AUTOSAR Security Modules

Current Status
# Agenda

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>AUTOSAR</td>
</tr>
<tr>
<td>2.</td>
<td>CAL &amp; CSM</td>
</tr>
<tr>
<td>3.</td>
<td>SecOC</td>
</tr>
</tbody>
</table>
Introduction

- **Automotive Open System Architecture**
  - Software for electronic control units (ECU)

Software architecture
Introduction

Software component (SWC) / Application
- Implementation of functionality of ECU
- Runs on microcontroller
- Sends & receives data to and from other ECUs (in network)
Introduction

Run time environment (**RTE**)  
- Provides interface to basic software (**BSW**)
System services (**SYS**) and libraries (**LIB**)  
- Cryptographic modules

Operating system (**OS**)  
Complex device drivers (**CDD**)
Introduction

Communication modules (COM)
- send & receive data on automotive bus systems
  - Controller Area Network (CAN)
  - Local Interconnect Network (LIN)
  - FlexRay
  - Ethernet
  - ...

![Diagram of SWC/Application, RTE, SYS, COM, LIB, CDD, Microcontroller]
Introduction

Microcontroller abstraction layer (**MCAL**)

- BSW & SWC independent of microcontroller
Motivation for security modules in AUTOSAR

New security challenges
- Automotive software plays central role in car innovations
- Car connectivity will provide an essential part for value-added features

Car security – strict and secure access control to...
- ... the car and its parts (ECU)
- ... sensitive car data (odometer, motor characteristic)
- ... passenger’s data (GPS)
- ... intellectual property of the OEM
AUTOSAR security modules

**CAL & CSM**
- Basic cryptographic primitives for BSW and application

**SecOC**
- Authenticated communication seamlessly integrated into the AUTOSAR communication stack
1. AUTOSAR

2. CAL & CSM

3. SecOC
Introduction

Crypto Abstraction Library – CAL
- BSW, CDD or SWC use CAL by inclusion
- Memory allocated by caller
  - Enables re-entrance

Crypto Primitive Library – CPL
- SW implementation of cryptographic primitives

Crypto Service Manager – CSM
- SWC use CSM through RTE
- BSW/CDD use CSM by inclusion
- Asynchronous operation possible
  - Callback indicates application

Crypto library module – CRY
- Implementation of cryptographic primitives
  - Usage of SW or crypto HW possible
Abstract definition of cryptographic services
- No definition for a concrete cryptographic algorithm

### Supported Cryptographic Services

<table>
<thead>
<tr>
<th>Basic Cryptography</th>
<th>Key Management</th>
<th>Miscellaneous</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Hash</td>
<td>- Key derivation function (KDF)</td>
<td>- Compression/ Decompression</td>
</tr>
<tr>
<td>- Message authentication code (MAC)</td>
<td>- Key generation, update*, export, import</td>
<td>- Checksum</td>
</tr>
<tr>
<td></td>
<td>- Generation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Verification</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Random number generation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Encryption/ Decryption</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Symmetric</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Asymmetric</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Signatures</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Key exchange protocols</td>
<td></td>
</tr>
</tbody>
</table>

*Csm only
Individual configuration of each required service
- Set of distinct configurations
- Specific implementation for each service configuration
Cryptographic Service Configuration

- AsymEncryptService
  - AsymEncrypt_1
    - RSA2048
  - AsymEncrypt_2
    - RSA4096

- SymEncryptService
  - SymEncrypt_1
    - AES
  - SymEncrypt_2
    - Serpent

- HashService

- Individual configuration of each required service
- Set of distinct configurations
- Specific implementation for each service configuration
- Implementations may change in future
Cryptographic Service Configuration

- AsymEncryptService
  - AsymEncrypt_1
    - ECC256
  - AsymEncrypt_2
    - ECC512

- SymEncryptService
  - SymEncrypt_1
    - AES
  - SymEncrypt_2
    - Serpent

- HashService

- Individual configuration of each required service
- Set of distinct configurations
- Specific implementation for each service configuration
- Implementations may change in future
- API compatibility not ensured
**CAL & CSM**

**General Usage**

**Streaming services**

- Indefinite long data stream

  - Start
  - Update
  - Update
  - ... 
  - Update
  - Finish

- Initialization with Start function (e.g. Csm_SymEncryptStart)
- Update function (e.g. Csm_SymEncryptUpdate)
- Finish function (e.g. Csm_SymEncryptFinish)

