


Cybersecurity Threat Analysis, Risk Assessment and Design Patterns for Automotive Networked Embedded Systems: A Case Study


Jürgen Dobaj

(Graz University of Technology, Graz, Austria,

 <https://orcid.org/0000-0001-6460-8080>, juergen.dobaj@tugraz.at)


Damjan Ekert

(ISCN GesmbH, Graz, Austria,

 <https://orcid.org/0000-0001-9301-242X>, dekert@iscn.com)


Jakub Stolf

(VSB TUO, Ostrava, Czech Republic

 <https://orcid.org/0000-0003-0599-6209>, jakub.stolfa@vsb.cz)


Svatopluk Stolf

(VSB TUO, Ostrava, Czech Republic

 <https://orcid.org/0000-0003-4125-1546>, jakub.stolfa@vsb.cz)


Georg Macher

(Graz University of Technology, Graz, Austria,

 <https://orcid.org/0000-0001-9215-3300>, georg.macher@tugraz.at)

Richard Messnarz

(ISCN GesmbH, Graz, Austria,

 <https://orcid.org/0000-0002-0555-3160>, rmess@iscn.com)

Abstract: Cybersecurity has become a crucial challenge in the automotive sector. At the current stage, the framework described by the ISO/SAE 21434 is insufficient to derive concrete methods for the design of secure automotive networked embedded systems on the supplier level. This article describes a case study with actionable steps for designing secure systems and systematically eliciting traceable cybersecurity requirements to address this gap. The case study is aligned with the ISO/SAE 21434 standard and can provide the basis for integrating cybersecurity engineering into company-specific processes and practice specifications.

Keywords: Cybersecurity, Threat Modeling, Risk Assessment, Verification, Validation, Design Patterns

Categories: D.2.1, D.2.2, D.2.5, D.4.6

DOI: 10.3897/jucs.72367

1 Introduction

Vehicles are moving from isolated and mostly electro-mechanical systems towards connected computers with wheels [Ebert 2009]. This trend is driven by the wish of the

automotive industry to offer innovative mobility services and further progress towards partially and fully automated vehicles. These goals are most probably only reachable with cooperative and automated vehicles [EC 2016]. Nevertheless, to achieve safe, automated, and interacting vehicles, cybersecurity needs to be further improved [BMW 2019][Macher 2019 2].

Recent evaluations and disclosures presented multiple vulnerabilities in almost all connected elements in current vehicles [Miller 2015][Ring 2015][Strobel 2018], as shown by the following example: Today, car manufacturers act as local internet registries (LIR) and maintain a cluster of IP addresses, certificates, and certificate management systems to secure remote access to external infrastructure and services from vehicles in their fleets. To that purpose, modern vehicles use a Linux-based gateway server to implement remote access to external services. Each vehicle's gateway server is configured with individual key material and certificates during the End-of-Line (EOL) production process. Besides the gateway servers, vehicles offer various other protocols to access on-board functions such as the on-board diagnostics (OBD) interface and services, WLAN, Bluetooth, and dedicated Vehicle2X communication protocols. These connectivity features broaden the attack surface of modern vehicles, making them susceptible to attacks from outside [Macher 2017, Macher 2017 2, Macher 2019, Messnarz 2017, Messnarz 2020].

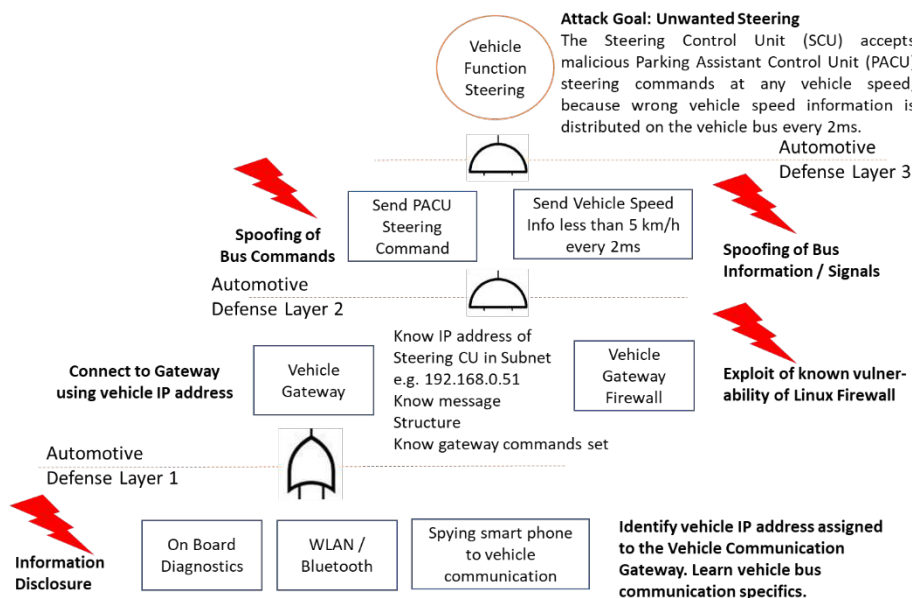


Figure 1: Example of a multi-layer remote cybersecurity attack on connected vehicles

Figure 1 shows such a remote cybersecurity attack that was successfully executed on a vehicle available on the market [Miller 2015]. In this multi-layer attack, vulnerabilities in the remote interfaces are exploited to gain access to the vehicle's on-board functions, ultimately leading to unwanted vehicle steering. The depicted attack consists of the following steps:

1. **Information disclosure:** In the first step, attackers use the same vehicle model for attack preparation. By sniffing vehicle bus messages via the OBD interface, the attackers can learn the specifics of the bus communication (i.e., parts of the message catalog). In addition, the IP address cluster used by the vehicle manufacturer can be identified. For a subsequent attack, the specific IP address of the vehicle under attack is obtained by sniffing the communication between the target vehicle and the vehicle owner's smartphone.
2. **Elevation of privilege:** In the next step, the known vulnerabilities of the Linux gateway's firewall are exploited to gain root privileges enabling the attacker to gain access to the vehicle's internal network. At this stage, it is possible to use the gateway to send malicious commands to the internal vehicle network from a remote device and deploy malicious software on the gateway.
3. **Spoofing:** In the final step, the vehicle is attacked by executing a script previously deployed on the gateway. This script sends messages to the vehicle's internal network containing malicious but appropriately assembled steering control commands and vehicle speed information. The message structure was identified in the first attack step and is now used in a so-called spoofing attack to emulate the behavior/messages of other bus participants. In the specific example, the emulated vehicle speed information (i.e., the vehicle is driving at a speed less than 5 km/h) tricks the steering control unit (SCU) to accept steering commands from the parking assistance control unit (PACU), even in a high-speed driving situation. Finally, enabling the attacker to steer the vehicle remotely.

1.1 Problem Statement

Nowadays, only a limited number of gateway server development companies closely cooperate with original equipment manufacturers (OEMs) in designing the vehicle electrical/electronic (E/E) architecture and the vehicle-fleet security service architecture (SSA). Therefore, most automotive development is still related to delivering electronically controlled networked embedded vehicle functions that implement, e.g., vehicle steering, acceleration, and braking.

