Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

Towards Smarter Vehicular Network Interfaces for The Software-Defined Vehicles Era

Shreejith Shanker
Reconfigurable Computing Systems Lab,
Trinity College Dublin

SCCM 2025, Leiden, NL

Software-Defined Vehicle

An SDV is a modern automobile in which ***core functions*** and ***features*** are ***controlled, updated and enhanced*** through ***software*** rather than fixed hardware systems[1]

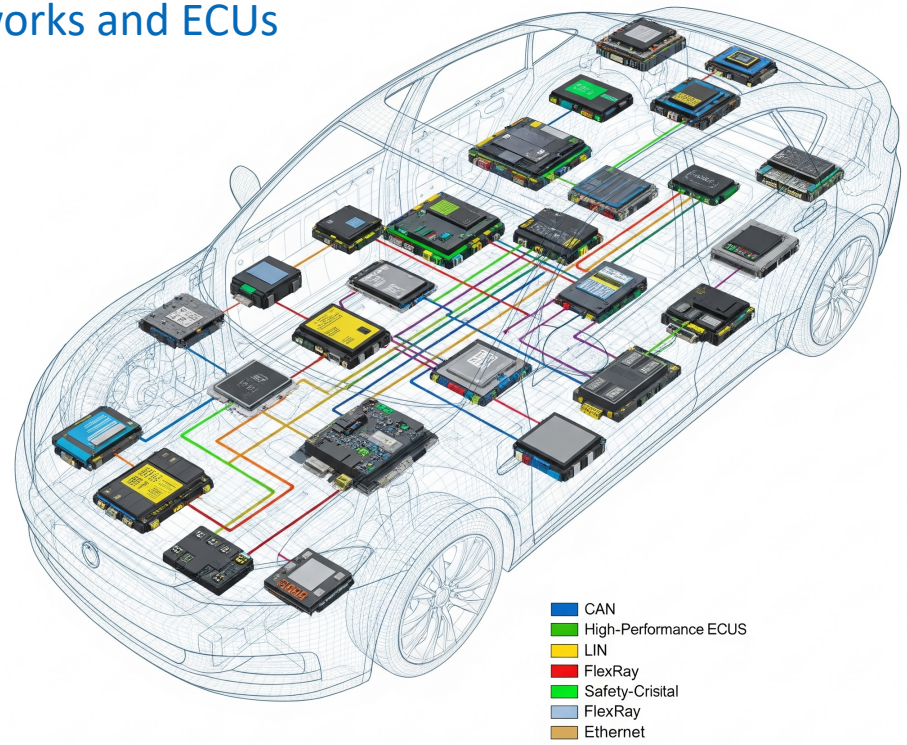
- The next evolution of Connected Autonomous Vehicles (CAVs)
  - Consolidated computing architecture
  - Scalable, modular software stack
  - Virtualisation at functional and network layers
  - Cybersecurity by design
    - From secure boot, to encrypted communication, real-time monitoring and intrusion detection systems

