Ref. Ares(2020)4250249 - 13/08/2020

SECREDAS

Product Security for Cross Domain Reliable Dependable Automated Systems

DELIVERABLE REPORT

Security analysis report v2

Document Type	Deliverable
Document Number	D 3.8
Primary Author(s)	Hanna AlZughbi, Tobias Braun
Document Date	11.08.2020
Document Version / Status	1.01
Distribution Level	Public
Reference DoA	April 2020

Project Coordinator	Patrick Pype, NXP Semiconductors, patrick.pype@nxp.com
Project Website:	www.secredas.eu
JU Grant Agreement Number	783119

SECREDAS has received funding from the Electronic Component Systems for European Leadership Joint Undertaking under grant agreement nr.783119. This Joint Undertaking receives support from the European Union's Horizon 2020 research and innovation programme and Austria, Belgium, Czech Republic, Germany, Finland, Hungary, Italy, The Netherlands, Poland, Romania, Sweden and Tunis

CONTRIBUTORS

Name	Organization	Name	Organization
Hanna AlZughbi	Fraunhofer IESE	Stelios Karagiannis	Beyond Vision
Tobias Braun	Fraunhofer IESE	Nadja Marko	VIF
Andrei Manta	Fraunhofer IESE	Bernhard Winkler	VIF
Behrooz Sangchoolie	RISE Research Institutes of Sweden	Joaquim Castella Triginer	VIF
Peter Folkesson	RISE Research Institutes of Sweden	Pedro Lousã	PDMFC
Alexandr Vasenev	TNO	Hayk Hamazaryan	ZF
Behnam Asadi Khashooei	TNO	Joerg Kemmerich	ZF
Andre Smulders	TNO	Mina Sheikhalishahi	TU/E
Sterre den Breeijen	TNO	Nicola Zannone	TU/E
Irina Chiscop	TNO	Calogero Calandra	MM
Cyrille FALCOU	Thales DIS	Salvatore Petroni	MM
Franck Morier	ISSM-THALES DIS Design Services SAS	Sara Potenza	MM

FORMAL REVIEWERS

Name	Organization	Date
Roy Pennings	NXP	15.06.2020, 11.08.2020
SECREDAS Steering Board	N/A	13.08.2020

DOCUMENT HISTORY

Revision	Date	Author / Organization	Description
0.1	01.12.2019	Tobias Braun	Initial Version
0.2	23.01.2020	Hanna AlZughbi	Initial review
0.3	12.05.2020	Hanna AlZughbi	Update APs matrix and other minor changes
0.4	13.05.2020	All	Integration of AP descriptions
0.5	25.05.2020	All	Improvement of AP descriptions, introduction, general alignment
0.6	08.06.2020	Hanna AlZughbi, Tobias Braun	Finalizing document for internal review
0.7	29.06.2020	All	Updates to address review comments
1.0	07.08.2020	Tobias Braun	Update of Section 1.1
1.01	11.08.2020	Tobias Braun	Formatting / Updated contributors list

Executive Summary

Deliverable D3.8 presents 11 analysis patterns that address problems and research questions such as securing safety related solutions, combined safety and security analysis and testing, improving security over the whole life-cycle of embedded systems, enhancing access control in terms of privacy as well as tracing security and privacy principles of the system architecture. Every Analysis Pattern (AP) is presented by the responsible project partner and applied to at least one Design Pattern (DP) of task 3.2. This second and final deliverable of task 3.3 focuses on the results of the development of these APs, along with the identified countermeasures and how they have been utilized for improving the DPs.

Table of Contents

1 Introduction 6
1.1 D3.7 vs. D3.8 7
1.2 Document Outline 8
2 Analysis Patterns 9
2.1 Analysis Patterns Overview 9
2.2.1 General structure of the AP descriptions 11
2.2.1 AP01: Security4Conserts 14
2.2.2 AP02: Secure delivery and flashing of binary update (ZF) 27
2.2.3 AP04: CCPP – Common Criteria Profile Protection 44
2.2.4 AP05: Privacy-preserving Multi-party Access Control 49
2.2.5 AP06: Safety Contents Management (SaCoMa) 56
2.2.6 AP07: Combined Safety and Security Analysis 63
2.2.7 AP08: Pre-injection analysis 82
2.2.7 AP09: Common Risk Assessment methodology 99
2.2.8 AP10: Security Zones approach 109
2.2.9 AP11: Tracing Security and Privacy Principles in Architectures 118
2.2.10 AP13: Embedding Identity and Access Management to the Event Log processes 125
3 Conclusion 134
Tables 135
References 136

Acronyms/Terminology

AES	Advanced Encryption Standard
API	Application Programming Interface
AP	Analysis Pattern
APP	Application
ASIL	Automotive Safety Integrity Level
CAN	Controller Area Network
C-ITS	Cooperative Intelligent Transport Systems
CPS	Cyber-Physical Systems
CTE	Common Technology Element
DFD	Data Flow Diagram
ECU	Embedded Control Unit
EKMS	Electronic Key Management System
eS	Embedded Secure Element
eNVM	embedded Non-volatile Memory
EVITA	E-Safety Vehicle Intrusion Protected Applications
FMEA	Failure Mode Effective Analysis
FTA	Fault Tree Analysis
FOTA	Firmware Over-The-Air
Gateway	A VCU that connects multiple networks within a car
GDPR	General Data Protection Regulation
GUI	Graphical User Interface
HARA	Hazard Analysis and Risk Assessment
HAZOP	Hazard and operability study
HEAVENS	HEAling Vulnerabilities to ENhance Software Security and Safety
HW	Hardware
IA	Improvement-AP
IC	Integrated Circuit
IoT	Internet-of-Things
ISE	Integrated SE
ISO	International Standardization Organization
IT	Information Technology
ITS	Intelligent Transport Systems
LTS	Long-Term Support
MA	Methodology-AP
MBSA	Microsoft Baseline Security Analyzer
MFA	Multi-Factor Authentication
OCTAVE	Operationally Critical Threat, Asset, and Vulnerability Evaluation
OEM	Original Equipment Manufacturer
OS	Operating System
OTA	Over-The-Air
OWASP	Open Web Application Security Project
PIA	Privacy Impact Assessment
PKI	Public Key Infrastructure
SAHARA	Security-Aware HARA
SAE	Society of Automotive Engineering
SE	Secure Element
SSL	Secure Sockets Layer
STAMP	Systems-Theoretic Accident Model and Processes
STPA	Systems-Theoretic Process Analysis
STPA-Sec	STPA for Security
STRIDE	Spoofing, Tampering, Repudiation, Information disclosure, Denial of Service And Elevation of Privilege
SUIT	System under Test
SW	Software
TARA	Threat Assessment and Remediation Analysis
TEE	Trusted Execution Environment
TLS	Transport Layer Security
TMR	Triple Modular Redundancy
TPM	Trusted Platform Module
TVRA	Threat, Vulnerabilities, and Implementation Risks Analysis
UC	Use Case
VCU	Vehicle Control Unit
VPN	Virtual Private Network
V2X	Vehicle-to-X, where X stands for either Vehicle or Infrastructure
WP	Work Package

1 Introduction

The main objective of Work Package 3 (WP3) is to develop and validate Common Technology Elements (CTEs) and Design Patterns (DPs) comprising safe, secure and privacy-preserving components, which will be used as a reference architecture for domain-specific solutions in the development of automated systems. The second objective of WP3 is to develop a framework for security and safety verification/testing of the DPs and CTEs.

WP3 tasks will use the reference architecture (i.e. design principles, from Task 2.3) and the corresponding WP2 security, safety and privacy requirements for the different Use Cases (from Work Package 1) as input for the creation of domain independent DPs. Task 3.1 (T3.1) describes the use of CTEs and shows how they can be used for the reference architecture. The DPs and their usage and combination of CTEs are analysed in this task (T3.3), against safety, security and privacy issues. The results will be used to improve the DPs in Task 3.2 (T3.2), e.g. by incorporating additional CTEs or modifying their usage within the DPs.

