting to restart],
event: [restart timer expired], action:

**[send message #0] succeeded.**

New state: [waiting to receive message #1].

***May 8 00:48:05.449: CTS SAP ev (Gi1/0/1): EAPOL-Key message from D0C7.8970.C381  <-- MAC of peer switc**

***May 8 00:48:05.449: CTS SAP ev (Gi1/0/1): EAPOL-Key message #0 parsed and validated.**

***May 8 00:48:05.449: CTS SAP ev (Gi1/0/1): Our MAC = 682C.7B9A.4D01  <-- MAC of local inte**

peer's MAC = D0C7.8970.C381.
CTS SAP ev (Gi1/0/1): Old state: [waiting to receive message #1],

**event: [received message #0], action: [break tie] succeeded.**

**New state: [determining role].**

***May 8 00:48:05.449: cts_sap_generate_pmkid_and_sci CTS SAP ev (Gi1/0/1) auth:682c.7b9a.4d01 supp:d0c7.8**
**AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA**

**CTS SAP ev (Gi1/0/1): Old state: [determining role],**

**event: [change to authenticator], action: [send message #1] succeeded.**

**New state: [waiting to receive message #2].**

***May 8 00:48:05.457: CTS SAP ev (Gi1/0/1): EAPOL-Key message from D0C7.8970.C381.**

CTS SAP ev (Gi1/0/1): New keys derived:
KCK = 700BEF1D 7A8E10F7 1243A168 883C74FB,
KEK = C207177C B6091790 F3C5B4B1 D51B75B8,
TK = 1B0E17CD 420D12AE 7DE06941 B679ED22,

***May 8 00:48:05.457: CTS SAP ev (Gi1/0/1): EAPOL-Key message #2 parsed and validated.**

***May 8 00:48:05.457: CTS-SAP ev: cts_sap_action_program_msg_2: (Gi1/0/1) GCM is allowed.**

*May 8 00:48:05.457: MACsec-IPC: sending clear_frames_option
*May 8 00:48:05.457: MACsec-IPC: geting switch number
*May 8 00:48:05.457: MACsec-IPC: switch number is 1
*May 8 00:48:05.457: MACsec-IPC: clear_frame send msg success
*May 8 00:48:05.457: MACsec-IPC: getting MACsec clear frames response
*May 8 00:48:05.457: MACsec-IPC: watched boolean waken up
*May 8 00:48:05.457: MACsec-CTS: create_sa invoked for SA creation
*May 8 00:48:05.457: MACsec-CTS: Set up TxSC and RxSC before we installTxSA and RxSA
*May 8 00:48:05.457: MACsec-CTS: create_tx_sc, avail=yes sci=682C7B9A
*May 8 00:48:05.457: NGWC-MACsec: create_tx_sc vlan invalid
*May 8 00:48:05.457: NGWC-MACsec: create_tx_sc client vlan=1, sci=0x682C7B9A4D010000
*May 8 00:48:05.457: MACsec-IPC: sending create_tx_sc

```
*May 8 00:48:05.457: MACsec-IPC: geting switch number
*May 8 00:48:05.457: MACsec-IPC: switch number is 1
*May 8 00:48:05.457: MACsec-IPC: create_tx_sc send msg success
*May 8 00:48:05.458: MACsec API blocking the invoking context
*May 8 00:48:05.458: MACsec-IPC: getting MACsec sa_sc response
*May 8 00:48:05.458: MACsec_blocking_callback
*May 8 00:48:05.458: Wake up the blocking process
*May 8 00:48:05.458: MACsec-CTS: create_rx_sc, avail=yes sci=D0C78970
*May 8 00:48:05.458: NGWC-MACsec: create_rx_sc client vlan=1, sci=0xD0C78970C3810000
*May 8 00:48:05.458: MACsec-IPC: sending create_rx_sc
*May 8 00:48:05.458: MACsec-IPC: geting switch number
*May 8 00:48:05.458: MACsec-IPC: switch number is 1
*May 8 00:48:05.458: MACsec-IPC: create_rx_sc send msg success
*May 8 00:48:05.458: MACsec API blocking the invoking context
*May 8 00:48:05.458: MACsec-IPC: getting MACsec sa_sc response
*May 8 00:48:05.458: MACsec_blocking_callback
*May 8 00:48:05.458: Wake up the blocking process
*May 8 00:48:05.458: MACsec-CTS: create_tx_rx_sa, txsci=682C7B9A, an=0
*May 8 00:48:05.458: MACsec-IPC: sending install_tx_sa
*May 8 00:48:05.458: MACsec-IPC: geting switch number
*May 8 00:48:05.458: MACsec-IPC: switch number is 1
*May 8 00:48:05.459: MACsec-IPC: install_tx_sa send msg success
*May 8 00:48:05.459: NGWC-MACsec:Sending authorized event to port SM
*May 8 00:48:05.459: MACsec API blocking the invoking context
*May 8 00:48:05.459: MACsec-IPC: getting MACsec sa_sc response
*May 8 00:48:05.459: MACsec_blocking_callback
*May 8 00:48:05.459: Wake up the blocking process
*May 8 00:48:05.459: MACsec-CTS: create_tx_rx_sa, rxsci=D0C78970, an=0
*May 8 00:48:05.459: MACsec-IPC: sending install_rx_sa
*May 8 00:48:05.459: MACsec-IPC: getting switch number
*May 8 00:48:05.459: MACsec-IPC: switch number is 1
*May 8 00:48:05.460: MACsec-IPC: install_rx_sa send msg success
*May 8 00:48:05.460: MACsec API blocking the invoking context
*May 8 00:48:05.460: MACsec-IPC: getting MACsec sa_sc response
*May 8 00:48:05.460: MACcsec_blocking_callback
*May 8 00:48:05.460: Wake up the blocking process
CTS SAP ev (Gi1/0/1): Old state: [waiting to receive message #2],
event: [received message #2], action: [program message #2] succeeded.
New state: [waiting to program message #2].
CTS SAP ev (Gi1/0/1): Old state: [waiting to program message #2],
event: [data path programmed], action: [send message #3] succeeded.
```

**New state: [waiting to receive message #4].**

***May 8 00:48:05.467: CTS SAP ev (Gi1/0/1): EAPOL-Key message from D0C7.8970.C381.**

***May 8 00:48:05.467: CTS SAP ev (Gi1/0/1): EAPOL-Key message #4 parsed and validated.**

***May 8 00:48:05.473: CTS-SAP ev: cts_sap_sync_sap_info: incr sync msg sent for Gi1/0/1**

***May 8 00:48:07.324: %LINK-3-UPDOWN: Interface GigabitEthernet1/0/1, changed state to up**

Étape 4. Vérifiez les traces au niveau de la plate-forme lorsque la liaison est établie.

<#root>

9300_stack#

**sh platform software fed switch 1 ifm mappings**

**Interface IF_ID Inst Asic Core Port SubPort Mac Cntx LPN GPN Type Active**

**GigabitEthernet1/0/1 0x8 1 0 1 0 0 26 6 1 1 NIF Y**

**Note the IF_ID for respective intf**

- This respective IF_ID shows in MACsec FED traces seen here.

9300_stack#

**set platform software trace fed switch 1 cts_aci verbose**

9300_stack#

**set platform software trace fed switch 1 MACsec verbose**

**<-- switch number with MACsec port**

9300_stack#

**request platform software trace rotate all**

**/// shut/no shut the MACsec interface ///**

9300_stack#

**show platform software trace message fed switch 1**

**2019/05/08 01:08:50.688 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): FED sent MACsec_**

**2019/05/08 01:08:50.688 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): FED sending MACs**

2019/05/08 01:08:50.688 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Running Instal
2019/05/08 01:08:50.688 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Processing job
2019/05/08 01:08:50.688 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Install RxSA c

2019/05/08 01:08:50.688 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Processing SPI
2019/05/08 01:08:50.688 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): MACSec install
2019/05/08 01:08:50.688 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): Entering ins_rx

**2019/05/08 01:08:50.688 {fed_F0-0}{1}: [l2tunnel_bcast] [16837]: UUID: 0, ra: 0, TID: 0 (ERR): port_idMA**

**2019/05/08 01:08:50.687 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): FED sent macsec_**

**2019/05/08 01:08:50.687 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): FED sending macs**

**2019/05/08 01:08:50.687 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): if_id = 8, cts_**

**2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Calling Install**

**2019/05/08 01:08:50.686 {fed_F0-0}{1}: [sec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): sci=0x682c7b9a4d01**

**2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Processing job**

**2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Create time of**

**2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): sci=0x682c7b9a4**

**2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Install TxSA ca**

**2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Processing SPI**

**2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): MACSec install T**

**2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): Entering ins_tx_**

**2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): FED sent macsec_**

**2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): FED sending macs**

2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Conf_Offset in

**2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Successfully in**

**2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Secy policy har**

**2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Install policy**

**2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Attach policy**

**2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Creating drop e**

**2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): if_id = 8, cts_**

**2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): sci=0x682c7b9a4**

2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Create RxSC ca
2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Processing SPI
2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): MACSec create R
2019/05/08 01:08:50.686 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): Entering cre_rx
2019/05/08 01:08:50.685 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): FED sent macsec
2019/05/08 01:08:50.685 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): FED sending mac
2019/05/08 01:08:50.685 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): txSC setting x