These functions need to be integrated into the SSA and E/E architecture in a secure manner to mitigate emerging security threats. Cybersecurity is thus becoming a cornerstone for the success of the automotive industry, alongside reliability and safety. This fact is mirrored in recent cybersecurity regulations that require proof of cybersecurity to introduce vehicle types to the market (i.e., UNECE Type Approval, particularly UNECE WP.29 / R155 and UNECE WP.29 / R.156).

In that respect, the automotive industry has advanced developing and implementing secure connected vehicles by leveraging experience from other sectors. However, the sector is still facing several specific challenges. Therefore, there is a considerable drive towards developing industry standards that address cybersecurity engineering to protect modern connected vehicles and the mobility infrastructure against cyberattacks [Schmittner 2019].

While these standards do not offer details on the development and implementation of cybersecurity, they represent a framework that provides general guidance. The upcoming ISO/SAE 21434 standard for road vehicle cybersecurity engineering was

developed with the understanding that for large parts of the cybersecurity process activities, there is a lack of established state of the art. Therefore only a framework can be given in the standard. This framework needs to be completed in the future to provide concrete guidance for establishing a secure development process.

1.2 Objective

This article aims to propose actionable steps and methods that are aligned with the automotive domain standards and regulations applicable to the secure design of electronically controlled networked vehicle functions. These steps should complement the standards and provide engineers practical guidance on the development of secure vehicle functions. The focus of this work is on engineering aspects, including the risk-driven system design and the systematic elicitation of requirements addressing the identified cybersecurity risks at different stages in the automotive development workflow. The workflow steps addressed include the system and architecture design phase, the HW/SW design phase, and the HW/SW detailed design phase, as shown in Figure 2. The case study shall demonstrate the application of security-driven analysis methods to create the necessary evidence (i.e., work products) to elicit cybersecurity requirements supported by a solid evidence-based and standard-aligned argument.

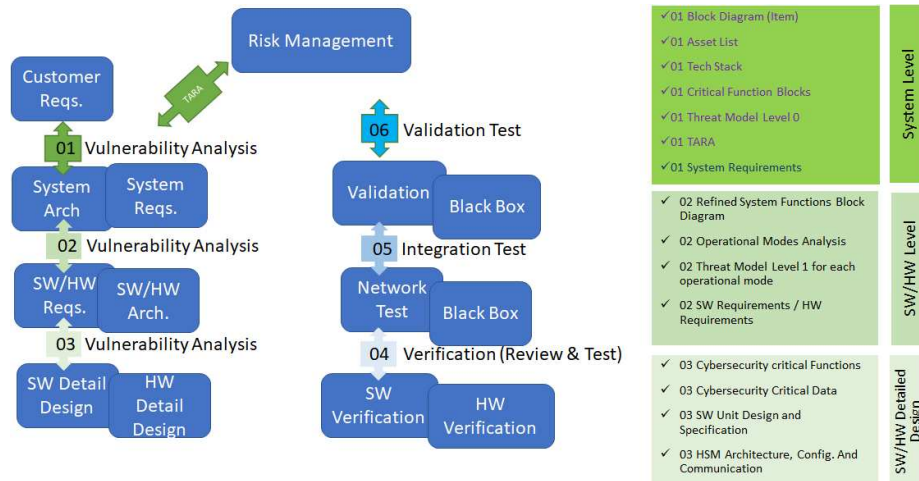


Figure 2: Cybersecurity development workflow and supporting work products

1.3 Case Study

The case study focuses on the development of a steering control unit (SCU) responsible for controlling vehicle steering. The SCU is highlighted in Figure 3, which exemplifies the scope of a typical automotive Tier 1 project targeted towards the development of a specific vehicle function. From a security point of view, the SCU shall be designed to be integrated into the given E/E architecture securely.

As depicted in Figure 3 and introduced at the beginning of Section 1, modern connected vehicles use gateway servers to establish remote connections to, e.g.,

roadside infrastructure and other vehicles. Today's vehicle E/E architectures cluster the on-board communication network into different subnetwork domains to separate the safety-critical networked vehicle functions from minor or non-critical entertainment and communication functions. Nevertheless, to ensure proper operation of the overall vehicle, the individual electronic control units (ECUs), responsible for controlling and coordinating the overall vehicle behavior, must be interconnected. To that purpose, the gateway server and the subnetwork domain controllers are interconnected via real-time Ethernet, CAN FD, or both. The individual ECUs within a specific network domain still rely on CAN FD communication to guarantee short start-up times and real-time control in millisecond time frames.

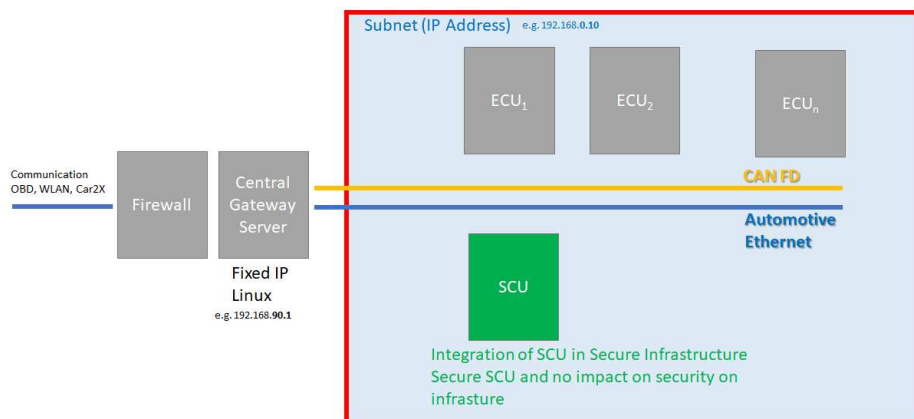


Figure 3: Typical supplier integration into a secure service architecture (SSA)

This clear separation into different network domains provides additional layers of defense against cyber-attacks. It also allows using already established automotive development processes and tools for the engineering of safety-critical electronically controlled vehicle functions. However, no established best-practices guidelines or security engineering processes for secure system design exist in the automotive domain. Hence, the automotive suppliers have to adapt their development lifecycles to also integrate cybersecurity engineering aspects into their processes landscape [ISO 21434 2020][ASPICE 2021][Messnarz 2016][Messnarz 2017][Riel 2018].

In the remainder of this work, the proposed security-driven engineering steps and methods for such an adaptation are described. These steps and methods are aligned to the ISO/SAE 21434 standard [ISO 21434 2020] (DIS version available since Feb. 2020). The main normative activities required by the standard include the item definition and the threat analysis and risk assessment (TARA).

The item definition of the electronically controlled electric power steering system is shown in Figure 4. It defines the project scope and the scope of the case study. In addition, it describes the system components and the internal and external interfaces of the system under development (SUD) on a conceptual level. From a security perspective, the item definition can support the systematic identification of critical assets. According to ISO/SAE 21434, an asset is defined as an element of the SUD whose compromise of cybersecurity properties may damage an item's stakeholder, i.e.,

a person or organization affected by the damage. Hence, elements in the item definition are often identified as assets that need protection because a successful attack on them typically significantly impacts the vehicle's functionality, safety, or cybersecurity.