CTEs are existing industrial technologies that can be used to develop new solutions, where design principles and best practices are combined and implementation recommendations for the correct use of CTEs are given. T3.1 will develop several CTEs for the reference architecture and support cooperative developments and knowledge exchange between the different domains present in this project (i.e. Automotive, Healthcare and Rail). This way, solutions that address similar problems in different domains can be analysed for common or synergistic approaches and subsequently improved for domain-specific use.

A DP is a best-practice solution template for a specific technical challenge; It describes the use and combination of CTEs (within the context of SECREDAS) as well as their interaction with other domain specific technical solutions. These DPs are then extended (in T3.2) for domain specific technology elements, which allow developers to identify solutions to common problems by adapting these DPs. Linking DPs to CTEs will reduce development effort and unnecessary duplication of work, by using concrete, trustworthy and proven solutions that have been already applied in an industrial context.

The scope of T3.3 is to analyse DPs (considering specific application scenarios) for the underlying safety, security and privacy issues, analyse their effect and assess the impact on the overall system. Corresponding countermeasures and enhancements will be developed and integrated into DPs to maintain the required

safety, security and privacy measures. By having several APs, various safety, security or privacy issues (of pre-selected DPs) can be addressed. This could also contribute to the improvement of different aspects of the same DP. Development of redundant and incompatible enhancements will be avoided by introducing a matrix showing the explicit relationships between developed APs and DPs. The enhanced DPs will later (in T3.4) become part of a combined safety, security and privacy test concept that will ensure a safe and secure use of future system components.

The focus of this task (T3.3) is to analyse the DPs (of T3.2), with the ultimate goal of enhancing those DPs (to be delivered in T3.2). To achieve that, standard analysis methods are used. However, in some cases, the existing analysis methods might not be sufficient to identify specific shortcoming in DPs. Especially if these shortcomings are in the area of tension between security, safety and privacy. To cover these cases, new analysis methods are introduced. To provide a direct link between T3.3 and T3.2, the new methods are applied to one or more DPs. The outcome here is twofold: first, the application on a concrete DP serves as evaluation for the new developed analysis method and example. Second, the results of this analysis can be used by the DP owner to improve their DP and/or as (additional) argument for the security/safety/privacy of their DP. The outcome of this task will be directly used in T3.2 to deliver enhancements for the related DPs. The description of the DP will be updated accordingly and presented in D3.6 of T3.2.

1.1 D3.7 vs. D3.8

Deliverable D3.8 is the successor of D3.7 and builds upon the started work. While D3.7 collected the initial ideas and scope of APs to be developed in T3.3, D3.8 provides a detailed description of the development done during the task runtime. During the refinement phase of the APs, AP03 “Vulnerability Database” and AP12 “Trust Establishment” had to be removed from this task, because the responsible partners who introduced these APs shifted their efforts within the project. Whilst the core topic of AP03 “Vulnerability Database” is still addressed within the project, the contribution of AP12 “Trust Establishment” has been withdrawn by the responsible partner (NOKIA) due to limited capacity.

The “Vulnerability Database” AP is strongly related to penetration testing, which is an important part of the *Security Testing* framework developed in T3.4. Similar to the developed *Threat Library*, the attack scripts also follow the original idea of AP03 regarding a structured way of selecting, tracking and executing attacks and penetration tests (reproducible) targeting known (potential) vulnerabilities.

Beyond Vision and PDMFC joined T3.3 after the last deliverable D3.7 had been published. They introduced the new analysis pattern “AP13: Embedding Identity and Access Management to the Event Log Process”. The scope of T3.3 as well as the interface with T3.2 was aligned between the members and task leaders of WP3 during the WP3-Face2Face meeting in Graz, Austria (October 2019). It was also agreed during this meeting that the main scope of T3.3 should be the improvement of DPs, while the description and improvement of CTEs should be part of T3.1.

1.2 Document Outline

Chapter 2 describes the AP concept along with the chosen template that we designed for collecting, listing and describing the developed AP by the responsible partners. It also provides an overview of the AP and their intended coverage of DPs within a matrix. Chapter 3 draws conclusions on the outcome of this task/deliverable and provides an outlook.

2 Analysis Patterns

In this deliverable, we provide a detailed description of the AP developed in T3.3, which have been introduced in D3.6.

2.1 Analysis Patterns Overview

In T3.3, we develop a set of APs for possible safety, security and privacy shortcoming in domain specific technology solutions. Thereby we are differentiating between two main approaches based on the scope/intention of the AP.

- The first approach has the aim to improve a single or a group of pre-selected DPs with respect to a specific problem, risk or threat, by applying well known analysis techniques and use the outcome to improve the analysed DP by tailored countermeasures.
- The second approach focuses on the development of new analysis techniques (or the extension/combination of existing ones) to support new or complementary aspects that are not yet considered (e.g. combining safety and security aspects) and therefore addresses mainly mutually related safety, security and privacy issues. These new analysis techniques are targeting to the identification of gaps and shortcomings, which are not yet covered by classical analysis approaches.

To distinguish between the two approaches, every AP is assigned to one of these two categories based on its scope. We refer to the first category as *Improvement-AP (IA)* and the second category as *Methodology-AP (MA)*.

Every AP covers at least one DP. Some APs, such as “Combined Safety and Security Analysis” (AP06) or “Security Zones” (AP10), are generic approaches and are not necessarily specific to a particular DP. While they are applied to one DP in this deliverable (as example application and for evaluation purposes), they are not limited to a specific DP or even application domain. This holds for all APs of the category “Methodology AP”. In contrast, APs of the category IA focus and contribute to a single DP and its improvement by enhancing the DP with concrete countermeasures for the addresses problem, as in the case with “Secure delivery and flashing of binary update” (AP02) that addresses DP19 (of the same name). The following matrix (Table 1) lists the APs (described below) and their relation to the addressed DPs (from T3.2) as well as the type of AP.

		Analysis Patterns (APs)										
	Name	AP01: Security4ConSerts	AP02: Secure delivery and flashing of binary update	AP04: Common Criteria Profile Protection	AP05: Privacy-preserving multi-party access control	AP06: Safety Contents Management	AP07: Combined Safety and Security Analysis	AP08: Pre-injection analysis	AP09: Common Risk Assessment Methodology	AP10: Security Zones approach	AP11: Tracing Security and Privacy Principles in Architectures	AP13: Embedding Identity and Access Management to the Event Log processes
	Type/Category of AP	**IA**	**IA**	**MA**	**IA**	**MA**	**MA**	**MA**	**IA**	**IA**	**IA**	**IA**
	Responsible	**IESE**	**ZF**	**ISSM Invia (Thales)**	**TU/eRADB**	**MM**	**VIF**	**RISE**	**GTO**	**TNO**	**TNO**	**PDMFC/BeyondVision**
Design Patterns (DPs) addressed by AP	DP01: Anomaly Detection											
	DP02: Automated Threat Detection and Vulnerability Management											x
	DP03: Bastion Host											x
	DP04: Cryptographic Erasure											
	DP05: Digital Identity Management and Smart Profiling								x			x
	DP06: Healthcare Data Exchange								x	x	x	
	DP07: Input Validation							x				
	DP08: Decentralized Key Distribution											
	DP09: Mutual Authentication								x			
	DP10: Authentication and Authorization module								x			
	DP11: Secure Embedded Network											
	DP12: End-2-End Security			x								
	DP13: Authentication and Authorization Protocol for SRWN											
	DP14: Long-term Support - SBoM											
	DP15: End-2-End Security for Constrained Devices											
	DP16: In-vehicle Network Intrusion Detection Systems				x							
	DP17: Secure key storage											
	DP18: Separation of networks											

	DP19: Secure delivery and flashing of binary update		x								
	DP20: Secure remote communication						x				
	DP21: Virtualized Embedded System										
	DP22: Security Test Studio										
	DP23: Sensor Data Analytics										
	DP24: Sensor Data Fusion Architecture					x					
	DP25: Conditional Safety Certificates	x									
	DP26: Privacy Preserving Multi-party Access Control				x						
	DP27: Anonymization and Data Protection										
	DP28: Anonymity Set										x
	DP29: Accountable Privacy Preserving Authentication										
	DP30: Data Usage Control										
	DP31: Security Testing Process										
	DP32: Derived Identity								x		
	DP33: Trusted Secondary OS										
	DP34: Privacy Preserving Path Detection										

Table 1: Overview of APs and related DPs

2.2.1 General structure of the AP descriptions

The description of each AP consists of two parts: First, an overview table providing a brief introduction into the content of the AP. This table contains a short description of the goal, the addressed problems, an abstract of the AP, a list of stakeholders and the definition of the type of the AP and the main scope - security, safety or privacy. The following table shows the template for the brief overview table, along with a short explanation of its entries. Based on this overview table, readers should be able to tell, whether the AP is relevant to their particular interest or not.