**Non-streaming services**

- Example: Csm_GenerateRandom
Hardware-based Security

- CSM services use cryptographic hardware or software implementation
Secure Hardware Extension (SHE)

- On-chip extension to microcontroller
- Memory for secure storage of (cryptographic) data
- Hardware extension for cryptographic primitives
- Specified by Hersteller Initiative Software (HIS)
SHE - Performance

- AES ECB Encryption: SHE vs. Software library

Measured on a Freescale MPC5646C (w/ CSE), MICROSAR Stack with CSM and SHE driver with the Vector 'AUTOSAR Measurement and Debugging (AMD) Runtime Measurement (Rtm)' Tool.

1 Block = 16 bytes
Agenda

1. AUTOSAR
2. CAL & CSM
3. SecOC
SecOC

Introduction

- SecOC is parallel to PDUR
  - PDUR routes PDUs
  - PDU is a message on a bus
SecOC

Introduction

- SecOC is parallel to PDUR
- PDUs are routed through SecOC
- PDU & authentication sent & received through IF or TP modules
  - COM module combines data into PDUs
  - IF modules send & receive atomic messages
  - TP modules manage messages longer than atomic messages
SecOC

Introduction

- SecOC is parallel to PDUR
- PDUs are routed through SecOC
- PDU & authentication sent & received through IF or TP modules
- SecOC uses Cal or Csm
- RTE-interface
- Authentication: MAC or signature
SecOC sends & receives secured PDUs
- Secured PDUs are protected against
  - Manipulation
  - Random errors
  - Replays
Sending a secured PDU

- DataID assigned to secured PDU
- Authentic PDU
Sending a secured PDU

- Freshness value
  - Monotonic counter to prevent replay attacks
- Implementation
  - Timestamp
  - Counter

SecOC

ECU 1

DataID 1
PDU 1
Fresh. Value

Secured PDU
Sending a secured PDU

- DataID, PDU, freshness value form input to MAC generator
- Symmetric key required for MAC generation
- SecOC may use CMAC to benefit from SHE
Sending a secured PDU

- PDU, truncated freshness value, truncated MAC form secured PDU
Sending a secured PDU

- NIST Special Publication 800-38B (CMAC)
  - Truncated MAC length \( \geq 64 \) bits

**Truncated MAC length must be thoroughly chosen dependent on network attributes and security requirements**
Reception of a secured PDU

- Authentic PDU is parsed
- DataID must be identical for sender and receiver
- Truncated freshness value is synchronized to form verification freshness value
Reception of a secured PDU

- Verification freshness value > stored freshness value (replay attacks)
  - If not: Increment MSBs of verification freshness value
- Synchronization between sender and receiver

```
Ver. Fresh. + 0..01 0..0 = Ver. Fresh.
```
Reception of a secured PDU

- DataID, PDU, verification freshness form input to MAC generator
- Symmetric key must be identical for sender and receiver
- MSBs of calculated MAC are compared to truncated MAC
  - If successful, PDU is forwarded
  - If not, PDU is dropped
System Configuration

ECU 1
- PDU 1
- PDU 2
- PDU 3

ECU 2
- PDU 1
- PDU 2

ECU 3
- PDU 1
- PDU 3

BUS
Assignment of DataIDs to the to-be-secured PDUs
System Configuration

- Specification of the layout of the secured PDU
Assignment of keys to the secured PDUs
- Initial keying
- Re-keying
System configuration

ECU 1
- DataID 1
  - PDU 1
    - Fresh. Value
    - MAC
  - DataID 2
    - PDU 2
      - Fresh. Value
      - MAC
- PDU 3

ECU 2
- DataID 1
  - PDU 1
    - Fresh. Value
    - MAC
- DataID 2
  - PDU 2
    - Fresh. Value
    - MAC

ECU 3
- DataID 1
  - PDU 1
    - Fresh. Value
    - MAC
- PDU 3

ECU1_Extract
ECU2_Extract
ECU3_Extract
For more information about Vector and our products please visit

www.vector.com

Author:
Philipp Werner, Armin Happel, Ralf Fritz, Steffen Keul
Vector Informatik GmbH