Secure Vehicle Communication

Secure Execution Environment
for V2V and V2I Communication

Antonio Kung

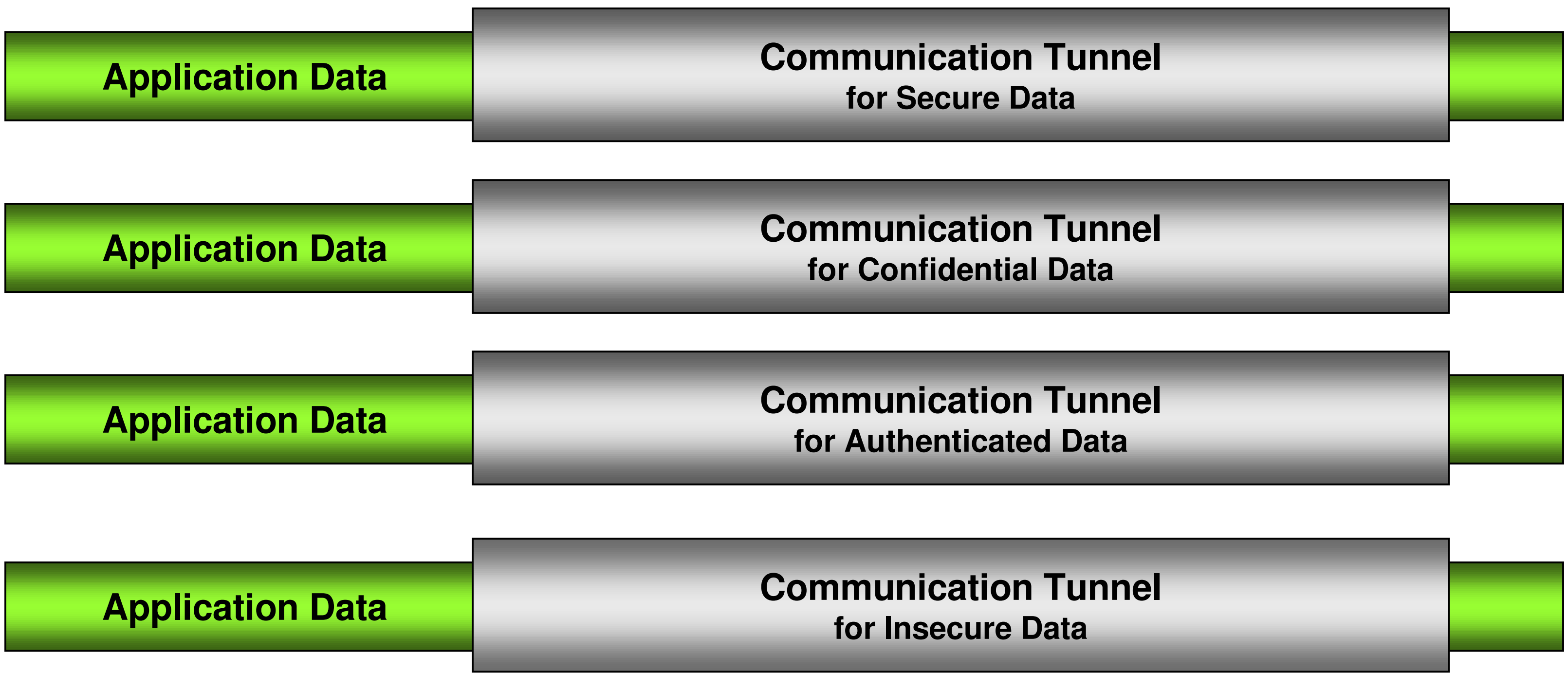
SEVECOM Project Co-ordinator

TRIALOG

Presentation Content

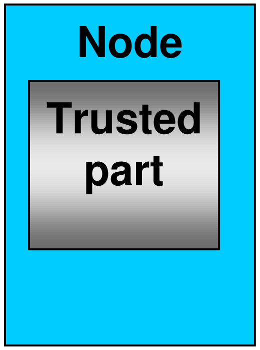
- Trust as a business requirement for execution platforms
  - Result from the GST project
- Security Module Approach
  - Result from the GST project (contribution from KU Leuven)
- Partitioning Approach
  - Result from the MILS Project

Color Notation for Trust

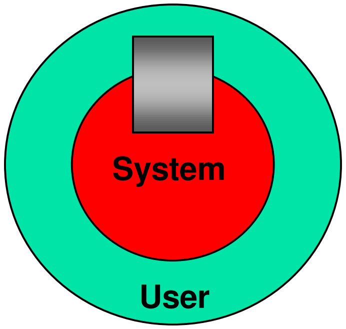


Trust in an Execution Environment

- A node is structured into a « trusted part » and a « non trusted part »

