New Features for Cisco IOS XE 17.10.1a

This chapter contains the following sections:

Software Supported MACsec

Overview

All existing Cisco IOS XE based router/switch use special transceiver to do MACsec encryption/decryption. This software MACsec uses CDAL infrastructure in QFP to do crypto operation. Comparing to the hardware choice, the way configuration/status/datapath is done is different thus creating some limitation on the functionality.

Release 17.10.1a only supports MACsec on L2 interfaces. The MACsec port must be put into access mode. As the encryption happens on the egress SVI interface, the vlan used for the port should be unique, meaning no other interface can use that vlan. This limitation is because the QFP does not have MAC table information.

Note Since MACsec is being done through software, performances are not line rate on L2 interfaces.

Note Cisco supports only the should secure MACsec mode for IR1101, which allows unencrypted traffic even in a secured state.

Limitation:

The IR1101 does not support the must secure mode.

For an egress packet, SVI only know the packet needs to go out on a vlan without info about any specific interface. It is up to the switch chip to decide which port to go. All the packets without MACsec tag can come in as usual. Outgoing L2 packet will also egress without encryption or modification.

Both the NE and NA license support GCM-AES-128. This feature is not available running the NPE image.

The MACsec protocol is defined in IEEE802.1AE.

Feature Limitations

- MACsec is not supported in controller mode in this release.
- There must be a unique vlan id for a MACsec interface.
- Only gcm-aes-128 is supported in this initial release.
- Both explicit and non-explicit SCI are supported on ingress side. The IR1101 sends out only explicit SCI packets as it is not an end system.
- The IR1101 does not support confidentiality offset.
- Integrity only is not supported in this first release.
- For gcm-aes-128, up to 32 bytes are added to an encrypted packet compared to a plain packet. So the MTU setup should add 32 for it to work properly.
- The MACsec key is managed by the MKA module. For that device, it requires a static key for MKA to negotiate MACsec key.
- There is no MIB support.

Related Documentation

Further information can be found at the following:

- MACsec and the MACsec Key Agreement (MKA) Protocol
- MACSEC and MKA Configuration Guide, Cisco IOS XE 17

Sample MKA Configuration

See the following example:

```
conf t
   aaa new-model
   mka policy p1
       key-server priority 1
       macsec-cipher-suite gcm-aes-128
       sak-rekey interval 3600
end
conf t
   key chain cak1 macsec
      key 414243
         cryptographic-algorithm aes-128-cmac
         key-string 0 12345678901234567890123456789012
         lifetime local 00:00:00 29 November 2021 infinite
end
conf t
   int fa 0/0/2
      switchport mode access
      switchport access vlan 77
      mtu 1532
      mka policy p1
      mka pre-shared-key key-chain cak1
      macsec network-link
```

```
      macsec replay-protection window-size 128
end
```

Show Commands

Show cpp_cp internal info:

```
show platform hardware cpp active feature soft-macsec server tx [dp] [item]
show platform hardware cpp active feature soft-macsec server rx [dp] [item]
show platform hardware cpp active feature soft-macsec server control [dp] [item]
```

Other show commands:

```
show macsec summary
show macsec status int fa 0/0/2
show macsec statistics int fa 0/0/2A
```

Clear Statistics

```
Clear macsec statis int fa 0/0/2
```

Test Command

Print 10 MKA packet for debug:

```
test platform software smacsec mka-ingress
```

High Security (HSEC) License

HSEC (High Security) license is a feature license that can be configured in addition to the network license (NE/NA). An HSEC license provides export controls for strong levels of encryption. HSEC is available to customers in all currently non-embargoed countries as listed by the U.S. Department of Commerce. Without an HSEC license, SEC performance is limited to a total of 250 Mbps of IPsec throughput in each direction. An HSEC license removes this limitation.

Command Line Interface

The configuration mode CLI to enable HSEC on the IR1101 is the following:

```
IR1101(config)# license feature hsec9
```

To benefit from the HSEC license, a new bandwidth will be available. The new bandwidth is called **uncapped**, and it is available with the following CLI from configuration mode:

```
IR1101(config)# platform hardware throughput level ?
250M throughput in bps
uncapped throughput in bps
IR1101# platform hardware throughput level uncapped
```

After performing the above commands, write mem and reload the router. The configuration will take effect when the router comes back up.

License Types

With this new feature, the IR1101 will support the following bandwidth/license types:

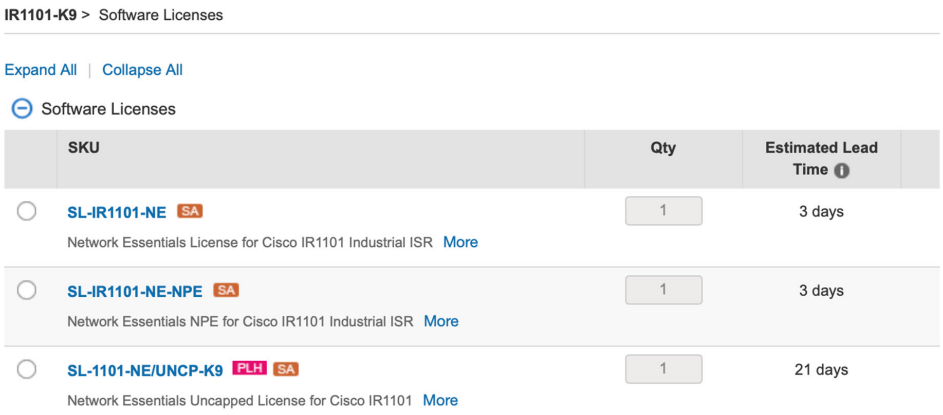
- Network-essentials 250 Mbps
- Network-advantage 250 Mbps

