# AUTOSAR communication groups,
## IEEE802.1X-2020 & EAP-TLS Complex Device Driver
## NA UG Overview

Robert Mansour, Supervisor, Vehicle Software Platform
Ford Motor Company.

September 21th,2023

# Agenda

- Background and Challenges.
- Rational for MacSec and port-access control.
- IEEE802.1X-2020 specification and EAP over LAN framing.
- AUTOSAR MacSec Key Agreement (MKA) with pre-shared key.
- MKA protocol with EAP participation.
- EAP with TLS authentication method.
- EAP-TLS, towards standardization.

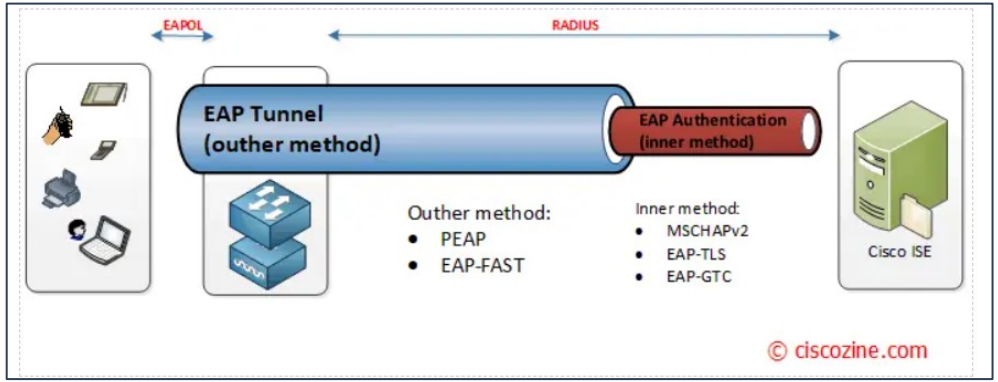
# Background & challenges

- What is MacSec, and why is it used?
  - Cyber Security considerations and more focus on port-based access control.
  - Port-based access control regulates access to the network and critical data.
  - IEEE802.1X-2020 provides:
    - specification on port access control and usage of 3 step security approach.
    - Definition of PAE (Port Access Entity), logon, authentication process & MacSec Key Agreement.
    - Definitions of EAPOL frame and PDU (Extensible Authentication Protocol Over LAN) as layer 2 frame.
  - AUTOSAR MacSec specification defined various part of the IEEE802.1X-2020.
- Challenges:
  - AUTOSAR Standard MacSec specification does not consider usage of EAP-TLS as the authentication method.
  - How can we fit EAP-TLS in an AUTOSAR CDD framework to work with stack
  - How to align IEEE802.1X supplicant design to align with AUTOSAR methodology

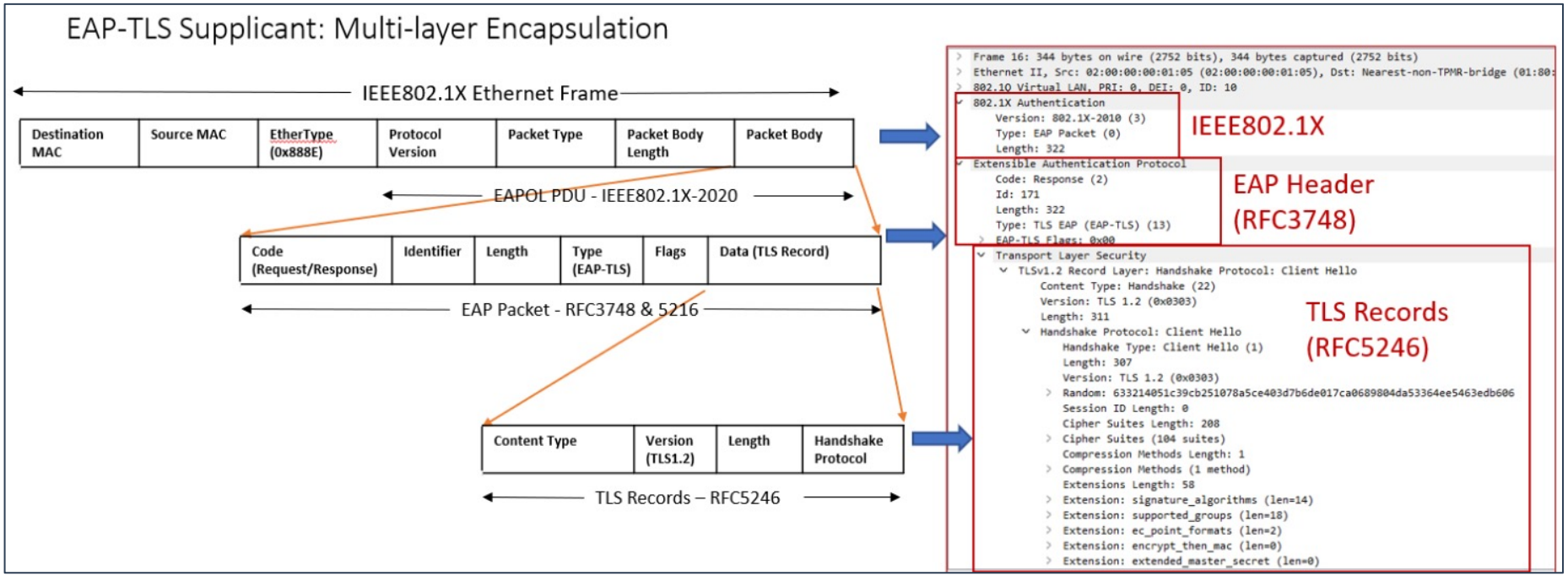
# Rationale for MacSec and port-access control