[1] https://www.ibm.com/think/topics/software-defined-vehicle

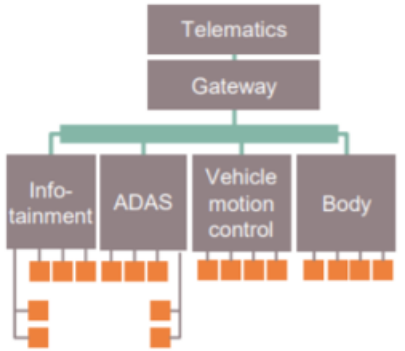
Image from tttech-auto

Security by design

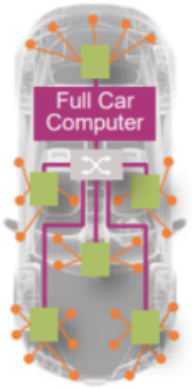
Vehicular networks and ECUs



Domain Architecture



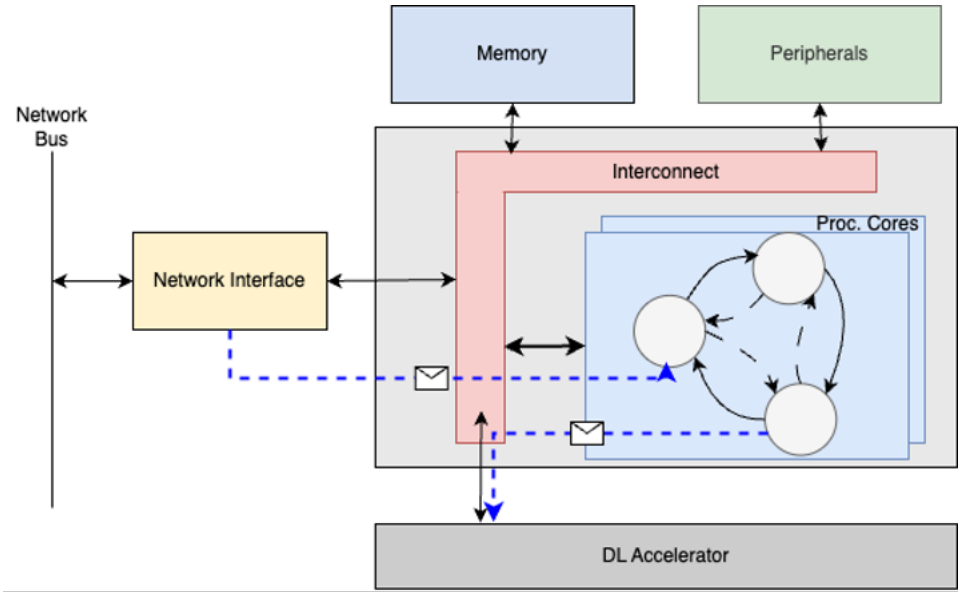
Zonal Architecture



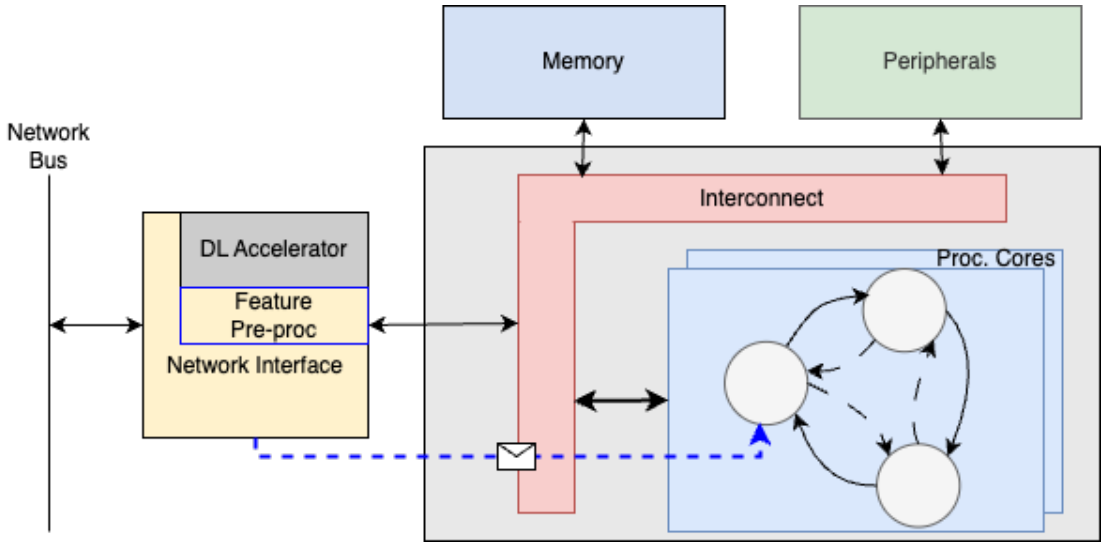
- Functions are mapped to electronic computing units (ECUs), networked by different protocols
- Based on the architecture, computing capabilities at ECUs and network protocol capabilities can vary
- Software-driven security approaches alone are not effective

Security by design

How to deal with Network security?



