HoliSec - HOLIstic Approach to Improve Data SECurity

Dnr 2015-06894

FORDONSSTRATEGISK FORSKNING OCH INNOVATION

| | |
|---|---|
| **Document Title** | D 3.2 – Secure Communication |
| **Document Type** | Deliverable |
| **Document Number** | D3.2 |
| **Document Responsible** | Thomas Rosenstatter, thomas.rosenstatter@chalmers.se, Chalmers |
| **Document Version** | 2.0 |
| **Document Status** | Final (Consortium) |
| **Dissemination Level** | Public |
| **Last Change** | September 6, 2019 |

| | |
|---|---|
| **Project Acronym** | HoliSec |
| **Project Title** | HOLIstic Approach to Improve Data SECurity |
| **Research Program** | Vinnova/FFI (Fordonsutveckling/Vehicle Development), Sweden |
| **Diary Number** | 2015-06894 |
| **Project Duration** | 2016 - 2019 |
| **Project Coordinator** | Lars-Olof Berntsson, lars-olof.berntsson@volvo.com, Volvo AB |

 

# Executive Summary

This deliverable *D3.2 – Secure Communication* presents the results and achievements of the sub-work package WP3.2 *Security mechanisms for the connected vehicle – Secure Communication* of the HoliSec project. Dissemination level of the current version is limited to the HoliSec project consortium.

This deliverable consists of eight publications and provides insight into the following topics in vehicular security:

- An evaluation of promising CAN message authentication solutions.
- Discussion about the possibilities of memory exploitation techniques on resource constrained automotive systems.
- Recommendations to improve vehicular security throughout the entire software development lifecycle based on an automotive use case.
- Identification of security issues in the automotive domain based on a survey for automotive security experts.
- A study on how security levels should be structured for the automotive domain.
- A mapping between automotive security levels and required mandatory security mechanisms and design rules.
- An analysis of AUTOSAR SecOC Profile 3, a method to provide freshness to authenticated messages, and a proposal on how to improve it.
- A preliminary assessment of the security implications of replacing 802.11p with cellular V2X.

 

# Contributors

| Editor(s) | Affiliation | Email |
|---|---|---|
| Tomas Olovsson | Chalmers | tomas.olovsson@chalmers.se |
| Thomas Rosenstatter | Chalmers | thomas.rosenstatter@chalmers.se |
| **Contributor(s)** | **Affiliation** | **Email** |
| Aljoscha Lautenbach | Chalmers | aljoscha@chalmers.se |
| Christian Sandberg | Volvo AB | christian.sandberg@volvo.com |
| Magnus Almgren | Chalmers | magnus.almgren@chalmers.se |
| Nasser Nowdehi | Chalmers, Volvo Car Corporation | nasser.nowdehi@volvocars.com |
| Romi Zaragatzky | Volvo Group Connected Solutions | romi.zaragatzky@volvo.com |
| Tomas Olovsson | Chalmers | tomas.olovsson@chalmers.se |
| Thomas Rosenstatter | Chalmers | thomas.rosenstatter@chalmers.se |

# HoliSec Consortium








 

# Document Change History

| Version | Date | Contributor | Description |
|---|---|---|---|
| 0.5 | 2018-02-12 | Thomas Rosenstatter | First draft created. |
| 1.0 | 2018-04-16 | Thomas Rosenstatter | Final V1.0. |
| 1.9 | 2019-02-15 | Thomas Rosenstatter | Final for consortium feedback. |
| 2.0 | 2019-09-06 | Thomas Rosenstatter | Incorporated consortium feedback. |

 

# Contents

 

# Acronyms

**ECU** Electronic Control Unit.

**RSU** Road-side Unit.

**V2I** Vehicle-to-Infrastructure.

**V2V** Vehicle-to-Vehicle.

 

# Chapter 1

# Introduction

The Internet has found its way into our vehicles and many new services will be introduced in the coming years. Some services target drivers and passengers, such as navigation and driver assistance systems, and others focus on the vehicle itself, such as remote diagnostics and remote software updates. Most vehicle manufacturers have plans to offer a fairly large number of services and we now face the challenge to implement new functionality without sacrificing traffic safety. The vehicle is a complex safety-critical system with components that must function at all times, and security problems should never result in safety problems or in an immediate halt of all systems. Instead the vehicle must operate in a degraded and fail-safe mode when under attack and when security problems have been detected.

Today's vehicles have an internal network consisting of 30 to 100 computers or Electronic Control Units (ECUs). The internal network is of the size of a small company and internal security is currently largely missing. The software in a modern vehicle contains tens of millions of lines of code with a total size of more than 100 MBytes [1]. These vehicles will now be connected to the infrastructure around it, i.e. to Road-side Units (RSUs), to other vehicles and to the internet. Therefore, it is imperative that security is properly addressed before these new services are introduced.

Moreover, the communication between vehicles and the outside world will in almost all cases be wireless. Exceptions may be found in repair shops and when vehicles are parked. It is possible to access the internal network by connecting a device directly to the internal busses of a vehicle, for example in a repair shop to diagnose problems and to update the firmware, but also by vehicle owners and others in order to "enhance" or change the functionality of the vehicle. With a sound security design, it should not matter whether the communication is wired or wireless. However, physical modification of ECUs and the possibility to physically attach devices to the internal network must be paid special attention to, as it cannot be ruled out that the vehicle owner modifies or adds equipment to the vehicle that interferes with its normal functionality.

The proposal of the U.S. Department of Transportation (DoT) highlights that national departments also see this promising technology (Vehicle-to-Vehicle (V2V) communication) as an enhancement for traffic safety. On the $13^{th}$ December 2016 the DoT proposed a rule that all light-duty vehicles are required to have an active V2V and Vehicle-to-Infrastructure (V2I) communication module in order to increase traffic safety and prevent accidents due to the exchange of information between the vehicles and infrastructure [2].