**2019/05/08 01:08:50.685 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Conf_Offset in**

**2019/05/08 01:08:50.685 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): if_id = 8, cts_**

**2019/05/08 01:08:50.685 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): secy created su**

**2019/05/08 01:08:50.685 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): if_id = 8, cts_**

**2019/05/08 01:08:50.685 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): if_id = 8, cts_**

**2019/05/08 01:08:50.685 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): is_remote is 0**

**2019/05/08 01:08:50.685 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Create TxSC cal**

2019/05/08 01:08:50.685 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Processing SPI
2019/05/08 01:08:50.685 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): MACSec create T
2019/05/08 01:08:50.685 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): Entering cre_tx
2019/05/08 01:08:50.685 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): FED sent clear_
2019/05/08 01:08:50.685 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): FED sending mac
2019/05/08 01:08:50.685 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Processing job
2019/05/08 01:08:50.685 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (debug): Processing SPI
2019/05/08 01:08:50.685 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): MACSec clear_fr
2019/05/08 01:08:50.685 {fed_F0-0}{1}: [MACsec] [16837]: UUID: 0, ra: 0, TID: 0 (info): Entering clear_
2019/05/08 01:08:50.527 {fed_F0-0}{1}: [pm_xcvr] [17885]: UUID: 0, ra: 0, TID: 0 (note): XCVR POST:XCVR
speed_auto Oper Speed:speed_gbps1 Autoneg Mode:Unknown autonegmode type
2019/05/08 01:08:50.525 {fed_F0-0}{1}: [xcvr] [17885]: UUID: 0, ra: 0, TID: 0 (note): ntfy_lnk_status:

**2019/05/08 01:08:48.142 {fed_F0-0}{1}: [pm_xcvr] [16837]: UUID: 0, ra: 0, TID: 0 (note): Enable XCVR for**

**2019/05/08 01:08:48.142 {fed_F0-0}{1}: [pm_tdl] [16837]: UUID: 0, ra: 0, TID: 0 (note): Received PM port**

Étape 5. Vérifiez l'état de l'interface MACsec dans le matériel.

<#root>

9300_stack#

**sh platform pm interface-numbers**

interface iif-id gid slot unit slun HWIDB-Ptr status status2 state snmp-if-index

---

**Gi1/0/1 8 1 1 1 1 0x7F2C90D7C600 0x10040 0x20001B 0x4 8**

9300_stack#

**sh pl software fed switch 1 ifm if-id 8 <-- iif-id 8 maps to gig1/0/1**

Interface IF_ID : 0x0000000000000008

**Interface Name : GigabitEthernet1/0/1**

Interface Block Pointer : 0x7f4a6c66b1b8
Interface Block State : READY

**Interface State : Enabled**

Interface Status : ADD, UPD
Interface Ref-Cnt : 8
Interface Type : ETHER
Port Type : SWITCH PORT

**Port Location : LOCAL**

Slot : 1
Unit : 0
Slot Unit : 1
SNMP IF Index : 8
GPN : 1
EC Channel : 0
EC Index : 0

**Port Handle : 0x4e00004c**

LISP v4 Mobility : false
LISP v6 Mobility : false

```
QoS Trust Type : 3
!
Port Information
Handle ............ [0x4e00004c]
Type .............. [Layer2]
Identifier ........ [0x8]
Slot .............. [1]
Unit .............. [1]

Port Physical Subblock
Affinity .......... [local]
Asic Instance ..... [1 (A:0,C:1)]
AsicPort .......... [0]
AsicSubPort ....... [0]
MacNum ............ [26]
ContextId ......... [6]
LPN ............... [1]
GPN ............... [1]
Speed ............. [1GB]
type .............. [NIF]
```

**PORT_LE ........... [0x7f4a6c676bc8]**

**<--- port_LE**

```
L3IF_LE ........... [0x0]
DI ................ [0x7f4a6c67d718]
SubIf count ....... [0]

Port L2 Subblock
Enabled ............. [Yes]
Allow dot1q ......... [Yes]
Allow native ........ [Yes]
Default VLAN ........ [1]
Allow priority tag ... [Yes]
Allow unknown unicast [Yes]
Allow unknown multicast[Yes]
Allow unknown broadcast[Yes]
Allow unknown multicast[Enabled]
Allow unknown unicast [Enabled]
Protected ............ [No]
IPv4 ARP snoop ....... [No]
IPv6 ARP snoop ....... [No]
Jumbo MTU ............ [1500]
Learning Mode ........ [1]
Vepa ................. [Disabled]

Port QoS Subblock
Trust Type ..................... [0x2]
Default Value .................. [0]
Ingress Table Map .............. [0x0]
Egress Table Map ............... [0x0]
Queue Map ...................... [0x0]
Port Netflow Subblock
Port Policy Subblock
List of Ingress Policies attached to an interface
List of Egress Policies attached to an interface
```

**Port CTS Subblock**

```
Disable SGACL ..................... [0x0]
Trust ............................. [0x0]
Propagate ......................... [0x0]
%Port SGT ......................... [-1717360783]
```

**Physical Port Macsec Subblock <-- This block is not present when MACsec is not enabled**

**MACsec Enable .... [Yes]**

**MACsec port handle.... [0x4e00004c] <-- Same as PORT_LE**

**MACsec Virtual port handles....**

**.....[0x11000005]**

```
MACsec Rx start index.... [0]
MACsec Rx end index.... [6]
MACsec Tx start index.... [0]
MACsec Tx end index.... [6]

Ref Count : 8 (feature Ref Counts + 1)
IFM Feature Ref Counts
FID : 102 (AAL_FEATURE_SRTP), Ref Count : 1
FID : 59 (AAL_FEATURE_NETFLOW_ACL), Ref Count : 1
FID : 95 (AAL_FEATURE_L2_MULTICAST_IGMP), Ref Count : 1
FID : 119 (AAL_FEATURE_PV_HASH), Ref Count : 1
FID : 17 (AAL_FEATURE_PBB), Ref Count : 1
FID : 83 (AAL_FEATURE_L2_MATM), Ref Count : 1
FID : 30 (AAL_FEATURE_URPF_ACL), Ref Count : 1
IFM Feature Sub block information
FID : 102 (AAL_FEATURE_SRTP), Private Data : 0x7f4a6c9a0838
FID : 59 (AAL_FEATURE_NETFLOW_ACL), Private Data : 0x7f4a6c9a00f8
FID : 17 (AAL_FEATURE_PBB), Private Data : 0x7f4a6c9986b8
FID : 30 (AAL_FEATURE_URPF_ACL), Private Data : 0x7f4a6c9981c8


9300_stack#
```

**sh pl hard fed switch 1 fwd-asic abstraction print-resource-handle 0x7f4a6c676bc8 1 <-- port_LE handle**

```
Handle:0x7f4a6c676bc8 Res-Type:ASIC_RSC_PORT_LE Res-Switch-Num:0 Asic-Num:1 Feature-ID:AL_FID_IFM Lkp-f
priv_ri/priv_si Handle: (nil)Hardware Indices/Handles: index1:0x0 mtu_index/l3u_ri_index1:0x2 sm handle
Detailed Resource Information (ASIC# 1)

**snip**
LEAD_PORT_ALLOW_CTS value 0 Pass
LEAD_PORT_ALLOW_NON_CTS value 0 Pass
```

**LEAD_PORT_CTS_ENABLED value 1 Pass <-- Flag = 1 (CTS enabled)**

**LEAD_PORT_MACsec_ENCRYPTED value 1 Pass <-- Flag = 1 (MACsec encrypt enabled)**

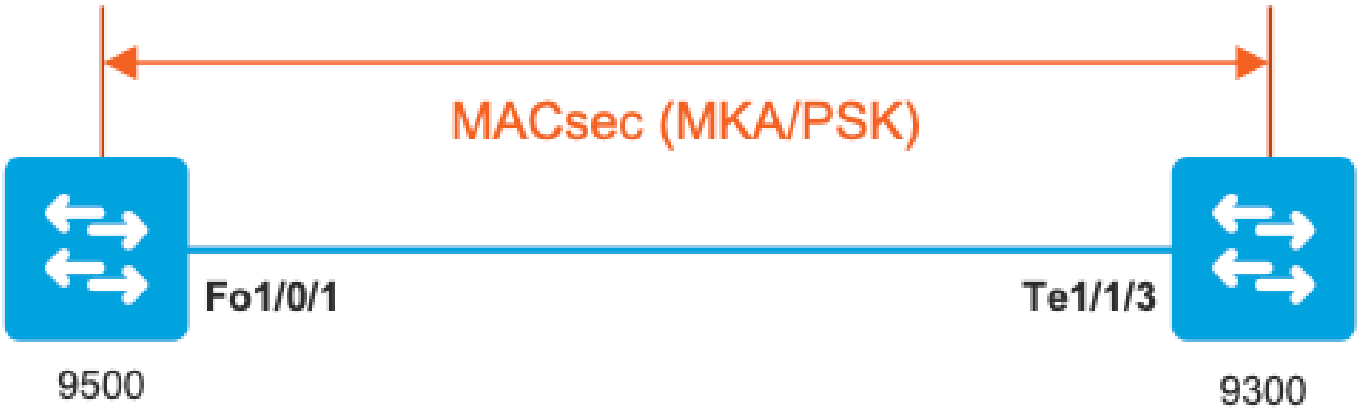
```
LEAD_PORT_PHY_MAC_SEC_SUB_PORT_ENABLED value 0 Pass
LEAD_PORT_SGT_ALLOWED value 0 Pass
```

