Vector Cyber Security Solution
vHSM – Optimized and Flexible Software for Hardware Security Modules

Vector Congress - Cybersecurity, 2019-10-09
Agenda

- Overview
  - Fundamentals
  - MICROSAR.HSM - Features and Architecture
  - MICROSAR.HSM - Architecture
  - MICROSAR.HSM - Application of vHSM in Featured Use Cases
  - MICROSAR.HSM - Integration and Workflow
  - Summary
Cyber Security in Automotive

Cyber security gains increasing importance in automotive industry due to highly connected vehicles and accessible customer interfaces. Vehicles become a part of the internet of things.

Therefore, highly complex algorithms have to be executed efficiently in an isolated portion within the ECUs. A secure storage for cryptographic secrets is needed as well.

To meet these needs, a hardware security module including its software is used to provide necessary performance and isolation with appropriate small footprints.
Layered Security Concept – Defense in Depth

Overview

Secure External Communication
- Secure communication to services outside the vehicle via TLS
- Intrusion detection mechanisms
- Firewalls
- Key Infrastructure / Vehicle PKI
- Synchronized secure time

Secure Gateways
- Message authentication codes (MAC)
- Freshness to ensure integrity of messages
- Encryption to ensure confidentiality of messages

Secure In-Vehicle Communication
- Secure key storage
- Secure boot and secure flash
- Crypto algorithm library
- HW trust anchor (HTA)
  - E.g. Hardware Security Modules (HSM)

Secure Platform
- Crypto algorithm library
- HW trust anchor (HTA)
  - E.g. Hardware Security Modules (HSM)
Security Mechanisms allocated in Example Architecture

Overview

- Firewall
- Key Infrastructure
- Crypto Primitives
- Monitoring / Logging
- Hypervisor
- Intrusion Detection / Prevention
- Secure On Board Com.
- Secure Off Board Com.
- Download Manager
- Secure Flash/Boot
- Secure Synchronized Time Manager
Agenda

Overview

▶ Fundamentals
MICROSAR.HSM - Features and Architecture
MICROSAR.HSM - Architecture
MICROSAR.HSM - Application of vHSM in Featured Use Cases
MICROSAR.HSM - Integration and Workflow
Summary
Pure Software Approach

- Executing cryptographic algorithms in software
- No hardware support / acceleration possible
- No isolated secure core or memory available
- May occupy many CPU resources (CPU time)
- Code size and speed highly depends on the microcontroller
General Considerations for Hardware Trust Anchors (HTA)

- **Hardware Trust Anchor (HTA)**
  - Provision of secure storage for secrets (e.g. Keys)
  - Possibilities to update secrets
  - Provision of accelerators for cryptographic algorithms
  - Provision of basic functions for integrity checks at start-up (a.k.a. “Secure Boot”)

- Different standardized **feature sets** for HTAs
  - Secure Hardware Extension (SHE)
  - Hardware Security Module (HSM)
  - Trusted Platform Module (TPM)
  - …

- Different **brand names** for HTA by different HW suppliers
  - Infineon: Aurix HSM / SHE+ driver
  - Reneas: Intelligent Cryptographic Unit (ICU)
  - Freescale: Crypto Service Engine (CSE)
  - ARM: Trust Zone
  - …
Based on Crypto Peripheral

- Accelerating cryptographic algorithms in crypto peripheral
- Hardware support / acceleration possible
- No isolated secure core or memory available
- Secrets and application data in same memory
- No updates of crypto hardware are possible
Fundamentals

Hardware Security Module (HSM)

- **HSM design objectives**
  - Harden ECUs against SW and selected HW attacks
  - Provide HW acceleration for crypto functions

- **EVITA HSM profiles**
  - **HSM full:**
    - Support strong authentication (e.g. via RSA, ECC)
    - Support complex block ciphers
    - High performance, updateable
  - **HSM medium:**
    - Secure ECU 2 ECU communication
    - Updateable
  - **HSM small:**
    - Secure critical sensors / actuators
    - Simple block ciphers
    - Low cost modules, not updateable
Fundamentals

**Hardware Security Module (HSM)**

- Software for the secure core needed to provide any cryptographic service
  - Secure Hardware Extension (SHE) emulation
  - Secure storage
  - Cryptographic basic functions and algorithms
  - ...