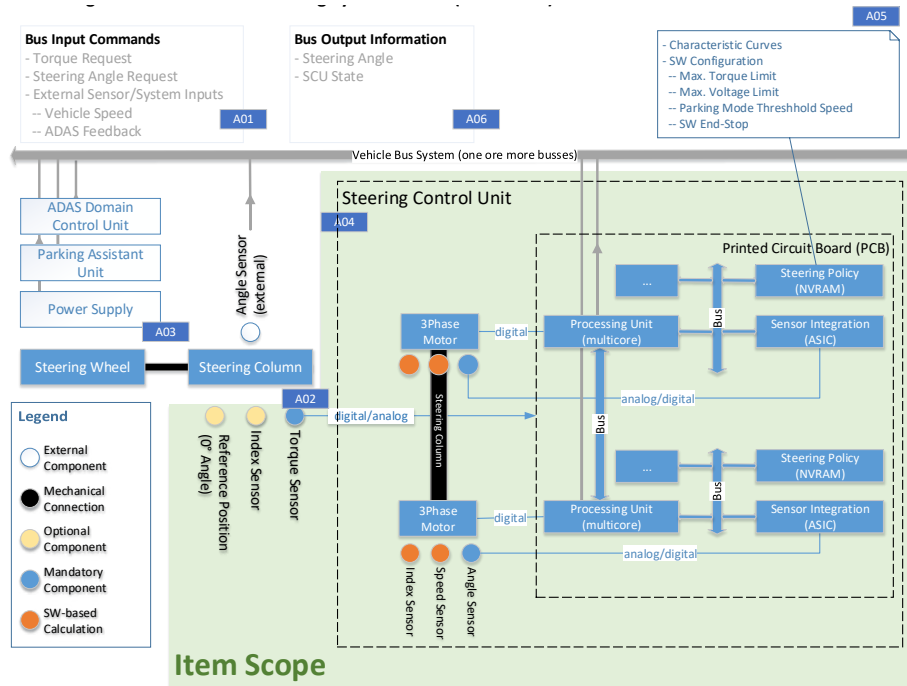


Figure 4: Item definition of the steering control unit (SCU)

The case study example is based on two parallel operating 3-phase motors, each mounted on the steering rack. Two master/slave processing units, both developed according to ASIL D constraints, are used to control the 3-phase motors. The processing units are connected to the vehicle bus for information exchange with other ECUs. The cybersecurity-critical system assets in the given item definition are labeled A_{xx} and are clustered into groups. The system architecture present is designed to meet the safety and fail-operational requirements of modern highly-automated vehicles [Dobaj 2020][Macher 2019 3][Messnarz 2019] [Veledar 2019].

2 System-Level Cybersecurity Engineering

A general strategy in cybersecurity engineering is related to the fact that every accessible system interface increases the potential attack surfaces. Hence, the better understanding of system interfaces and their associated cybersecurity threats, the higher the likelihood of appropriately addressing cybersecurity risks and defending the system when under attack. In addition, the analysis of potential attack paths starting at the

interface level helps to understand and assess risks and the impact of attacks on stakeholders.

2.1 TARA Preparation

The threat analysis and risk assessment (TARA) goal is to identify potential cybersecurity threats of the SUD and the subsequent assessment of risks associated with the identified threats. Before conducting the TARA, some pre-analysis steps shall be performed, including (a) the cybersecurity item definition described in the previous section, the derivation of additional information from (b) the critical function block analysis, which describes the primary system functions, (c) the analysis of the technology stack in use, detailing the hardware and software building blocks for implementing specific cybersecurity mechanisms, and (d) threat modeling to represent the adversarial cyber threats inherent to every cyber(-physical) system.

These pre-analysis steps help obtain a broader understanding of the system, its potential attack interfaces, and the required knowledge and technical skills an attacker may need to achieve its goal. In addition, system assets and cybersecurity-related risks can be elaborated more appropriately, which also facilitates the proper definition of cybersecurity goals and the selection of mitigation measures.

In short, before conducting the TARA, first, the system assets shall be identified using the (semi) structured approaches (a) to (d). In the following, the outcome of these approaches is explained.

Figure 5 shows an excerpt of assets identified through the cybersecurity item definition, critical function analysis, and technology stack analysis. The resulting system-level asset list includes, e.g., interfaces, firmware, configuration data, cryptographic key material, function blocks, sensors, and actuators. The asset analysis is the primary input to the subsequent TARA.

Trace to System Diagram	Asset ID	Ref	Asset Name	Asset Type	Direction	ASIT Rating	Analysis Level	Vehicle Operating Mode	Description & Conditions	Applicable Threats
A01	1	A01.1	ADAS Steering Angle Request	External Interface	Digital Input	D	Item	- Normal Operation - Parking Mode	Active Steering Mode (Vehicle Speed > 10 km/h)	-T-ID-
A01	2	A01.2	ADAS Steering Torque Request	External Interface	Digital Input	D	Item	- Normal Operation - Parking Mode	Active Steering Mode (Vehicle Speed > 10 km/h)	-T-ID-
A01	3	A01.3	Parking Assistant Steering Angle Request	External Interface	Digital Input	D	Item	- Normal Operation - Parking Mode	Parking Mode (Vehicle Speed <= 10 km/h)	-T-ID-
A01	4	A01.4	Parking Assistant Steering Torque Request	External Interface	Digital Input	D	Item	- Normal Operation - Parking Mode	Parking Mode (Vehicle Speed <= 10 km/h)	-T-ID-

Figure 5: Excerpt of the asset list

The system-level threat model (level 0 threat model) supports developing a representation of adversarial threats inherent to the SUD. Thus, system-level threat models are valuable tools for structuring and guiding the asset identification process. In addition, threat models facilitate the systematic (and automated) derivation of potential threats associated with each element in the model. The aim, therefore, is that system-level threat models shall be developed to represent the system-level assets (e.g., data storages, data flows, services, people, and functions) identified during the previous brainstorming-like steps and missing elements/assets shall be added.

The system-level threat model of the SCU is exemplified in Figure 6. It reflects the system structure of the item definition and including the previously identified asset groups. The depicted threat model is based on the STRIDE approach and allows only a

limited set of model elements, which reduces the model complexity and enables the automated derivation of threats associated with each model element representing a potential system asset.

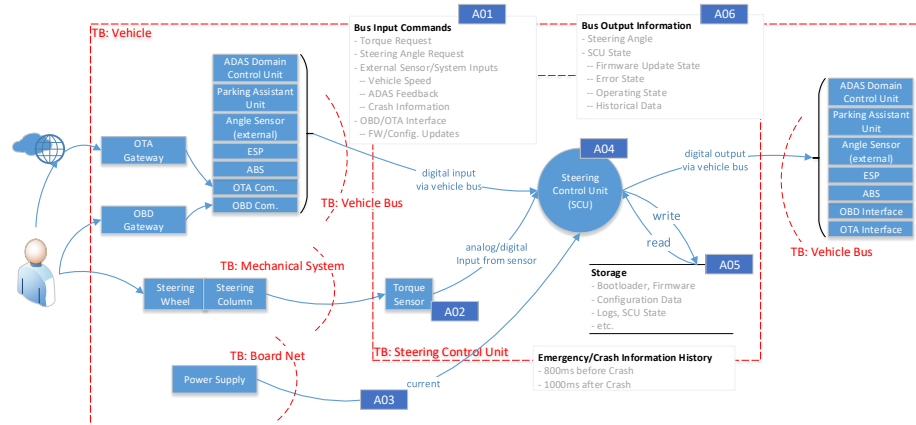


Figure 6: Level 0 threat model of the steering control system

Initially intended for IT system threat analysis, the STRIDE approach has been adopted for automotive applications [Macher 2019] [Messnarz 2020]. Table 1 gives an overview of the STRIDE threats, the cybersecurity property affected by a specific threat, and an automotive-specific example for each threat group. Using the STRIDE approach, threats applicable to specific elements/assets contained within the system-level threat model can be (automatically) identified. The risk associated with each threat is analyzed in the subsequent TARA.