**LEAD_PORT_EGRESS_MAC_sec_ENABLE_WITH_SCI value 1 Pass  <-- Flag = 1 (MACsec with SCI enabled)**

```
LEAD_PORT_EGRESS_MAC_sec_ENABLE_WITHOUT_SCI value 0 Pass
LEAD_PORT_EGRESS_MAC_sec_SUB_PORT value 0 Pass
LEAD_PORT_EGRESS_MACsec_ENCRYPTED value 0 Pass
**snip**
```

# Scénario 2 : sécurité de liaison commutateur à commutateur MACsec avec MKA en mode clé prépartagée (PSK)

## Topologie




Étape 1. Validez la configuration des deux côtés de la liaison.

<#root>

C9500#

**sh run | sec key chain**

```
key chain KEY MACsec
 key 01
   cryptographic-algorithm aes-256-cmac
  key-string 7 101C0B1A0343475954532E2E767B3233214105150555030A0004500B514B175F5B05515153005E0E5E505C52
```

**lifetime local 00:00:00 Aug 21 2019 infinite <-- use NTP to sync the time for key chains**

**mka policy MKA**

```
key-server priority 200
```

```
MACsec-cipher-suite gcm-aes-256
confidentiality-offset 0

C9500#
sh run interface fo1/0/1

interface fo1/0/1
MACsec network-link

mka policy MKA

mka pre-shared-key key-chain KEY

C9300#
sh run interface te1/1/3

interface te1/1/3
MACsec network-link

mka policy MKA

mka pre-shared-key key-chain KEY
```

## Étape 2 : validation de MACsec et de l'exactitude de tous les paramètres/compteurs

<#root>

```
### This example shows the output from one side, verify on both ends of MACsec tunnel ###

C9500#
sh MACsec summary

Interface                   Transmit SC      Receive SC
FortyGigabitEthernet1/0/1       1                1

C9500#
sh MACsec interface fortyGigabitEthernet 1/0/1
```

**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-256**

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**

**Transmit Secure Channels**

SCI : 0CD0F8DCDC010008
SC state : notInUse(2)

**Elapsed time : 00:24:38**

Start time : 7w0d
Current AN: 0
Previous AN: -
Next PN: 2514
SA State: notInUse(2)
Confidentiality : yes
SAK Unchanged : yes

**SA Create time : 1d01h**

**SA Start time : 7w0d**

**SC Statistics**

Auth-only Pkts : 0
Auth-only Bytes : 0

**Encrypt Pkts : 3156  <-- can increment with Tx traffic**

Encrypt Bytes : 0

**SA Statistics**

Auth-only Pkts : 0

**Encrypt Pkts : 402  <-- can increment with Tx traffic**

**Port Statistics**

Egress untag pkts 0
Egress long pkts 0

**Receive Secure Channels**

SCI : A0F8490EA91F0026
SC state : notInUse(2)

**Elapsed time : 00:24:38**

Start time : 7w0d
Current AN: 0
Previous AN: -
Next PN: 94
RX SA Count: 0
SA State: notInUse(2)
SAK Unchanged : yes
SA Create time : 1d01h
SA Start time : 7w0d

**SC Statistics**

Notvalid pkts 0
Invalid pkts 0
Valid pkts 0
Valid bytes 0
Late pkts 0
Uncheck pkts 0
Delay pkts 0
UnusedSA pkts 0

NousingSA pkts 0
Decrypt bytes 0

**SA Statistics**

Notvalid pkts 0
Invalid pkts 0

**Valid pkts 93**

UnusedSA pkts 0
NousingSA pkts 0
!

**Port Statistics**

Ingress untag pkts 0

**Ingress notag pkts 748**

Ingress badtag pkts 0
Ingress unknownSCI pkts 0
Ingress noSCI pkts 0
Ingress overrun pkts 0

C9500#

**sh mka sessions interface fortyGigabitEthernet 1/0/1**

Summary of All Currently Active MKA Sessions on Interface FortyGigabitEthernet1/0/1...
==========
Interface Local-TxSCI

**Policy-Name**

Inherited Key-Server
Port-ID Peer-RxSCI MACsec-Peers Status CKN
==========

**Fo1/0/1 0cd0.f8dc.dc01/0008**

**MKA**

NO YES

**8 a0f8.490e.a91f/0026 1 Secured01 <-- CKN number must match on both sides**

0cd0.f8dc.dc01

**<--**

**MAC of local interface**

a0f8.490e.a91f

**<--**

**MAC of remote neighbor**

8

**<-- indicates IIF_ID of respective local port (here IF_ID is 8 for local port fo1/0/1)**

C9500#

**sh platform pm interface-numbers | in iif|1/0/1**

interface

**iif-id**

  gid  slot  unit  slun    HWIDB-Ptr     status   status2  state  snmp-if-index

**Fo1/0/1**

**8**

     1    1     1      1   0x7EFF3F442778  0x10040  0x20001B  0x4        8

C9500#

**sh mka sessions interface fortyGigabitEthernet 1/0/1 detail**

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

**Local Tx-SCI..... 0cd0.f8dc.dc01/0008**

**Interface MAC Address.... 0cd0.f8dc.dc01**

**MKA Port Identifier..... 8**

**Interface Name..... FortyGigabitEthernet1/0/1**

Audit Session ID.....
CAK Name (CKN)..... 01
Member Identifier (MI)... DFDC62E026E0712F0F096392

**Message Number (MN)..... 536     <-- can increment as message numbers increment**

```
EAP Role.................. NA
Key Server................ YES
```

**MKA Cipher Suite......... AES-256-CMAC**

```
Latest SAK Status........ Rx & Tx
Latest SAK AN............ 0
Latest SAK KI (KN)....... DFDC62E026E0712F0F09639200000001 (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)
SAK Rekey Time........... 0s (SAK Rekey interval not applicable)

MKA Policy Name.......... MKA
Key Server Priority...... 200
Delay Protection......... NO
Delay Protection Timer.......... 0s (Not enabled)

Confidentiality Offset... 0
Algorithm Agility........ 80C201
SAK Rekey On Live Peer Loss........ NO
Send Secure Announcement.. DISABLED
SAK Cipher Suite......... 0080C20001000002 (GCM-AES-256)
MACsec Capability........ 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired........... YES
```

**# of MACsec Capable Live Peers............ 1 <-- Peers capable of MACsec**

**# of MACsec Capable Live Peers Responded.. 1 <-- Peers that responded to MACsec negotiation**

**Live Peers List:**

```
MI                        MN    Rx-SCI (Peer)        KS           RxSA
                                                   Priority      Installed
--------------------------------------------------------------------------------
```

**ACF0BD8ECCA391A197F4DF6B 537  a0f8.490e.a91f/0026    200            YES  <-- One live peer**

!

**Potential Peers List:**

```
MI                        MN    Rx-SCI (Peer)        KS           RxSA
                                                  Priority      Installed
--------------------------------------------------------------------------------
```

**Check the MKA policy and ensure that it is applied to expected interface**

C9500#

**sh mka policy MKA**

MKA Policy defaults :
Send-Secure-Announcements: DISABLED
!
MKA Policy Summary...
!
Codes : CO - Confidentiality Offset, ICVIND - Include ICV-Indicator,
SAKR OLPL - SAK-Rekey On-Live-Peer-Loss,
DP - Delay Protect, KS Prio - Key Server Priority