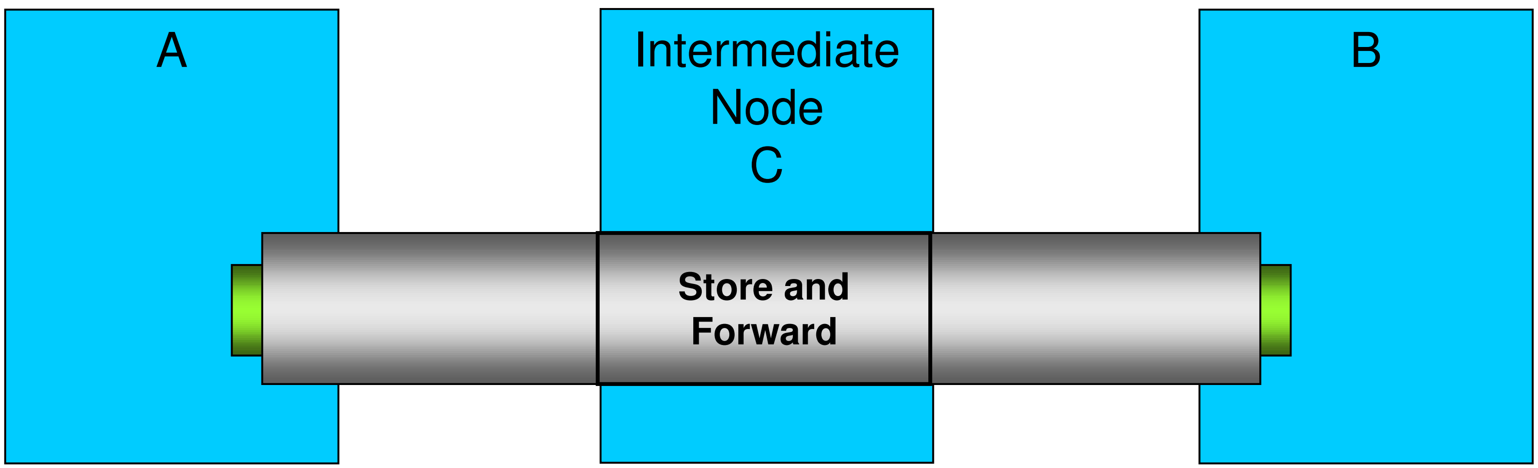


- Very classical in an OS with privileged mode management
  - But Security is transversal



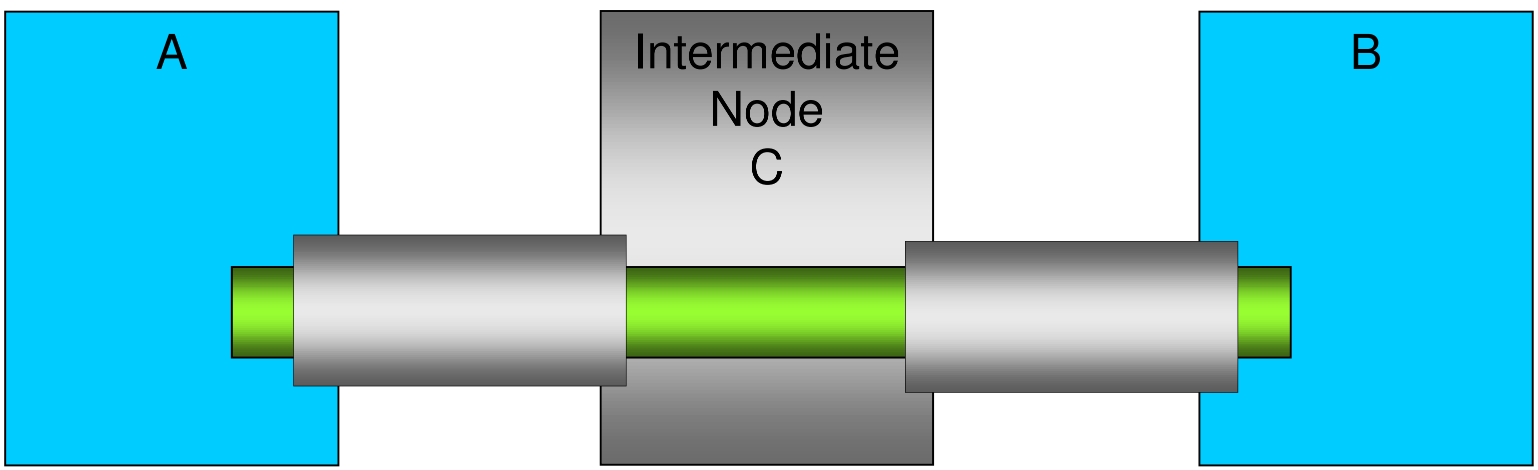
Case 1 of E2E Security

- A and B communicate via C
- C does not have access to application data
  - If C is malicious it can deny transmission