Coupled accelerator: Blue path shows message flow



Smart Network Interface: Virtualised security module at Network interface

Traditional security schemes for network-driven messages rely on coupled accelerators

- Data movement incurs significant latency
- Acceleration can be containerised, but software still aware and involved
- How to virtualise security extensions and make them transparent? — Smart Network Interfaces

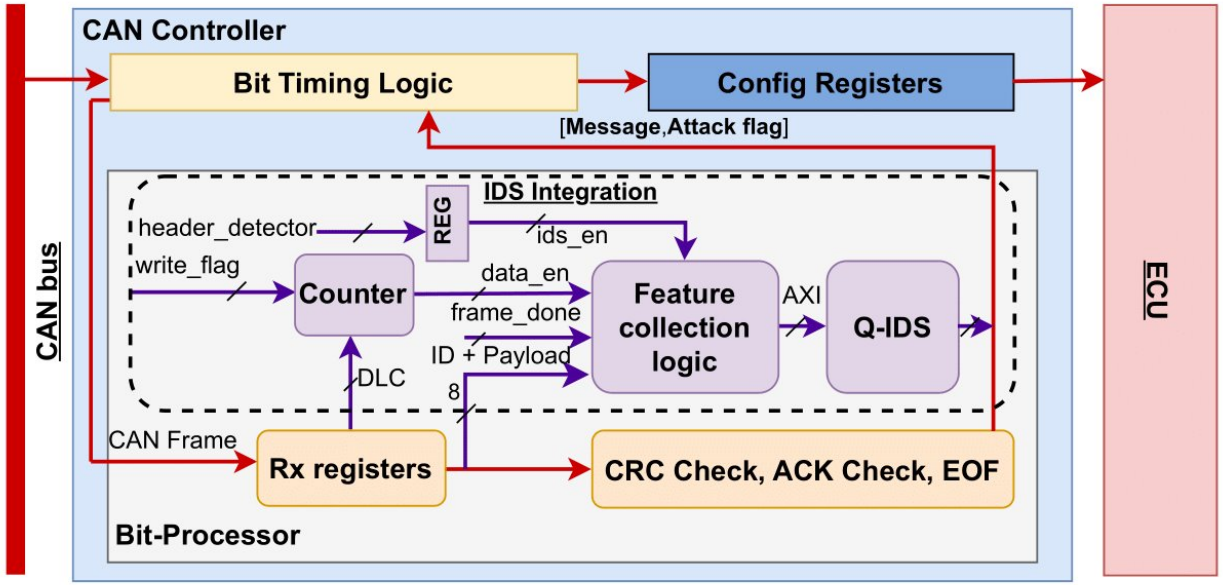
SecCAN: An Extended CAN interface with embedded IDS

Virtualising IDS capabilities

Integrate IDS engine into a CAN controller's internal data path

- Protocol implemented in three blocks
  - Bit timing logic (physical signalling)
  - Bit-processor (bit-packing and signalling)
  - Register interface (configuration/status registers, ECU interface)

- Taps into the bit-packing logic
  - Taps the bits as received and decoded
  - Features extracted at byte level & fed to the IDS engine
  - Overlaps with bit reception timing, and completes within the window — completely transparent to the network & ECU

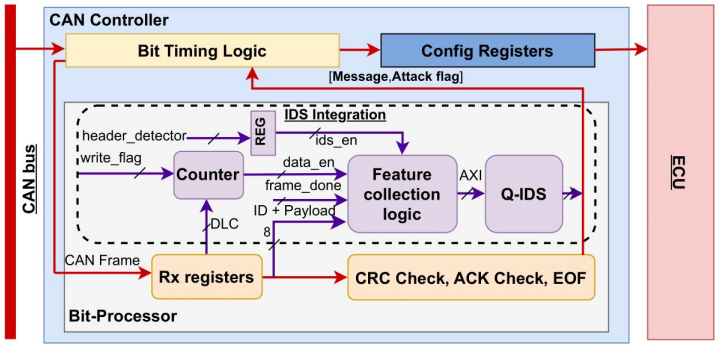


SecCAN: Embedding IDS engine within the Bit-processing logic of CAN controller

S. Khandelwal and S. Shreejith, "SecCAN: An Extended CAN Controller With Embedded Intrusion Detection," in *IEEE Embedded Systems Letters*, 2025