- Network-essentials uncapped
- Network-advantage uncapped
- HSEC

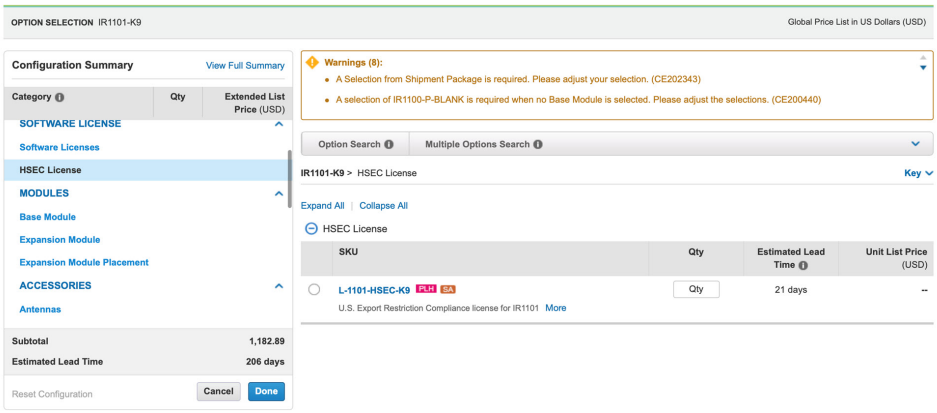
Ordering

The following is an example from the IR1101-K9. The license will be available on the IR1101-A-K9 as well.

In the following example, select the SL-1101-NE/UNCP-K9 (Network Essentials Uncapped License):



The L-1101-HSEC-K9 license will get auto included when you select the uncapped license, as shown in the following:



Cisco Software Central

This guide provides information on how to order, activate, and manage your Cisco Smart Licenses.

https://software.cisco.com/software/csws/ws/platform/home?locale=en_US&locale=en_US&locale=en_US#

Enable Secure Data Wipe Capabilities

Secure data wipe is a Cisco wide initiative to ensure storage devices on all the IOS XE based platforms to be properly purged using NIST SP 800-88r1 compliant secure erase commands. Whenever possible, IoT platforms will leverage the corresponding ENG design and implementation available so far on their platforms.

This feature is supported on the following IoT platforms:

- IR1101
- IR1800
- IR8140
- ESR6300

When the enable secure data wipe is executed, the following will get wiped out:

- IR1101, IR1800, IR8140: NVRAM, rommon variables, and bootflash
- ESR6300: NVARM, rommon variables, bootflash

The router will be in rommon prompt with default factory settings (baud rate 9600) after the command is executed. The bootflash will not get formatted until booting with IOS image thru usbflash or tftp download if the platform is supported.

Performing a Secure Data Wipe

To enable the feature, perform the following:

```
Router#factory-reset all secure
The factory reset operation is irreversible for securely reset all. Are you sure? [confirm]Y
```

Important This operation may take hours. Please do not power cycle.

To check the log after the command is executed, and booting up IOS XE, perform the following:

```
Router#show platform software factory-reset secure log
Factory reset log:
#CISCO DATA SANITIZATION REPORT:# IR1800
Purge ACT2 chip at 12-08-2022, 15:17:28
ACT2 chip Purge done at 12-08-2022, 15:17:29
mtd and backup flash wipe start at 12-08-2022, 15:17:29
mtd and backup flash wipe done at 12-08-2022, 15:17:29.
```

Rawsocket Keepalive Configuration CLI

Rawsocket keepalive for async interfaces is a feature that existed in classic IOS platforms. As part of 17.10.1a, the feature will be extended to IOS-XE based platforms. A new CLI with the following syntax will be added under rawsocket.

```
Router(config-line)#raw-socket tcp keepalive interval
```

CLI Changes

On IOS-XE platforms starting from 17.10.1a, there is a CLI correction and an additional CLI was added as part of raw-socket.

The correction is for the **raw-socket idle timeout** command. There is now an option to configure the timeout based on minutes and seconds, whereas the previous configuration used only minutes.

```
Router(config-line)# raw-socket tcp idle-timeout [0-1440] [<0-59> | cr]
```

The additional CLI is for clearing the raw-socket TCP clients. The command syntax is **clear raw-socket line** *[1-145|tty|x/y/z]* for example:

```
Router# clear raw-socket line 0/2/0
```

Note When initiating clear raw-socket line, raw-socket sessions will be cleared for raw-socket clients from the **show raw-socket tcp sessions** command. Connections will be re-established after a TCP hand-shake, which can be done by doing shut/no shut on TCP connection interface.