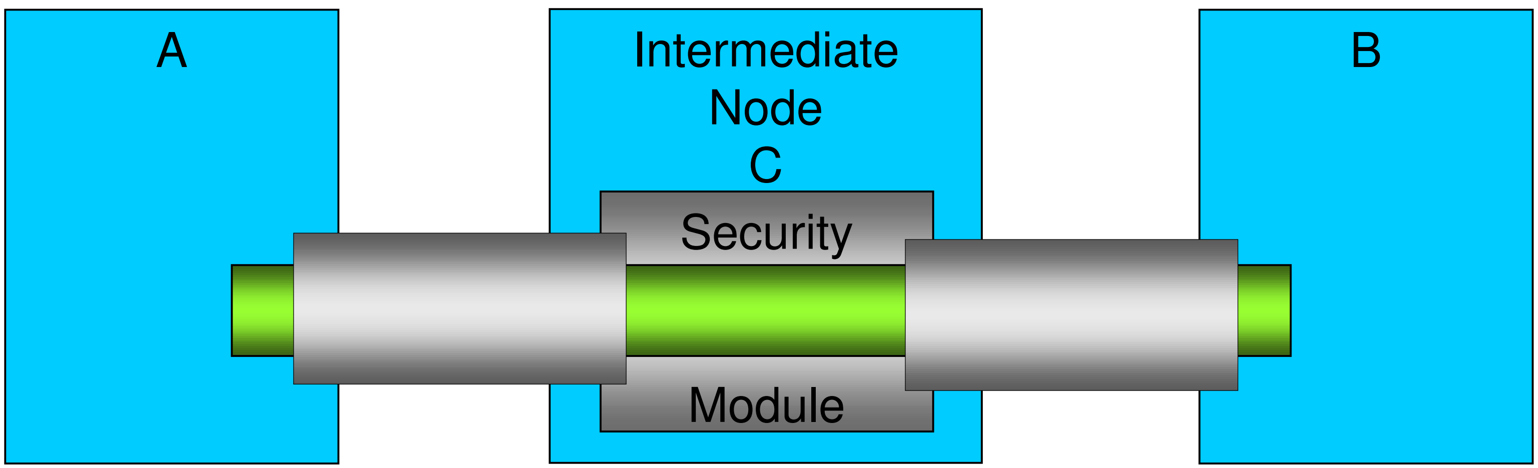
Case 2 or E2E Security

- A communicates with C, C communicates with B
- C has access to application data
  - it is a trusted node
- Crypto mechanisms can be different in A-C and in C-B