This document explores questions about the usability of proposed CAN message authentication solutions, memory exploitation on resource constrained automotive systems, how security should be classified for the automotive domain, and security issues during the development lifecycle. Given the fundamental differences of the automotive domain and the well-established IT security, the challenges need to be identified and further taken into consideration. Chapter 2 lists the publications written in the course of the HoliSec project.

 

# Chapter 2

# Publications

## 2.1 Paper I: In-vehicle CAN message authentication: An evaluation based on industrial criteria [3]

*Presented at IEEE 86th Vehicular Technology Conference, VTC-Fall 2017, Toronto, Canada, 2017-09-24 - 2017-09-27*

### Authors

*Nasser Nowdehi, Aljoscha Lautenbach, and Tomas Olovsson*

### Abstract

*Vehicles have evolved from mostly mechanical machines into devices controlled by an internal computer network consisting of more than 100 interconnected Electronic Control Units (ECUs). Moreover, modern vehicles communicate with external devices to enable new features, but these new communication facilities also expose safety-critical functions to security threats. As the most prevalent automotive bus, the Controller Area Network (CAN) bus is a prime target for attacks. Even though the computer security community has proposed several message authentication solutions to alleviate those threats, such solutions have not yet been widely adopted by the automotive industry. We have identified the most promising CAN message authentication solutions and provide a comprehensive overview of them. In order to investigate the lack of adoption of such solutions, we, together with industry experts, have identified five general requirements they must fulfill in order to be considered viable in industry. Based on those requirements, we analyze and evaluate the identified authentication solutions. We find that none of them meet all the requirements, and that backward compatibility and acceptable overhead are the biggest obstacles.*

## 2.2 Paper II: What the Stack? On Memory Exploitation and Protection in Resource Constrained Automotive Systems [4]

*Presented at Critical Information Infrastructures Security: 12th International Conference, CRITIS 2017, Lucca, Italy, 2017-10-08 - 2017-10-13*

### Authors

*Aljoscha Lautenbach, Magnus Almgren and Tomas Olovsson*

### Abstract

*The increased connectivity of road vehicles poses significant challenges for transportation security, and automotive security has rapidly gained attention in recent years. One of the most dangerous kinds of security relevant software bugs are related to memory corruption, since their successful exploitation would grant the attacker a high degree of influence over the compromised system. Such vulnerabilities and the corresponding mitigation techniques have been widely studied for regular IT systems, but we identified a gap with respect to resource constrained automotive systems. In this paper, we discuss how the hardware architecture of resource constrained automotive systems impacts memory exploitation techniques and their implications for memory protection. Currently deployed systems have little to no protection from memory exploitation. However, based on our analysis we find that the simple and well-known measures like stack canaries, non-executable RAM, and to a limited extent memory layout randomization can also be deployed in this domain to significantly raise the bar for successful exploitation.*

## 2.3 Paper III: Secure software development for automotive systems – Implementation pitfalls in AUTOSAR

*Technical Report 2017:06, ISSN 1652-926X – Can be found in the appendix, Chapter 4*

### Authors

*Aljoscha Lautenbach, Magnus Almgren and Tomas Olovsson*

### Abstract

*With the advent of ubiquitous communication in cyber-physical systems such as vehicles, the importance of secure software is hard to overstate. When the willful manipulation of software can endanger lives, security must be taken seriously. In this paper, we show the need for security at all levels of the automotive software development lifecycle and give recommendations how to concretely improve vehicular security. For this purpose, we develop a simple AUTOSAR application, and highlight security issues which must be addressed during the development process. Based on the identified issues, we recommend steps to alleviate the security risks.*

 

## 2.4 Paper IV: Understanding Common Automotive Security Issues and Their Implications [5]

*Presented at International Workshop on Interplay of Security, Safety and System/Software Architecture,Barcelona, , 2018-09-07 - 2018-09-07*

### Authors

*Aljoscha Lautenbach, Magnus Almgren and Tomas Olovsson*

### Abstract

*With increased connectivity of safety-critical systems such as vehicles and industrial control systems, the importance of secure software rises in lock-step. Even systems that are traditionally considered to be non safety-critical can become safety-critical if they are willfully manipulated. In this paper, we identify 8 important security issues of automotive software based on a conceptually simple yet interesting example. The issues encompass problems from the design phase, including requirements engineering, to the choice of concrete parameters for an API. We then investigate how these issues are perceived by automotive security experts through a survey.*

*The survey results indicate that the identified issues are indeed problematic in real industry use-cases. Based on the collected data, we draw conclusions which problems deserve further attention and how the problems can be addressed. In particular, we find that key distribution is a major issue. Finally, many of the identified issues can be addressed by improved documentation and access to security experts.*

 

## 2.5 Paper V: Open Problems when Mapping Automotive Security Levels to System Requirements [6]

*Presented at 4th International Conference on Vehicle Technology and Intelligent Transport Systems, VEHITS 2018 (http://vehits.org), Funchal, Madeira, Portugal, 2018-03-16 - 2018-03-18*

*Can be downloaded at https://research.chalmers.se/en/publication/501558*

### Authors

*Thomas Rosenstatter and Tomas Olovsson*

### Abstract

*Securing the vehicle has become an important matter in the automotive industry. The communication of vehicles increases tremendously, they communicate with each other and to the infrastructure, they will be remotely diagnosed and provide the users with third-party applications. Given these areas of application, it is evident that a security standard for the automotive domain that considers security from the beginning of the development phase to the operational and maintenance phases is needed. Proposed security models in the automotive domain describe how to derive different security levels that indicate the demand on security, but do not further provide methods that map these levels to predefined system requirements nor security mechanisms. We continue at this point and describe open problems that need to be addressed in a prospective security framework for the automotive domain. Based on a study of several safety and security standards from other areas as well as suggested automotive security models, we propose an appropriate representation of security levels which is similar to, and will work in parallel with traditional safety, and a method to perform the mapping to a set of predefined system requirements, design rules and security mechanisms.*

 