**Policy**

KS DP CO SAKR ICVIND Cipher Interfaces

**Name**

Prio OLPL Suite(s) Applied
==========

**MKA**

200 FALSE 0 FALSE TRUE

**GCM-AES-256**

**Fo1/0/1 <-- Applied to Fo1/0/1**

**### Ensure that PDU counters are incrementing at Tx/Rx at both sides.**

**This is useful to determine the direction of issues at transport. ###**

C9500#

**sh mka statistics | sec PDU**

MKPDU Statistics

**MKPDUs Validated & Rx..... 2342 <-- can increment**

"Distributed SAK"..... 0
"Distributed CAK"..... 0

**MKPDUs Transmitted..... 4552 <-- can increment**

**### MKA Error Counters ###**

```
C9500#
```

**show mka statistics**

```
** snip***
```

**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
```

Étape 3 à Étape 5

Suivez les instructions du scénario 1.

---

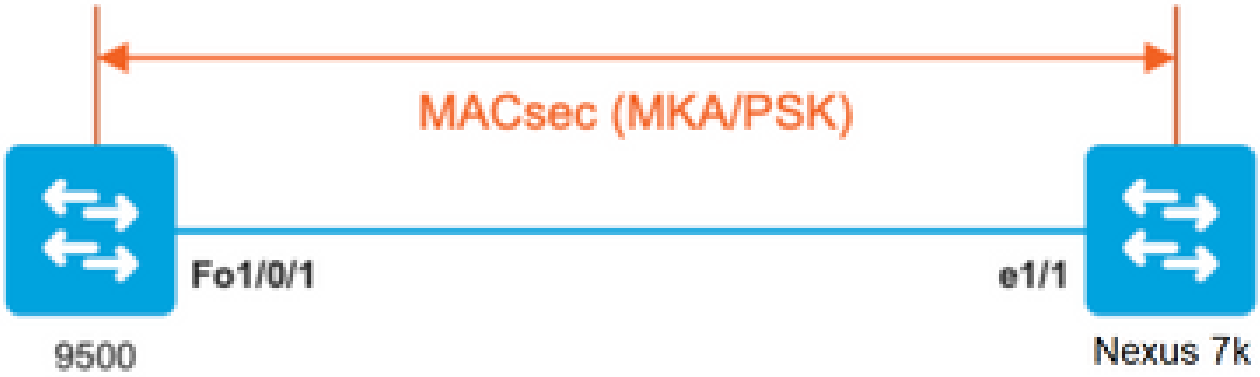
⚠ Avertissement : à des fins d'interopérabilité, sachez que certaines plates-formes font du remplissage et d'autres pas. Cela peut conduire à des problèmes clés où la session mka reste à l'état Init. Vous pouvez vérifier ceci avec la commande show mka sessions.

---

## Exemple de problème de remplissage

Cet exemple d'utilisation présente un Catalyst 9500 et un Nexus 7k dans NX-OS 8.2(2), mais peut également se produire avec des périphériques Catalyst tels que C3560CX.

(L'ID de bogue Cisco CSCvs92023 documente le problème).




- Si vous utilisez la configuration présentée dans le scénario 2, MKA ne peut pas établir le tunnel en raison d'une non-concordance de clé.
- Vous devez compléter manuellement la clé avec des 0 sur le côté 9500 puisque ce périphérique ne fait pas de remplissage.

Catalyst 9500

<#root>

```
conf t
  key chain MACsec1 MACsec
     key

0100000000000000000000000000000000000000000000000000000000000000 --> device does not do padding automati

       key-string 12345678901234567890123456789012
       end
```

Nexus 7K

<#root>

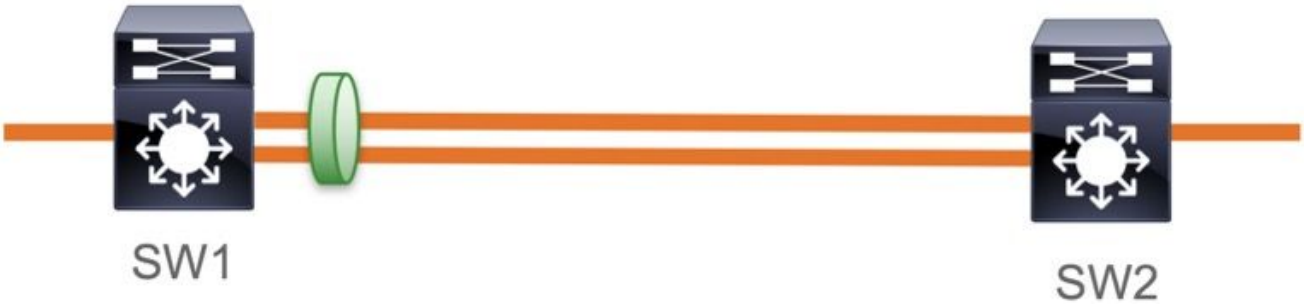
```
conf t
  key chain MACsec1 MACsec


key 01  --> Device does automatic padding.

      key-octet-string 12345678901234567890123456789012
      end
```

# Autres options de configuration

## Sécurité de liaison commutateur à commutateur MACsec avec MKA sur l'interface groupée/Port-Channel




- Canaux de port L3 et L2 (LACP, PAgP et mode ON)
- Types de cryptage (AES-128 et AES-256, AES-256 est applicable pour la licence Advantage)
- Échange de clés MKA PSK uniquement

Plates-formes prises en charge:

- Catalyst 9200 (AES-128 uniquement)
- Catalyst 9300
- Catalyst 9400
- Catalyst 9500 et Catalyst 9500H
- Catalyst 9600

Exemple de configuration de commutateur à commutateur Etherchannel

La configuration de la chaîne de clés et de la stratégie MKA reste identique à celle illustrée précédemment dans la section Configuration MKA.

<#root>

```
interface <>  <-- This is the physical member link. MACsec encrypts on the individual links


 MACsec network-link
```

```
    mka policy <policy-name>
    mka pre-shared-key key-chain <key-chain name>
    macsec replay-protection window-size frame number
```

**channel-group**

