F2010-E-035

SECURE AUTOMOTIVE ON-BOARD ELECTRONICS NETWORK ARCHITECTURE

[1]Apvrille, Ludovic, [2]El Khayari, Rachid, [2]Henniger, Olaf[*], [3]Roudier, Yves, [3]Schweppe, Hendrik, [4]Seudié, Hervé, [5]Weyl, Benjamin, [6]Wolf, Marko

[1]Telecom ParisTech, France,
[2]Fraunhofer Institute for Secure Information Technology, Germany,
[3]EURECOM, France,
[4]Robert Bosch GmbH, Germany,
[5]BMW Group Research and Technology, Germany,
[6]escrypt GmbH, Germany

KEYWORDS – automotive on-board network; security architecture; hardware security module, embedded systems, vehicle communication systems

ABSTRACT – This paper introduces hardware and software components for secure automotive on-board networks providing the basis for the protection of external vehicle communication. It is based on work done within the European research project EVITA (http://evita-project.org). It provides a framework that covers cross-layer security, targeting platform integrity, communication channels, access control and intrusion detection and management. We present a modular hardware/software co-design: Hardware security modules (HSM) provide means to protect the platform integrity, to ensure the integrity and confidentiality of key material and to enhance cryptographic operations, thereby protecting critical assets of the architecture. In order to provide cost-effective hardware solutions, three different variants of HSMs have been specified: The full HSM for protecting external communication interfaces, the medium HSM for protecting the on-board communication between electronic control units (ECUs), and the light HSM for protecting the on-board communication with sensors and actuators. Application specific interfaces are provided by the software framework that interacts with the HSMs. High-level design considerations, such as least privilege design and separation principles have been followed throughout the work. We provide an outlook on deployment scenarios.

MOTIVATION

Automotive applications based on vehicle-to-vehicle and vehicle-to-infrastructure (V2X) communications have been identified as a means for decreasing the number of fatal traffic accidents in the future and for intelligent traffic management. However, malicious attacks on embedded IT systems and networks implementing those functionalities and malicious encroachments on messages transiting between vehicles and infrastructure, such as sending fake messages and spoofing over the wireless network, may have a severe impact. Thus, the on-board network needs to provide appropriate security measures in order to protect against malicious messages. Sensitive in-vehicle data must be trustable and protected from modification.

A list of potential attacks and related security requirements (1) served as starting point for designing the secure on-board architecture. The attacks have been classified according to their risk level in order to choose adequate levels of protection against them. We derived in-car security mechanisms out of the security requirements (2). Security functions are partitioned between software and hardware with cost and security levels as major criteria. The secure storage of secret keys together with secure and trustworthy communication among in-car electronic components lays the foundation for sound data exchange between vehicles or infrastructure services. Therefore, we place the “root of trust” in hardware security modules realized as an on-chip extension to automotive ECUs. This enables the reliable enforcement of application-specific security properties such as authenticity, confidentiality, or freshness as well as dependable access control.

The rest of this paper is organised as follows: After giving an overview of related work in the field of V2X and on-board communications and summarizing the security requirements from (1), we present our security architecture. The paper concludes with a deployment overview and a summary and outlook.

RELATED WORK

The past decade has seen a tremendous growth in the vehicular communication domain, yet no comprehensive security architecture solution has been defined that covers all aspects of on-board communication (data protection, secure communication, secure and tamper proof execution platform for applications). On the other hand, several projects, namely GST (3), C2C-CC (4), IEEE Wave (5) and SeVeCOM (6) have been concerned with inter-vehicular communication and have come up with security architectures for protecting vehicle-to-vehicle and vehicle-to-infrastructure communications. These proposals essentially aim at communication-specific security requirements in a host-based security architecture style, as attackers are assumed to be within a network where no security perimeter can be defined (ad-hoc communication).