Analysis Aim:	The aim of this AP and its contribution to SECREDAS goals.
Abstract:	Description of the AP and its application with regards to the related DP(s).
Addressed Problem:	Summarization of the problem addressed by the AP.
Stakeholder:	Identified stakeholders to be considered as part of the analysis (e.g. context information based on the use cases, design principals or DPs, etc.).
Categories and Scope:	
Scope:	Type of the AP (allowed values: Improvement-AP/Methodology-AP)
Safety:	

Security:	What problem domain(s) are addressed by this AP, what is the main focus and what should be improved by this AP.
Privacy:	
Input:	
Description:	Short description of the required input for this AP and an example, if available.
Example(s):	
Addressed Elements:	
Related DP:	List of related DPs.
Contributors:	
Responsible:	Main contact and responsible partner for this AP.
List of Contributors:	List of other contributors

Table 2: Template for the AP overview.

The second part (of the AP description) contains a detailed self-contained description of the AP, which can be read independently from the description of the other APs. This allows selective, problem-oriented reading of the document, without too many cross references to other sections. The following table shows the structure of the AP description, depending on the type of the pattern. For the following description of the structure, we are assuming the AP number Y performing an Analysis for DP Z:

<table>
<tr><th colspan="2">Structure of the detailed AP description depending on the type/scope of AP</th></tr>
<tr><th>Improvement-AP (IA)</th><th>Methodology-AP (MA)</th></tr>
<tr><td colspan="2">Introduction
Description of the problem providing a motivation for the development of this AP either by illustrating the required improvements/shortcomings of concrete DP within the usage context, or showing the requirement for a new analysis methodology (e.g. by identifying gaps of currently used methods or identifying new requirements).</td></tr>
<tr><td></td><td>Description of the Y analysis method
Definition of the scope and goals of the new analysis method as well as requirements. Description of the method and background as well as developed tools.</td></tr>
<tr><td>Analysis of DP Z
Identification of shortcoming and risks considering the identified usage context of the DP Z. Definition of the scope of improvement as well as the addressed problems/risks.</td><td>Analysis of DP Z
Results of applying the described analysis to DP Z, as well as illustrating (potential) tool support for applying the AP.</td></tr>
<tr><td>Developed solution
Description of the developed solution and integrated counter measures. The range of solutions presented can range from conceptual solutions to concrete implementations.</td><td>Developed solution [optional, only if applicable]
Developed solution (for the analysed DP Z) based on the previous chapter.</td></tr>
<tr><td>Evaluation of Improvement
Evaluation or argument showing that the developed improvement solves the identified issues.</td><td>Evaluation of Analysis method Y
Illustrating the benefits of using this method (i.e. the added value).</td></tr>
<tr><td colspan="2">Related Work
References to existing work or related work (e.g. related standards, literature considered/used or differentiation to existing analysis methods.</td></tr>
<tr><td colspan="2">Conclusion
Short conclusion of developed improvement or benefits of using the new developed analysis methodology.</td></tr>
<tr><td colspan="2">References</td></tr>
</table>

2.2.1 AP01: Security4Conserts

2.2.1.1 Brief Overview

Analysis Aim:	The aim of this AP is to identify security threats as well as countermeasures, which will give confidence in using Conditional Safety Certificates (ConSerts) within insecure and open environments. Our goal is to secure ConSerts as an infrastructure, that is, secure the issuing (e.g. using certificate management systems) and the exchange (e.g. using secure V2X communication) of ConSerts and demands/guarantees between collaborating systems.
Abstract:	ConSerts enable runtime negotiation of safety-related properties among collaborative systems; it assures safety of open-adaptive systems under predefined conditions that need to be met at runtime. These conditions are manifested in the relationship between safety demands, of a particular system, and the corresponding safety guarantees, of the system to be integrated or collaborated with. Since the assured guarantees can change during runtime, ConSerts support systems adaptation (to such changes) through runtime evidences, by mapping the required demands (to the assured guarantees) from a predefined set of various configurations. Example: In a platooning scenario, where a group of vehicles are driving close to each other (in a sequence), a collaboration among the vehicles would enable them to maintain a safe distance. Using sensor data and other information gathered at runtime (e.g. reliability of the sensor values, such as ensured ASIL levels), vehicles exchange safety guarantees and demands beside their safety certificates (generated at development time) to form a formal safety case and enable a safe collaboration. In case of the platooning, a reduced reliability of a distance measuring sensor or a braking system might result in switching the configuration and (application dependent) counter measures, such as increasing the distance between the vehicles in the platoon or terminating the collaboration. This AP focuses on analyzing security threats that may violate such a collaboration, as well as developing and integrating corresponding measures to counteract those threats.
Addressed Problem:	In the current state, ConSerts do not deal with security incidents. For instance, a manipulation attempt of ConSerts is not yet investigated. Such security issues will be addressed within this AP.
Stakeholder:	OEMs, Vehicle Owners (i.e. drivers), Other (cooperating) vehicles, Roadside units, Railway infra/assets (vehicle, driver), sstandardization body, Developers of updates, In-manufacturer specialists responsible for Safety and/or integration (e.g., architect), attackers.
Categories and Scope:	
Scope:	Improvement-AP
Safety:	ConSerts currently only focus is Safety.
Security:	Security measures are the main scope of this AP and will be developed and integrated into the ConSerts DP.
Privacy:	Privacy is out of scope and will only be addressed, where related to security.

Input:	
Description:	Different scenarios where ConSerts may be applied as well as the main building blocks of ConSerts (as an infrastructure). Selected examples, where ConSerts can be applied to (and security is important): Platooning and exchange of (safety related) sensor data, in general.
Example(s):	
Addressed Elements:	
Related DP:	ConSerts (DP25) is a DP that operates at the level of safety requirements (i.e. safety demands and guarantees), which are specified at development time, based on a sound and comprehensive safety argumentation (e.g. an assurance case). ConSerts conditionally certifies that an associated system will provide specific safety guarantees. Conditions are manifested in the fulfilment of specific demands from other collaborative systems or the environment in which the system operates. With ConSerts, systems become aware of the safety-relevant conditions of their environment during runtime and accordingly adapt to them. This allows each system to offer additional guarantees depending on the degree of fulfilment of the demands provided by the collaborative systems.
Contributors:	
Responsible:	Fraunhofer IESE: • Tobias Braun (tobias.braun@iese.fraunhofer.de) • Hanna AlZughbi (hanna.alzughbi@iese.fraunhofer.de)
List of Contributors:	N/A

2.2.1.2 Detailed description

Introduction

In task 3.2, we introduced Conditional Safety Certificates (ConSerts) as a safety Design Pattern (DP) for the safety assurance of open adaptive systems. Our DP describes the use of safety certificates during runtime, to achieve safety among open-adaptive (collaborative) systems.