SecCAN: An Extended CAN interface with embedded IDS

Virtualising IDS capabilities



Integrate IDS engine into a CAN controller's internal data path

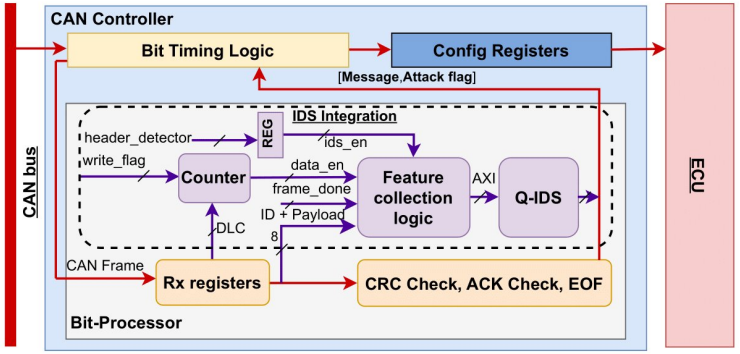
- Custom quantised MLP-based IDS engine
  - 4-bit weights/activations
  - Multi-class threat identification - DoS, Fuzzing, Flooding, Malfunction attacks
  - Brevitas + FINN flow for optimised dataflow model
  - Classification result appended as a flag to the message for upstream ECU

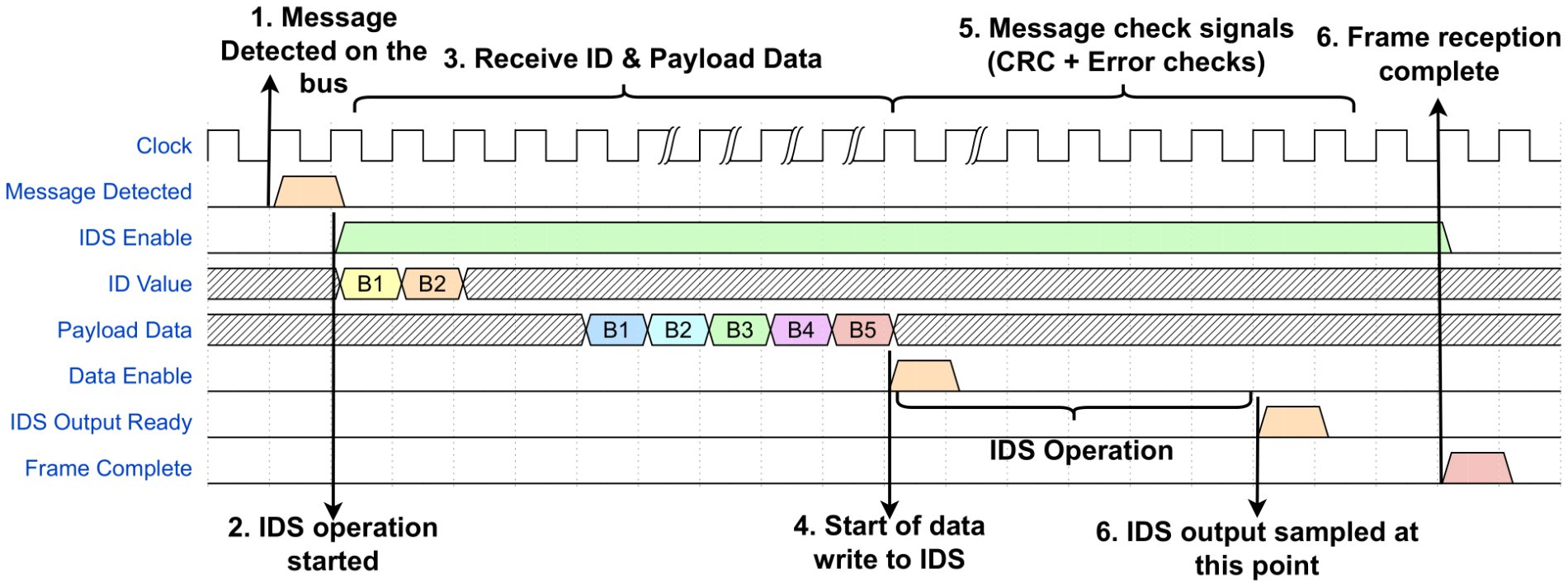
- Key requirements of IDS engine:
  - End-to-end latency < CAN message window ~37μs
  - Operate at native controller frequency ~ 16 MHz

S. Khandelwal and S. Shreejith, "SecCAN: An Extended CAN Controller With Embedded Intrusion Detection," in *IEEE Embedded Systems Letters*, 2025

