Project

Applying Your Learning

Lab9

The Instructors Design’s for Students to Analyze

Remember

There is no 100% security

Security, like all engineering, involves tradeoffs

Know what you are trying to secure

The adversary…

State Sponsored

Schedule

- 12 APR Topic Review / Project Launch
- 19 APR [DEFEND] Design Review: **Key Management**
- 26 APR [DEFEND] Design Review: **Secure Message Exchange**
- 03MAY [ATTACK] Analysis: **Attack Plan for Two Designs**

Biased toward attacking Instructor designs

Project -- Randomizer

Network	Number of Nodes per Session	Nature of Security	Channel	Hardware Acceleration	Message Rate
CAN	Fixed; N=5	Secure each message	1	Symmetric only	10 msg/sec
CAN FD	Dynamic; N=3..15	Secure the control loop	8	Symm & Asymm	1,000 msg/sec

Project – Instructor Constraints

Instructor A	Instructor B
CAN FD Fixed; N = 5 Secure Control Loop 8 Channel Symm and Asymm 1000 msg/sec	CAN FD Dynamic; N = 3..15 Secure Control Loop 8 Channel Symm and Asymm 10 msg/sec

NOTE!

- Each design has at least one huge security gap.
- Can you identify them?

Instructor A: Fast Multi-Loop

Key Management

Key Management is handled in two stages: by

- **Joining** (done once in an ECU's "life"), which is driven by PoP CWTs and uses NaCl Box() to move network channel key, S_{NCH}.
- **Session** (done on every power cycle, or more often)

An : Valid PoP CWT
Az : ECUs must be for the VIN and assigned a channel(s) – signed by OEM

P/K_{OEM} – for signing/validating PoP CWT (ECU's do NOT have K_{OEM})
P/K_A -- public/private key agreement keypair for ECU "A".
S_{NCH} – long-lived network channel key
S_{VCH} – short-lived session channel key

Message Exchange

```
[ F32|32 I24 ] 64
```

F : 32-bit freshness value, unique to each Tx. Rx must track FV of each ECU they receive from.
I : 24-bit truncated CMAC covers PGN, SA, CH, FV and Data
C : none
An : must have signed CWT to get keys
Az : keys are distributed by channel, so CWT must contain channel(s) the ECU is authorized to

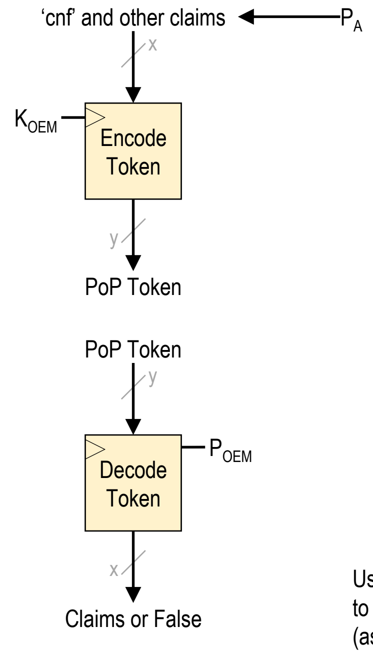
Instructor A

CAN FD
Fixed; N = 5
Secure Control Loop
8 Channel
Symm and Asymm
1000 msg/sec

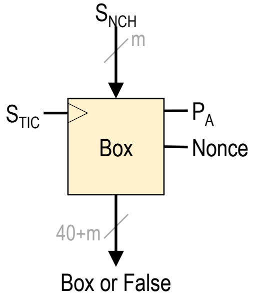
Instructor A – Key Management

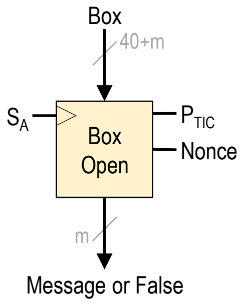
- **"Joining" is a one-time event**
  - TIC is the network leader
  - Uses CWT signed by OEM
    - CWT includes AZ for VIN and Channel
  - Mutual authentication of CWT
    - TIC validates ECU CWT using P_{OEM}
    - ECU validates TIC CWT using P_{OEM}
    - Validation includes Proof-of-possession step
      - Send a nonce in a box, require nonce+1 to come back in another box
  - TIC gives out long-lived channel key, S_{NCH} using Box()
    - Message size = (16+24)+16 = 56 bytes
    - (16+24) is MAC and nonce
    - 16 is AES-128 key, S_{NCH}.