ConSert conditionally certifies that the underlying system (or service) will provide specific safety guarantees. Conditions are manifested in the fulfillment of demands from other collaborative systems or environment (in which the system operates). The safety requirements themselves (i.e. safety demands and guarantees) are specified at design time, based on a sound and comprehensive safety argumentation (e.g. an assurance case). Matching safety demands and guarantees during runtime allows the consideration of highly dependent dynamic aspects, such as environmental aspects as well as dynamic collaboration between systems. This is necessary, since only static system properties are either not sufficient to decide whether a collaboration among open adaptive systems is safe, or worst-case assumptions have to be considered during design time to ensure safe collaboration (not knowing the concrete conditions during runtime). This often causes flaws regarding the overall performance of the

provided service. By considering runtime aspects, it is often not necessary to limit the performance, in general, unless required to ensure safety. The results of these checks are used to decide whether a collaboration (among systems) is safe and can be granted or otherwise rejected.

Additionally, ConSerts support systems adaptability (to such changes) through runtime evidences, by mapping the required demands (to the assured guarantees) from a predefined set of configurations (specified at design time). During runtime, sensor data is measured, processed and communicated to other systems (e.g. vehicles). Consequently, each system becomes aware of the safety-relevant conditions of their environment and accordingly adapt to them, which allows each system to offer additional guarantees depending on the degree of fulfillment of the demands provided by the collaborative systems. Depending on the assured guarantees and required demands, the most optimal configuration is activated to enable high service quality, while ensuring safety.

The shortcoming of ConSert DP in its current state is that it is only designed as safety concept and has not been designed to deal with security incidents or threats that may be imposed to the underlying system or service. For example, if an attacker would manipulate guarantees and demands that are being exchanged between two systems (e.g. vehicles), this would probably go unnoticed and have a severe consequence on the overall collaborative systems safety. Therefore, we need to analyze ConSert DP against such threats in an effort to understand how secure it is against such incidents and implement countermeasures to improve our Consert DP in terms of security aspects.

In this AP, we use an existing approach to analyze and assess security risks of applying ConSerts in real world applications or environments. Depending on the identified threats, we suggest countermeasures and integrate them in our DP. These are implemented and evaluated in our demonstrator of platooning scenario.

This chapter is structured as followed: In the first Section, we perform a threat analysis and risk assessment regarding the application of ConSerts. After that, we describe the concept of developed counter measures for the identified threats while proceed with the description of the implementation and evaluation of the counter measures. Related work is linked, while the last Section draws conclusion and describes future work.

Analysis of DP25: ConSerts

In the introduction, we describe ConSert as a DP, but from an operational perspective, we need to differentiate between two concepts: ConSertTree and runtime generated guarantees/demands.

ConSertTree describes the services provided among systems' sub-components, and is specified at design time based on a sound and comprehensive argumentation. It consists of a set of configurations allowing adaptation during runtime. Every configuration describes a mapping between required safety demands from other services (e.g. provision or fusion of sensor values), and the resulting guarantees ensured for the provided services. ConSertTrees are exchanged during the initial communication between collaborative systems, referred to as the setup phase.

Runtime aspects are represented by guarantees (and demands) that are evaluated during runtime, based on sensor data and other properties, and are continuously exchanged among systems. This information is used to verify, whether an ongoing collaboration still satisfies the safety requirements formulated in the design time safety case. Depending on the provided guarantees, the underlying system would adapt by switching its configuration accordingly.

Therefore, it is important to consider both ConSertTree and demands/guarantees in the security analysis, to differentiate between the threats that both might be exposed to. While we try to perform the analysis independent from a concrete use case or application scenario, it helps to keep the platooning scenario in mind, to better understand the possible impacts.

In the following, we present our analysis table, based on the Security-Aware Hazard Analysis and Risk Assessment (SAHARA) approach [1] that helps in determining the impact of security threats on the overall system safety.

Threat Analysis				SAHARA						
				Result	Likelihood				Impact	
ID	Threat Type (STRIDE)	Threat Scenario/Description	Attacker Motivation	Security Level	Level of Knowledge	Description	Resources required	Description	Threat Criticality	Impact Description
1	Spoofing	Send fake ConSertTree from a masqueraded address	To cause damage	2	1	Technical knowledge required	2	Simple analysis and injection tools required	3	Having a spoofed system might cause damage to other systems that depend on the received ConSertTree for deriving safety-critical actions (e.g. by initiating unsafe collaboration).
2	Spoofing	Send fake guarantees/demands from a masqueraded address	To cause damage	2	1	Technical knowledge required	2	Simple analysis and injection tools required	3	Having a spoofed system might cause damage to other systems that depend on the received guarantees/demands for performing safety-critical actions.
3	Tampering	Manipulate the content of ConSertTree	To cause damage	3	1	Technical knowledge required	1	Standard tools required	3	Having manipulated ConSertTree may have direct impact on safety (e.g. by initiating unsafe collaboration).
4	Tampering	Manipulate the run-time generated guarantees/demands	To cause damage	3	1	Technical knowledge required	1	Standard tools required	3	Having manipulated guarantees/demands may have direct impact on safety (e.g. by potentially performing unsafe actions).
5	Repudiation	Deny sending (fake) ConSertTree	To avoid being blamed in case of a damage	3	0	No prior knowledge required	1	Standard tools required	2	Having a system that is able to deny the issuance of a certain ConSertTree would result in a low confidence in the overall systems collaboration.
6	Repudiation	Deny sending a specific(fake) guarantees/demands	To avoid being blamed in case of a damage	3	0	No prior knowledge required	1	Standard tools required	2	Having a system that is able to deny the issuance of certain guarantee/demand would result in a low confidence in the overall systems collaboration.
7	Information disclosure	Capture ConSertTree from storage	To launch other attacks	0	1	Technical knowledge required	2	Simple analysis tools required	1	Having in depth information about the system might help to enable attacks. However, since ConSertTrees are used to negotiate collaboration

										(exchanged and agreed during development between different companies), they do not contain sensitive data in terms of privacy.
8	Information disclosure	Capture ConSertTree during transmission	To launch other attacks	0	1	Technical knowledge required	2	Simple analysis tools required	1	Having in depth information about the system might help to enable attacks. However, since ConSertTrees are used to negotiate collaboration (exchanged and agreed during development between different companies), they do not contain sensitive data in terms of privacy.
9	Information disclosure	Capture guarantees/demands during transmission	To launch other attacks	1	1	Technical knowledge required	2	Simple analysis tools required	2	Having leaked information about the system, during runtime, might make it easier to launch tailored attacks.
10	Denial of Service	Flood a communication channel with randomly generated (fake) ConSertTree	To prevent legitimate systems from acquiring ConSertTree, and therefore preventing collaboration	1	1	Technical knowledge required	2	Simple injection tools required	2	Having a communication channel down would prevent or degrade collaboration between safety-critical systems.
11	Denial of Service	Flood a communication channel with randomly generated (fake) guarantees/demands	To prevent legitimate systems from acquiring guarantees or demands, and therefore degrade service performance or prevent collaboration	1	1	Technical knowledge required	2	Simple injection tools required / depends on communication technology used	2	Having a communication channel down would degrade collaboration between safety-critical systems.
12	Elevation of Privilege	Gain unauthorized permission to issue ConSertTree	To cause damage or gain illegitimate advantage	2	1	Technical knowledge required	2	Simple analysis tools required	3	Having unauthorized entities, generating unreliable ConSertTree, might cause unsafe collaboration between safety-critical systems.
13	Elevation of Privilege	Gain unauthorized permission to issue Guarantees	To cause damage or gain illegitimate advantage	2	1	Technical knowledge required	2	Simple analysis tools required	3	Having unauthorized components, generating unreliable guarantees, might cause unsafe collaboration between safety-critical systems.

Table 3 Security Analysis of DP25 ConSerts [10]

Denial of Service attacks can be used to limit the performance or prevent collaborative services. Assuming a wireless communication between collaborative partners, a shared medium is used. Therefore, denial of service attacks are always possible by jamming the signal/allocated frequency. How easy such an attack is, depends on the communication technology in place. However, since (wireless) communication has always to be considered unreliable, a failsafe operation/safe degradation for this type of failure has always to be defined in the functional safety concept to prevent harm (from this point of view a denial of service attack is just an additional root cause for unreliability). From this point on, we will focus on security threads with level 2 and higher: These are related to tampering, spoofing, repudiation and Elevation of Privilege.