For instance, (7) presents the C2C communication consortium’s solution integrating previous approaches (8)(15)(16)(17) for secure vehicular communications. These proposals consider the car mostly as a single entity, communicating with other cars using secure protocols. In particular, this architecture relies on a complex security back-end infrastructure (including authorities, notably implementing PKIs, e.g., for pseudonym and identity management). This is necessary for protecting the identity of a car yet making it possible to manage its identifiers when required. However, no specific execution platform requirements are put forward by these proposals, except for the need to protect node identifiers: All proposals mention that data such as vehicular registration and cryptographic material should be stored in a tamper-resistant manner. Unfortunately, this requirement is not accompanied by any further analysis of the particular threats to data integrity and authentication within the vehicle that might guide the design of such storage. The EVITA project fills this gap, by providing an in-vehicle platform with secure storage, a trusted execution environment, among other qualities such as cryptographic processors in hardware and a security framework for communication, authentication and authorization as well as intrusion detection and management in software.

SECURITY REQUIREMENTS

At the highest level, the security objectives that are covered are:

- to prevent unauthorized manipulations of vehicular on-board communication networks,
- to prevent unauthorized modifications of vehicle applications especially regarding safety and m-commerce applications,
- to protect privacy of vehicle drivers,
- to protect intellectual property of vehicle manufacturers and suppliers,
- while maintaining the operational performance of applications and security services.

The EVITA project has inferred the following set of security requirements and related functional requirements in order to satisfy the stated security objectives (1):

- **Integrity/authenticity of e-safety related data**: Actions depending on critical information should be decided based on assurances about integrity and authenticity in terms of origin, content, and time. Forgery of, tampering with, or replay of such information should at least be detectable.
- **Integrity/authenticity of ECU/firmware installation/configuration**: Any replacement or addition of an ECU and/or its firmware or configuration to the vehicle shall be authentic in terms of origin, content, and time. In particular, the upload of new security algorithms, security credentials, or authorizations should be protected.
- **Secure execution environment**: Compromises to ECUs should not result in system wide attacks, primarily with regard to e-safety applications. Successful ECU attacks should have limited consequences on separate and/or more trusted zones of the platform.
- **Vehicular access control**: Access to vehicular data and functions should be controlled (e.g. for diagnosis, resources, etc.)
- **Trusted on-board platform**: The integrity and authenticity of operated software shall be ensured. An altered platform might be prevented from running in an untrusted configuration (e.g. via comparison with a trusted reference) if so required.
- **Secure in-vehicle data storage**: Applications should be able to use functionality in order to ensure access control to as well as the integrity, freshness and confidentiality of data stored within a vehicle, especially for personal information and security credentials.
- **Confidentiality of certain on-board and external communication**: The confidentiality of existing software/firmware as well as updates and security credentials shall be ensured. Some applications might additionally require that part of the traffic they receive or send internally or externally should remain confidential.
- **Privacy**: A privacy policy shall be enforceable on personal data stored within a vehicle or contained in messages sent from a vehicle to the outside. For example, some applications should limit the ability to link sent messages.
- **Interference of security functionality**: The operation of security services must not negatively affect the availability of bus systems, CPUs, RAM, or of the radio medium.

The above stated requirements are constraints that arose from security concerns. How the constraints are satisfied is described in the following sections.

SECURITY ARCHITECTURE

Partitioning in Hardware and Software

Partitioning a system in software and hardware means finding an optimum solution for executing a set of application functions – including functions related to security – over various candidate hardware architectures (11). Those architectures are typically built upon hardware nodes such as CPUs, buses, memories, hardware accelerators and sensors/actuators. Criteria for selecting an optimal architecture are typically the cost of hardware elements, the power consumption, the execution time or throughput of the system, and the ability to change the functionality of the system without incurring heavy costs.

In software/hardware partitioning, a typical methodology relies on the description of application functions, of candidate architectures, of the mapping of application functions onto the architectures and then the simulation of the mapping. The analysis of simulation traces resulting from the simulation generally leads to the selection of the “best” architecture.

We have used a model that included hardware security functions, software security functions, security requirements, attack probabilities and risk as well as preliminary protocol definitions. We used software called TTool (12) for this purpose, which allows us to answer questions regarding deployment and design decisions and tailor the software and hardware security architecture for specific vehicle types and models according to specific security requirements.