- MacSec provides end-to-end encryption at the MAC layer.
- Prevents data tampering by any intermediate device or network.
- MacSec offers high performance and low latency.
- MacSec can exist with other security protocols (SSL/TLS, IPsec)
- Can operate at higher layers to provide added features (Tunneling, certificate-based authentication).

Source: MACsec: A Guide to LAN-WAN Security at the MAC Layer (linkedin.com)



# IEEE802.1X-2020 Frame Structure



# MacSec Key Agreement (MKA) with pre-shared key (PSK)

- MKA protocol allows PAEs to confirm mutual possession of secure CAK and agree on MacSec Symmetric shared keys.
- The Root of MKA sessions is the CAK (Connectivity Association Key), a secret key. (IEEE802.1X-2020 section 9.3 MKA key hierarchy)

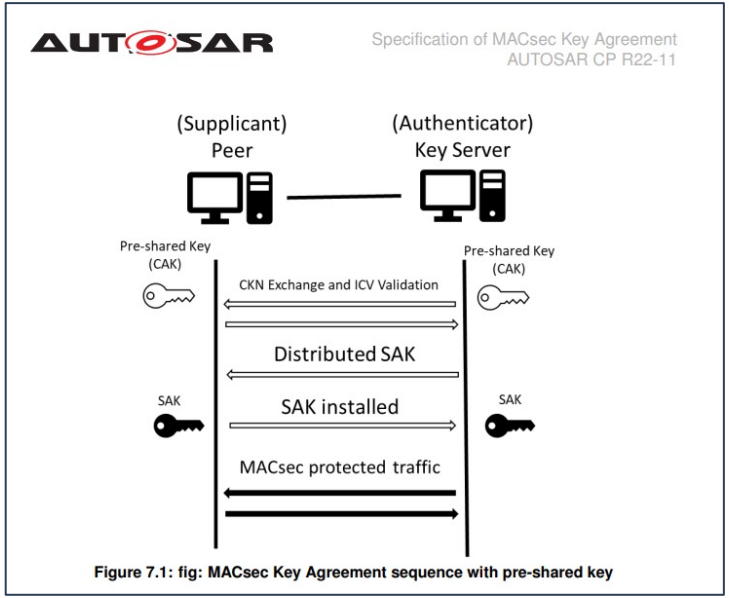


Figure 7.1: fig: MACsec Key Agreement sequence with pre-shared key

Source: Specification of MACsec Key Agreement (autosar.org)

# MacSec Key Agreement (MKA) with CAK acquired through participation in EAP

- EAP (Extensible Authentication Protocol) can be used to mutually authenticate a supplicant PAE.
- EAP shall support key derivation, to generate MSK that is 64 bytes long. (IEEE802.1X-2020 MKA and EAP methods)