## 2.6 Paper VI: Towards a Standardized Mapping from Automotive Security Levels to Security Mechanisms [7]

*Presented at 21st IEEE International Conference on Intelligent Transportation Systems, ITS-C, Maui, Hawaii, USA, 2018-11-04 - 2018-11-07*

*Can be downloaded at https://research.chalmers.se/publication/507336*

### Authors

*Thomas Rosenstatter and Tomas Olovsson*

### Abstract

*Modern vehicles are becoming targets and need to be secured throughout their lifetime. There exist several risk assessment models which can be used to derive security levels that describe to what extent components, i.e., functions and signals, need to be protected. These models provide methods to gather application specific security requirements based on identified threat and item combinations that need to be coped with, however, a standardized mapping between security levels and required mandatory security mechanisms and design rules is missing. We address this problem first by suggesting that the risk assessment process should result in five security levels, similar to the functional safety standard ISO 26262. Second, we identify suitable security mechanisms and design rules for automotive system design and categorize them in appropriate security levels. Our proposed methodology is as much as possible aligned with ISO 26262 and we believe that it should therefore be realistic to deploy in existing organizations.*

## 2.7 Paper VII: Extending AUTOSAR's Counter-based Solution for Freshness of Authenticated Messages in Vehicles

*Submitted to the 24th IEEE Pacific Rim International Symposium on Dependable Computing (PRDC 2019)*

### Authors

*Thomas Rosenstatter, Christian Sandberg and Tomas Olovsson*

### Abstract

*Nowadays vehicles have an internal network consisting of more than 100 microcontrollers, so-called Electronic Control Units (ECUs), which provide core functionalities, active safety, diagnostics, comfort and infotainment. The network technology which connects these ECUs with each other depends on the data they exchange. The Controller Area Network (CAN) bus is one of the most widespread bus technologies in use, and thus became the primary target for attackers. Means to protect the CAN bus have been proposed by research as well as industry. AUTOSAR, an open system platform for vehicles, introduced in version 4.3 SecOC*

 

*Profile 3, a counter-based solution to provide freshness in authenticated messages to protect the system against replay attacks. In this paper, we analyse and assess this method regarding safety constraints and its usability, and discuss design considerations when implementing such a system. Furthermore, we propose a new profile considering the identified deficiencies which allows a faster synchronisation of the counter between senders and receivers. Finally, we evaluate our solution in an experimental setup in regards to bandwidth usage and time to synchronise the freshness counter.*

## 2.8 Paper VIII: A Preliminary Security Assessment of 5G V2X [8]

*Presented at IEEE 89th Vehicular Technology Conference: VTC2019-Spring 28 April – 1 May 2019, Kuala Lumpur, Malaysia*

### Authors

*Aljoscha Lautenbach, Nasser Nowdehi, Tomas Olovsson and Romi Zaragatzky*

### Abstract

*Research on intelligent transport systems (ITS) for improved traffic safety and efficiency has reached a high level of maturity and first applications will hit the market in 2019. Since 2004, the wireless standard 802.11p has been developed specifically for ITS services. Since then new telecommunication standards have been devised, and the new 5G telecommunication standard is nearing completion. Due to its technological advantages such as higher speeds and reliability, it is being considered to be used for ITS services. The new radio technology "New Radio (NR)", which is being developed as part of 5G, can complement or replace 802.11p in V2X applications. While there has been some work to compare 802.11p and 5G New Radio in terms of performance and applicability for safety-critical use cases, little work has been done to investigate the implications for security. In this paper, we provide an overview of the security requirements of known ETSI ITS use cases, and based on those use cases we compare and assess the security implications of replacing 802.11p with cellular V2X. We find that due to the use of millimeter waves, beamforming and massive MIMO, there will be an implicit improvement for confidentiality and privacy, and it may also be possible to shorten authentication procedures in certain cases. When a fully network-assisted C-V2X mode is chosen, it is also possible to outsource several of the ITS security requirements to the cellular network.*

 

# Chapter 3

# Conclusions

The current work highlights the different areas of secure communication within the automotive domain. Starting from measures against memory exploitation of resource constrained automotive systems and continuing with identifying security issues and further proposing a model for identifying security mechanisms based security levels which are the result of a Threat Analysis and Risk Assessment (TARA).

In paper I [3], we identified the most promising CAN message authentication solutions and further provide a comprehensive overview of them. The proposed methods have been evaluated based on five industrial criteria.

Paper II [4] discusses how the hardware architecture of resource constrained systems impacts memory exploitation techniques. Based on the analysis we concluded that well-known and yet simple techniques like stack canaries and non-executable RAM can be implemented on such an architecture as well.

Papers III and IV [5] focus on issues that occur during the development of secure automotive systems. Identifying these issues and finding suitable counter measures is an important step towards a secure software development lifecycle. Furthermore, the survey highlights the need for solutions in the area of the distribution of cryptographic keys.

In paper V [6], we focus on the challenges in the automotive domain, "What makes the automotive domain different?", in order to propose the optimal number and structure of security levels which is based on a study on standards and models from different domains.

Paper VI [7] identifies appropriate security mechanisms and design rules for the automotive domain that are directly linked with security levels. Such a mapping to required mandatory security mechanisms and design rules, increases the efficiency during the design phase for the reason that this mapping already covers a large number of threats and lets the engineers focus on specific security requirements that are strongly depending on the application. Furthermore, such a mapping eases the communication between

manufacturers and suppliers, as they have a common ground when talking about the required security measures for modules and functions.

Paper VII analyses SecOC Profile 3, a method to provide freshness to authenticated messages, regarding the synchronisation of the freshness value. The analysis includes how the freshness value is reconstructed in case only a truncated freshness value is being transmitted, the total waiting time for a synchronisation message and a discussion about design considerations and limitations of such a counter-based solution. Finally, we propose ways to improve SecOC Profile 3 and compare our improved method with it.