```
      mode active <-- Adding physical member to the port-channel
```

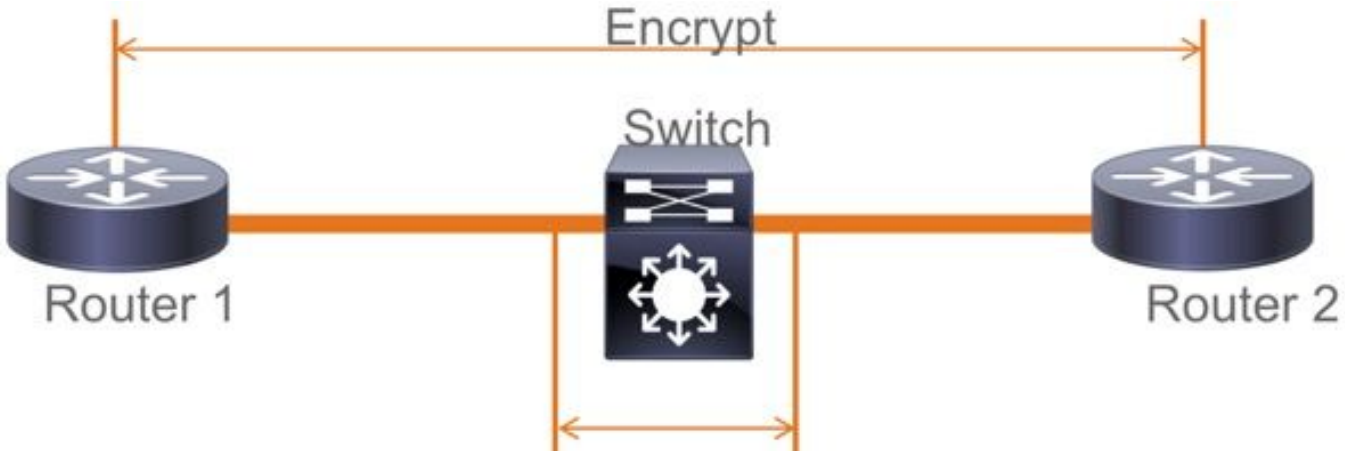
## Sécurité de liaison commutateur à commutateur MACsec sur les commutateurs intermédiaires L2, mode PSK

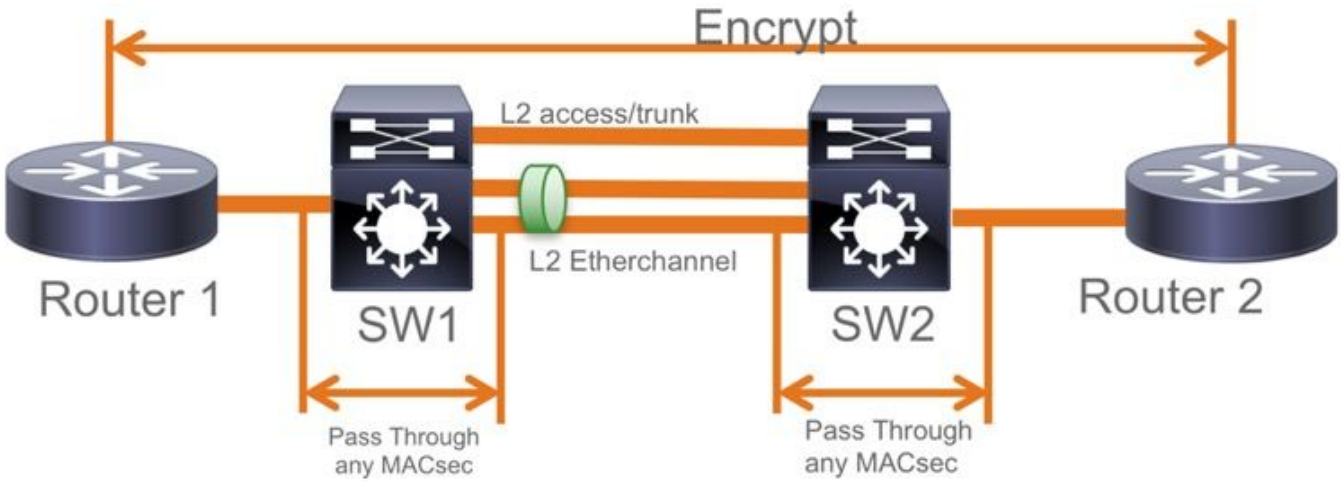
Cette section couvre certains des scénarios WAN MACsec pris en charge dans lesquels Cat9K doit transmettre des paquets chiffrés de manière transparente.

Dans certains cas, les routeurs ne sont pas directement connectés, mais ils disposent de commutateurs intermédiaires de couche 2, et les commutateurs de couche 2 peuvent contourner les paquets chiffrés sans aucun traitement du chiffrement.

Les commutateurs Catalyst 9000 transfèrent de manière transparente les paquets dont l'étiquette Clear débute dans la version 16.10(1)

- La fonctionnalité Pass-through est prise en charge pour MKA/SAP
- Prise en charge sur l'accès L2, trunk ou Etherchannel
- Pris en charge par défaut (pas de CLI de configuration à activer/désactiver)
- Assurez-vous que les routeurs envoient des trames EAPOL avec un éther-type autre que par défaut (0x888E)

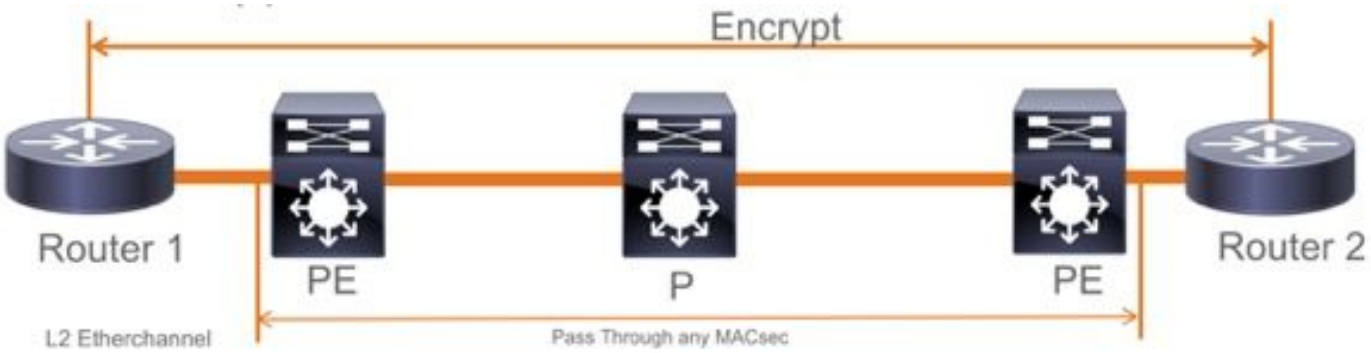





Topologie EoMPLS / VPLS

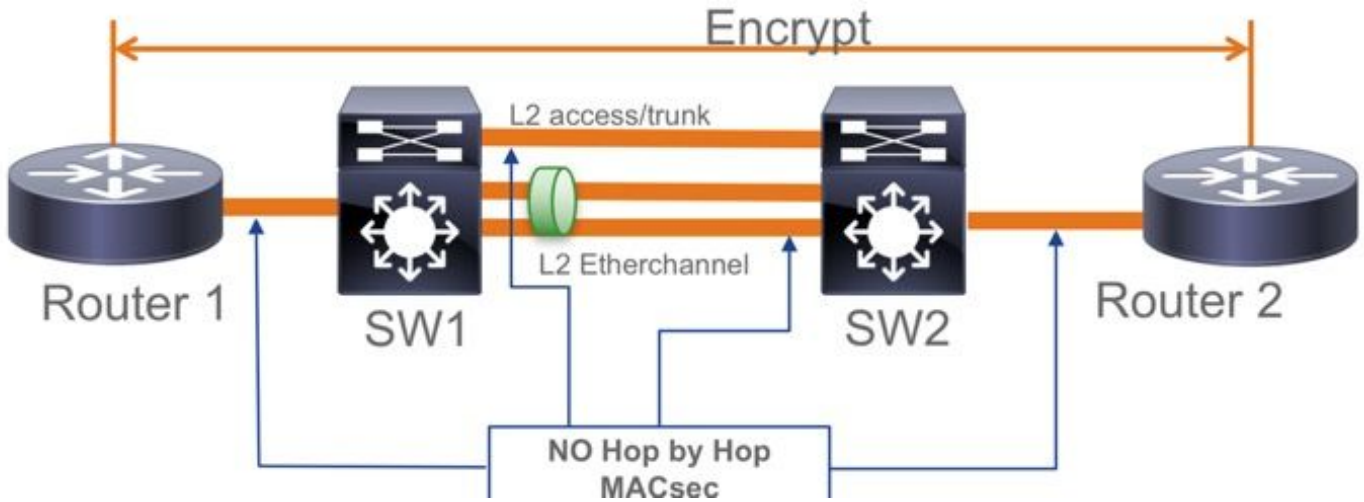
Plates-formes prises en charge Cat 9300/9400,9500/9500H en tant que périphériques PE ou P

- VPLS
- EoMPLS
- Pris en charge par défaut (pas de CLI de configuration à activer/désactiver)
- Début 16.10(1)

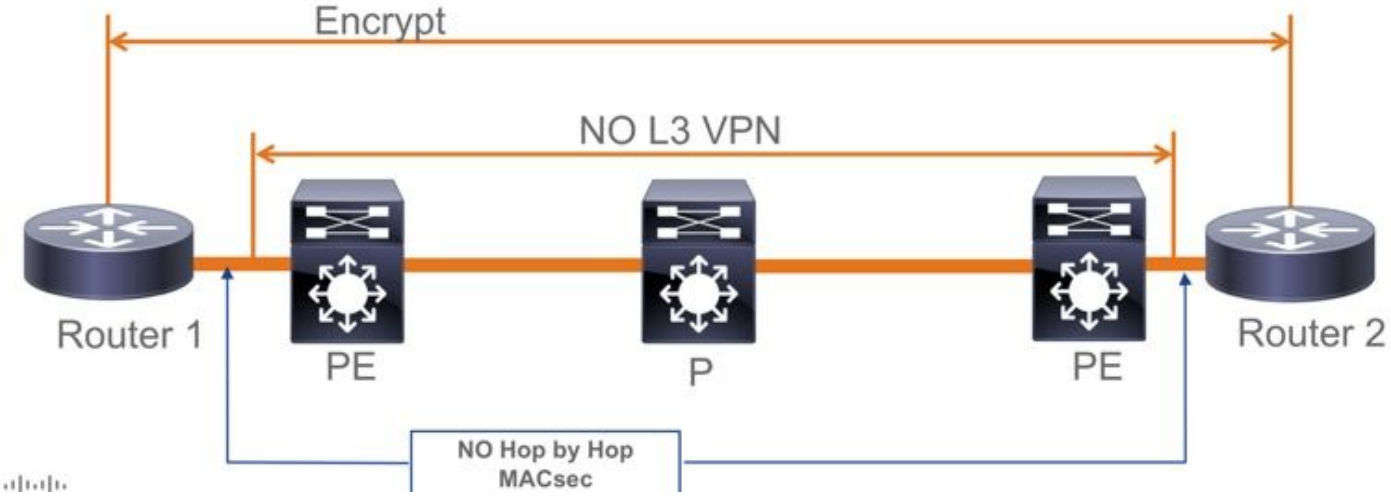



# Restrictions

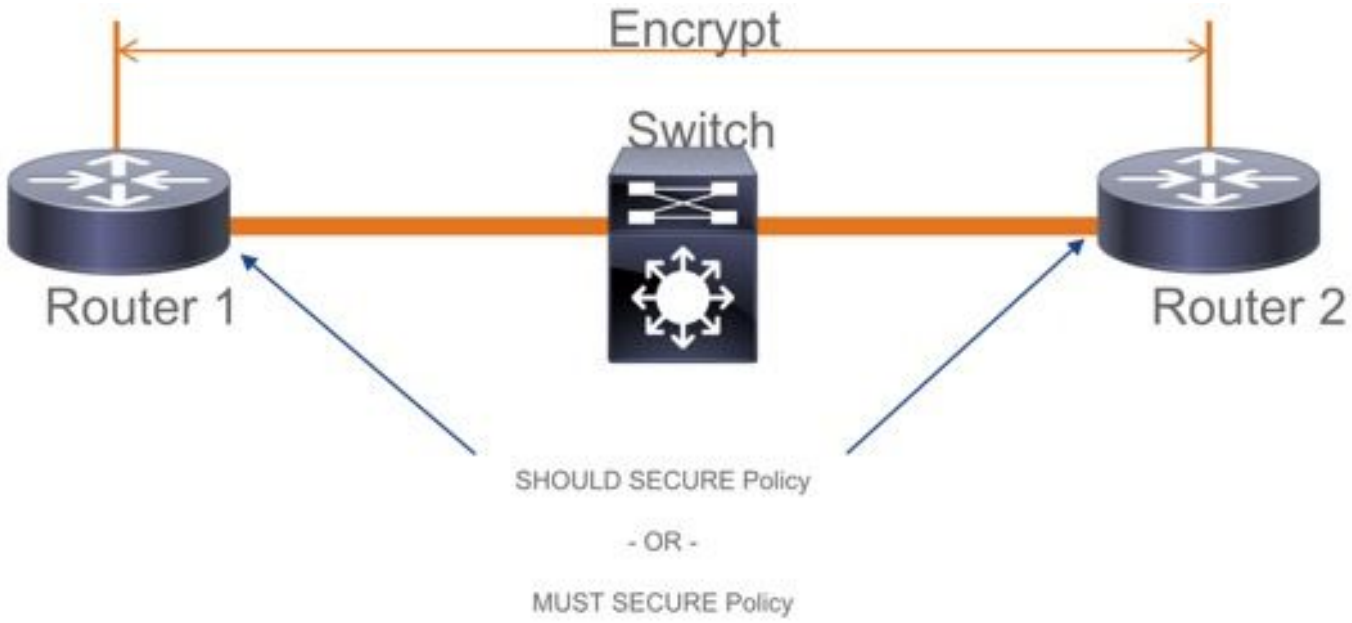
Le double chiffrement n'est pas pris en charge. Les adresses MACsec de bout en bout avec la balise Clear nécessitent que les commutateurs Hop by Hop ne soient pas activés sur les liaisons L2 directement connectées.




- ClearTag + EoMPLS avec commutateurs intermédiaires de couche 2 uniquement, MACsec ne peut pas être activé sur la liaison CE-PE
- ClearTag + L3VPN avec commutateurs intermédiaires non pris en charge



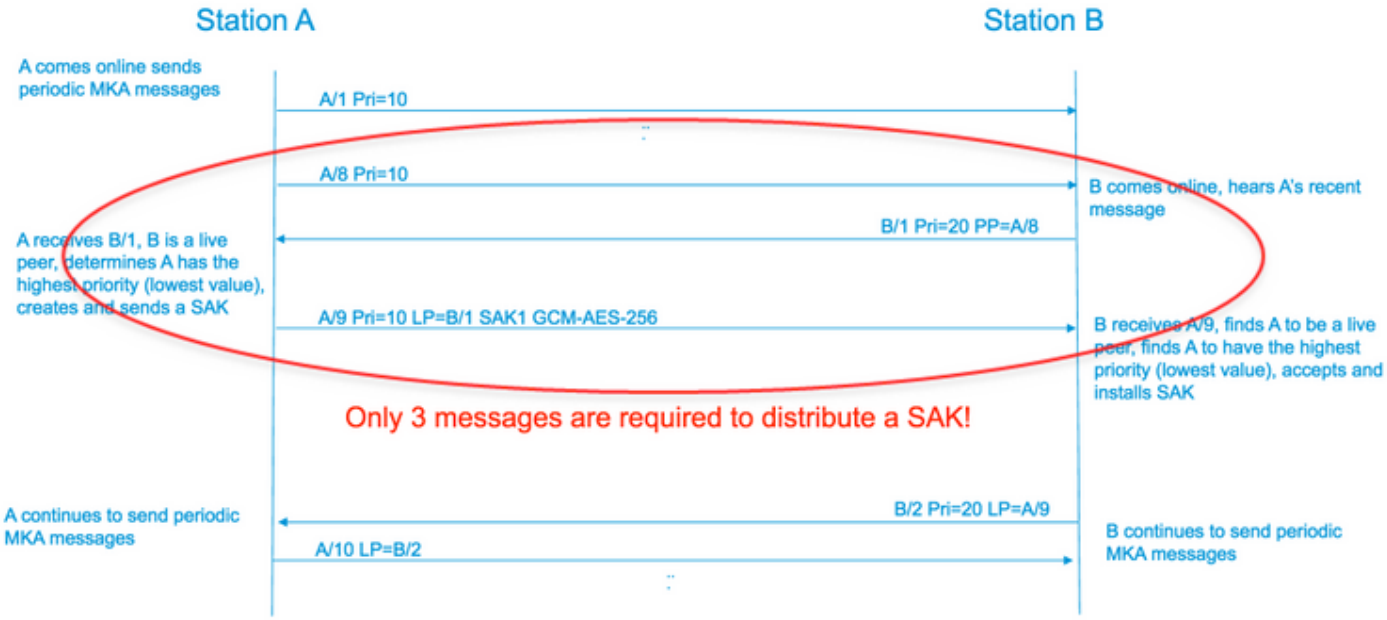

- La fonction Devrait sécuriser en mode PSK n'est pas prise en charge. Le mode par défaut est Must Secure.
- La stratégie Must Secure ne chiffre pas uniquement EAPoL pour négocier les paramètres MACsec.




# Informations opérationnelles MACsec

## Séquence de fonctionnement

1. Lorsque la liaison et les deux périphériques finaux sont activés, ils échangent des trames MKA (ethertype = 0x88E, identique à EAPOL avec le type de paquet MKA). Il s'agit d'un protocole de négociation multipoint à multipoint. La valeur de clé CAK (normalement statique pré-partagée), le nom de clé (CKN) doivent correspondre et ICV doit être valide pour que les homologues soient détectés et acceptés.