EVITA Hardware Security Modules

Figure 1 presents the general architecture of an automotive HSM. With the security requirements of (1) as basis, in (2) a cost-effective HSM design has been chosen that provides enough security and flexibility. In this architecture the HSM resides in the same chip as the application CPU core.

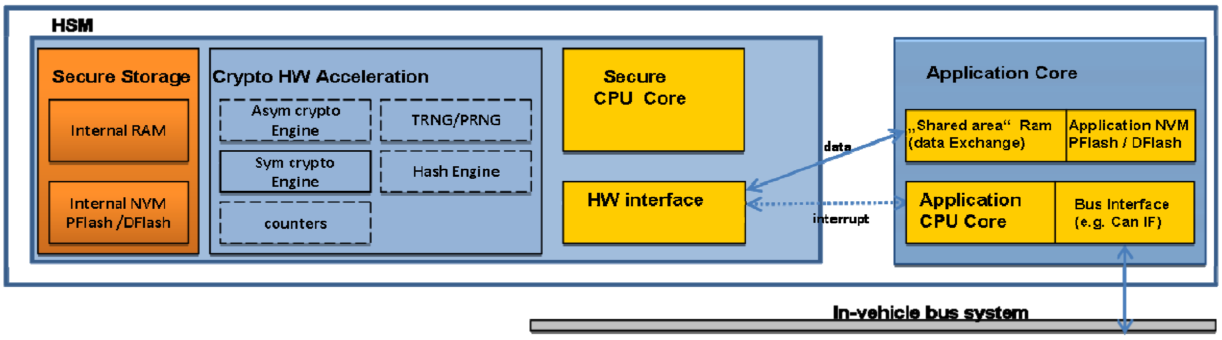

Figure 1: General structure of an automotive HSM

The components of the HSM are divided into mandatory and optional components because, depending on the use case, different security requirements have to be fulfilled. The optional components are represented in Figure 1 with dashed lines. In order to provide cost-effective hardware solutions, we specify three different EVITA HSM variants that meet different security needs (2):

- **Full EVITA HSM**: This HSM focuses on protecting the in-vehicle domain against security vulnerabilities of V2X communications. This requires creating and verifying electronic signatures. In order to satisfy the performance requirements, a very efficient asymmetric cryptographic engine is needed. The full HSM provides the maximum level of functionality, security, and performance of all the different HSM variants. It further aims to provide a security lifetime of at least 20 years, which means ECRYPT II Level 7 "Long-term protection" (13) and/or NIST 2030+ (14).
- **Medium EVITA HSM**: This HSM focuses on securing the on-board communication. Except for the asymmetric cryptographic building block and a little less performing CPU (e.g. 25 MHz vs. 100 MHz), the medium HSM resembles the full HSM. The medium HSM has no asymmetric cryptographic building block in hardware; however, it is able to perform some non-time-critical asymmetric cryptographic operations in software, e.g. for the establishment of shared secrets. As for efficiency and cost reasons virtually all internal communication protection is based on symmetric cryptographic algorithms, leaving out the asymmetric crypto engine is reasonable to save costs and hardware size.
- **Light EVITA HSM**: This HSM focuses on securing the interaction between secured ECUs and sensors and actuators. It is only required to contain a symmetric cryptographic engine and the corresponding functionally shortened hardware interface. Hence, the light HSM is able to fulfill the strict cost and efficiency requirements (e.g., regarding message size, timings, protocol limitations or processor consumption) that are typical for sensors

and actuators. The necessary shared secrets can be established in different ways, e.g. by pre-configuration during manufacturing, by self-initialization (e.g., based on physically unclonable functions) or even by running a key establishment protocol in software at the attached application processor.

Table 1 presents the components of the different HSM variants. All technical details such as RAM size, clock frequencies etc. are current estimates and subject to change.