OEM issues PoP CWT



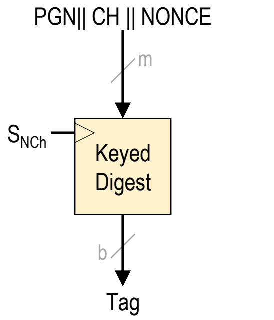
Use P_{ECU} and own K to confirm possession (ask ECU to sign a nonce)



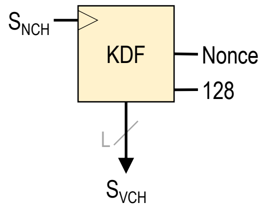


Instructor A – Key Management

- **“Session” happens on every power-cycle (or more often, if needed)**
  - TIC waits for all 5 members to complete NAME process before starting…
  - On each session & for each channel
    - TIC sends 8-bit channel, 128-bit nonce and 128-tag
    - Tag = CMAC(S_{NCH}, PGN || CH || NONCE)
    - Nonce Message = CH || NONCE || Tag
  - Each member validates the nonce, if valid
    - Calculate the session key for the channel(s) it participates in
    - S_{VCH} = KDF(S_{NCH}, NONCE)

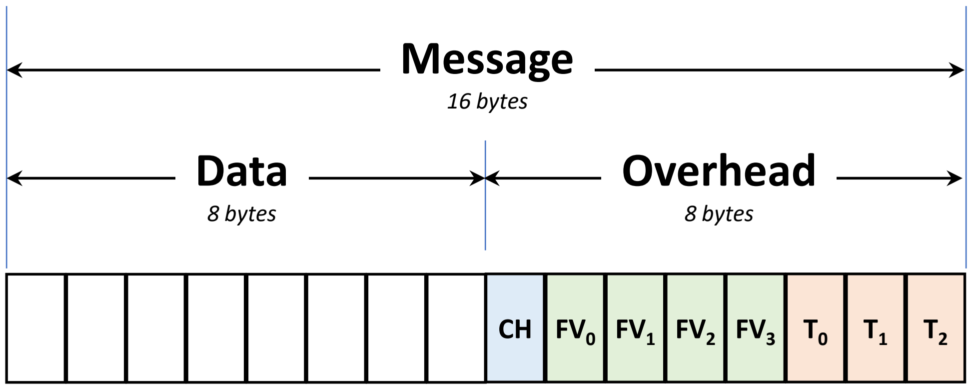


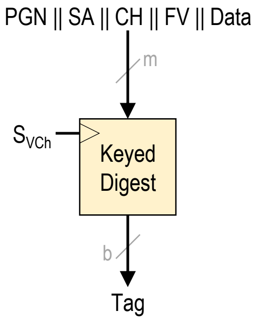
- Used by TIC to create nonce message.
- Used by member to validate the nonce message.



- Used by all participants to create the session key for the channel

Instructor A – Message Exchange





- Tag used to prove freshness, integrity, An and Az
- Only ECUs authorized to the channel have access to the key required to create and validate tags

CH – channel
FV – 32-bit freshness value, unique to each Tx
T – 24-bit truncated CMAC for tag, use CMAC(S_{VCH}, PGN || SA || CH || FV || Data)

Opted for the simplicity allowed by CAN FD and symmetric acceleration to tag each message (as opposed creating epochs and wrapping security around the stream)

Instructor B: SlowSigned

Key Management

Key Management (on every power cycle): by

- **Recognition,** ECU sends 32-byte public key.
- If other ECU doesn't recognize the Public key then it triggers mutual An/Az via PoP CWT.
- ECUs remember P_{ECU} and Authorized channels for subsequent power cycles.

An : Valid PoP CWT
Az : ECUs must be assigned a channel(s) – signed by OEM

P/K_{OEM} – for signing/validating PoP CWT (ECU's do NOT have K_{OEM})
P/K_A – public/private signing keypair for ECU "A".