Paper VIII [8] analyses the security impact of using the new 5G telecommunication standard for V2X applications. The findings are that the use of millimeter waves, beamforming and massive MIMO implicitly improves the confidentiality and privacy. Moreover, having a fully network-assisted C-V2X would also allow to outsource several of the ITS security requirements to the cellular network.

 

# Bibliography


[1] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage, "Experimental security analysis of a modern automobile", in *2010 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2010, pp. 447–462, ISBN: 1424468949. DOI: 10.1109/SP.2010.34.

[2] National Highway Traffic Safety Administration (NHTSA), *U.s. dot advances deployment of connected vehicle technology to prevent hundreds of thousands of crashes*. [Online]. Available: https://www.nhtsa.gov/press-releases/us-dot-advances-deployment-connected-vehicle-technology-prevent-hundreds-thousands (visited on 12/19/2016).

[3] N. Nowdehi, A. Lautenbach, and T. Olovsson, "In-vehicle can message authentication: An evaluation based on industrial criteria", in *2017 IEEE 86th Vehicular Technology Conference (VTC-Fall)*, Sep. 2017, pp. 1–7. DOI: 10.1109/VTCFall.2017.8288327.

[4] A. Lautenbach, M. Almgren, and T. Olovsson, "What the stack? on memory exploitation and protection in resource constrained automotive systems", in *Critical Information Infrastructures Security*, G. D'Agostino and A. Scala, Eds., Cham: Springer International Publishing, 2018, pp. 185–193, ISBN: 978-3-319-99843-5.

[5] ——, "Understanding common automotive security issues and their implications", in *Security and Safety Interplay of Intelligent Software Systems*, B. Hamid, B. Gallina, A. Shabtai, Y. Elovici, and J. Garcia-Alfaro, Eds., Cham: Springer International Publishing, 2019, pp. 19–34, ISBN: 978-3-030-16874-2.

[6] T. Rosenstatter and T. Olovsson, "Open problems when mapping automotive security levels to system requirements", in *Proceedings of the 4th International Conference on Vehicle Technology and Intelligent Transport Systems - Volume 1: VEHITS,*, INSTICC, SciTePress, 2018, pp. 251–260, ISBN: 978-989-758-293-6. DOI: 10.5220/0006665302510260.

[7] T. Rosenstatter and T. Olovsson, "Towards a standardized mapping from automotive security levels to security mechanisms", in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, Nov. 2018, pp. 1501–1507. DOI: 10.1109/ITSC.2018.8569679.

[8] A. Lautenbach, N. Nowdehi, T. Olovsson, and R. Zaragatzky, "A preliminary security assessment of 5g v2x", in *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*, Apr. 2019, pp. 1–7. DOI: 10.1109/VTCSpring.2019.8746547.

 

# Chapter 4

# Appendix

This chapter contains papers that are not published anywhere else.

# Secure software development for automotive systems

## Implementation pitfalls in AUTOSAR

Aljoscha Lautenbach, Magnus Almgren, Tomas Olovsson

***Abstract*—With the advent of ubiquitous communication in cyber-physical systems such as vehicles, the importance of secure software is hard to overstate. When the willful manipulation of software can endanger lives, security must be taken seriously. In this paper, we show the need for security at all levels of the automotive software development lifecycle and give recommendations how to concretely improve vehicular security. For this purpose, we develop a simple AUTOSAR application, and highlight security issues which must be addressed during the development process. Based on the identified issues, we recommend steps to alleviate the security risks.**

### I. Introduction

Imagine when driving on the highway, the driver seat suddenly starts to slide back and forth, and the seat adjustment controls are not responding. Clearly, the driver will have problems to drive the car safely: she may be unable to reach the brakes in a critical moment, or her movement may be so restricted that it is impossible to steer correctly. Perhaps the driver would be able to handle the situation for a short amount of time, but after a while she would become fatigued from having to constantly adjust her body to the changing seat position, which can obviously lead to dangerous situations.

There are many safety-critical systems in a car, but as the above example shows, even not directly safety-critical systems such as the seat adjustment system can have a negative impact on safety when attacked.

In the last few years, a growing number of (cyber) security problems have been discovered in automotive systems. The first systematic investigations started early this millennium [14], but only the more recent works by Checkoway et al. [4], Miller and Valasek [6], [7], [11] and others brought the problem to the attention of a wider audience.

Automotive systems face the same challenge as all embedded systems in the wake of ubiquitous connectivity: their technology was designed when connectivity was limited, and malicious attackers were not seen as a serious threat since local access was required to do any damage. Therefore, security mechanisms are missing and must now be added in retrospect.

Automotive software engineers are well trained in the safety aspects of their work, but few have security training. This is further complicated by the fact that, depending on the context, the same terms may have different meanings. For instance, for a safety engineer, data integrity means protection from *random* transmission faults via error detection or error correction codes, such as cyclic redundancy checks (CRCs). However, for a security engineer, data integrity means protection from *intentional* manipulation, which requires stronger guarantees such as cryptographically secure hashes using secret keys. Without security training, CRCs can easily be misconstrued as sufficient for security as well [15]. It is well known that implementing secure systems and using cryptographic libraries correctly requires finesse and training [1], [5].

In an effort to increase component interoperability and to allow for third-party software integration, the AUTOSAR consortium was formed. The consortium develops a platform and methodology for vehicular software development, simply called AUTOSAR. There are about 100 partners involved in AUTOSAR, including several major vehicle manufacturers and suppliers. AUTOSAR has been widely adopted in the USA, Japan and Europe [3].

Since AUTOSAR provides a common and open platform, it allows reasoning about security in a broad automotive context. AUTOSAR already has many safety mechanisms built into the development process which also improve security significantly. Improving the security in AUTOSAR has a large potential impact, since it is widely used. Moreover, by ensuring good security in AUTOSAR, it can also serve as an example for other platforms. It also offers support for writing secure applications.

In this paper, we show with a simple example using AUTOSAR that due to the complexity and safety implications of automotive software, it is necessary to make security a first-order concern. In order to focus on the security issues, we keep the example as simple as possible.