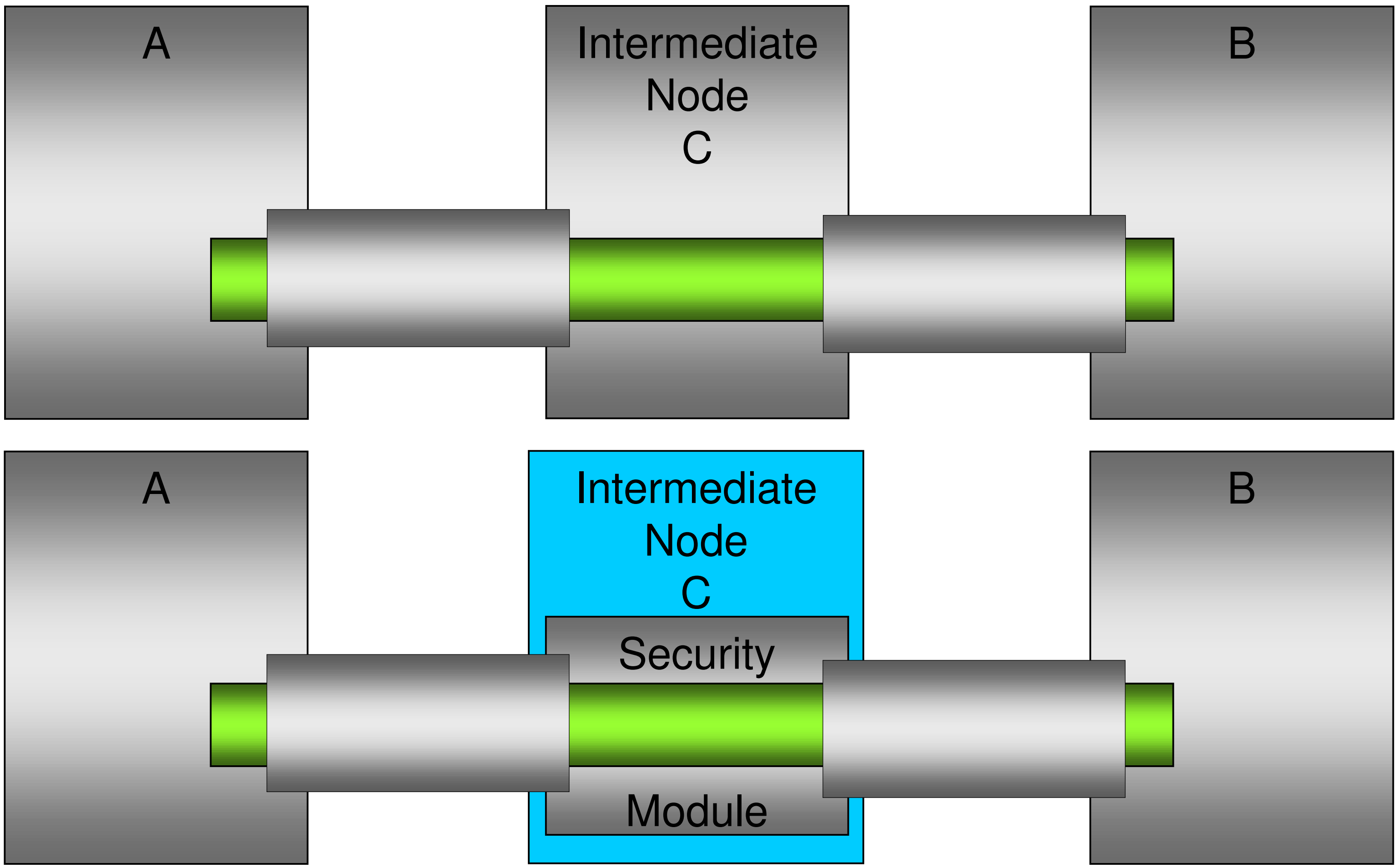


Case 3 of E2E Security

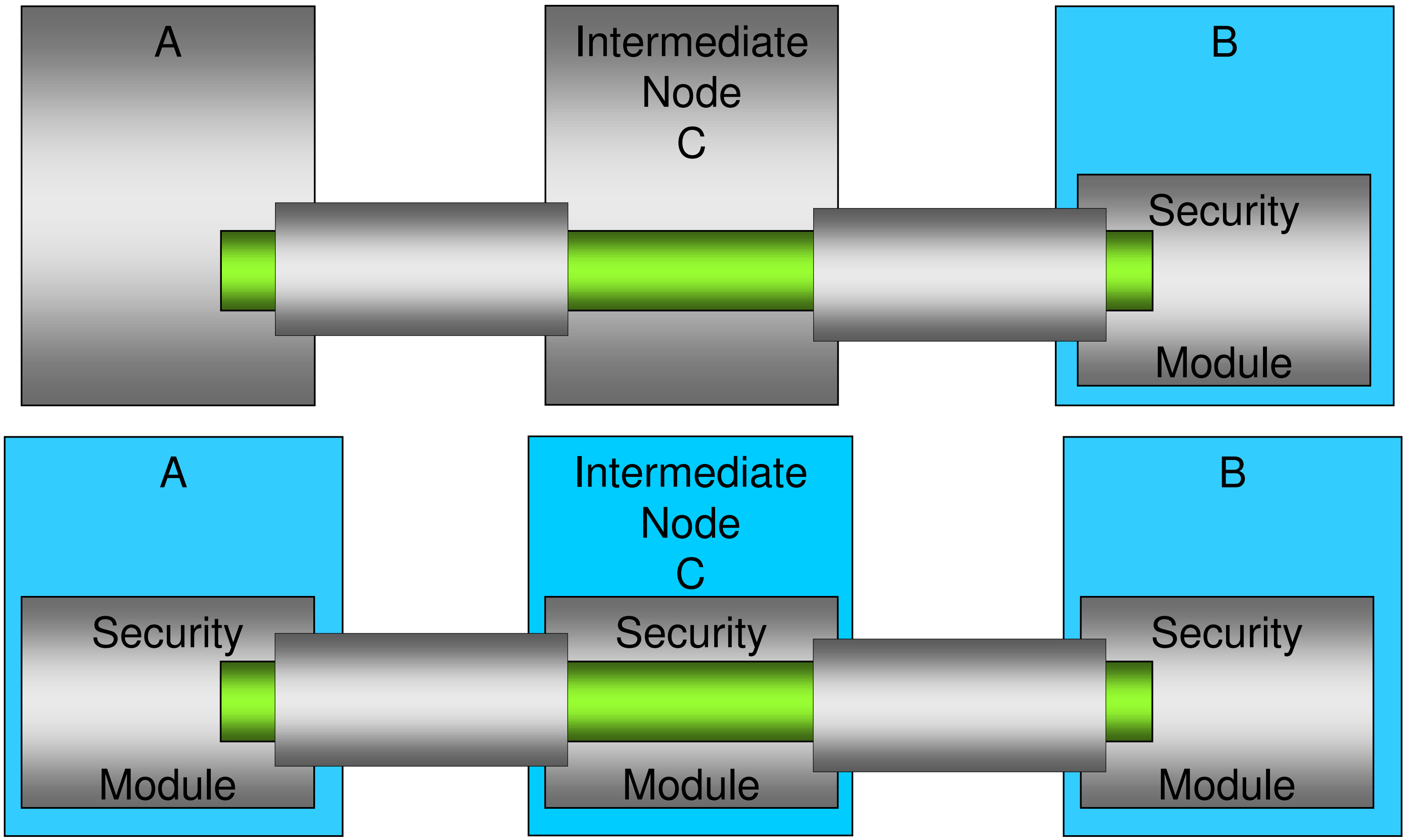
- A communicates with C, C communicates with B
  - Only a trusted part of C has access to application data, called security module
- Crypto mechanisms can be different in A-C and in C-B segments