	Full EVITA HSM	**Medium EVITA HSM**	**Light EVITA HSM**
Internal RAM	✓ (e.g. 64 kByte)	✓ (e.g. 64 kByte)	optional
Internal NVM (Non-volatile memory)	✓ (e.g. 512 kByte)	✓ (e.g. 512 kByte)	optional
Symmetric Cryptographic Engine (e.g. AES-128 CCM, GCM f/AE)	✓	✓	✓
Asymmetric Cryptographic Engine (e.g. ECC-256-GF(p) NIST FIPS 186-2 prime field)	✓		
Hash engine (e.g. Whirlpool)	✓		
Counters	✓ (e.g. 16 × 64-bit monotonic counter)	✓ (e.g. 16 × 64-bit monotonic counter)	optional
Random Number Generator	✓ (e.g. AES-PRNG with TRNG seed)	✓ (e.g. AES-PRNG with TRNG seed)	optional
Secure CPU (e.g. ARM Cortex-M3 32 bit, 50–250 MHz)	✓	✓	
Hardware Interface	✓	✓	✓

Table 1: Components of different EVITA HSMs (technical details may change)

Hardware Interface

The EVITA hardware security module provides an asynchronous (i.e., non-blocking), almost completely multi-session-capable (i.e., interruptible), and partly also multi-threading-capable hardware interface. It is compliant to the Secure Hardware Extension (SHE) specification proposed by the automotive HIS consortium (10).

The hardware interface covers the invocation specification of all cryptographic hardware security blocks, higher-level security functionality (e.g., secure boot, secure time-stamping) and all necessary security management functionality (e.g., device administration, key creation, key import/export). It furthermore defines all hardware interpreted data structures and direct interdependencies, such as key hierarchies, replay protection counters or basic access logic.

A distinctive feature of the EVITA hardware interface is the inherent fine-grained application-specific authorization in contrast to, e.g. general-purpose authorizations for using internal secret(s). Thus the same secret can have different authorizations for different usage purpose (so called use-flags). A symmetric key, for instance, can have different authorizations for using the key for encryptions, decryption, MAC generations, or MAC verifications. These authorizations in turn, can be based either directly on the passwords or indirectly based on the individual ECU platform configuration similar to the Trusted Computing (TC) approach (cf. TC authenticated boot) or could be even a combination of both (i.e., configuration and password). Moreover, each individual use flag can have its individual export constraints. Hence, a cipher interface is defined as follows (here simplified):

```
EVITA_RETURNCODE cipher_init(
  in algorithm_identifier,         // reference to hardware cipher algorithm
  in cipher_mode{encrypt|decrypt}, // indicate use-flag ‘encrypt’ or ‘decrypt’
  in operation_mode{CBC|GCM|..},   // indicate cipher mode of operation
  in padding_scheme{none|bit|..},  // indicate padding scheme (if required)
  in initialization_vector,        // initialization vector (if required)
  in key_handle,                   // reference to internal key to be used
  in key_authorization_size,       // key authorization for requested use-flag
 out max_chunk_size,               // maximum size of a chunk on process()
 out session_handle );             // multi-session authentication handle

EVITA_RETURNCODE cipher_process(
  in session_handle,               // session reference from init()
  in input_data_size,              // input data to become encrypted or decrypted
 out output_data );                // output data being encrypted or decrypted

EVITA_RETURNCODE cipher_finish(
  in session_handle,               // release session handle
 out final_output );               // last output chunk (e.g., from padding scheme)
```

The cipher becomes initialized first by defining all relevant operation parameters and providing (directly or indirectly) the necessary key usage authorization if corresponding authorizations have been set during the creation of the referred secret key. The initialization then returns all relevant processing parameters (e.g., maximum chunk size) and a session identifier for interruption and parallel processing of the corresponding cipher operation(s). After all data chunks have been encrypted or decrypted, the cipher session is closed and the session handle will be released by invoking the finalization command.

Security Software

We employ a flexible and modular security framework that allows distributed deployment (2). As use cases may demand a subset of security requirements, a modular architecture has the advantage of reusing security components where possible. The security framework offers the following functionalities.

- **Access control:** Management and enforcement of policies,
- **Authentication services:** Depending on requirements,
- **Secure communication:** Establishment of authenticated and/or confidential communication channels,
- **Intrusion detection:** Modules provide means to detect and manage intrusions at different abstractions levels.