Our contributions are:

1) We demonstrate that security is a pervasive design issue, even for functions which may not seem safety-critical at first glance.
2) We highlight several security considerations and issues for secure automotive software development.
3) We give recommendations for the previously identified security issues.

To make these points clear for a wide readership, we use the motivation example of the seat sliding attack to emphasize the issues before we discuss their consequences.

### II. The AUTOSAR Methodology

Modern vehicles include around 100 **electronic control units (ECUs)**, which control every conceivable system, from engine control to the windshield wipers. Each ECU runs one or more applications, and they are connected through several different in-vehicle networks, using bus technologies such as LIN, CAN, FlexRay, MOST or Ethernet.

One of the key ideas in AUTOSAR is abstraction for component decoupling: to an application, there is no difference between communicating with another application running on the same electronic control unit and communicating with

 

an application remotely via a network. Conceptually, the AUTOSAR platform consists of three main layers: the **basic software components (BSW)**, the **runtime environment (RTE)**, and different application **software components (SWC)**. The basic software components provide the basic operating system services and device drivers and they serve as interface to the hardware. The runtime environment is the link between the basic software and the applications. The RTE implements an abstract communication bus between the different software components, called the **virtual function bus (VFB)**, to make the communication hardware independent.

The AUTOSAR methodology defines the following basic steps in the development of an application:

1) Develop an abstract functional view of the system (Abstract System Description)
2) Develop the virtual function bus (VFB) model
3) Develop the system

We will explain each of these steps in more detail in the next section.

## III. Secure Software Development with AUTOSAR

The described attack of the sliding driver seat will serve as a running example to illustrate the different security issues one can encounter during software development. Whenever a new concept is introduced, we will highlight how this relates to the driver seat sub-system, and we will expand the details of the example as needed. We return to the identified issues in section IV with a discussion.

In the next subsection, we outline common security concerns. The remaining three subsections clarify the three development steps of the AUTOSAR methodology using the running example. In each step, we discuss the security issues an application developer encounters.

### A. Overview of Security Concerns

There are many different security concerns in software development and they can be found at different levels. We will distinguish three conceptual levels: *(1) Organizational*, *(2) Architectural* and *(3) Implementation*.

At the organizational level, the issues concern the structure and culture of the organization, and how they impact security. Among others, the answers to the following questions are of interest: Is there a budget for security testing? Is there systematic security testing? Has every development team access to a security expert? Are security requirements communicated to suppliers? How is key management handled with suppliers?

At the architectural level, the focus lies on questions concerning the software and network architecture, such as: Are the software components partitioned in a way that takes security into account? Is the communication between components secured? Do the used protocols have known vulnerabilities?

Finally, most security vulnerabilities are found at the implementation level. Broadly speaking, the types of vulnerabilities encountered at this level are: incorrect use of cryptography, insufficient randomness, missing input validation, memory access errors and timing vulnerabilities [1], [5], [10], [12].

### B. Abstract Functional View

The abstract functional view is simply an abstract system description. For the running example, the sub-system description is taken from the "Explanation of Application Interfaces of the Body and Comfort Domain" document in AUTOSAR. We chose to highlight the driver seat, because an attack may have a big safety impact. Note that we provide a highly simplified example to keep the illustrations and discussions on point.

The abstract functional view for the seat example can be described as follows. In the seat adjustment sub-system, there is a seat adjustment manager **software component (SWC)** (`SeatAdjustmentManager`) which controls all movement of the seat. Each axis has a corresponding actuator SWC (`Actuator`), and each actuator has an adapter SWC (`ActuatorAdapter`) which ensures that requested positions confirm to the electrical and mechanical restrictions of the actuator. The adapter also translates the requests from the `SeatAdjustmentManager` into commands understood by the actuator.

### C. Virtual Function Bus

Once the abstract system description has been established, the next step is to define the **Virtual Function Bus (VFB)** application model.VFB models depict only the software components and their interactions, independent of their physical location or underlying communication system.

The VFB model we developed for the example application is depicted in Fig. 1. In the VFB model, the *black triangles* represent a *data transfer* from a sender port to a receiver port. The *circle* (server port) and *half circle* (client port) represent a *command invocation* from the client on the server. Our example is limited to three software components: `SeatAdjustmentManager`, `ActuatorAdapter` and `Actuator`.

The `SeatAdjustmentManager` has five ports: (1) The `ManualSeatAdjustment` server port accepts manual commands to adjust the seat, (2) the `OperatingMode` receiver port receives data on the current operational mode, (3) the `VehicleSpeed` receiver port receives data on the current vehicle speed, (4) the `SeatAxisMove` client port can call `SeatAxisMove` on the slide adapter, and (5) the `SeatAxisPosition` receiver port receives data on the current position of the slide actuator.

The `ActuatorAdapter` has three ports: (1) The `SeatAxisMove` server port offers the service to move the seat by a given amount, (2) the `SeatAxisAction` client port invokes the correct move command on the actuator, and (3) the `SeatAxisPosition` receiver port receives data from the actuator on the current position.

Finally, the `Actuator` has only two ports: (1) The `SeatAxisAction` server port offers the service to move the seat by a given amount, and (2) the `SeatAxisPosition` sender port sends the current position to the adapter and the seat manager.