STRIDE Threat	Security Property	Threat Example
Spoofing	<i>Authentication</i> : Imitation of something/someone different	An ECU pretends to be a different ECU by sending CAN-messages with a sender-ID of another ECU
Tampering	<i>Integrity</i> : Manipulation of data at rest (e.g., code) or in transition (e.g., CAN-msgs.)	Unprotected CAN-messages may be intercepted, changed/manipulated, and resent.
Repudiation	<i>Non-Repudiation</i> : Claim to have something (not) done	An ECU could claim to have sent a message even this is not true by, e.g., changing log files.
Information Disclosure	<i>Confidentiality</i> : Disclosure of secret information	A communication protocol may leak confidential information or key material.
Denial of service	<i>Availability</i> : Denial or degradation of a service	An ECU continuously holds the CAN bus data lines "high".

Table 1: STRIDE threats and their affected security properties

2.2 TARA Execution

The threat analysis and risk assessment (TARA) shall be performed based on identified assets and associated threats. A schema shall be used to classify threats and to estimate and classify the threat risk appropriately. Since the ISO/SAE 21434 does not provide a normative classification schema for risk assessment, methods such as Heavens [Mafijul, 2014] or SAHARA [Macher et.al. 2015] may be applied. In this work, the TARA steps are based on the Heavens method. The TARA execution's individual steps and work products are shown in Figure 7 to Figure 11.

Figure 7 shows an excerpt of an asset description, the STRIDE threats applicable to the assets, and the description of specific threat scenarios. For each asset, multiple STRIDE threats may be applicable. Plausible threat scenarios shall be derived from the system-level threat model and described for each applicable threat.

In the particular example shown in Figure 7, the steering angle request transmitted by the ADAS domain controller via the vehicle bus depicts the asset to be analyzed. This asset is modeled in Figure 6 as a data flow element that is part of the asset group A01. According to the STRIDE approach, the following three threats apply to a data flow (i.e., the asset to be analyzed): tampering, information disclosure, and denial of service (TID). Figure 7 describes two threat scenarios, one for the tampering and one for the information disclosure threat. In general, a threat scenario description shall not contain (technical) details of the envisioned attack. Instead, a threat scenario shall be a high-level description of an envisioned threat that shall be checked for its plausibility using the threat models created.

Asset Description										Threat Analysis	
Asset ID	HEAV	Asset Name	Description	Asset Type	Direction	ASIL Rating	Analysis Level	Vehicle Operating Mode	Description & Conditions	Threat Type	Threat Scenario
ID	ENS										
A01.1	1	ADAS Steering Angle Request		External Interface	Digital Input	D	Item	- Normal Operation - Parking Mode	Active Steering Mode (Vehicle Speed > 10 km/h)	-T-ID-	Tampering An attacker pretends to be the ADAS system and sends malicious steering angle request to the SCU.
A01.1	2	ADAS Steering Angle Request		External Interface	Digital Input	D	Item	- Normal Operation - Parking Mode	Active Steering Mode (Vehicle Speed > 10 km/h)	-T-ID-	Information Disclosure An attacker sniffs the messages on the bus and can learn about the command structure and implemented security measures.

Figure 7: Threat scenario description

The impact on stakeholders (i.e., the impact level (IL)) and the likelihood of a successful attack (i.e., the threat level (TL)) shall be estimated for each threat scenario. The two obtained ratings are combined to a single risk value (i.e., the security level (SecL)) that classifies the risk of a specific threat scenario and its associated asset (i.e., the risk linked to a specific asset and threat pair).

To appropriately estimate the impact of a specific threat scenario on stakeholders, it is recommended to define potential damage scenarios describing the anticipated damage to the system and the effect on stakeholders. An example of such a damage scenario description is shown in Figure 8.

			
Damage ID	Ref	Damage Scenario Short Description	Damage Scenario Long Description
D001	D001 - Item Function: Unwanted steering/vehicle behavior	Unwanted steering/vehicle behavior	Manipulation of SCU functionality could result in unwanted steering behavior.
D012	D012 - Item Input: Non-Repudiation	Non-Repudiation	It can not be trustworthy shown that the input commands received by the SCU were wrong or manipulated.

Figure 8: Description of potential damage scenarios for impact analysis

The Heavens method uses four classification parameters to estimate the damage scenario's impact and the expected loss for different stakeholders: (a) impact on system safety, (b) financial impact, (c) impact on vehicle operation/behavior/function, and (d) privacy and legislative impacts. These impacts are estimated using a qualitative scale from 'no impact' to 'high impact' and are then combined to the final impact level rating. An example classification of the "Unwanted steering/vehicle behavior" damage scenario, including a justification, is shown in Figure 9.

Impact Analysis					
Safety Damage to Road User	Financial Damage to Customer or Supplier	Operational Damage to Vehicle or Road User	Privacy & Legislation Damage to Customer, Supplier, or Road User	Justification	IL [HUM] Impact Level
Life-threatening	Medium	High	No Impact	S: Manipulation of item functionality leads to life-threatening situations F: Manipulation of item functionality leads to product recalls O: Manipulation of item functionality leads to unwanted vehicle behavior P: Abuse or loss of item functionality leads to extensive media coverage	1200 High
No injury	Medium	No Impact	High	S: Absence of non-repudiation has no impact on system safety F: Absence of non-repudiation might have an impact if logging data is wrong or if invalid commands are accepted by the item or other items and the customer/supplier can not trustworthy show that the blamed item operated correctly O: Absence of non-repudiation has no impact on vehicle operation P: Absence of non-repudiation might result in legislative issues if life-threatening accidents happened and e.g., logging data can not be found or is not trustworthy	200 Low

Figure 9: Classification of a damage scenario's impact level rating

In general, identified damage scenarios can be mapped to multiple asset/threat pairs and their associated threat scenarios. The damage/impact analysis solely considers the most severe damage scenarios. Consequently, the highest impact level rating is assigned to the specific threat scenario, as shown in Figure 10.

Threat Analysis		Damage Impact Analysis									
Applicable Threat (STRIDE)	Threat Scenario	Damage Scenario 1	DS 1 Impact Level	Damage Scenario 2	DS 2 Impact Level	Damage Scenario 3	DS 3 Impact Level	Damage Scenario 4	DS 4 Impact Level	DS 5	DS 5 Impact Level
Tampering	An attacker pretends to be the ADAS system and sends malicious steering angle request to the SCU.	D002 - Item Input: Unwanted steering/vehicle behavior	High	D012 - Item Input: Non-Repudiation	High						
Information Disclosure	An attacker sniffs the messages on the bus and can learn about the command structure and implemented security measures.									No	

Figure 10: Linking of potential damage scenarios to threat scenarios

Besides the impact rating, the likelihood of a successful attack shall be estimated, which is represented by the threat level (TL) rating. This rating is based on the estimation of the following four parameters: (a) the expertise required by an attacker, (b) the attacker's required knowledge about the target system, (c) the given window of

opportunity to mount an attack, and (d) the equipment and tools required to mount the attack. The system-level threat model shall be used as a supportive tool to estimate and justify the threat level classification.