SecCAN: An Extended CAN interface with embedded IDS

Virtualising IDS capabilities

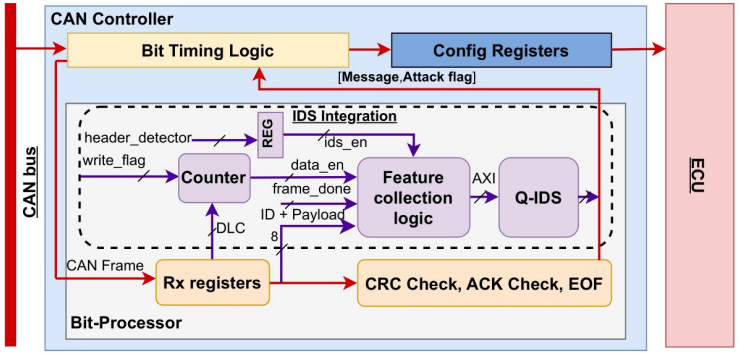




SecCAN: Low-level timing diagram capturing the IDS operating window for a 5-byte CAN message

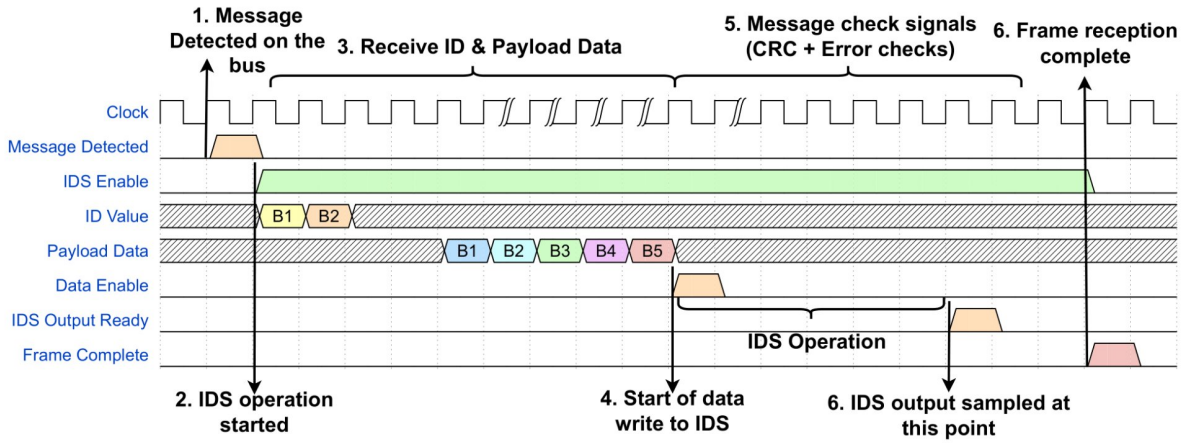
SecCAN: An Extended CAN interface with embedded IDS

Virtualising IDS capabilities



Key takeaways:

- Line-rate, real-time message identification and tagging at full speed CAN (with IDS running at native CAN frequency)
- Further exploit CAN signalling to indicate faults to downstream ECUs
- Custom quantised IDS accelerator is extremely energy efficient with high multi-class detection accuracy
  - 73 μJ per inference, 99.99% inference accuracy across multiple datasets and attacks*
- IDS is completely virtualised and transparent
- Openly available: https://github.com/RCSL-TCD/SecCAN