Several observations can be made about the security requirements of this application at this point. Most importantly, unless the messages between the software components are

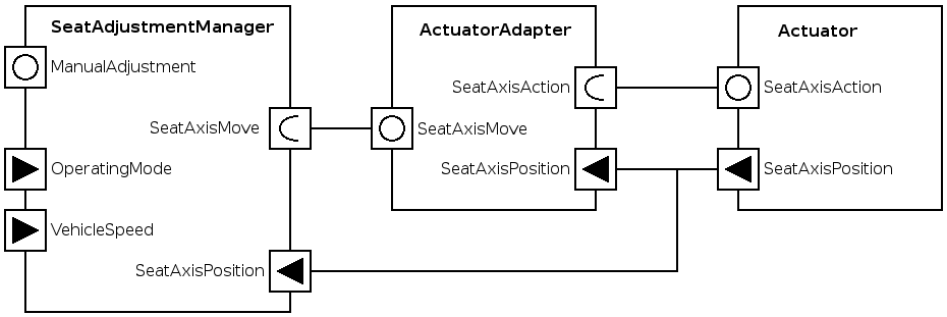


Fig. 1. Seat Adjustment Application Model for the Virtual Function Bus

authenticated, they can be spoofed by an attacker. There are many different ways to gather the security requirements of an application, but they all require some experience and training. Therefore, the first security issue is the identificiation of potential vulnerabilities.

**Issue #1:** Identification of potential vulnerabilities.

*D. System Development*

Obviously, the system development step can be arbitrarily complex. It includes the definition of a topology of the Electronic Control Units (ECUs), networks and communication matrices, and the deployment of the software components to the ECUs. In case of a single development organization, the development process looks as follows:

1) Create a System Configuration Description: includes ECU and network topologies.
2) Create ECU Extracts: the parts of the system configuration for one specific ECU.
3) Develop the software components: mostly independent from ECU configuration.
4) Integrate the AUTOSAR ECUs: ECUs are configured in this step. This includes task scheduling and BSW configuration.

The steps above are not necessarily sequential, and steps 2) and 3) are repeated as often as necessary.

*1) System Configuration Description:* The system configuration description must include the network design, the hardware choices, and the ECU composition, i.e., the choice which software component is put on which ECU. The choices pertaining to security are usually a trade-off: higher security often means lower usability and/or higher costs.

For the seat adjustment sub-system the network bus can either be a CAN, LIN or any other bus. The bus choice has an impact on the security protocols which can be used. For instance, there are a large number of proposed authentication protocols for CAN, but LIN has received comparatively little attention.

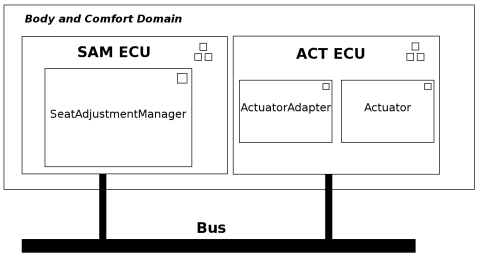


Fig. 2. ECU compositions

We will use two ECUs: the `SeatAdjustmentManager` SWC goes onto one seat adjustment manager ECU (SAM ECU), and the `ActuatorAdapter` and `Actuator` SWCs go onto the actuator ECU (ACT ECU), as illustrated in Fig. 2. The placement in this example is somewhat arbitrary, and there are several equally viable alternative placements. The placement generally depends on the concrete actuators being used, and the physical restrictions within the vehicle. However, the placement does make a difference for the security of the system. If it was possible to put all SWCs onto the same ECU, the signals would not traverse a bus, and could not be spoofed easily.

For the seat adjustment ECU, we opt for an ECU in the medium price- and performance range, since it is likely the ECU would also host several additional functions. For the ACT ECU we assume a low-end ECU. Due to cost pressure, neither ECU has support for hardware accelerated cryptographic functions.

*2) ECU Extracts:* When the system description is complete, the configuration description for the individual ECUs are extracted, so called ECU extracts.

An **ECU extract** is the complete architectural model of one ECU containing the allocated software components and related signals. At this point, the choice of basic software modules (BSWs) must be made for this ECU. This includes the OS components and device drivers which are needed for

all applications on the ECU to work correctly. Depending on the functionality that is needed, different BSW modules are included in the binary of the ECU. A minimal setup includes around 15 - 20 BSW modules.

*3) Software Component Development:* After the configuration for an ECU has been extracted, the software components can be developed for it. Note that the application code should not rely on a particular ECU configuration or available hardware.

From the VFB model described in section III-C, C template files are generated, which form the basic framework of the application. The rest is straight-forward development work.

Given the architecture presented in Figure 1 (the VFB model from III-C), the seat-sliding attack can be executed by spoofing a single message: the message which invokes `SeatAxisMove` on the `ActuatorAdapter` to adjust the seat position. Since there is no authentication of the message, the receiver has no way to distinguish correct messages from spoofed messages.

To authenticate a message, a so called *authenticator* is added to the message. The authenticator is an encrypted hash of the message. Only the party with the right key would have been able to encrypt the hash. Obviously, the receiving party must have the right key to decrypt the has.

In order to prevent the seat sliding attack, authentication should be added. AUTOSAR provides two different BSW modules to access cryptographic primitives: the **Crypto Abstraction Library (CAL)** and the **Crypto Service Manager (CSM)**. Both provide an interface to cryptographic primitives, decoupling the interface from the underlying implementation. They differ in how they can be used and interact with other software components [2], which we will return to at the end of this section. In contrast, the **Secure Onboard Communication (SecOC)** module, added recently in AUTOSAR release 4.2.1, allows to add authentication to applications without having to change the application code. It does so by moving the authentication onto the communication stack, so that for specifically chosen messages, authentication will be added automatically before sending them over the network. This has the advantage that the application developer only needs to configre the SecOC module correctly, rather than having to implement an authentication protocol from scratch.

The choice whether to use CAL, CSM or SecOC depends on the use-case, the available hardware, and the AUTOSAR platform version. However, almost all of the issues regarding authentication are the same, no matter which of the three modules is chosen. Should symmetric or asymmetric cryptography be used? How large should the keys be? How should the keys be distributed to the ECUs? Which algorithm should be used for the authenticator Should the transmitted authenticator be truncated? Should a freshness value be used? How large should that value be? And how much of it should be transmitted? We will shortly discuss each of these in turn.