These modules can be configured in a policy-oriented way. In order to enable a maximum level of flexibility and adaptability of the security functionality, all security modules can be configured (statically or even dynamically) by accepting and enforcing individual security policies such as the following:

- Authorization policies that specify to which extent a certain entity is allowed to access and use a specific resource under a certain condition, e.g. for:

  - file, network and input/output device access,
  - external and internal communications (i.e., filter policies that define, for instance, which contents, sources, protocols, or services can be communicated between certain communication endpoints),
  - usage of connected peripherals

- Privacy policies that define to which level data and information needs to be hidden against third parties or at least need to be anonymized for a certain level of privacy,
- Authentication policies that define what level of authentication (e.g., password, smartcard) is required for corresponding role authorizations,
- Intrusion detection policies that define certain attack scenarios and attack heuristics,
- Intrusion response policies that define what kind of countermeasures has to executed against a certain kind of detected attack.

Modules of the software security framework can be deployed to individual ECUs, depending on the role, requirements and performance of an ECU.

DEPLOYMENT

We outline a deployment of hardware security modules based on an e-safety scenario that is part of the use cases defined in (9). The scenario is called “Active Brake” and uses inter-vehicle communication. In Figure 2, the ECUs within the receiving vehicle taking part in communication for this scenario are shown. Data is analyzed for authenticity and authorization at the communication control unit (CCU) and is then distributed to the individual domains and ECUs where actions are taken. Authenticity of data (i.e., whether they originate at the CCU) is checked at each receiving ECU. For this purpose, these ECUs are equipped with HSMs. Depending on the performance and runtime environment of the ECUs, medium or light HSMs are used (for the sensors only the light HSM is feasible, due to constraints of the platform).

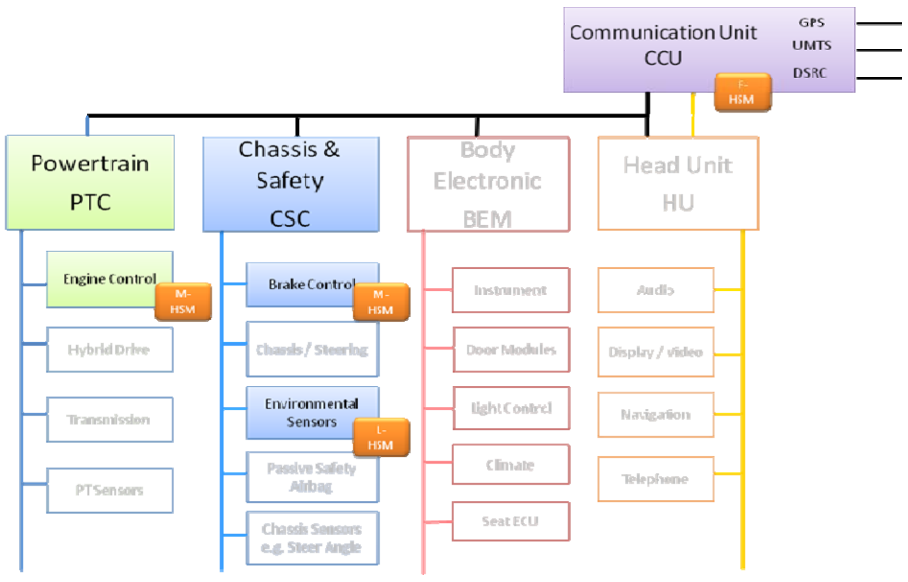

Figure 2: Deployment example of HSMs

SUMMARY AND OUTLOOK

We have designed and specified a modular security hardware and software concept that is going to be used for automotive on-board networks and the corresponding electronic control units. The different variants of security hardware make a deployment cost-effective. They reflect security requirements of different domains and electronic components.

The EVITA project will define secure protocols between the on-board components. Formal methods and simulations will be used to verify the protocols before their implementation and integration into automotive operating systems and runtime environments. The HSMs will be prototyped using FPGA logic boards and automotive microcontrollers from Infineon. Finally, hardware and software will be deployed into a demonstrator vehicle.