2. Le périphérique dont la priorité de serveur de clés est la plus faible (valeur par défaut = 0) est sélectionné comme serveur de clés. Le serveur de clés génère le SAK et le distribue via des messages MKA. En cas de temps, la valeur la plus élevée de l'identificateur de canal sécurisé (SCI) est gagnante.
3. Par la suite, toutes les trames sécurisées MACsec sont chiffrées avec la cryptographie symétrique (SAC). Des canaux sécurisés TX et RX distincts ont été créés. Mais la même clé SAK est utilisée pour le chiffrement et le déchiffrement.
4. Lorsqu'un nouveau périphérique est détecté dans un réseau local à accès multiple (via des messages EAPOL-MKA), le serveur de clés génère une nouvelle clé qui sera utilisée par tous les périphériques. La nouvelle clé entre en service après avoir été reconnue par tous les périphériques (reportez-vous à la section 9.17.2 de la norme IEEE 802.1X-2010).




## Paquets MACsec

Trame de contrôle (EAPOL-MKA)

- MAC de destination EAPOL = 01:80:C2:00:00:03 pour la multidiffusion des paquets vers plusieurs destinations
- Type d'éther EAPOL = 0x888E

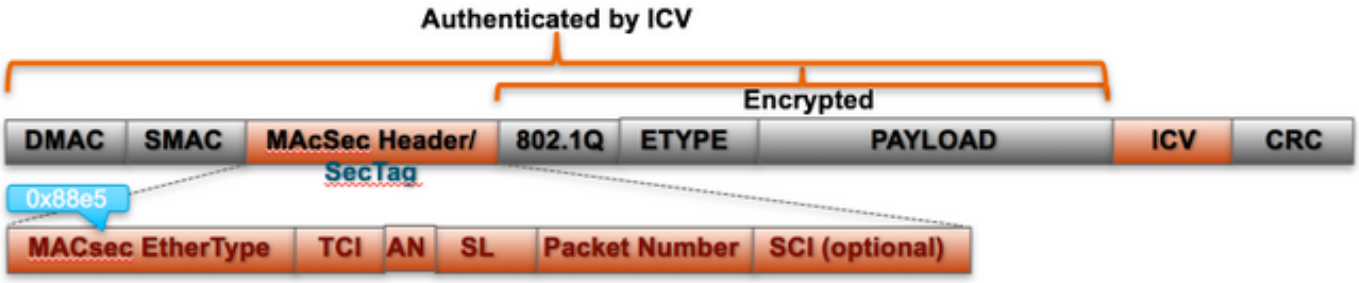
Charge utile L2 au format de trame de contrôle.

| | | |
|---|---|---|
| Protocol Version | | |
| Packet Type = EAPOL-MKA | | |
| Packet Body Length | | **Size** |
| Packet Body (MKPDU) | Basic Parameter Set | **Multiple of 4 octets** |
| | Parameter Set | **Multiple of 4 octets** |
| | Parameter Set | **Multiple of 4 octets** |
| | ICV | **16 octets** |

Trame de données

MACsec insère deux balises supplémentaires sur les trames de données avec une surcharge maximale de 32 octets (16 octets minimum).

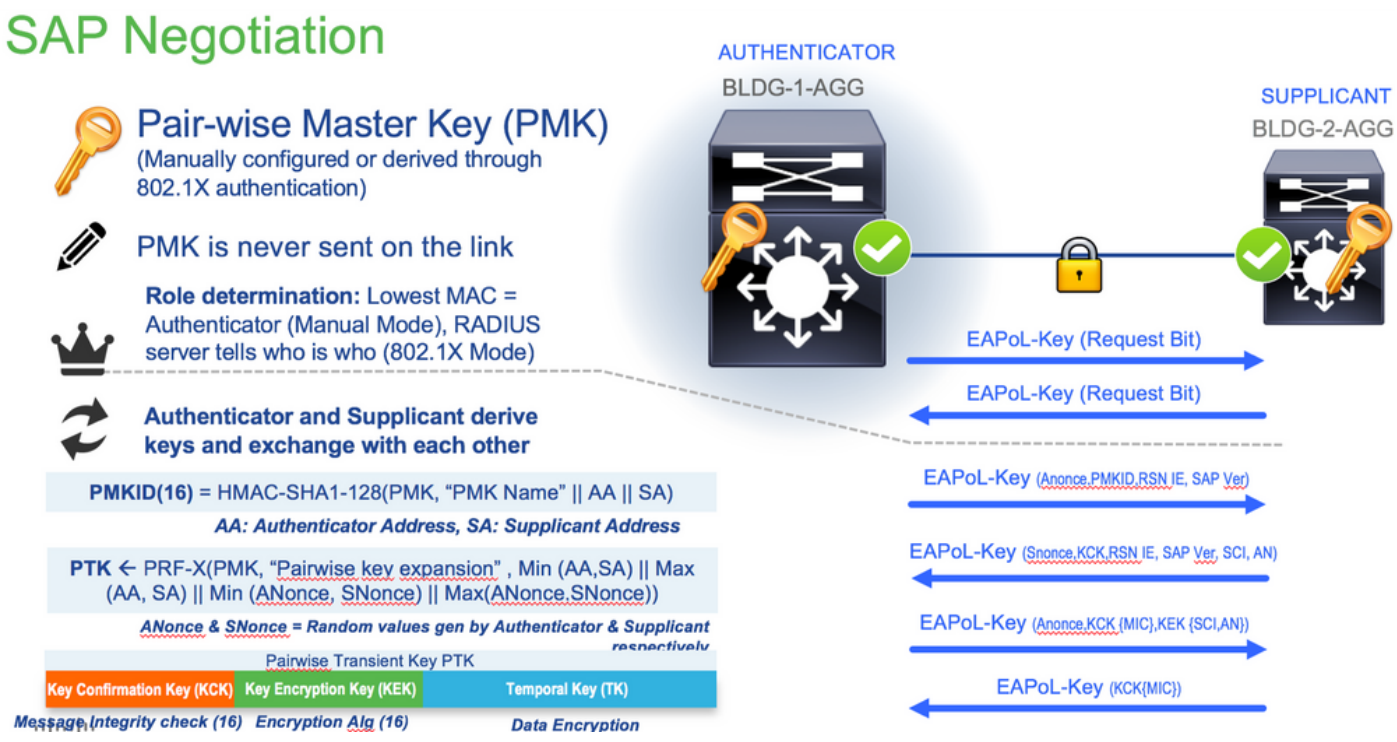
- SecTag = 8 à 16 octets (SCI de 8 octets est facultatif)
- ICV = 8 à 16 octets en fonction de la combinaison de chiffrement (AES128/256)




MACsec Tag Format

| Field | Size | Description |
|---|---|---|
| Ethertype | 16 bit | MAC length/type value for MACsec packet<br>Ethertype = 88-E5 |
| TCI | 6 bit | Tag control info contains:<br>Version, ES, SC, SCB, E, C (indicates how frame is protected) |
| AN | 2 bit | Association number |
| SL | 8 bit | Short Length<br>Indicates MSDU length of 1-48 octets<br>0 indicates MSDU length > 48 octets |
| PN | 32 bit | Packet sequence number |
| SCI | 64 bit | Secure channel identified (optional) |

Négociation SAP




Échange De Clés

# MACsec Key Derivation Schemes

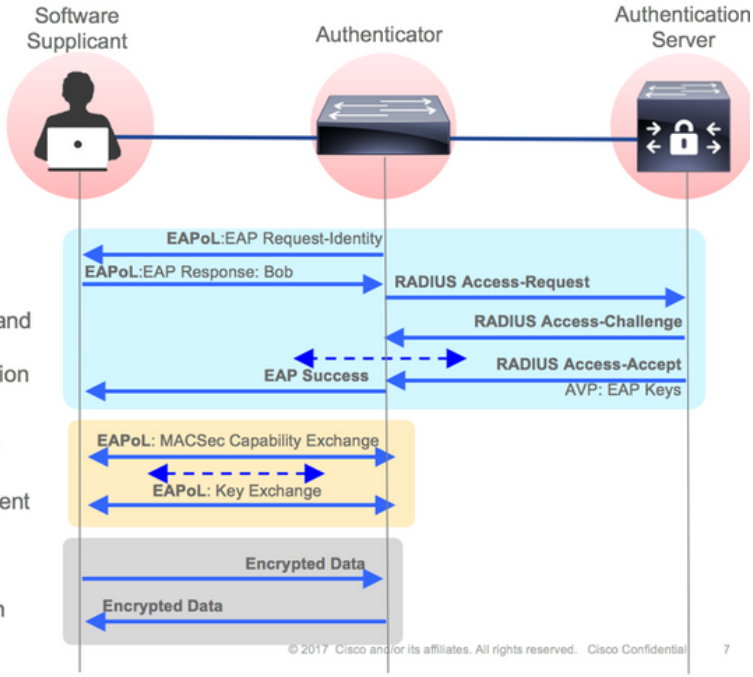