Developed Solution

We emphasise that our approach's main focus does not deal with securing data exchange among internal components (e.g. ECUs), within a system (e.g. on the field bus level of a vehicle), but rather the exchange of safety related data (i.e. guarantees and demands, needed for system collaboration) across systems.

For simplicity, we consider the platooning scenario for descripting our approach. In this scenario, required data might be collected or processed within the vehicle, by various sensors and ECUs, and communicated over field buses (e.g. CAN or FlexRay) to the Gateway ECU, to be transmitted to other vehicles. For our analysis, we assume that safety related components (i.e. ECUs, sensors, actuators) are connected via protected and separated field buses. We also assume an attacker with limited access to the physical resources (i.e. the attacker is not able to inject messages on separated field buses related to safety critical functions, see DP18 "Separation of networks"). If the attacker gains control over an ECU, an intelligent sensor or an actuator, this means that they may be able to generate falsified (sensor) data and send it over the field bus, to be further processed. In contrast, we assume an attacker with unlimited access to the communication channels (between vehicles). In other words, the attacker may not only listen to the information being communicated, but also can inject or manipulate communicated messages on a shared communication medium.

While in general our presented solution also works for intra-vehicle networks (exchanges safety related demands and guarantees between ECUs of the same vehicle), the limited bandwidth of classical field buses like CAN and FlexRay as well as communication related real-time constraint could prevent a practical application. By replacing or complement classical field buses with techniques like real-time Ethernet, this limitation could be resolved. Regarding security of in-vehicle networks measures described in DP11 or DP15DP15 as well as the developed measures from WP 6.2 (spoofing protection for CAN/CAN FD) must be additionally taken, but are out of scope for this AP.

Since the threat types spoofing, tampering, repudiation and elevation of privilege have the highest security levels according to our SAHARA analysis, shown in Table 3, we focus our solution on addressing these threat types for the application of ConSerts. In this AP, the realization and usage of the offered collaborative services is out of scope. Nevertheless, the collaboration and information exchange of such services must also be secured. Therefore, additional measured such as DP12 and DP15 [10] have to be applied.

Digital Signatures

The tampering threat type needs to be addressed by maintaining the integrity property of the exchanged data, under which, no unauthorized party may alter the data in transit, without being detected. Besides that, the spoofing threat type needs to be addressed by maintaining the authenticity property of the received message, under which, no sender may successfully fake an authorship of a message, without being detected. These two properties form the data origin authentication property. On the other hand, the repudiation threat type needs to be addressed by maintaining the non-repudiation property, under which, no sender may successfully dispute an authorship of a message, which they have sent over. Incidentally, both data origin authentication and non-repudiation properties can be achieved under the same technique, which is, Digital Signatures.

Digital signature is a scheme, by which, the received message can be associated with the sender with high confidence, and the receiver is able to authenticate the origin of the message, that is, verify the identity of the sender and the integrity of the message.

Associating a signed message with the sender, implies that the sender has a parameter, that is only private to him, i.e. Private Key (hence the name), by which they can sign the message under their name (or identity). In other words, the sender cannot deny sending a message that is signed by their private key, as long as the key is not forged (by a third-party) or revoked (by the entity that issued it). Therefore, secure storage of private keys is of an eminent importance. We assume that all private keys (used for signing) are stored within a secure module (e.g. TPM). We highlight here, that if an attacker is able to hack this unit/module, then they may be able to generate, sign and send fake messages. We recommend CTE-05 “Secure Element” [9] as well as DP17 “Secure Key Storage” [10] to prevent against such attacks. Including these measures also helps to mitigate the threat Elevation of Privilege (see Table 3, threat 13).

Given that the private key is owned only by the sender (or signer), their signed message can be authenticated, by checking against the signature, to verify the identity of the sender and the integrity

of the message. By adding timestamp and sequence number[1] to the message's signature, the receiver can detect replay attacks that may be part of a spoofing attack.

We implement our digital signatures on top of the generic Public Key Infrastructure, presented by Bißmeyer et al [2], according to which, we assume an existence of a Root Certificate Authority (RCA) that can issue (and revoke) certificates to Certificate Authorities (CAs). Revocation of certificates can be realized via revocation lists (which need to be updated on a regular basis) [RFC5280] or via dedicated protocols like OCSP (Online Certificate Status Protocol, RFC 6960), requiring an online connection. A combination of both approaches may be used, depending on the domain and preconditions (e.g. online connection available, constraints regarding latency, etc.).

For clearness, we illustrate the general concepts using a platooning use case as collaboration scenario. In this scenario, CAs can be OEMs or suppliers that issue (and revoke) certificates directly to (their) vehicles. Where the RCA is the entity under which all (European) OEMs and suppliers operate.

Figure 1, shows an example, how this public key infrastructure could be instantiated in terms of realizing collaborative functions among vehicles. The concept can easily be extended to include RSUs (and their suppliers[2]) to enable a safe collaboration with vehicles within a V2X communication scenario.

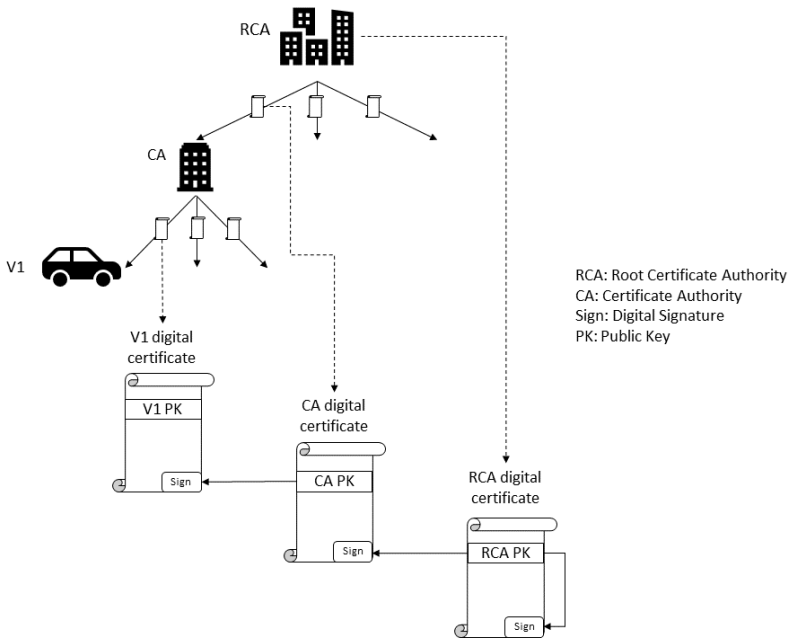


Figure 1 Public Key Infrastructure

[1] The timestamp must be based on a common synchronized clock between the collaboration partners, such as the time derived from GPS. The sequence number is increased with every transmitted message, while paying special attention to counter overflows.

[2] Producers of RSUs could be integrated into the CA infrastructure on the level of OEMs.

Development Phase

At development time, OEMs (i.e. the responsible CA in terms of safety assurance) generate (and sign) ConSertTrees for every vehicle. The signed ConSertTree is directly embedded into the vehicle, before shipping, as shown in Figure 2. By checking the signature of ConSertTrees before using them (see communication setup phase) and assuming, that OEMs implement appropriate measures to separate and protect their private keys, this combination provides a good protection against the threat Elevation of Privilege (see Table 3, threat 12).

The signed ConSertTree may contain either the underlying certificate or its hash. While the first variant allows direct communication link, it increases the size of ConSertTree, and therefore the required communication bandwidth. The second variant is better in terms of bandwidth usage but requires an additional data connection for retrieving the certificate from a public server (based on the embedded hash). By implementing one of these variants, new OEMs (that join the market) may be able to participate in secure collaborations.