Figure 11: Classification of the threat level rating and the resulting security level

The left part of Figure 11 shows the classification and justification of the threat level rating linked to the tampering threat associated with the ADAS steering angle request message (i.e., the asset). The right part of Figure 11 shows the resulting security level rating that is a combination of the threat level and impact level. The security level rating describes the cybersecurity risk associated with a given asset/threat pair. For each applicable threat, a cybersecurity goal shall be defined. This cybersecurity goal defines the high-level cybersecurity properties that shall be implemented to reduce the risk of the identified threats. The pair of security level and cybersecurity goal can be considered similar to the concept of the ASIL rating and safety goal pair known from automotive safety engineering.

Cybersecurity goals can be translated to high-level cybersecurity requirements that shall reduce the cybersecurity risk by preventing the exploitation of potential threats. It should be noted that the assessment of several asset/threat pairs may result in the definition of similar cybersecurity goals. The cybersecurity requirements derived from such cybersecurity goals shall inherit the highest determined cybersecurity level associated with these cybersecurity goals.

3 Cybersecurity Requirements Elicitation and Traceability

Cybersecurity requirements arise from cybersecurity goals and customer demands, such as the BMW group standard GS 95014 or the VW KGAS standard state requirements for cyber-security (and functional safety) engineering. These requirements can include functional and non-functional requirements, which can also be related to specific development processes to, e.g., ensure compliance with automotive (cybersecurity) standards. Such demands shall be integrated into the project's requirements management system as cybersecurity customer requirements.

The cybersecurity goals identified within the TARA depict an essential source to establish a common understanding of cybersecurity between customer and supplier, which is considered especially important at the beginning of a project. Therefore, the

cybersecurity goals shall be aligned with the customer and subsequently management like customer requirements. To ensure traceability between requirements and design decision, the cybersecurity customer requirements shall be linked to the cybersecurity goals and the intermediate work products used to support the TARA process. In an Automotive SPICE-aligned development process, typically, a template for requirements specification is used. This template shall be extended to include relevant cybersecurity attributes such as the cybersecurity goal identifier and the security level. These attributes can support prioritizing cybersecurity-related development activities and can be used to demonstrate the coverage of cybersecurity requirements, e.g., during a homologation assessment [Ekert et.al. 2020].

A link/traceability model is exemplifying in Figure 12. The model shows the trace links between work products created during system analysis and the requirements derived from the analysis results (i.e., the work products). The numbers in green circles indicate the cybersecurity engineering workflow steps. The red lines indicate links and traceability identifiers between the steps. The first four steps are explained in Section 2. The remaining steps are explained in the following. Before that, a summary of the requirements elicitation workflow steps is given:

1. The asset shall be identified using brainstorming and system modeling, as described in Section 2.1.
2. Threats to assets shall be identified and analyzed using threat modeling techniques. In addition, the risks associated with these threats shall be assessed, as described in Section 2.1 and Section 2.2.
3. Cybersecurity goals shall be derived from the TARA (i.e., steps 1 and 2), as described in Section 2.2.
4. Customer cybersecurity requirements shall be derived from cybersecurity goals and the customer's requirements specification, as described in Section 3.
5. Cybersecurity customer requirements shall be refined/translated to (technical) cybersecurity system requirements.
6. HW/SW functions that affect system cybersecurity (e.g., sending/receiving data from the bus, cryptographic algorithms) shall be identified. These functions are subsequently denoted as security-critical functions (SCF).
7. The purpose of each function is to processes information/data. Suppose the information/data processed by a SCF affects system cybersecurity, such as cryptographic material, configuration data, and bus messages. In that case, this information/data shall be considered as security-critical data/information (SCD) and linked to the SCF that is processing the SCD.
8. Each SCD shall be linked to its SCFs processing the SCD and vice versa.
9. HW/SW cybersecurity objectives shall be assigned to the SCF and SCD elements. The assignment of the cybersecurity objectives can be used to identify weaknesses in the system and its design by comparing the assigned cybersecurity objectives with the SUD's cybersecurity properties determined by the cybersecurity goals identified during the TARA. The identified weaknesses shall drive the improvement of the system design and elicitation of HW/SW requirements.
10. Appropriate security design patterns and mitigation techniques shall be identified and integrated into the system design to address identified weaknesses. These patterns and techniques shall be documented as HW/SW

requirements and linked to the corresponding SCD and SCF elements that shall implement the cybersecurity properties achieved through a specific pattern.

11. Attack patterns describe strategies and techniques used in known cybersecurity attacks. These attack patterns can be used to support the derivation of appropriate design patterns and mitigation techniques against known attacks.
12. Cybersecurity design patterns and attack patterns can be used for systematic security test derivation based on known attacks and exploits targeted explicitly towards the SUD and its implemented mitigation techniques.

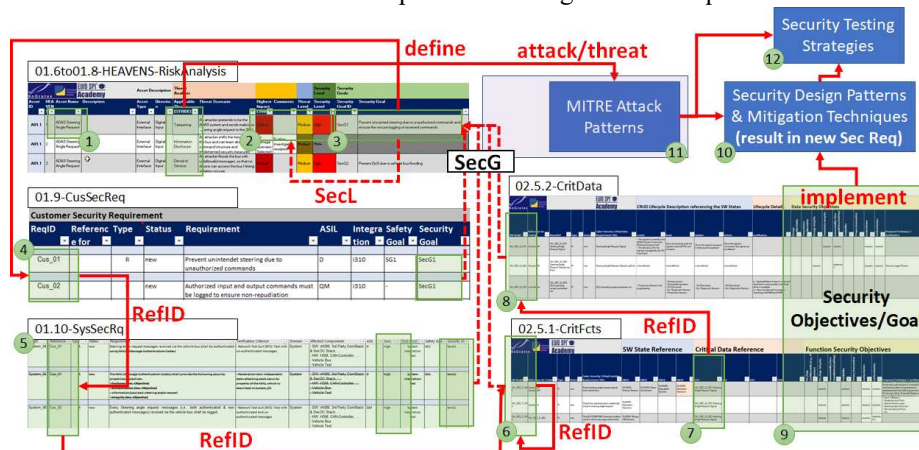


Figure 12: Link/traceability model including cybersecurity requirements and work products from cybersecurity analysis

In the following, the SCU example is continued with workflow step 5, i.e., the refinement of cybersecurity customer requirements to (technical) cybersecurity system requirements. At this stage, cybersecurity goals have been defined and linked to assets. In addition, these goals have been integrated into the requirements management system as cybersecurity customer requirements. These steps relate to the entries one to four in Table 2, which provides an excerpt of dependent work products and requirements.

In the example given, the ADAS steering angle request is classified as a high-risk asset. The high-level security goal SecG_01 is defined to address this risk. Next, the security goal is split into two customer requirements and their corresponding system requirements Sys_01 and Sys_02. Following the workflow steps 7 to 10, the SCF and SCD elements shall be identified next. The SCF and SCD elements are intended to support the system analysis and requirements elicitation process. Hence, these elements are not necessarily requirements describing the implementation of specific mechanisms. Instead, these elements can also be high-level concepts, functions, techniques, and information that can be used to build a structured argument for design decisions. Based on such an argumentation, the (detailed) HW/SW design shall be developed and the corresponding requirements can be derived, i.e., steps 9 and 10. These steps are explained more explicitly in the following paragraphs.