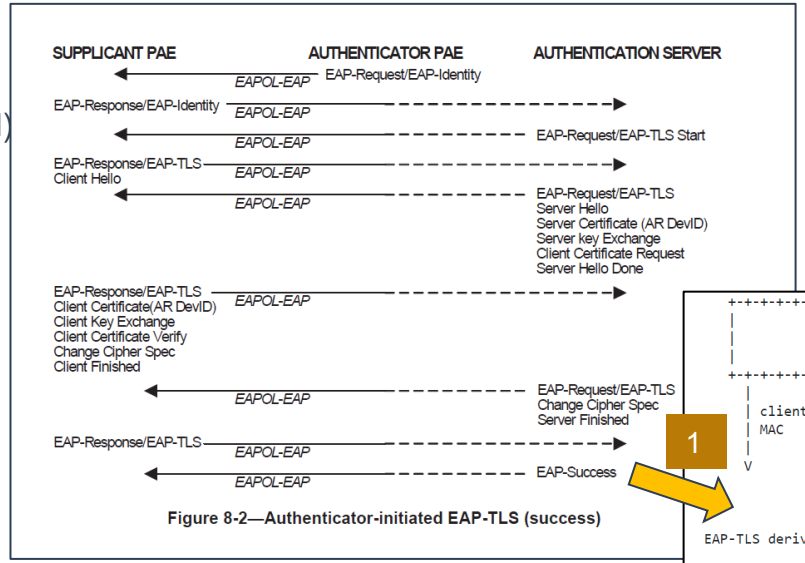



Figure 8-2—Authenticator-initiated EAP-TLS (success)

Source: IEEE802.1X-2020 Section 8. Authentication using EAP

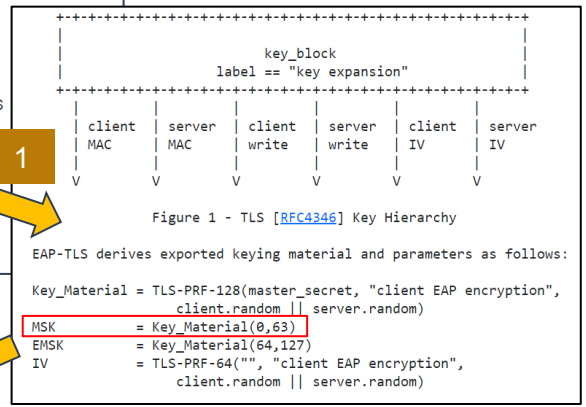



Figure 1 - TLS [RFC4346] Key Hierarchy

EAP-TLS derives exported keying material and parameters as follows:

```
Key_Material = TLS-PRF-128(master_secret, "client EAP encryption",
                           client.random || server.random)
MSK          = Key_Material(0,63)
EMSK         = Key_Material(64,127)
IV           = TLS-PRF-64("", "client EAP encryption",
                          client.random || server.random)
```

Source: RFC 5216 - The EAP-TLS Authentication Protocol (ietf.org)

**6.2.2 Using EAP for CAK key derivation**

A pairwise CAK is derived directly from the EAP MSK using the following transform:

CAK = KDF(Key, Label, mac1 | mac2, CAKlength)

where

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 addresses used in the EAPOL-EAP exchange (11.1.2)

mac2 = the greater of the two source MAC addresses 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