For in-vehicle communication, the question of symmetric vs asymmetric cryptography is relatively simple to answer: symmetric cryptography is strongly preferred due to much better performance. Asymmetric cryptography still has a place in the vehicle for functions where speed is not critical, e.g., signatures for remote software updates, or when pre-shared keys are impractical, e.g., when communicating on the internet. For the remainder of this article, we focus on symmetric cryptography.

Since the key (also known as "secret") is needed to authenticate the message, the length of the used keys is, together with the used algorithm, one of the biggest factors for the security of the scheme. Generally speaking, the longer the key, the more secure the mechanism. So choosing the right key length is important.

**Issue #2:** Choosing the right key length

When symmetric cryptography is used, the keys must be available at both communication ends before communication starts. There are several ways key distribution can be done. One possibility is to use an out-of-band channel, i.e., to communicate the keys outside the communication system (pre-shared keys). This is usually done during production. The other way is to use public key schemes such as the Diffie-Hellman key exchange protocol or certificates. The choice of the key distribution mechanism has both practical and security implications.

**Issue #3:** Choice of the right key distribution mechanism.

As mentioned earlier, the cryptographic algorithm choice is of paramount importance. Cryptographers constantly try to find weaknesses in published algorithms, and an algorithm which was considered secure five years ago may not be so today, although this is often a gradual process. A good example is the SHA-1 cryptographic hash algorithm: first attacks have already been discovered in 2005 [13], and it has been considered weak for many years, but only very recently the first collision was publicized [9]. Given the long lifetime of automotive products, the right algorithm must be chosen to guarantee security. This also includes the choice whether or not the authenticator should be truncated, to gain bandwidth in exchange for security.

**Issue #4:** Choice of the right cryptographic algorithm.

In order to guard against replay attacks, in which an attacker records a previously recorded message, and simply resends it to achieve a particular result, the concept of freshness is used. This can be a monotonic counter or a timestamp, and is added as input to the authenticator. That way, two messages which are otherwise identical will have different authenticators. The counter must therefore be chosen appropriately large. It is important that the counters are synchronized, that is, only particular counter values are accepted at particular times, otherwise replay attacks will work again.

**Issue #5:** How to guarantee freshness.

Since the crypto abstraction library is an AUTOSAR library, its functions can be called directly from a SWC or BSW module as synchronous function calls. In other words, CAL functions are always executed in the context of the caller and bypass

the runtime environment (RTE). However, AUTOSAR libraries must be stateless and reentrant, and are not allowed to access any hardware, which clearly implies that *the CAL does not support the use of cryptographic hardware* [2]. The crypto abstraction library only defines a standardised interface, the underlying library with the software implementation of the crypto primitives is not standardised. Cryptographic libraries should always be written by cryptographers or security experts, otherwise there is a high probability that they are insecure. If this is not immediately obvious, consider the Debian SSL bug discovered in 2008: two small, superficially harmless changes by the Debian maintainers significantly lowered the entropy during SSL key generation, which led to a huge number of insecure keys.

**Issue #6:** The correct implementation of cryptographic primitives is notoriously difficult.

An AUTOSAR service manager, like the crypto service manager (CSM), needs to be accessed via the runtime environment, because it arbitrates access for one or more clients to one or more services. Function calls to the CSM are essentially remote procedure calls via message passing, even if the components are placed on the same ECU. The calls can be synchronous or asynchronous (configured at compile time), and just like the CAL, the CSM only defines the standardised interface, the implementation depends on the AUTOSAR supplier. The Crypto Service Manager can use cryptographic hardware by using a special device driver.

To summarize, the crypto service manager is slightly harder to use and the implementation is somewhat more complex than the crypto abstraction library, but it is also more powerful because it can use hardware support and offers asynchronous function calls, if desired. Contrasting CAL and CSM with SecOC, there is another source of difficulty in using the cryptographic primitives: Since they are used directly by the SWC, the application programmer may use the API incorrectly. SecOC on the other hand only needs to be configured correctly, and the communication stack will handle the rest.

**Issue #7:** Incorrect API use.

In addition to these specific challenges regarding authentication, there are also typical development pitfalls that apply to any program written in C. It is easy to write insecure code, e.g., due to faulty memory management, pointer handling or lack of input validation.

**Issue #8:** Writing secure C code is difficult.

*4) ECU integration:* During the ECU integration phase, the SWCs are integrated and the ECUs are configured and tested. This includes the BSW configuration of the ECUs, and the task scheduling model.

As mentioned previously, the Secure Onboard Communication (SecOC) module is integrated into the communication stack, so that using it is simply a matter of configuration, and no additional programming is required. However, all issues related to authentication in general also apply to using SecOC for authentication purposes. SecOC allows to configure which subset of messages should be authenticated.

In the latest AUTOSAR release 4.3.0, SecOC also includes a number of so called security profiles, which are preset configurations of parameters, to simplify the choices for the developers.

## IV. Discussion and Recommendations

As we have shown, security is a pervasive design issue which affects every level of the development process, and even trivial systems can be dangerous when exploited by an attacker. In the following we will discuss the consequences, and give some recommendations how they can be handled.

Currently, software developers in the automotive industry are usually well trained in safety, but they often have little or no training in security. As a result, they may inadvertently introduce security relevant bugs into their code. Therefore, it should be made as easy as possible for automotive software developers to write secure code. This can in part be achieved through improved documentation. For instance, the difficulties in choosing the right key length (issue #2), choosing the right cryptographic algorithm (issue #4) and choosing good parameters to guarantee freshness (issue #5) can be alleviated by adding more security related documentation.

**Recommendation #1:** Improve documentation by adding look-up tables for recommended key lengths, algorithms, MAC length and freshness parameters.

The topic of key management (issue #3) deserves special attention, because of its wide ranging implications. If symmetric keys or a public key infrastructure setup are chosen, the vehicle manufacturer must maintain a central infrastructure to store and retrieve those keys on demand in a secure manner. The key management also needs to be coordinated with suppliers to clarify how and when the keys are installed. Furthermore, the keys must be accessible to licensed workshops for repair and maintenance.