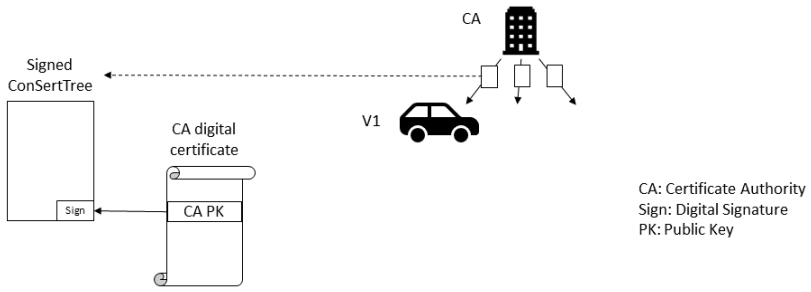


Figure 2 Development Phase

Communication Setup Phase

At runtime, and during the initial communication between vehicles, every vehicle (V1) double signs its (previously signed) ConSertTree, using its private key, and sends it over, along with its underlying certificate (or its hash), to other vehicles, as shown in Figure 3. Since new OEMs may later join the market, their certificates may be fetched (using the given hash), from a central server.

The receiver vehicle (V2), verifies (or authenticates) the received certificate of V1 (against the root-of-trust, i.e. RCA). RCA certificate, along with certificates of all OEMs, are pre-loaded onto vehicles, to facilitate the validation of the exchanged certificates (among vehicles) against the root-of-trust (i.e. RCA).

Upon successful verification (of V1's certificate), V2 uses the public key (attached to the received certificate) of V1, to verify the signature of ConSertTree (i.e. the integrity of the received message, from V1). Likewise, V2 uses the OEM's public key to further verify that the received ConSertTree has

not been manipulated (by V1) after initially signed by the OEM (upon issuing). Prior to using V1 and CA certificates for verification, V2 checks whether these certificates are still valid or have been revoked. To prevent replay attacks, the messages used to exchange the ConSertTrees contain a timestamp and a message sequence number (using a random start value for the first message of each node) to prevent replay attacks. Therefore both, timestamp and sequence number, are part of the payload covered by the signature.

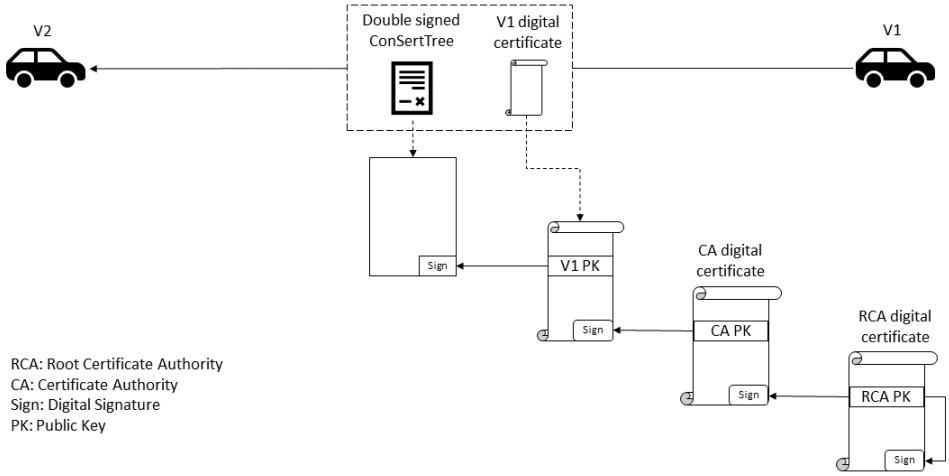


Figure 3 Communication Setup Phase

Collaboration Phase

Vehicles continuously generate guarantees (and demands), sign them, with their private key, and send them to other vehicles. As shown in Figure 4, the receiver vehicle (V2) validates the signature of the received guarantees, using the public key of the sender vehicle (V1), which it received, previously, during the communication setup phase. Upon successful verification, V2 shall proceed with the processing of runtime evidences and the overall system collaboration.

To prevent replay attacks, also the message exchanging guarantees and demands include timestamps and message sequence numbers.

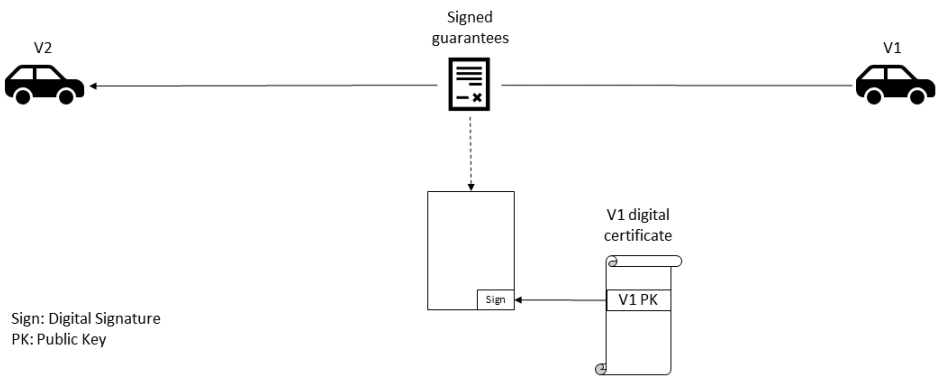


Figure 4 Collaboration Phase

Evaluation of Improvement

In the previous section, we illustrated our digital signatures approach to address the main issues identified in our analysis, namely, spoofing, tampering and repudiation, since they pose high security threats.

Using digital signatures, we showed that the receiver is able to authenticate the origin of the message, that is, verify the identity of the sender and the integrity of the message. Moreover, the received message can be associated with the sender with high confidence. We integrated our security improvements into the ConSerts design pattern (DP25) and will update DP description for the next deliverable D3.6 of WP3.2. To demonstrate the improvements, we are using the ConSerts demonstrator of the platooning scenario. The demonstrator is based on CARLA [7], an open-source simulator for autonomous driving research using the Unreal 3D Engine for visualization. We extended the demonstrator by adding a component for simulating the communication between the vehicles (allowing the consideration of attacks, such as frame injection or denial of service attacks) and integrated our solution in the used ConSerts (proof of concept) runtime environment based on the ROS framework [8].It is worth mentioning that using the generic PKI approach (introduced by Bißmeyer et al [2]) would introduce higher cost of data processing (on each vehicle) and communication overhead (among vehicles). However, it is more crucial to us to properly address the underlying threats, defined in our analysis, on the expense of computation and communication costs.

Related Work

In our analysis we use SAHARA [1], which is a combined analysis approach of the automotive Hazard Analysis and Risk Assessment (HARA) method, and Microsoft's threat modelling approach, STRIDE that comprises of Spoofing, Tampering, Repudiation, Information disclosure, Denial of service and Elevation of privilege. This combination, of HARA and STRIDE, enables thorough insight into the impact of security threats on the overall system's safety.

We based our solution on the generic PKI approach, introduced by Bißmeyer et al [2], which is designed to secure the V2X communication. The authors introduce the whole PKI infrastructure, including the issuance and revocation of digital certificates. Using this approach, one can also deploy encryption, to address confidentiality issues. But here one should keep in mind that there is a trade-off between confidentiality and cost. For our specific use case of ConSerts, we concluded that encryption is not crucial, since our exchanged data does not contain privacy-related information.

We refer to in-vehicle security techniques from [3]–[6]. In [3] the authors focus on masquerade and replay attacks. They use pair-wise secret keys among ECUs (sender and receiver would share the same key) and uses a counter for every sent and received message to prevent against replay attacks. They also introduce a method for implementing the counter in a way to prevent counter overflow.

Conclusion

The aim of this AP is to analyze ConSert (DP25, [10]) against security threats that may be imposed to the underlying system or service. Based on the identified threats, we implement countermeasures and integrate them in the DP. This will give high confidence in using ConSerts within insecure and open environments. In our analysis, we use SAHARA [1] to analyze and assess security risks of ConSerts DP. We identify the threat types spoofing, tampering, repudiation and elevation of privileges as being the ones with the highest security level. The threat types spoofing and tampering need to be addressed by maintaining the data origin authentication property. On the other hand, repudiation threat type requires the non-repudiation property.