Source: IEEE802.1X-2020 Section 6.2.2. Using EAP for CAK key derivation

# IEEE802.1X-2020: Authentication using EAP-TLS

Supplicant PAE CDD

Authenticator PAE CDD

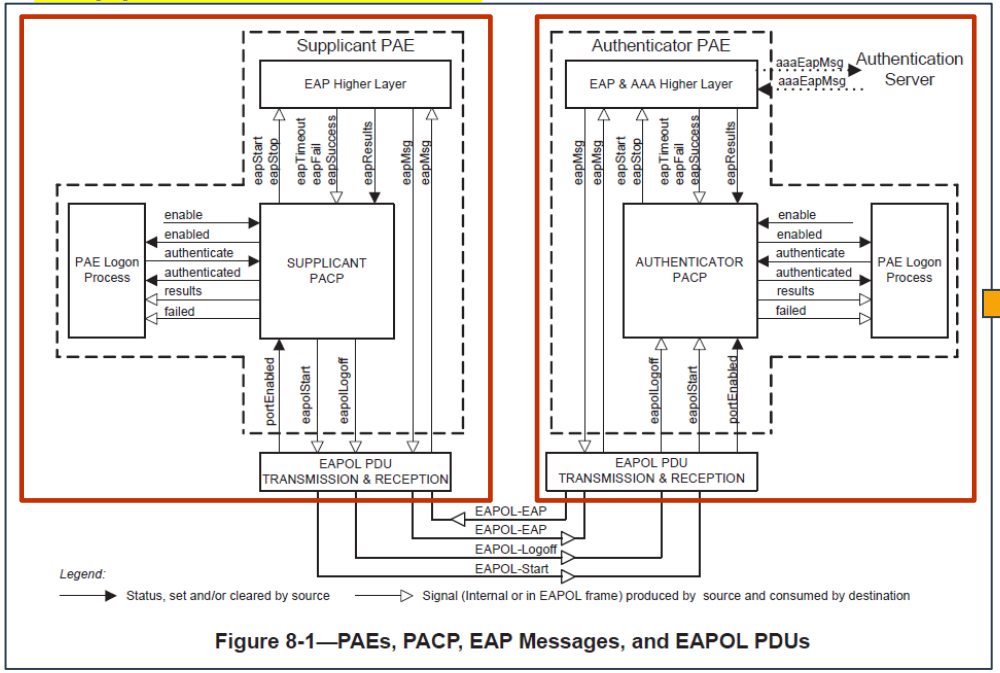
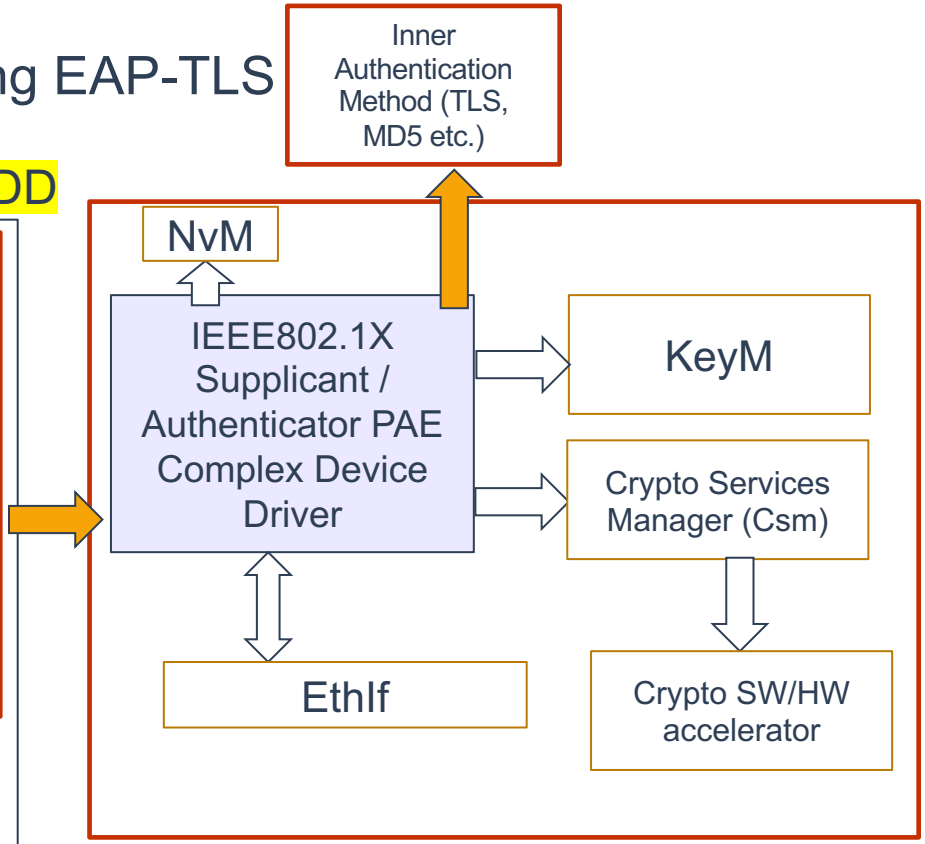