*Open Car Hacking dataset and Survival Analysis dataset

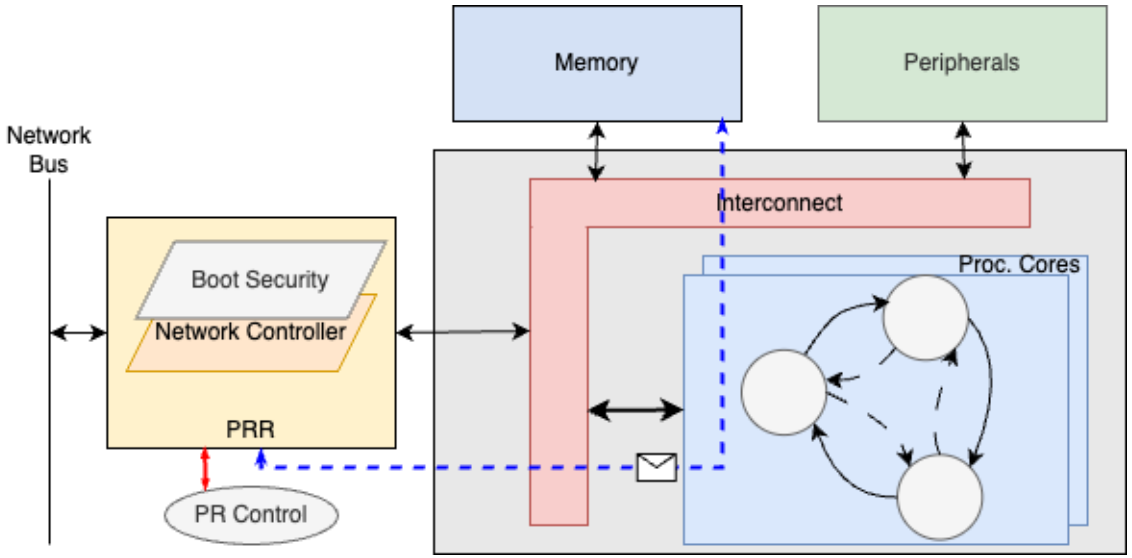
Security by design: Smart Network Interfaces

Trust-based network activation

ECUs network messages are exploited for most critical attacks

- Malicious code injected to ECU to introduce wider attacks

- Trusted execution modes can be expanded to prevent Network Interface activation
  - Partial reconfiguration can be exploited to completely deactivate the interface in case of malicious code on FPGA-based ECUs



Utilising PR for Network Interface activation based on Boot-time/ run-time code/system integrity checks

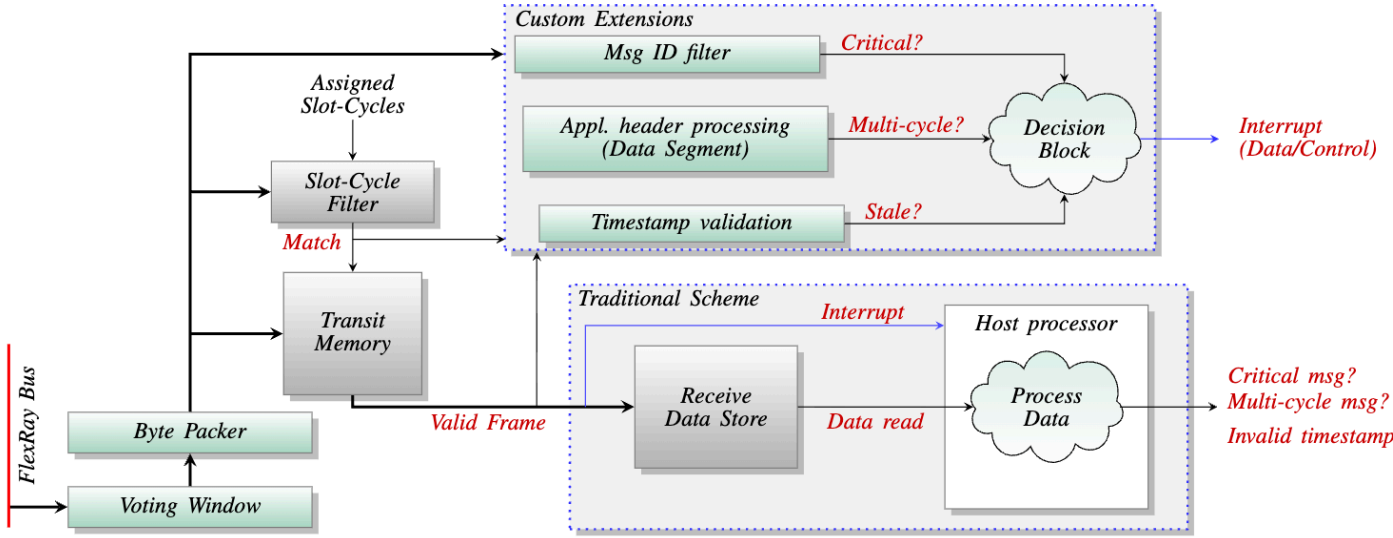
Shanker Shreejith and Suhaib A. Fahmy. “Security aware network controllers for next generation automotive embedded systems” DAC 2015