Table 2 states that the reading of messages from the vehicle bus is identified as security-critical function SCF_01. To meet Sys_01, critical vehicle bus messages must be authenticated in a secure manner, which results in the definition of the software requirement SW_Req_01 stating that the standard AUTOSAR interfaces shall be used

for message authentication. In addition, hardware requirements one and two are defined to ensure proper support for the implementation of cryptographic algorithms (e.g., the secure message authentication algorithm) on the hardware level.

A similar approach is applied to derive the HW/SW requirements implementing the system requirement Sys_02, which states that vehicle bus messages shall be securely logged to ensure the non-repudiation property (i.e., SCF_02). To that purpose, every vehicle bus message (i.e., SCD_01) is embedded into a log entry (i.e., SCD_02), which shall be written to an in-memory first-in-first-out (FIFO) queue (i.e., SW_Req_02). This FIFO queue shall only be stored to the secure flash memory at every system shut down to prevent the destruction of the flash memory due to heavy usage (i.e., SW_Req_03). To ensure that the content of the FIFO queue is written to flash memory even under harsh conditions (e.g., loss of power supply), the hardware security module (HSM) shall provide hardware-accelerated writing to the flash memory (i.e., HW_Req_03).

#	Type	Description
1	Asset	ADAS Steering Angle Request (part of asset group 1, i.e., the input messages received via the vehicle bus)
2	SecL	High
3	Security Goal	<i>SecG_01</i> : Prevent unwanted steering due to <u>unauthorized commands</u> and <u>ensure the secure logging</u> of commands received via the vehicle bus
4	Customer Requirements	<i>Cus_01</i> : Prevent unintended steering due to <u>unauthorized commands</u> <i>Cus_02</i> : All received commands shall be <u>securely logged</u> to <u>ensure non-repudiation</u>
5	System Requirements	<i>Sys_01</i> : Security-critical messages received via the vehicle-bus shall be <u>authenticated</u> to ensure <u>message integrity</u> <i>Sys_02</i> : Security-critical messages received via the vehicle-bus shall be <u>securely logged</u> to <u>ensure non-repudiation</u>
6	SCF and (detailed) HW/SW Requirements	<i>SCF_01</i> : Read critical message from the vehicle bus <i>SCF_02</i> : Secure logging of critical vehicle bus <i>SW_Req_01</i> : Use the AUTOSAR Secure On-board Com. (SOC) runtime interface to validate the <u>authenticity</u> of security-critical vehicle bus messages <i>HW_Req_01</i> : The hardware security module (HSM) shall provide a <u>secure storage</u> to ensure <u>confidentiality</u> of stored cryptographic key material. <i>HW_Req_02</i> : The HSM shall provide a <u>secure execution zone</u> (trust zone) for critical cryptographic algorithms, e.g., to ensure <u>confidentiality</u> of keys.
7	SCD and (detailed) HW/SW Requirements	<i>SCD_01</i> : Steering angle request signal embedded into a particular vehicle bus message <i>SCD_02</i> : Vehicle bus message log entry incl. the steering angle request <i>SW_Req_02</i> : Critical vehicle bus messages shall be written into an in-memory FIFO queue that is stored in the secure flash memory on shut down. <i>SW_Req_03</i> : The in-memory FIFO queue shall be capable of holding messages received within a 2 second time windows (i.e., to store messages received between +/- 1000 ms). <i>HW_Req_03</i> : The HSM shall support hardware-accelerated reading/writing to <u>secure flash memory</u> .

Table 2: Example of associated analysis results and requirements. The cybersecurity properties to be achieved are highlighted through underlining

The described example should demonstrate how the above-defined workflow can be used to build traceable arguments to support design decisions and elicit related

requirements in a structured manner. The workflow steps 9 to 12 are intended to support the requirements elicitation process by providing guidance in identifying design weaknesses and selecting appropriate cybersecurity measures and techniques addressing these weaknesses to meet the desired system cybersecurity properties (i.e., the security goals).

To identify design weaknesses (i.e., step 9), first, the cybersecurity properties provided by the current system design shall be identified and assigned to the SCF and SCD elements. It is recommended to use system and threat models for the identification of these properties. Second, the cybersecurity objectives (i.e., the desired system cybersecurity properties) shall be assigned to the SCF and SCD elements. The assigned desired and provided properties shall be compared to identify gaps/weaknesses in the system design in the third step.

So far, it has been discussed how cybersecurity properties can be used for the systematic and traceable identification of design weaknesses. In step 10, cybersecurity properties are used to improve system security through the systematic selection of security design patterns based on the cybersecurity properties the system shall meet. This approach is similar to the one proposed in [Scan. 2008], where patterns are selected to achieve specific cybersecurity properties. Table 3 shows an excerpt of such security design patterns (i.e., high-level security controls) that can be used for implementing a specific cybersecurity property on a particular system level. Some of these security controls are already discussed above, such as control-flow integrity (i.e., Sys_01 and related requirements), secure storage, and encryption of data at rest (i.e., Sys_02 and related requirements).

Summing up, by selecting appropriate security design patterns and mitigation mechanisms, the security of the SUD can be improved. In addition, the patterns can support the elicitation of proper HW/SW security requirements and security test scenarios in a structured and traceable manner (see Figure 12).

Security Property	Environmental Level Security Control	Vehicle Level Security Control	Component Level Security Control
Integrity	Integrity management of access rights	- Secure communication, TLS, IPsec, etc. - Functional separation and trusted execution of the control flow	- Access control - Control-flow integrity - Trust anchor
Authenticity	- Access control to dev. and production sites - Secure communications	- Message authentication codes etc.	- Secure boot with a trust anchor, e.g., public key hash in OTP fuses
Availability	- Intrusion detection mechanisms to react to potential attacks	- Congestion control on gateways/routers	- Rate limiting on network interfaces - Deterministic scheduling
Confidentiality	- Access control to documentation	- Encryption of data in flight - TLS, IPsec, etc.	- Encryption of data at rest - Secure storage

Table 3: Example for security design patterns and security controls applicable at various system levels (adapted from [BMW 2019])

4 Cybersecurity Prospects on Hardware and Software

The previous section discusses the process of systematic cybersecurity systems and requirements engineering, which includes the derivation of appropriate cybersecurity controls to meet specific cybersecurity goals. This section briefly discusses how cybersecurity controls, such as those provided in Table 3, are integrated into a typical AUTOSAR-based system stack commonly used within automotive embedded systems.

Figure 13 shows such a system stack, which is structured into multiple layers. The bottom layer consists of the hardware elements for code execution, data storage, and connectivity. The hardware security module (HSM) is also located on this layer. The HSM provides hardware support to securely store and access private key material, which is essential for adequately implementing various higher-level security controls. In addition, an HSM may also provide hardware-accelerated cryptographic algorithms that can be leveraged to meet the stringent real-time requirements of automotive networked systems, even if secure protection of cyclic communication in the millisecond and sub-millisecond ranges is required.