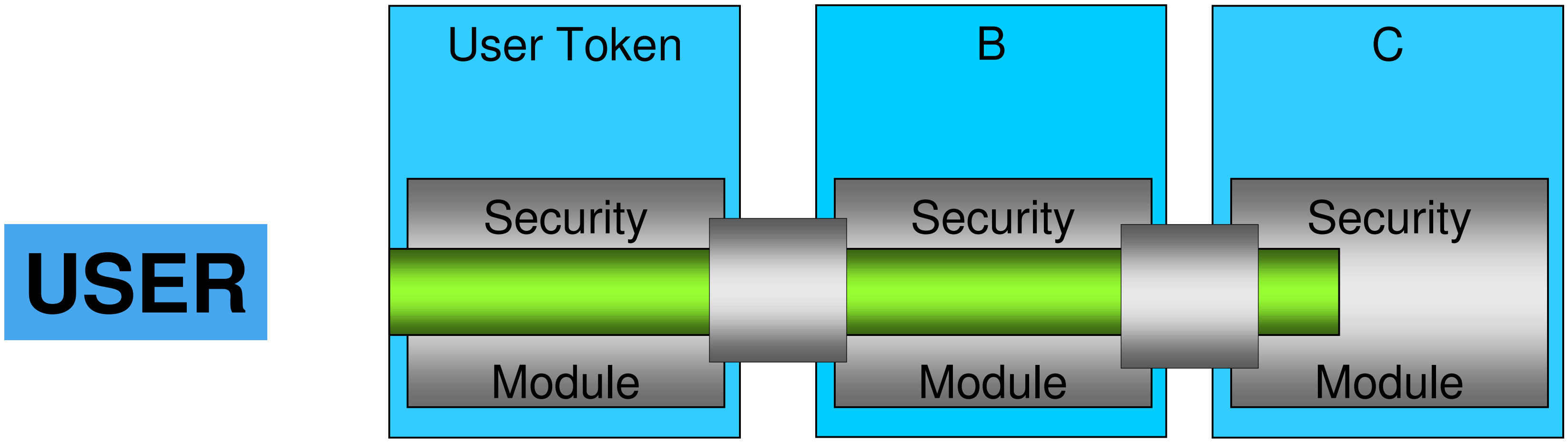
Endpoints Might also be Secure



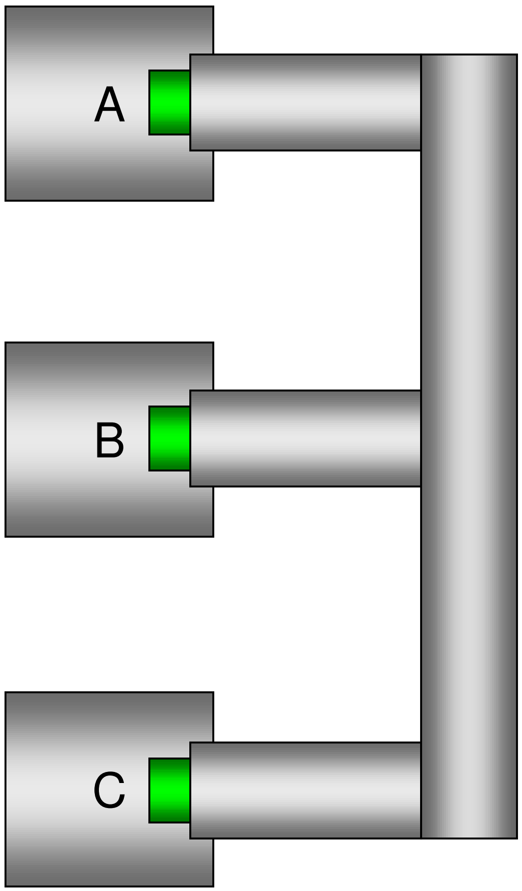
Endpoints Must also be Secure



Including Input-Outputs



Peer-to-peer Security == Circle of Trust?



Secure Execution Environment
for V2V and V2I Communication

Vehicle ⟷ Vehicle   Vehicle ⟷ RSE   RSE ⟷ Control center

- Vehicle TCU trusted part?
  - TCU includes a trusted part
    - e.g. non trusted part is PC centric part of TCU, trusted part is CALM implementation part of TCU
- RSE trusted part?
- Control Center and Service Center trusted part?

Security Module – High Level

API

Can be used for
- Applications
- Secure Communications

Could be sealed in a tamper evident enclosure, e.g., Integrity-protected log file or database, hardware enclosure,…

Functionality

- Authenticate data
- Verify authenticated data
- Decrypt encrypted data
- Encrypt plaintext data
- Generate key pair
- Generate secret key
- Play key agreement protocol
- Generate random data
- Compare Local vs. Reference time
- Convert security mechanism

Implementation relies on

Inner Kernel with security features

Device/user certificate(s)
Trusted (CA) certificates
Device/user/system data
Session data (keys, logs)

Cryptographic kernel
- Signing primitives and keys
- Decryption primitives and keys
- Secret master keys
- Decrypt and re-encrypt (optional)