- Supported features highly depend on
  - Available hardware resources
  - Software implementation

- Performance highly depends on
  - Hardware architecture (e.g. clock frequency)
  - HSM Software architecture
  - Interfacing with application cores
Overview

Fundamentals

- **MICROSAR.HSM - Features and Architecture**
  - MICROSAR.HSM - Architecture
  - MICROSAR.HSM - Application of vHSM in Featured Use Cases
  - MICROSAR.HSM - Integration and Workflow

Summary
Core Features

- (HW accelerated*) Cryptographic basic functions and algorithms
- Streaming of crypto jobs
- Isolated secure key storage
- Secure error log
- Development error handling and debugging
- Customization and extension support

Featured use cases:

- Secure software download and code signing support
- Secure communication support
  - On board (e.g. SecOC)
  - Off board (e.g. V2G)
- Secure boot support

*Depending on available accelerators in HW
Basic functions and symmetric crypto algorithms
- Hash (e.g. SHA-1, SHA-256,...)
- Random number generation
- MAC Generate and Verify
  - CMAC
  - HMAC
- Cipher
  - AES in the modes ECB, CBC, GCM
- Key derivation functions (KDF) and key exchange
  - KDF in counter mode
  - Concatenation KDF
  - Key exchange protocol EC-DHE
- Asymmetric crypto algorithms\(^1\)
  - RSA (Generation and Verification of Signatures)
  - RSA (Encryption and Decryption)
  - ECDSA (Generation and Verification of Signatures)
  - EdDSA
  - RSASSA-PSS

---

This feature requires vHSM Add-On Asymmetric Crypto
Key Storage

- Any Keys can be passed in plaintext and stored inside the HSM
  - Includes symmetric/asymmetric keys and certificates
  - Any other security relevant data (e.g. mileage) can be stored as well in secure storage!

- Key Installation of symmetric keys according SHE 1.1
  - Support of counter handling
  - UID can be read out

- Extensive configuration options:
  - Free choice if a key is stored in flash or only in RAM
  - Keys can be stored redundantly and reset safe
  - Keys are pre-loaded / cached on startup to avoid loading with every use
  - Number of keys only limited by available RAM and data flash of platform
  - Keys can be locked until secure boot has finished
  - Keys can be configured as write once
  - Keys can be persisted immediately or delayed to be able to persist multiple keys at once
Secure Boot

- Can be split up in several jobs/phases
- One job is started automatically, e.g. for bootloader verification
- Bootloader can start one or more further jobs
  - Sequential secure boot: Jobs can be executed as a sequence so that application is not started before finishing secure boot jobs
  - Parallel secure boot: Jobs can be executed in parallel to application start to enable fast boot times

- Configurable secure boot sanctions are:
  - System reset
  - Only logging
  - Do not unlock keys (read/write/use)

- Depending on platform support:
  - Configuration of two init vectors for A/B flashing of application
    - Vector is changed automatically after a failed secure boot
    - Vectors can be set from the application if needed

Example: Signatures for code signing and secure boot update
vHSM Error Log, DETs and Debugging

- vHSM provides an error log, which can be used to log error events which occur on the HSM.
- Errors can be written to secure data flash and read out by application.
- The maximum number of log entries is configurable.

- Reporting of Det Errors on vHSM
  - Can be treated as error and logged
  - Can be forwarded to application
  - Det of vHSM works as proxy of the host DET
    - Errors are forwarded and reported on the host
  - Due to source code delivery, vHSM is fully debuggable
Agenda

Overview
Fundamentals
MICROSAR.HSM - Features and Architecture

MICROSAR.HSM - Architecture

MICROSAR.HSM - Application of vHSM in Featured Use Cases
MICROSAR.HSM - Integration and Workflow

Summary
Internal Architecture of the vHSM

- The vHSM firmware consists of vHSM dedicated and Vector standard modules
- Modular and configurable architecture
  - In analogy to AUTOSAR Crypto stack
    - CryIf channels
    - CryDrv driver objects
    - Crypto primitives
    - Keys, key types and key elements
- Communication between host and vHSM is done by IPC / shared memory
- Synchronous and asynchronous job processing supported
- Notification about job completion by polling or interrupts on host side
Internal Architecture of the vHSM

- AUTOSAR 4.3 BSW
  - SWC / Application
    - RTE
    - SYs
    - COM
    - OS
  - MCAL
    - Crypto(vHSM)
- Microcontroller
  - Flash Bootloader
  - RAM
  - HSM Channels
- vHSM
  - vHSM Job Processor
  - vHSM KeyM
  - Crypto Software Library
  - Custom Crypto Job
  - Crypto Hardware Accelerator
  - Secure Boot and Update Support

Application core subsystem
Shared memory
HSM subsystem
Agenda

Overview
Fundamentals
MICROSAR.HSM - Features and Architecture
MICROSAR.HSM - Architecture

► MICROSAH.R.HSM - Application of vHSM in Featured Use Cases
MICROSAR.HSM - Integration and Workflow
Summary
Use Case: Secure OnBoard Communication (SecOC)

SecOC is used to authenticate the communication between two ECU’s to prevent a third party from intervening or pretending to be the correct communication partner.