In our solution, we maintain both data origin authentication and non-repudiation properties by implementing digital signatures on top of the generic PKI, presented by Bißmeyer et al [2]. For clearness, we illustrate the application of the general concepts using the platooning scenario, where CAs can be OEMs or suppliers that issue (and revoke) certificates directly to (their) vehicles, and the RCA is the entity under which all OEMs and suppliers in Europe operate.

Using digital signatures, we show that the receiver is able to authenticate the origin of the message, that is, verify the identity of the sender and the integrity of the message. Moreover, the received message can be associated with the sender with high confidence. We illustrate our approach by implementing the identified measured on top of the platooning demonstrator of DP26.

Future Work

In future work, we plan to extend our approach to realize secure logging to allow for post-mortem investigation (e.g. in case of an accident). Therefore, the receiver has to upload the received (and signed) guarantees (as a proof of data authenticity), along with the ConSertTree, to a blockchain (or trusted cloud). Following the processing of runtime evidences, the selected configuration (selected mapping between guarantees and demands) are stored on a consortium blockchain. The consortium blockchain contains records of all interactions (among systems), and is kept private from the outside world, while being transparent among the internal OEMs/suppliers that manages it. Updates of ConSertTree can be tracked on the blockchain. Thereby, the updated ConSertTree must also include a unique version number to keep track of the latest version and prevent installing outdated/expired ConSertTrees.

References

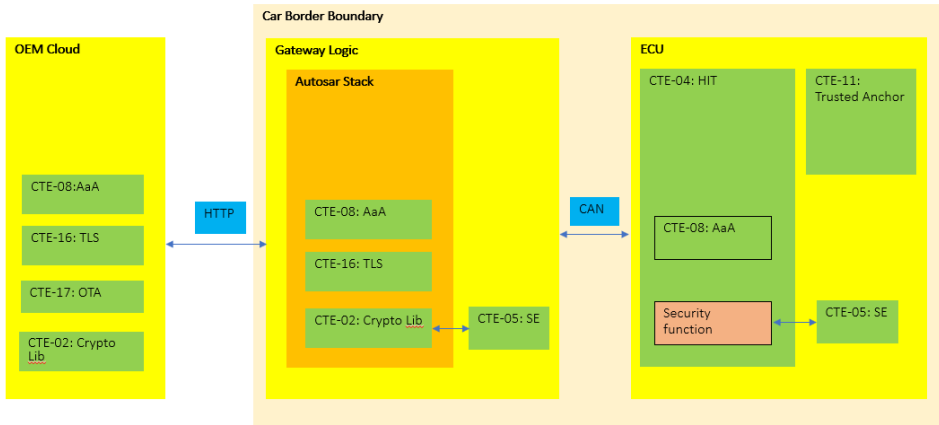
[1] Macher, G., Sporer, H., Berlach, R., Armengaud, E., & Kreiner, C. (2015). SAHARA: A security-aware hazard and risk analysis method. 2015 Design, Automation & Test in Europe Conference & Exhibition (DATE), 621-624.

[2] Bißmeyer, N., Stubing, H., Schoch, E., Gotz, S., Stotz, J.P., & Lonc, B. (2011). A Generic Public Key Infrastructure for Securing Car-to-X Communication.
[3] C. Lin and A. Sangiovanni-Vincentelli, "Cyber-security for the controller area network (can) communication protocol," in 2012 International Conference on Cyber Security, Dec 2012, pp. 1–7.
[4] D. Zelle, C. Krauß, H. Strauß, and K. Schmidt, "On using tls to secure in-vehicle networks," in Proceedings of the 12th International Conference on Availability, Reliability and Security, ser. ARES '17. New York, NY, USA: ACM, 2017, pp. 67:1–67:10. [Online]. Available: http://doi.acm.org/10.1145/3098954.3105824
[5] M. Wolf, A. Weimerskirch, and C. Paar, "Security in automotive bus systems," in IN: PROCEEDINGS OF THE WORKSHOP ON EMBEDDED SECURITY IN CARS (ESCAR)'04, 2004.
[6] U. E. Larson and D. K. Nilsson, "Securing vehicles against cyber attacks," in Proceedings of the 4th Annual Workshop on Cyber Security and Information Intelligence Research: Developing Strategies to Meet the Cyber Security and Information Intelligence Challenges Ahead, ser. CSIIRW '08. New York, NY, USA: ACM, 2008, pp. 30:1–30:3. [Online]. Available: http://doi.acm.org/10.1145/1413140.1413174
[7] Alexey Dosovitskiy and German Ros and Felipe Codevilla and Antonio Lopez and Vladlen Koltun, "CARLA: An Open Urban Driving Simulator", Proceedings of the 1st Annual Conference on Robot Learning, 2016, pp. 1-16
[8] Stanford Artificial Intelligence Laboratory et al., "ROS: Robotic Operating System"
[9] SECREDAS, D3.2: Common Technology Element Change Report v1, 2020
[10] SECREDAS, D3.6: Design Pattern Descriptions v1, 2020

2.2.2 AP02: Secure delivery and flashing of binary update (ZF)

2.2.2.1 Brief Overview

Analysis Aim:	The aim of this AP is to improve DP19, by considering its cybersecurity aspects, and applying SAHARA to improve the analysed DP by countermeasures. Thereby, we investigate the applicability of specific used CTEs for the use in secure vehicle networks, identify required communication scenarios and analyze performance impacts.as well as communication scenarios which must be supported by CTEs.
Abstract:	From the UC 3 ("keep car secure"), we select as scope the secure delivery of SW update from update infrastructure through gateway to ECU and flashing of the delivered binary update. For this purpose, the OEM could communicate with vehicle through http with usage of TLS to deliver the binary update to the vehicle gateway. The gateway realized the flashing of ECU software (SW) via CAN- BUS. The Flashing of delivered binary update is realised with help of ARM Trust Zone.
Addressed Problem:	Secure OEM-ECU communication and secure flashing of software update on ECU(Automotive Domain).
Stakeholder:	OEM, Manufacturer (Gateway, ECU)
Categories and Scope:	
Scope:	Improvement-AP 1. Secure delivery of binary update 2. Secure onboard communication 3. Flashing of delivered binary update
Safety:	Security is the focus of the proposed architecture.
Security:	
Privacy:	
Input:	
Description:	We used the UC3 "keep car secure"-TARA and expanded it with usage of TLS (by HTTP(S) communication), secure CAN communication and ARM Trust Zone focusing on the threats related to the application DP19.
Example(s):	

	
Addressed Elements:	
Related DP:	DP19: Secure delivery and flashing of binary update
Contributors:	
Responsible:	Hayk Hamazaryan (ZF, Hayk.Hamazaryan@zf.com) Joerg Kemmerich (ZF, Joerg.Kemmerich@zf.com)
List of Contributors:	N/A

2.2.2.2 Detailed description

Introduction

This AP analysis in T3.2 introduces DP19 (Secure delivery and flashing of binary update) to collect and prove all threat mitigations applied/combined by DP. DP19 addresses the secure delivery (of SW from Update infrastructure to the vehicle) and flashing of binary update to an ECU. Afterwards, the gateway reports the update status to update the infrastructure. The design pattern contains mitigation mechanisms against:

- Spoofing of messages or data received by the vehicle.
- Information disclosure.
- Privileged access to vehicle systems.
- Misuse or compromise of update procedures.
- Extraction of vehicle data/code.
- Erasure of data/code.

In addition, the design pattern provides:

- Authentication and integrity of software.
- Data protection best practices are followed.
- The encryption of data and communication in V2X.
- Authenticity, integrity and non-repudiation of message exchange between the system and another entity on the network/bus.

- Secure Key Storage.
- Secure Onboard Communication (SecOC).
- Update binary verification.
- Device validity.
- Strong defenses against illegally manipulation of application resources.