Security by design: Smart Network Interfaces

Payload extensions - Message time-stamping

Datapath extensions can enable augmented features to alleviate some attacks

- Payload could be augmented with a synchronised timestamp managed at the Network interface
- Extensions can also be used for network level capabilities such as message segmentation, (re-) ordering and packing



Data-layer extensions within FlexRay interface to enable numerous data-layer capabilities with zero ECU overhead

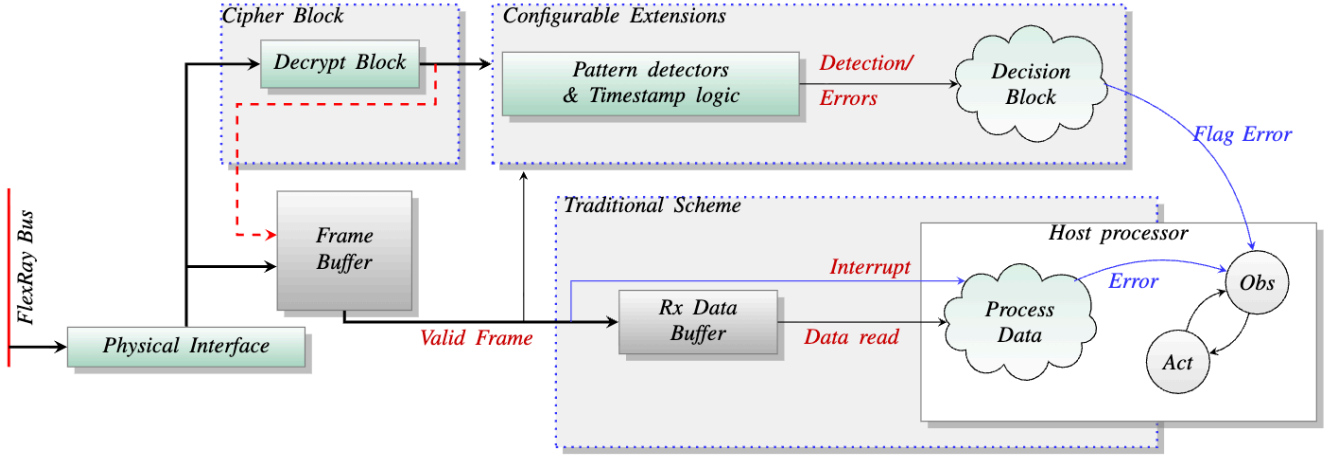
Shanker Shreejith and Suhaib A. Fahmy. “Extensible FlexRay Communication Controller for FPGA-Based Automotive Systems”, TVT 2015

Security by design: Smart Network Interfaces

Trust-based network access control

With time-triggered networks, network access can be restricted through protocol header obfuscation

- FlexRay example - protocol headers are used to extract timing information for network integration
- Obfuscating the header using simple symmetric cipher (PRESENT in this case) will prevent unauthorised nodes to integrating to the network
- Such offloading can only be done through network controller data path extensions



Exploiting protocol-level features to restrict network access to only authorised devices. In this case, the FlexRay protocol headers are obfuscated using lightweight cipher blocks, preventing unauthorised nodes from integrating on the FlexRay bus. The data layer extension within the Controller implements the entire flow.

Shanker Shreejith and Suhaib A. Fahmy. “Security aware network controllers for next generation automotive embedded systems” DAC 2015

Shanker Shreejith and Suhaib A. Fahmy. “"Smart Network Interfaces for Advanced Automotive Applications” IEEE Micro, 2018

Concluding thoughts

- Network controllers for future SDVs will need to be designed/adapted with security as a corner stone
  - Using software integration and offload will not scale to the data rates, autonomous capabilities and safety requirements

- Extending data path of protocol controllers can open up exciting avenues
  - Allows exploitation of network and physical - level signatures for true cross-layer security

- Combining reconfigurable platforms capabilities such as partial reconfigurability with trusted execution frameworks can offer physical isolation in case of malicious conditions

- Virtualising these capabilities at the network interface will allow easy portability for software-driven solutions and stacks - complying with AUTOSAR and other standards