The SecOC module offers the following functions:

- Transmission of authenticated and integrity-protected I-PDU's.
- Authentication with a Message Authentication Code (MAC). The actual generation and validation of the Message Authentication Code is performed by vHSM.
- Prevention of replay attacks via freshness value manager (FVM).

vHSM supports SecOC in:

- Key storage
- MAC generation / verification
- Performance optimization via
  - Configurable key caching
- Speed up of crypto processing
- Isolation of host and secure domains

⚠️ Advantage: MAC generation and verification on isolated, secure core with secure memory for key storage and higher performance due to hardware acceleration.
**Use Case: Secure Boot 1/2**

- Prevent execution of tampered ECU software by means of a chain of trust
- Integrity check is performed at ECU startup
- Each software unit involved in the boot process validates the integrity of the subsequent software unit, forming a chain of trust
- Validation can be done via checking signature / MAC
- Keys and MAC must be stored in a secure area

**vHSM supports Secure Boot in:**

- Secure key and MAC storage
- Signature / MAC verification
- 1..n secure boot slots with
  - Configurable keys
  - Configurable sanctions
- Configurable performance improvement options
  - Sequential or parallel secure boot
- Isolation of host and secure domains
Use Case: Secure Boot 2/2

- Can be split up in several jobs/phases
- One Job is started automatically, e.g. for bootloader verification
- Bootloader can start one or more further jobs
  - Sequential secure boot: Jobs can be executed as a sequence so that application is not started before finishing secure boot jobs
  - Parallel secure boot: Jobs can be executed in parallel to application start to enable fast boot times
- Sanction for failed secure boot configurable (e.g. System reset, Only logging, Do not unlock keys, HSM halt)
- Support of two init vectors for A/B flashing of application

- **Advantage:** Secure Key and MAC storage as well as signature and MAC verification on isolated, secure core with better security and faster startup due to hardware acceleration
Agenda

Overview
Fundamentals
MICROSAR.HSM - Features and Architecture
MICROSAR.HSM - Architecture
MICROSAR.HSM - Application of vHSM in Featured Use Cases

▸ MICROSAR.HSM - Integration and Workflow

Summary
Integration Scenarios

**vHSM + MICROSAR**

- **Advantages**
  - Better tool support → DaVinci Configurator provides consistency between MSR and vHSM configurations
  - Combination already in use
  - Better technical support possible
  - Easier debugging

**vHSM + 3rd Party AUTOSAR stack**

- **Compatibility to AUTOSAR 4.3 compliant BSW**
  - Usage and interaction via crypto driver Crypto(vHSM)
Comprehensive configuration with DaVinci Configurator Pro

- DaVinci Configurator Pro provides pre-config file to sync vHSM configuration with MICROSAR configuration including:
  - Available primitives
  - Available and configured keys

![Diagram showing the integration and workflow between Developer 1 and Developer 2 using DaVinci Configurator Pro for vHSM configuration.]

- Generate pre-config file as output
- Input as Additional Definitions
- Available primitives
- Available and configured Keys

*arxml
Agenda

Overview
Fundamentals
MICROSAR.HSM - Features and Architecture
MICROSAR.HSM - Architecture
MICROSAR.HSM - Application of vHSM in Featured Use Cases
MICROSAR.HSM - Integration and Workflow

Summary
Summary

MICROSAR.HSM: vHSM – What it actually is and does.

**MICROSAR.HSM – Functionality**

- Cryptographical services on secure core with isolated memory
- Large library of crypto algorithms
  - Crypto Basic functions (hash, random numbers)
  - Message authentication code (HMAC, CMAC)
  - Symmetric and asymmetric crypto ciphers
  - Supporting signature generation and verification
- Providing secure key storage, update and handling
- Providing secure boot protocol
- Supporting hardware acceleration for better performance
- Providing possibilities to update vHsm software

**MICROSAR.HSM – Configuration**

- Modular architecture with extensive configuration space
- Adaptable HSM firmware to match use case requirements and footprint
- Comprehensive configuration tool DaVinci Configurator

**MICROSAR.HSM – Example Use Cases**

- Secure boot in combination with flash bootloader
- Secure software update and code signing
- Secure OnBoard Communication (SecOC)

**vHSM** is an efficient and flexible firmware for hardware security modules that is adaptable to your use case in order to improve cybersecurity.
For more information about Vector and our products please visit

www.vector.com

Learst, Jason
Vector North America