**Figure 8-1—PAEs, PACP, EAP Messages, and EAPOL PDUs**



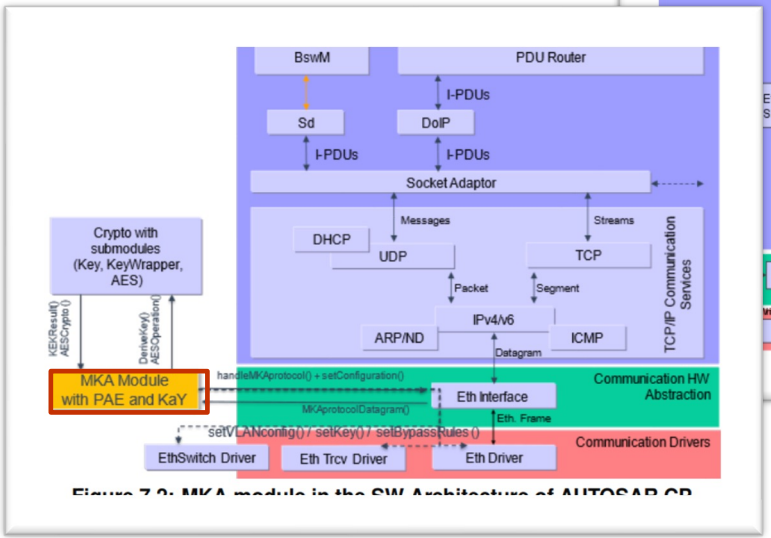
# What are the PROS/CONS of EAP-TLS

| PROS | CONS |
| --- | --- |
| Enhanced security with certificate based-authentication | Slower Ethernet Learning boot time |
| Dynamic CAK generation | |
| End-to-end security at MAC level (device protection) | |
| EAP can provide flexibility on inner authentication methos (TLS, MD5, chacha etc.) | |

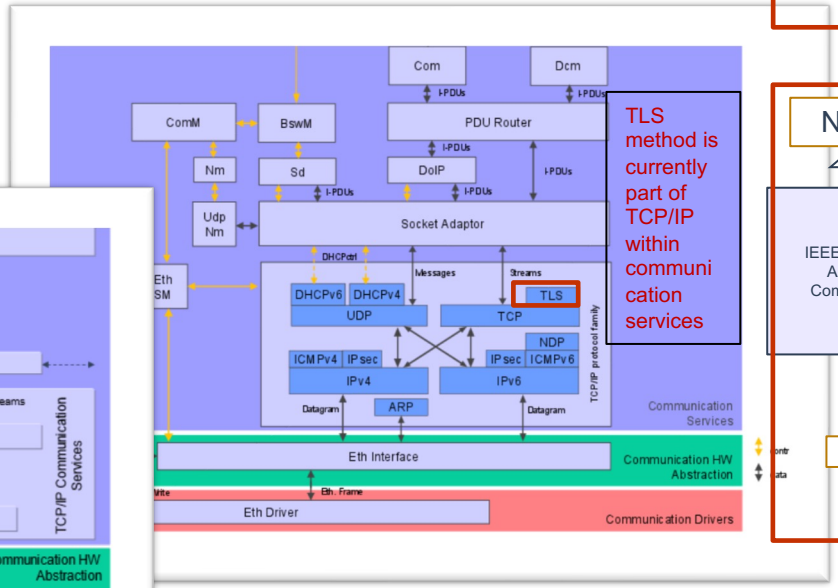
# How EAP-TLS can fit in the AUTOSAR Stack ?

Ford's current approach:
- Implementation as a CDD

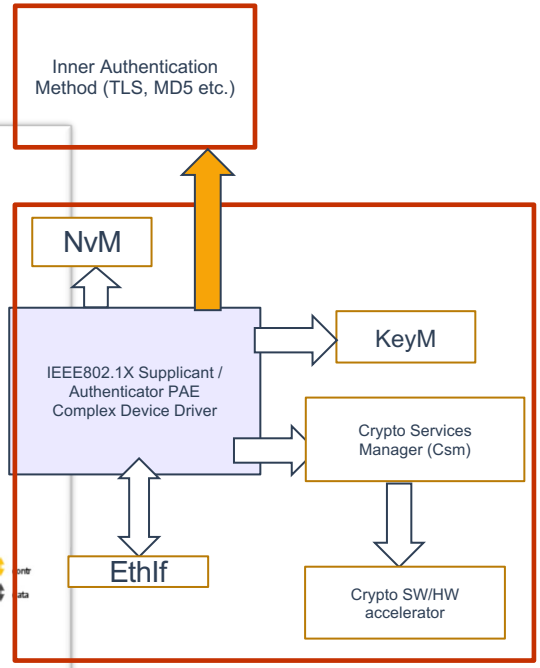
Other solutions (towards standardization):
- Part of BSW layer, adjacent to MKA




Specification of MACsec Key Agreement (autosar.org)




TLS method is currently part of TCP/IP within communication services




Source: Specification of TCP/IP Stack (autosar.org) R22-11

Interface Considerations:
- EthIf (EAPOL frame routing).
- KeyM (Certificates Management).
- Csm (Crypto operations).
- NvM (Data persistency).
- TLS (Inner Authentication Method).