Message Exchange

```
[ F32|32 I24 ] 64
```

F	:	32-bit freshness value, unique to each Tx. Rx must track FV of each ECU they receive from.
I	:	24-bit truncated CMAC covers PGN, SA, CH, FV and Data
C	:	none
An	:	must have signed CWT to get keys
Az	:	keys are distributed by channel, so CWT must contain channel(s) the ECU is authorized to

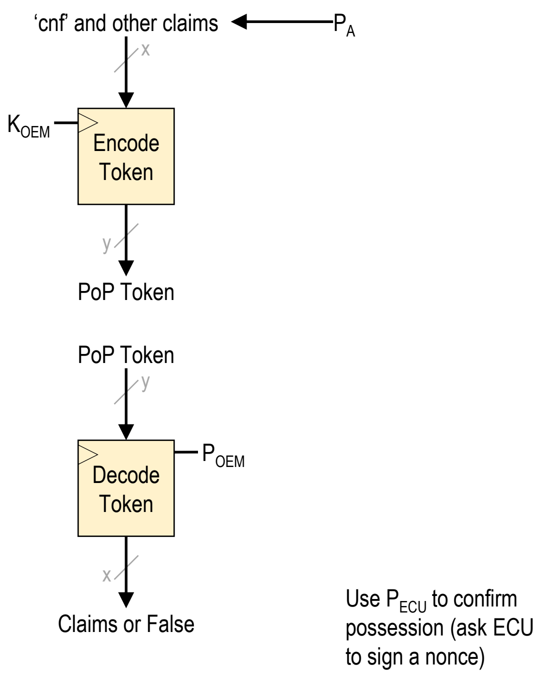
Instructor B

CAN FD
Dynamic; N = 3..15
Secure Control Loop
8 Channel
Symm and Asymm
10 msg/sec

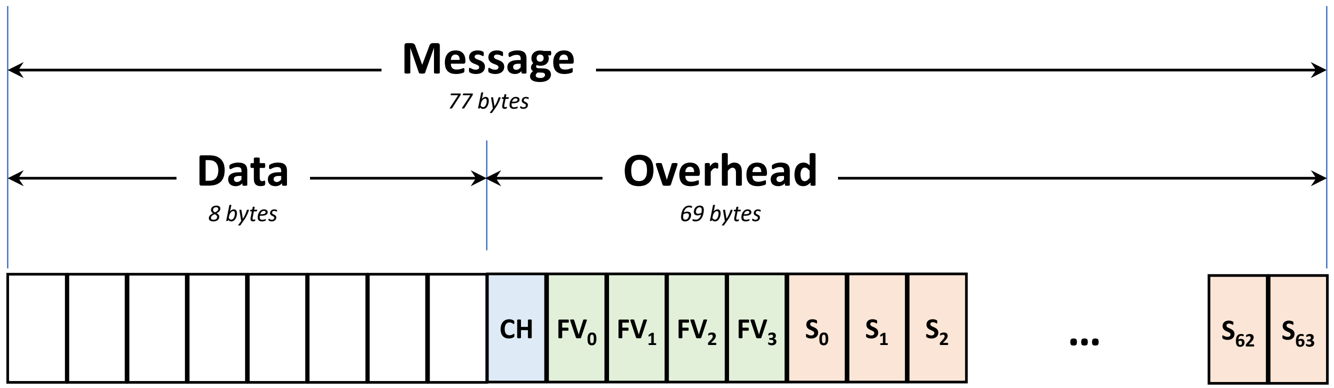
Instructor B – Key Management

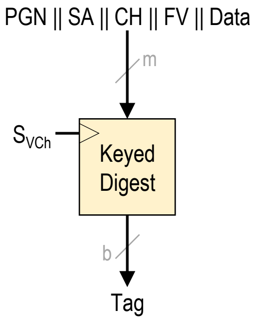
- **“Recognition” happens on every cycle**
  - ECU_A sends its public key, P_A, as 32-byte CAN FD frame
    - If everyone else recognizes the ECU then go to message exchange
  - If ECU_B does not recognize ECU_A:
    - ECU_B responds with “unrecognized” PGN, where the frame contains P_A.
    - ECU_B and ECU_A exchange PoP CWT, validate them, and then do Proof-of-Possession step.
      - Proof-of-Possession requires signing a nonce provided by the other ECU
    - ECUs then store the other ECU’s public key and channels in flash memory, to be used in future recognition events.

OEM issues PoP CWT



Instructor B – Message Exchange





- Tag used to prove freshness, integrity, An and Az
- Only ECUs authorized to the channel have access to the key required to create and validate tags

CH – channel
FV – 32-bit freshness value, unique to each Tx
T – 512-bit (64-btye) Ed25519 signature = Ed25519(K_A, PGN || SA || CH || FV || Data)

The receiver uses P_A to validate the signature.

NOTE This approach is acceptable given: a) the slow message rate and b) the availability of hardware accelerated asymmetric cryptography.