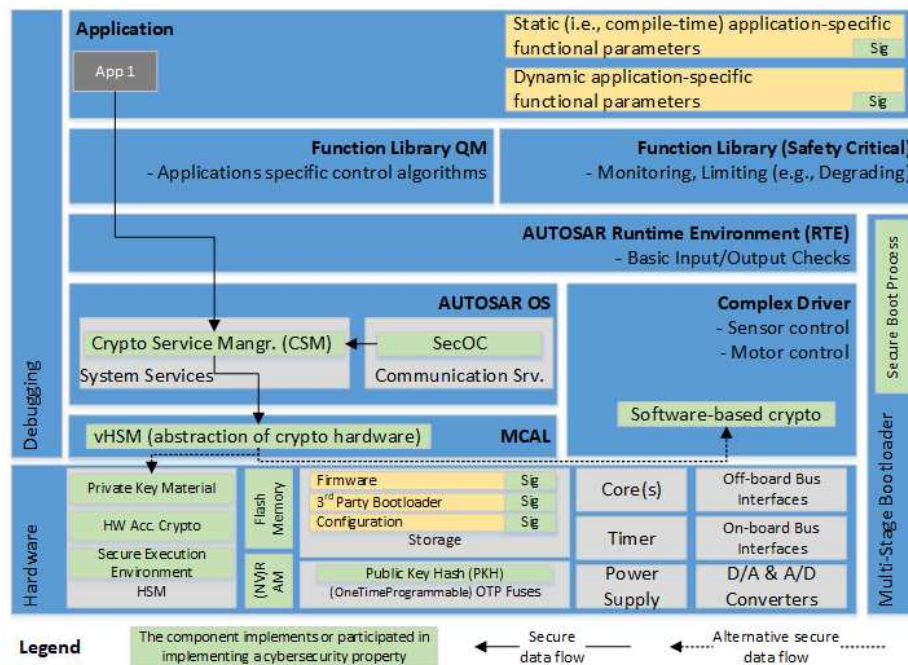


Figure 13: AUTOSAR-based layered HW/SW stack

Like other hardware components, the HSM can be integrated into the software stack via a driver located at the microcontroller abstraction layer (MCAL). In Figure 13, this driver is denoted as virtual HSM (vHSM). Suppose there is no dedicated hardware HSM available. In that case, the HSM functionality can be emulated in software. Therefore, the vHSM can forward service requests to the software-based

HSM implementation located, e.g., at the complex driver layer. It is important to note that the software-based HSM approach is less secure than using a dedicated HSM.

An application (e.g., App 1) can access dedicated cryptographic services of the HSM via the runtime environment (RTE) layer that specifies the interface to the crypto service manager (CSM). The RTE forwards service requests from the application to the system services and the CSM that are located at the AUTOSAR core operating system (OS) layer.

Besides accessing the HSM directly via the RTE, it is also possible to configure the communication services located at the AUTOSAR OS layer, to manage HSM service usage for establishing secure on-board communication (SecOC) sessions. In this case, the SecOC services are configured to, e.g., check and ensure the authenticity of specific messages, making the HSM-based security mechanism transparent to the application layer.

Besides the HSM, several other components and security controls are leveraged to ensure system security. For example, implementing a secure boot process shall ensure the integrity of the overall software system, including system firmware and persistent data. To that purpose, a so-called trust anchor can be established during the EoL production process (see also Section 1) by, e.g., storing a public key hash (PKH) of the primary bootloader within the OTP fuses of the microcontroller. At every system boot, the microcontroller first uses the PKH to verify the integrity of the bootloader. The boot process only continues if the integrity of the bootloader can be verified, i.e., that the bootloader has not been manipulated. In the subsequent boot-stages, firmware and data can be verified by checking the validity of their signature using, for example, the key material stored within the HSM during the EoL production process.

5 Conclusion

Currently, the integration of cybersecurity into the automotive industry's development processes is in its early stages, and there is no generally accepted state of the art yet. The upcoming ISO/SAE 21434 standard is an essential step in this direction. However, at the current stage, the ISO/SAE 21434 provides only rough guidance on how such development processes could be designed. Therefore, this article attempts to provide actionable steps and methods to help engineers integrate secure system design techniques and systematic cybersecurity requirement elicitation approaches into their established development processes. This work uses a case study to demonstrate the application of the proposed steps and methods on the example of an electric power steering system. The case study and workflow presented have been derived from practical expert knowledge combined with results from previous research that has been adapted to the domain of automotive systems engineering. Our future work consists of validating and refining the presented approach in further projects and feeding back the results and insights into the different standardization working groups. In addition, we aim to develop a generic security-driven development lifecycle model that can be used for the development of current and future automotive electronic systems and architectures.

Acknowledgements

This project has received funding from the BLUEPRINT project DRIVES (2018 – 2021) under grant agreement No 591988-EPP-1-2017-1-CZ-EPPKA2-SSA-B and from the H2020 project TEACHING (n. 871385) - www.teaching-h2020.eu. The publication reflects only the author's view and that the funding agency is not responsible for any use that may be made of the information it contains.

We are grateful to the EuroSPI community and conference series (www.eurospi.net) for sharing experience since 1994 and the EU Project ECQA Certified Cybersecurity Engineer and Manager – Automotive Sector, Erasmus+ Programme, Grant Agreement No. 2020-1-CZ01-KA203-078494. This research has been supported by Grant of SGS No. SP2020/62, VŠB - Technical University of Ostrava, Czech Republic.

Many thanks to the SoQrates working party (<https://soqrates.eurospi.net>) for knowledge sharing and exchange. Special thanks goes to: Aschenberger Werner (KTM AG), Bauer-Wukitsevits Robert (msg Plaut), Böhner Martin (Elektrobit), Brasse Michael (HELLA), Bressau Ernst (BBraun), Dallinger Martin (ZF), Dorociak Rafal (HELLA), Dreves Rainer (Continental), Ekert Damjan (ISCN), Forster Martin (ZKW), Geipel Thomas (BOSCH Engineering), Grave Rudolf (Elektrobit), Griessnig Gerhard (AVL), Gruber Andreas (Kreisel Electric), Habel Stephan (Continental), Hällmayer Frank (SF), Haunert Lutz, Karner Christoph (KTM AG), Kinalzyk Dietmar (AVL), König Frank (ZF), Lichtenberger Christoph (MAGNA), Lindermuth Peter (MAGNA), Macher Georg (TU Graz), Maier Bjoern (AVL), Mandic Irenka 8MAGNA), Maric Dijaz (LORIT Consultancy), Mayer Ralf (BOSCH Engineering), Mergen Silvana (TDK EPCOS), Messnarz Richard (ISCN), Much Alexander (Elektrobit), Nikolov Borislav (msg Plaut), Oehler Couso Daniel (MAGNA), Riel Andreas (ISCN Group), Rieß Armin (Braun), Santer Christian (AVL), Schlager Christian (MAGNA), Schmittner Christoph (AIT), Schubert Marion (ZKW), Sechser Bernhard (Process Fellows), Sokic Ivan (Continental), Sporer Harald (Infineon), Stahl Florian (msg Plaut), Wachter Stefan (msg Plaut), Walker Alastair (LORIT Consultancy), Wegner Thomas (ZF), Geyer Dirk (AVL), Dobaj Jürgen (TU Graz), Wagner Hans (msg Group), Aust Detlev (AustCON).