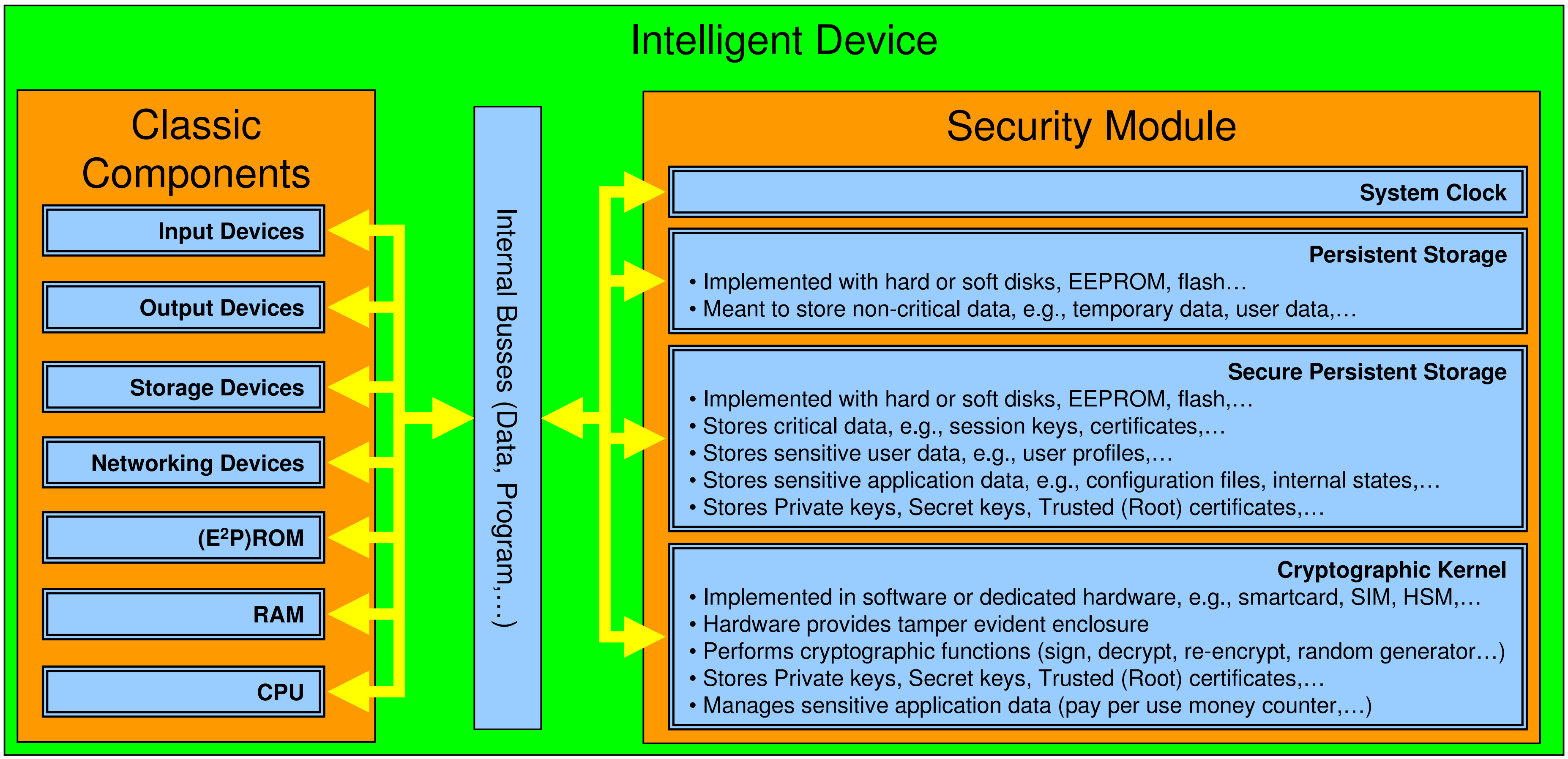
Example

- Secure messaging:
  - Key agreement phase:
    - Ping pong messages
      - sendPing, receivePing, preparePong, receivePong
  - After key agreement:
    - byte[] dataToShip=prepareForSend(SecurityLevel, Data, SessionAlias)
    - byte[] receivedData=processIncoming(incomingData, SecurityContext)
  - Receiver engine:
    - Endless loop:
      - Message incomingData=receiveData()
      - Case(incomingData.type){
        - Ping: { Message pong=preparePong(ping);send(incomingData); }
        - Pong: { processPong(incomingData); }
        - Insecure: {…}
        - Confidential: {…}
        - Authenticated: {…}
        - Secure: {…}
      - }

Example

- Secure data storage:
  - storeData(SecurityLevel, Data, Alias, OverwriteIfExists)
    - SecurityLevel: plaintext, encrypted, integrity protected, confidential
    - Alias: (unique) reference to retrieve the data later on
    - OverwriteIfExists: self-explanatory boolean
  - byte[] fetchedData=retrieveData(Alias, SecurityContext) throws noSuchAlias
    - SecurityContext: if the Alias refers to data which should not be made available given the current SecurityLevel, it will not successfully be fetched

Common Device Components



Examples of Security Modules

- Hardware security module (most expensive)
  - Used for high-bandwidth communications, secure payments, etc.
- Smartcard, SecurID token, SIM card
  - Commonly used to provide strong user, service and device authentication
- Trusted platform module (TPM)
  - By default built into many new laptops and desktops
  - Lacks features necessary for GST, e.g., authentication of users, application data, etc.
  - TPM only authenticates the device
- Software key store (cheapest)
  - Cryptography-related data is stored in persistent memory (flash, magnetic,…)
  - Non-secure microcontroller operates on this data

Security Modules Form Factors

- Dedicated coprocessor
  - Pluggable (e.g., reader for smartcard/memory card, SIM lock for SIM card, socket for chip
  - Fixed, e.g., soldered secure microprocessor (similar to smartcard, TPM)
- Using the main processor for functionality, coprocessor for important processes (e.g., payable services)
- Using the main processor only
  - Software-only security
  - Privileged mode (e.g. Arm with TrustZone)

Example of Use for V2V and V2I Communication

- Car A wishes to exchange data with Car B
- Car A steps
  - Use the Security Module of A to authenticate data
  - Send the authenticated data to B
- Car B steps:
  - Use the Security Module of B to validate the authenticity of received data
  - If authentication is OK, B processes data

MILS

- Multiple Independent Levels of Security
- Security Architecture for Middleware
- Based on military classification of security levels classifications
  - TS: top secret
  - S: secret
  - C: classified
  - U: unclassified