ACKNOWLEDGMENTS

This work has been part of the collaborative project “EVITA – E-safety vehicle intrusion protected applications” (http://evita-project.org), co-funded by the European Commission under the Seventh Framework Programme. The authors thank the other project participants from BMW Group Research and Technology, Continental Teves AG & Co. oHG, escrypt GmbH, EURECOM, the Fraunhofer Institutes for Secure Information Technology and for Systems and Innovation Research, Fujitsu Services AB, Infineon Technologies AG, Institut Telecom, Katholieke Universiteit Leuven, MIRA Ltd, Robert Bosch GmbH and Trialog for the fruitful cooperation.

REFERENCES

(1) A. Ruddle, D. Ward, B. Weyl, S. Idrees, Y. Roudier, M. Friedewald, T. Leimbach, A. Fuchs, S. Gürgens, O. Henniger, R. Rieke, M. Ritscher, H. Broberg, L. Apvrille, R. Pacalet, G. Pedroza, “Security requirements for automotive on-board networks based on dark-side scenarios”, EVITA Deliverable D2.3, 2009.

(2) B. Weyl, M. Wolf, F. Zweers, T. Gendrullis, M.S. Idrees, Y. Roudier, H. Schweppe, H. Platzdasch, R. El Khayari, O. Henniger, D. Scheuermann, A. Fuchs, L. Apvrille, G. Pedroza, H. Seudié, J. Shokrollahi, A. Keil, “Secure on-board architecture specification”, EVITA Deliverable D3.2, 2010.

(3) GST, Global Systems for Telematics, EU FP6 project, http://www.gst-forum.org/

(4) C2C-CC, Car2Car Communication Consortium, http://www.car-to-car.org/

(5) IEEE WAVE, Wireless Access in Vehicular Environments, IEEE standard 1609.2.

(6) SeVeCOM, Secure Vehicle Communication EU FP6 project, http://www.sevecom.org/

(7) M. Gerlach, A. Festag, T. Leinmüller, G. Goldacker and C. Harsch, “Security Architecture for a Vehicular Communication”. In International Workshop on Intelligent Transportation (WIT), 2005.

(8) T. Kosch: Local Danger Warning based on Vehicle Ad-hoc Networks: Prototype and Simulation. In Proceedings of 1st International Workshop on Intelligent Transportation (WIT), Hamburg, Germany, March 2004.

(9) E. Kelling, M. Friedewald, T. Leimbach, M. Menzel, P. Saeger, H. Seudié, B. Weyl, “Specification and evaluation of e-security relevant use cases”, EVITA Deliverable D2.1, 2009.

(10) R. Escherich, I. Ledendecker, C. Schmal, B. Kuhls, C. Grothe, F. Scharberth: SHE – Secure Hardware Extension – Functional Specification Version 1.1. Hersteller-Initiative Software (HIS) AK Security, 2009.

(11) S. Künzli. Efficient Design Space Exploration for Embedded Systems. PhD thesis. 2006.
(12) TTool, the TURTLE Toolkit. http://labsoc.comelec.enst.fr/ttool.html
(13) ECRYPT II: Yearly Report on Algorithms and Keysizes (2008-2009), D.SPA.7 Rev. 1.0, ICT-2007-216676, 07/2009.
(14) NIST: Recommendation for Key Management, Special Publication 800-57 Part 1, 03/2007.
(15) M. Raya, P. Papadimitratos, and J.-P. Hubaux, "Securing vehicular communications," IEEE Wireless Communications Magazine, Special Issue on Inter-Vehicular Communications, October 2006.
(16) M. Raya, D. Jungels, P. Papadimitratos, I. Aad, and J.-P. Hubaux, "Certificate Revocation in Vehicular Networks," Laboratory for Computer Communications and Applications (LCA), EPFL, Tech. Rep. LCAREPORT-2006-006, 2006.
(17) P. Papadimitratos, V. Gligor, and J.-P. Hubaux, "Securing Vehicular Communications – Assumptions, Requirements, and Principles," in Proc. 4th Workshop on Embedded Security in Cars (escar), 2006.