Other Considerations:
- Isolation of EAP protocol from the authentication method ?

# How EAP-TLS can fit in the AUTOSAR Stack ?

**Thought Starters & open forum discussion:**

- EAP-TLS as 1 module within communication services ?
- Split of EAP from TLS authentication method ?'
- MKA to be within Communication HW Abstraction to interface with EthIf + Ethernet MacSec Drivers ?
- Both MKA + EAP-TLS to be within Communication Services & interface with EthIf?
- Single routing of IEEE802.1X (0x888E) frames from EthIf to upper layer module owner ?
- Other ?

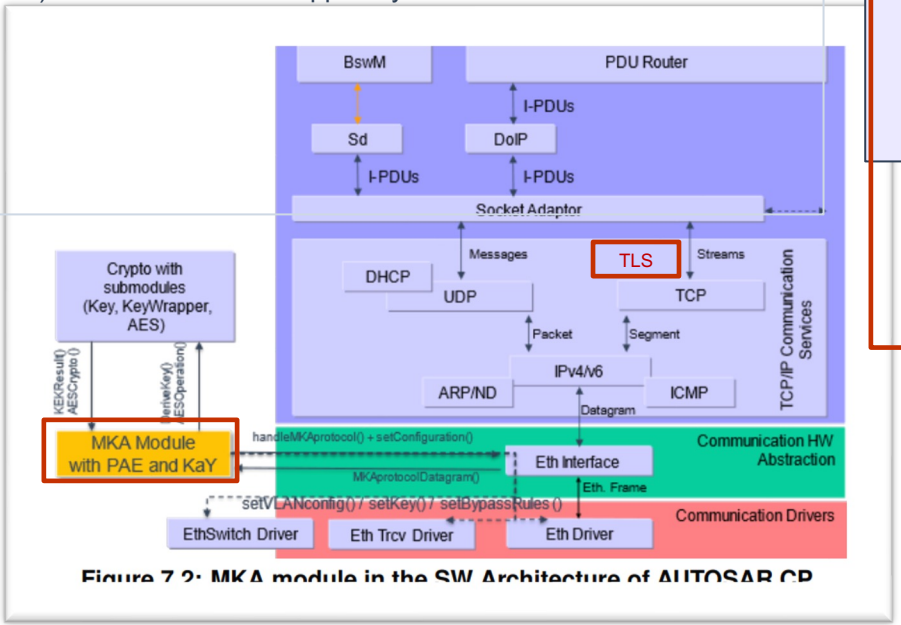
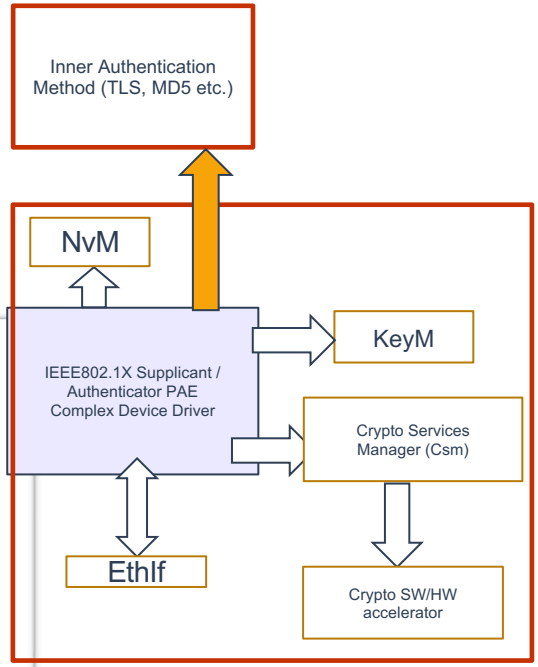



Figure 7.2: MKA module in the SW Architecture of AUTOSAR CP




Specification of MACsec Key Agreement (autosar.org)

# Key Takeaways:

- Integration of EAP-TLS CDD within AUTOSAR stack allowed interface with standard AUTOSAR interfaces.
- Maximized usage of standard Csm, KeyM for cryptographic operations, and certificate management.
- EAPOL framing leveraged EthIf by defining frame owners for various EAPOL packet types (EAP, MKA, Logoff, Start, etc).

# Thank you!