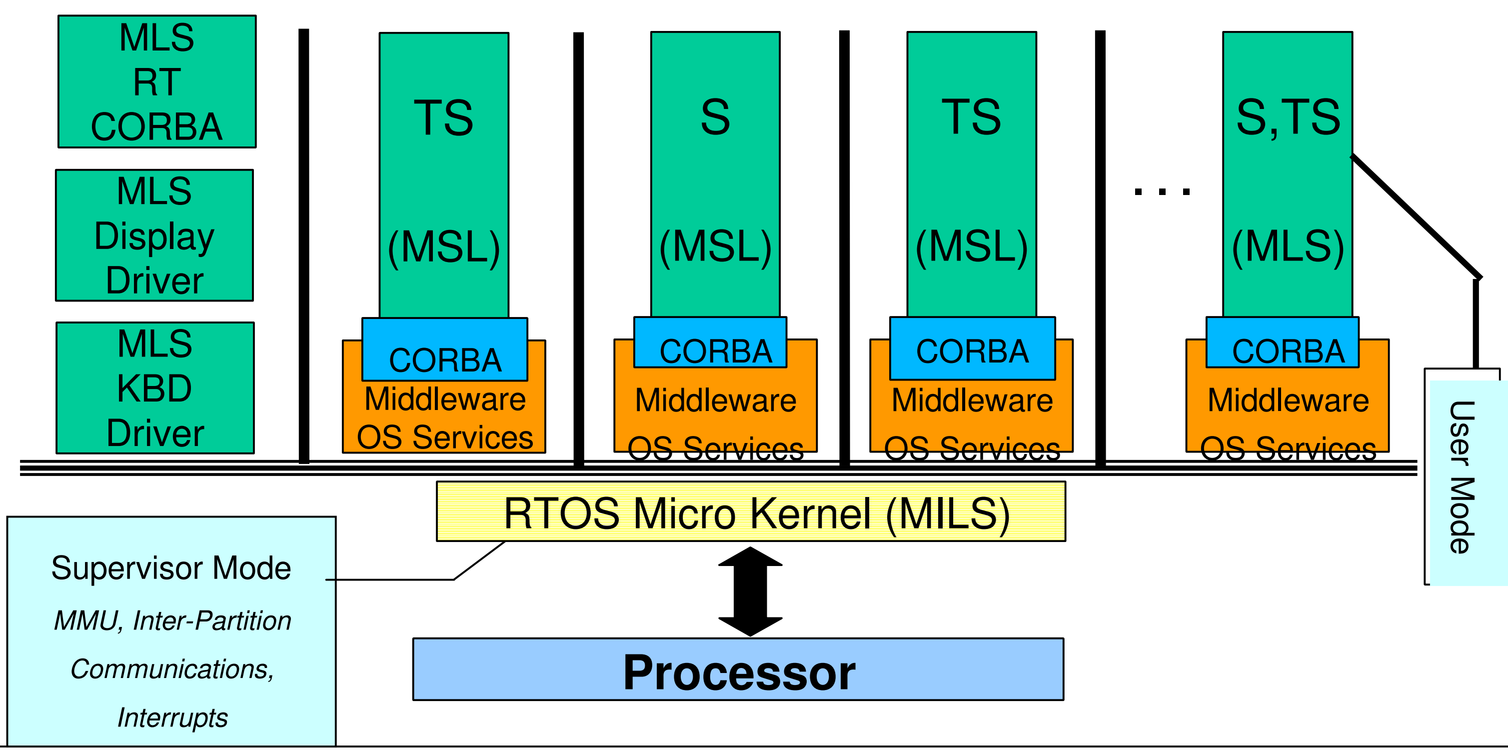
High Assurance MILS Architecture

MILS - Multiple Independent Levels of Security

MSL - Multi Single Level

MLS - Multi Level Secure

Application Partitions



MILS

- 3 independent layers:
  - Partitioning kernel
    - Offers process separation, in space and time
    - Small footprint => easier certification
  - MILS middleware layer
  - MILS application layer
    - Implement own security policies using provided protected mechanisms

Protection Mechanisms

- Data isolation
  - Information in the state of one partition must not be accessible to other partitions
- Information flow
  - Only authorized communication between partitions can occur
- Periods processing
  - Sanitization of shared resources between context switches
- Damage limitation
  - Failure in one partition is contained, so it does not affect other partitions

Independent Components for V2V and V2I Communication

- Which kind of independence
- Which kind of protection
  - In a typical microcontroller, a thread have access to the whole memory
    - can read sensitive data
    - can modify sensitive data

Secure Vehicle Communication

Thanks

Antonio Kung

antonio.kung@trialog.com