## Session Key Agreement Protocols

**SAP** — **Security Association Protocol** is Cisco proprietary protocol for MACSec Key negotiation.

Used only for Switch-to-Switch encryptions.

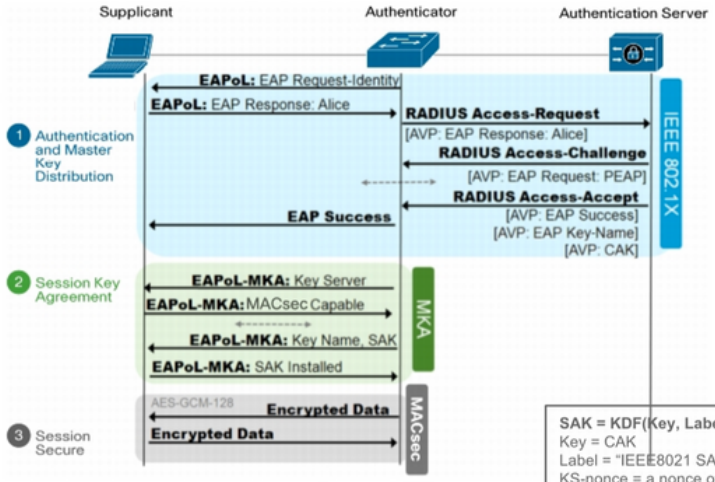
**MKA** — MKA (**MACsec Key Agreement**) is defined in IEEE 802.1X-2010.

Used today for Switch-to-Host encryptions. Router MACsec uses MKA






# MKA Exchange




ICV = AES-CMAC(ICK, M, 128)
M = DA + SA + (MSDU – ICV)

A pairwise CAK (Connectivity Association Key) is derived directly from the EAP MSK:
**CAK = KDF(Key, Label, mac1 | mac2, CAKlength)**

Key = MSK[0-15] for a 128 bit CAK, MSK[0-31] for a 256 bit CAK
Label = "IEEE8021 EAP CAK"
mac1 = the lesser of the two source MAC addr used in the EAPOL-EAP exchange
mac2 = the greater of the two source MAC addr used in the EAPOL-EAP exchange
CAKlength = two octets representing an integer value (128 for a 128 bit CAK, 256 for a 256 bit CAK) with the most significant octet first

The KEK(Key Encryption Key) is derived from the CAK using the following transform:
**KEK = KDF(Key, Label, Keyid, KEKLength)**

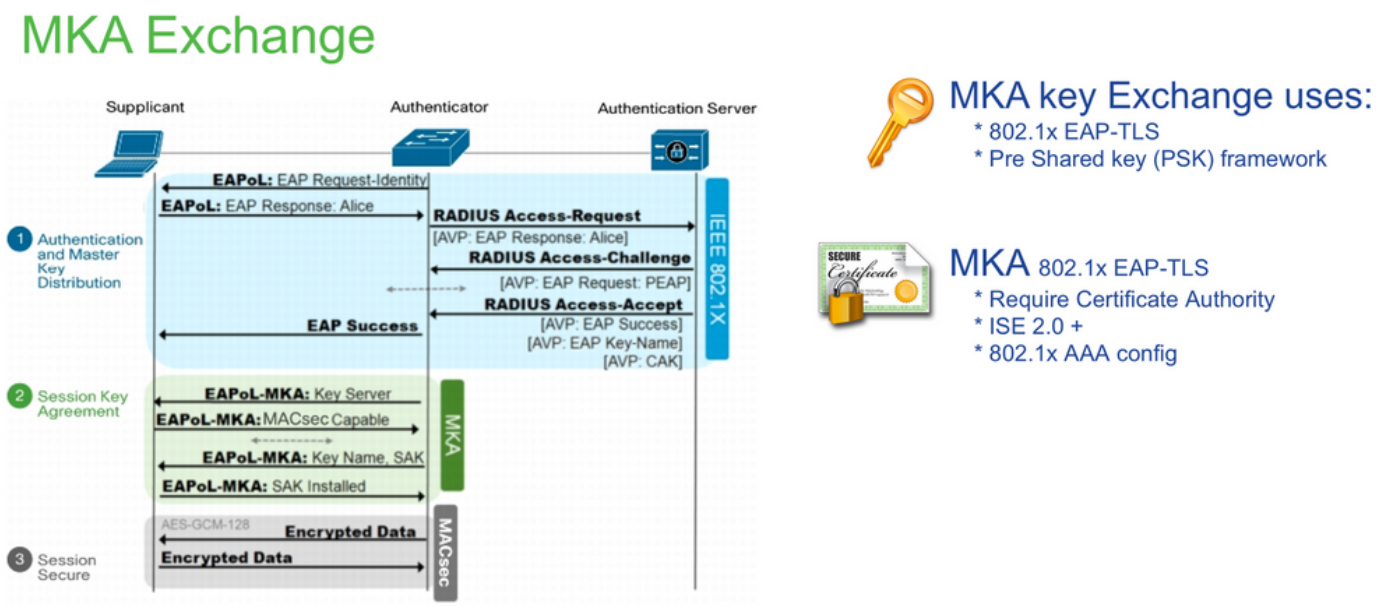
Key = CAK
Label = "IEEE8021 KEK"
Keyid = the first 16 octets of the CKN, with null octets appended to pad to 16 octets
KEKLength = two octets representing an integer value (128 for a 128 bit KEK, 256 for a 256 bit KEK) with the most significant octet first

The ICK (ICV Key) is derived from the CAK using the following transform:

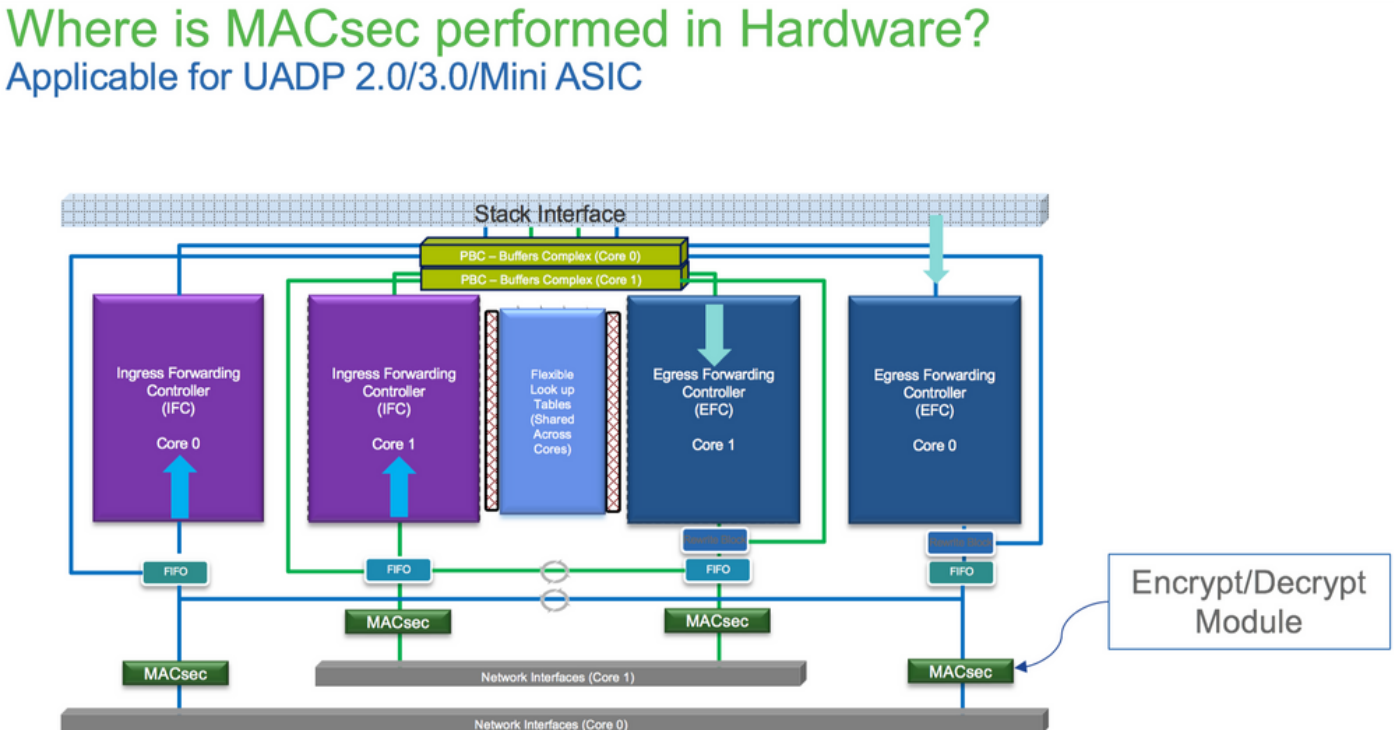
**ICK = KDF(Key, Label, Keyid, ICKLength)**

Key = CAK
Label = "IEEE8021 ICK"
Keyid = the first 16 octets of the CKN, with null octets appended to pad to 16
ICKLength = two octets representing an integer value (128 for a 128 bit ICK, 256 for a 256 bit ICK) with the most significant octet first

**SAK = KDF(Key, Label, KS-nonce | MI-value list | KN, SAKlength)**
Key = CAK
Label = "IEEE8021 SAK"
KS-nonce = a nonce of the same size as the required SAK, obtained from an RNG each time an SAK is generated.
MI-value list = a concatenation of MI values (in no particular order) from all live participants
KN = four octets, the Key Number assigned by the Key Server as part of the KI
SAKlength = two octets representing an integer value (128 for a 128 bit SAK, 256 for a 256 bit SAK) with the most significant octet first.




MACsec sur la plate-forme




Matrice de compatibilité des produits

## LAN MACsec Support per Platform

| | MACsec | Cat 9200 | | Cat 9300 | | Cat 9400 | | Cat 9500 | | Cat 9500H / 9600 | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | SW | License | SW | License | SW | License | SW | License | SW | License |