**Recommendation #2:** The vehicle manufacturer should develop a process for key management.

Some of the identified issues can be alleviated by providing developer training or by providing access to security or cryptography experts. For example, for identifying potential security vulnerabilities at the architectural stage (issue #1), a security expert should be available to provide an analysis. For the implementation of cryptographic libraries (issue #6), cryptographers should be used, and developers should confirm that the library they use was developed by experts. Finally, API misuse (issue #7) can obviously be limited through developer training, too.

 

**Recommendation #3:** Every development team should have access to at least one security expert and every team should have at least one developer who is trained in security.

This leaves the point that writing secure C code is difficult (issue #8). Since AUTOSAR was developed for the automotive industry, it includes several requirements and processes to enhance the safety of the resulting code. One such requirement is that basic software modules (BSW) must be compliant to the MISRA C guidelines [8], and when they are not, the exception must be clearly documented. Those guidelines were developed specifically to make the C programming language safer to use in critical systems.

One effective result of requiring conformance to MISRA C is that all unsafe C library functions are implicitly forbidden to be used in production code. There are several commercial compilers which check MISRA C code compliance, but MISRA C contains many items which can not be checked automatically, or which require additional formal verification tools. Moreover, regular programming mistakes which lead to security vulnerabilities can still happen. For example, it is possible to allocate a fixed-size buffer and accidentally write beyond its boundaries due to missing or insufficient run-time checks. Nevertheless, strict adherence to the MISRA C guidelines strengthens the security of the code considerably. AUTOSAR only requires that BSW modules adhere to the MISRA C guidelines, but it is sensible to also require this of other software components. Regular AUTOSAR applications are free to ignore the MISRA C guidelines, but obviously following them is also a good idea for application SWCs.

**Recommendation #4:** Adhere to the MISRA C guidelines.

## V. Concluding Remarks

As we have demonstrated, security is a design issue that permeates all parts of automotive software design and development. Even systems which are generally perceived to have no safety-critical components can pose dangers when exploited by an attacker. With the increased communication of cyber-physical systems, securing software is of ever-increasing importance.

Automotive software developers are well trained in writing software according to safety requirements, but writing secure software requires additional knowledge and skills. Consequently, new frameworks, platforms and standards should make it as easy as possible to write secure code, and they should foster an environment which supports secure development: it should be hard to make bad decisions, and easy to make good ones. This can be partially achieved with supporting documentation to facilitate informed choices about security measures. However, improved documentation alone is not enough. In order to integrate security into the entire development process, cultural and organizational changes are needed. For instance, every development team should have access to a security expert. Using a structured development approach such as AUTOSAR can be a first step in the direction of more security aware development practices.

In order to achieve such a security conducive environment, several aspects must come together. First and foremost, there must be organizational support. Secure development can not be done without a budget for security. Then there are the complex interactions of OEMs and suppliers which must be coordinated. More documentation how to securely use existing security functions should be added. Finally, development processes must be adapted to include security reviews and security testing. All of the above entails a cultural change, so a concerted effort of all involved partners in the automotive industry is needed to secure future vehicles.

## Acknowledgments

We would like to thank all anonymous reviewers for their valuable feedback. The research leading to these results has been partially supported by the HoliSec project (2015-06894) funded by VINNOVA, the Swedish Governmental Agency for Innovation Systems, and by the Swedish Civil Contingencies Agency (MSB) through the project “RICS”.

## References

[1] R. Anderson. Why cryptosystems fail. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, CCS ’93, pages 215–227, New York, NY, USA, 1993. ACM.

[2] AUTOSAR. Utilization of Crypto Services, AUTOSAR Release 4.2.1.

[3] AUTOSAR consortium. AUTOSAR, http://www.autosar.org/, last accessed: 2017-03-26.

[4] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno. Comprehensive experimental analyses of automotive attack surfaces. In *Proceedings of the 20th USENIX Security Symposium*, page 7792, San Francisco, CA, USA, Aug. 2011.

[5] D. Lazar, H. Chen, X. Wang, and N. Zeldovich. Why does cryptographic software fail?: A case study and open problems. In *Proceedings of 5th Asia-Pacific Workshop on Systems*, APSys ’14, pages 7:1–7:7, New York, NY, USA, 2014. ACM.

[6] C. Miller and C. Valasek. A survey of remote automotive attack surfaces. Technical report, Defcon 22, Aug. 2014.

[7] C. Miller and C. Valasek. Remote Exploitation of an Unaltered Passenger Vehicle. Technical report, Defcon 23, Aug. 2015.

[8] MISRA. MISRA C:2012 - Guidelines for the use of the C language in critical systems, 2013.

[9] M. Stevens, E. Bursztein, P. Karpman, A. Albertini, Y. Markov, A. P. Bianco, and C. Baisse. Announcing the first SHA1 collision, https://security.googleblog.com/2017/02/announcing-first-sha1-collision.html, February 2017.

[10] L. Szekeres, M. Payer, T. Wei, and D. Song. Sok: Eternal war in memory. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 48–62, May 2013.

[11] C. Valasek and C. Miller. Adventures in Automotive Networks and Control Units. Technical report, Defcon 21, Aug. 2013.

[12] V. Van der Veen, N. dutt-Sharma, L. Cavallaro, and H. Bos. Memory errors: the past, the present, and the future. In *Proceedings of the 15th International Symposium on Research in Attacks, Intrusions, and Defenses*, pages 86–106. Springer, 2012.

[13] X. Wang, Y. L. Yin, and H. Yu. *Finding Collisions in the Full SHA-1*, pages 17–36. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[14] M. Wolf, A. Weimerskirch, and C. Paar. Security in automotive bus systems. In *Workshop on Embedded Security in Cars*, 2004.

[15] R. Zalman and A. Mayer. A secure but still safe and low cost automotive communication technique. In *Proceedings of the 51st Annual Design Automation Conference*, DAC ’14, pages 43:1–43:5, New York, NY, USA, 2014. ACM.

 

 