References

- [ASPICE 2021] Automotive SPICE for Cybersecurity, 1st Edition, Feb. 2021, VDA QMC Working Group 13, Feb. 2021
- [BMW 2019] BMW et al., Safety First for Automated Driving, 2019
- [Dobaj 2020] Dobaj J., Seidl M., Krug T., Krisper M., Macher G. (2020) A Seamless Self-configuring EtherCAT Master Redundancy Protocol. In: Yilmaz M., Niemann J., Clarke P., Messnarz R. (eds) Systems, Software and Services Process Improvement. EuroSPI 2020. Communications in Computer and Information Science, vol 1251. Springer, Cham. https://doi.org/10.1007/978-3-030-56441-4_28
- [Ebert 2009] C. Ebert, C. Jones, Embedded Software: Facts, Figures, and Future, Computer 42 (2009) 42–52. <https://doi.org/10.1109/MC.2009.118>.

- [Ekert 2020] Ekert D., Messnarz R., Norimatsu S., Zehetner T., Aschbacher L. (2020) Experience with the Performance of Online Distributed Assessments – Using Advanced Infrastructure. In: Yilmaz M., Niemann J., Clarke P., Messnarz R. (eds) Systems, Software and Services Process Improvement. EuroSPI 2020. Communications in Computer and Information Science, vol 1251. Springer, Cham. https://doi.org/10.1007/978-3-030-56441-4_47
- [EC 2016] European Commission, A European strategy on Cooperative Intelligent Transport Systems, a milestone towards cooperative, connected and automated mobility, 2016
- [ISO 21434 2020] ISO 21434/SAE 21434 DIS, Road vehicles – Cybersecurity engineering, DIS version, Feb 2020
- [Macher 2015] Macher, G., Sporer, H., Berlach, R., Armengaud, E., Kreiner, C.: SAHARA: a security-aware hazard and risk analysis method. In: 2015 Design, Automation Test in Europe Conference Exhibition (DATE), pp. 621–624, March 2015
- [Macher 2017] G. Macher, R. Messnarz, C. Kreiner, et. al, Integrated Safety and Security Development in the Automotive Domain, Working Group 17AE-0252/2017-01-1661, SAE International, June 2017
- [Macher 2017 2] Macher G., Much A., Riel A., Messnarz R., Kreiner C. (2017) Automotive SPICE, Safety and Cybersecurity Integration. In: Tonetta S., Schoitsch E., Bitsch F. (eds) Computer Safety, Reliability, and Security. SAFECOMP 2017. Lecture Notes in Computer Science, vol 10489. Springer, Cham
- [Macher 2019] Macher, G.; Diwold, K.; Veledar, O.; Armengaud, E. & Römer, K. The Quest for Infrastructures and Engineering Methods Enabling Highly Dynamic, Autonomous Systems European Conference on Software Process Improvement, 2019, 15-27
- [Macher 2019 2] Macher G., Schmittner C., Dobaj J., Armengaud E., Messnarz R. (2020). An integrated view on automotive spice, functional safety and cyber-security. In 2020 SAE Technical Papers 2020-01-0145, SAE International
- [Macher 2019 3] Macher, G.; Druml, N.; Veledar, O. & Reckenzaun, J., Safety and Security Aspects of Fail-Operational Urban Surround perception (FUSION), International Symposium on Model-Based Safety and Assessment, 2019, 286-300
- [Macher 2020] Macher G., Schmittner C., Veledar O., Brenner E. (2020). ISO/SAE DIS 21434 Automotive Cybersecurity Standard-In a Nutshell. In 2020 International Conference on Computer Safety, Reliability, and Security, pages 123-135, Springer International
- [Mafijul, 2014] Islam Mafijul, Deliverable d2-security models: HEAVENS Project, 2014
- [Messnarz 2012] Messnarz R., König F., Bachmann V.O. (2012) Experiences with Trial Assessments Combining Automotive SPICE and Functional Safety Standards. In: Winkler D., O'Connor R.V., Messnarz R. (eds) Systems, Software and Services Process Improvement. EuroSPI 2012. Communications in Computer and Information Science, vol 301. Springer, Berlin, Heidelberg
- [Messnarz 2016] Messnarz R., Kreiner C., Riel A., Integrating Automotive SPICE, Functional Safety, and Cybersecurity Concepts: A Cybersecurity Layer Model, Software Quality Professional . Sep2016, Vol. 18 Issue 4, p13-23. 11p., 2016
- [Messnarz 2017] Messnarz R., Much A., Kreiner C., Biro M., Gerner J. (2017) Need for the Continuous Evolution of Systems Engineering Practices for Modern Vehicle Engineering. In: Stolf J., Stolf S., O'Connor R., Messnarz R. (eds) Systems, Software and Services Process Improvement. EuroSPI 2017. Communications in Computer and Information Science, vol 748. Springer, Cham. https://doi.org/10.1007/978-3-319-64218-5_36

- [Messnarz 2019] Messnarz R., Macher G., Stolfa J., Stolfa S. (2019) Highly Autonomous Vehicle (System) Design Patterns – Achieving Fail Operational and High Level of Safety and Security. In: Walker A., O'Connor R., Messnarz R. (eds) Systems, Software and Services Process Improvement. EuroSPI 2019. Communications in Computer and Information Science, vol 1060. Springer, Cham. https://doi.org/10.1007/978-3-030-28005-5_36
- [Messnarz 2020] Messnarz R. et al. (2020) Automotive Cybersecurity Engineering Job Roles and Best Practices – Developed for the EU Blueprint Project DRIVES. In: Yilmaz M., Niemann J., Clarke P., Messnarz R. (eds) Systems, Software and Services Process Improvement. EuroSPI 2020. Communications in Computer and Information Science, vol 1251. Springer, Cham. https://doi.org/10.1007/978-3-030-56441-4_37
- [Miller 2015] C. Miller, C. Valasek, Remote Exploitation of an Unaltered Passenger Vehicle (2015).
- [Riel 2018] Andreas Riel, Christian Kreiner, Richard Messnarz, Alexander Much, An architectural approach to the integration of safety and security requirements in smart products and systems design, CIRP Annals, Volume 67, Issue 1, 2018, Pages 173-176, 2018, <https://doi.org/10.1016/j.cirp.2018.04.022>
- [Ring 2015] M. Ring, J. Durrwang, F. Sommer, R. Kriesten, Survey on vehicular attacks - building a vulnerability database, in: 2015 IEEE International Conference on Vehicular Electronics and Safety (ICVES), Yokohama, IEEE, 11/5/2015 - 11/7/2015, pp. 208–212
- [Scan. 2008] R. Scandariato, K. Yskout, T. Heyman, W. Joosen, Architecting software with security patterns, 2008
- [Schmittner 2019] Schmittner, C. & Macher, G. Automotive Cybersecurity Standards-Relation and Overview International Conference on Computer Safety, Reliability, and Security, 2019, 153-165
- [Strobel 2018] S. Strobl, D. Hofbauer, C. Schmittner, S. Maksuti, M. Tauber, J. Delsing, Connected cars - Threats, vulnerabilities and their impact, in: 2018 IEEE Industrial Cyber-Physical Systems (ICPS), St. Petersburg, IEEE, 5/15/2018 - 5/18/2018, pp. 375–380
- [Veledar 2019] Veledar, O.; Damjanovic-Behrendt, V. & Macher, G., Digital Twins for Dependability Improvement of Autonomous Driving, European Conference on Software Process Improvement, 2019, 415-426