| **Switch to Switch** | 128 Bits SAP | 16.10.1 + | NE | 16.6.1 + | NE | 16.10.1 + | NE | 16.6.1 + | NE | 16.9.1 + / 16.11.1 + | NE |
| | 128 Bits MKA | 16.10.1 + | NE | 16.6.1 + | NE | 16.10.1 + | NE | 16.6.1 + | NE | 16.9.1 + / 16.11.1 + | NE |
| | 256 Bits MKA | Not Supported | | 16.6.1 + | NA | 16.10.1 + | NA | 16.6.1 + | NA | 16.9.1 + / 16.11.1 + | NA |
| | **ClearTag** Pass Through | 16.10.1 + | NE | 16.10.1 + | NE | 16.10.1 + | NE | 16.10.1 + | NE | 16.10.1 + / 16.11.1 + | NE |
| | | | | | | | | | | | |
| **Host to Switch** | 128 Bits MKA | 16.10.1 + | NE | 16.8.1 + | NE | 16.9.1 + | NE | 16.8.1 + | NE | 16.9.1 + / 16.11.1 + | NE |
| | 256 Bits MKA | Not Supported | | 16.9.1 + | NA | 16.10.1 + | NA | 16.9.1 + | NA | 16.9.1 + / 16.11.1 + | NA |

**NE – Network Essentials. NA – Network Advantage.**
**C9300 Stackwise 480 / C9500 SWV High Availability is not supported for MACsec**
**C9400 Sup 1XL-Y does not Support MACsec on any Supervisor ports**
**C9400 Sup 1 and 1XL support MACsec for only for interfaces with speed 10/40 Gbps**

## LAN MACsec Performance Data

| | MACsec | Cat 9200 | Cat 9300 | Cat 9400 | Cat 9500 | Cat 9500H / 9600 |
|---|---|---|---|---|---|---|
| **Switch to Switch** | 128 Bits SAP | Line Rate | Line Rate | Line Rate | Line Rate | Line Rate |
| | 128 Bits MKA | Line Rate | Line Rate | Line Rate | Line Rate | Line Rate |
| | 256 Bits MKA | Not Supported | Line Rate | Line Rate | Line Rate | Line Rate |
| | | | | | | |
| **Host to Switch** | 128 Bits MKA | Line Rate | Line Rate | Line Rate | Line Rate | Line Rate |
| | 256 Bits MKA | Not Supported | Line Rate | Line Rate | Line Rate | Line Rate |

**C9400 Sup 1XL-Y does not Support MACsec on any Supervisor ports**
**C9400 Sup 1 and 1XL support MACsec for only for interfaces with speed 10/40 Gbps**

**NE – Network Essentials. NA – Network Advantage.**
**Line rate is calculated with the additional MACsec header overhead**

## Informations connexes

[Guide de configuration de la sécurité, Cisco IOS® XE Gibraltar 16.12.x (commutateurs Catalyst 9300)]

À propos de cette traduction
Cisco a traduit ce document en traduction automatisée vérifiée par une personne dans le cadre d'un service mondial permettant à nos utilisateurs d'obtenir le contenu d'assistance dans leur propre langue.

Il convient cependant de noter que même la meilleure traduction automatisée ne sera pas aussi précise que celle fournie par un traducteur professionnel.