The design pattern describes how to use a combination of CTEs for a secure delivery and flashing of binary updates in Automotive. The OEM could communicate with vehicle through http with usage of TLS to deliver the binary update to the gateway. The gateway establishes the flashing of ECU SW via CAN- BUS. The keys will be injected into the ECU or agreed with the rest of the vehicle by using a scheme which is outside the scope of this document. They must be stored in a protected area of flash which is inaccessible to normal code (in our case secure element - SE) – to ensure that the keys retain the properties of authenticity, integrity and confidentiality.

This design pattern combines three main functions to realize a secure SW update of ECUs within the vehicle:

1. Secure delivery of binary update (with usage of TLS).
2. Secure onboard communication.
3. Flashing of delivered binary update (with usage of SE and Arm TrustZone).

The following figure represent the main participants of DP19, CTEs and communication channels.

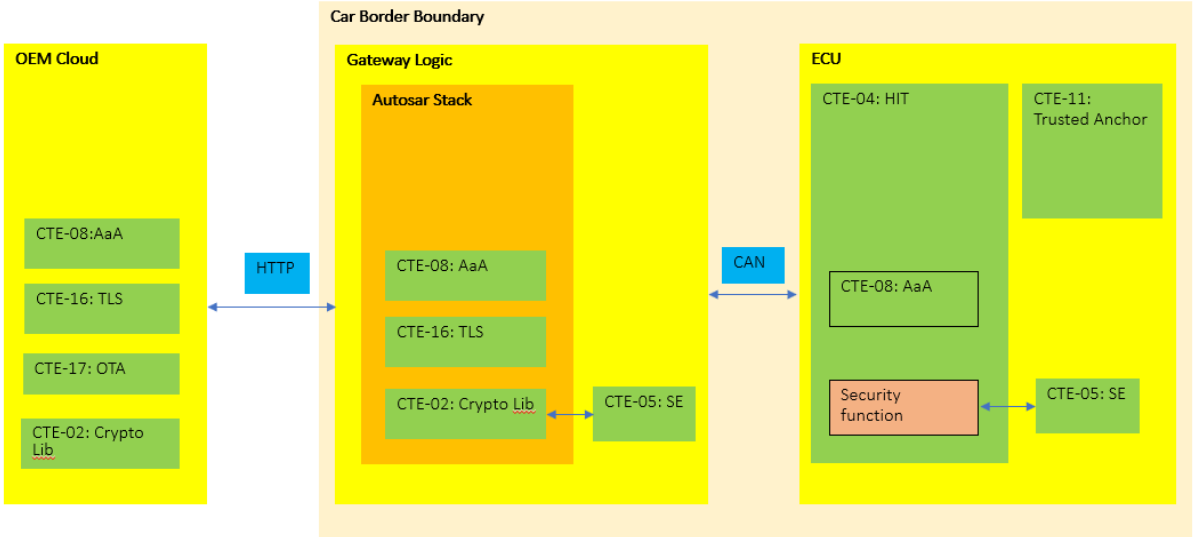


Figure 5: Secure delivery and flashing of binary update design

Analysis of DP 19

In WP1 we presented the SAHARA for UCP3. [8] The following figure shows on the UC3 data flow diagram the application scope of DP19. DP19 realizes a special concept for implementing secure delivery of update package from OEM infrastructure through gateway to ECU and flashing the software on ECU.

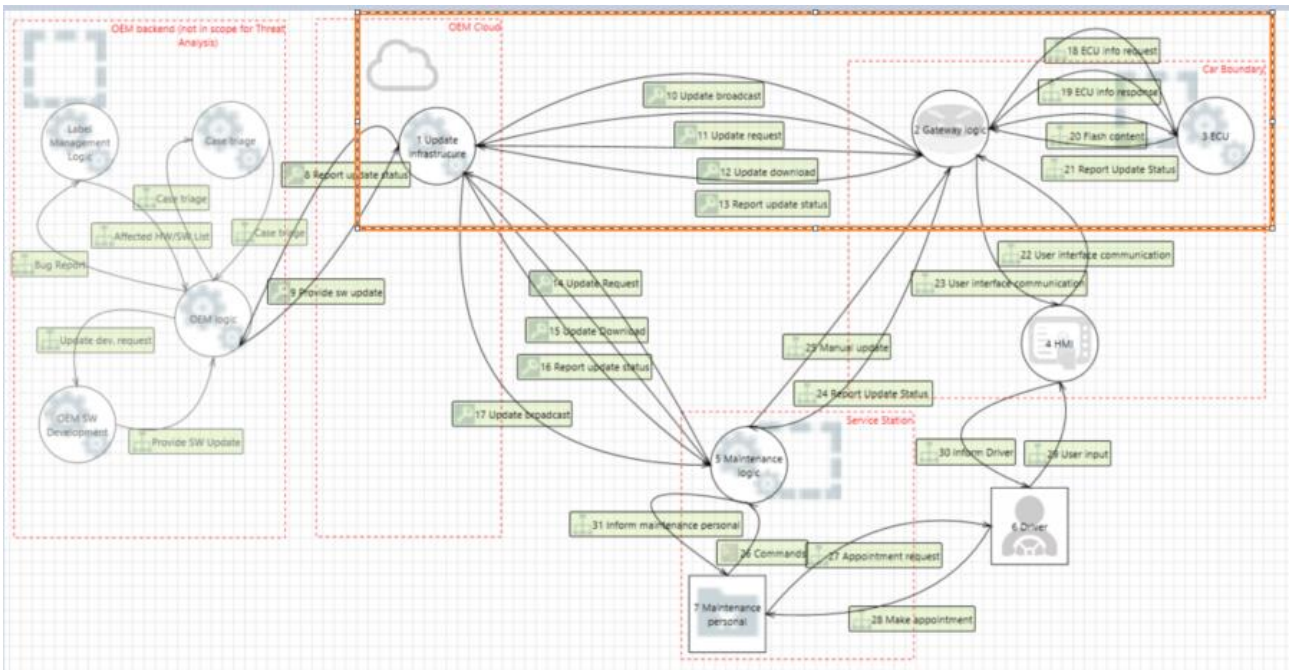


Figure 6: UC3 data flow diagram the application scope of DP19

DP19 is a special concept for implementation of selected parts of UC 3 shown in the figure above. Accordingly, all risks and their mitigations related to the selected part of UC3 can be utilised in DP19 analysis. Our steps for SAHARA realisation on DP19 were:

1. Take Assets from UC3 SAHARA[8] for Nodes: 1, 2, 3, 10-13, 18-21, + Extra, ODB, V2x, Wider vehicle network
2. Add new/redefine existing Assets for TLS, ARM Trust Zone, Secure Boot, SE
3. Consider the mitigations implemented in CTEs.

The following table shows the results from step 1 and 2. To identify assets, we performed a SAHARA risk assessment. The table shows an excerpt of this analysis, only listing threats with Security level above 2.

Ref.No.	Old Ref.No.	Use Case	Impacted stakeholders	Asset	Threat type (STRIDE)	Threat Scenario	Security Level (SL)	Level of Knowledge	Resoures required	Threat Criticality	Assumptions / comments (optional)	Security Goals / Mitigation	Target
1	4	UC3	Communicatio n channels used to attack a vehicle	ECUs(2,3)	Spoofing	Spoofing of messages (e.g. 802.11p V2X during platooning, etc.) by impersonation	3	1	1	3	Safety relevant	Keys,IP/ SE	ARM TZ
2	4	UC3	Communicatio n channels used to attack a vehicle	ECUs(2,3)	Spoofing	Spoofing of messages (e.g. 802.11p V2X during platooning, etc.) by impersonation	3	1	1	3	Safety relevant	Keys,IP/ SE	Sec.Boot
3	20	UC3	Human factor and social engineering	ECUs(2,3)	Social engineering (several)	Erroneous use or administration of devices and systems (inc. OTA updates)	3	0	1	2		root access of fleet of vehicles/4 eyes principle	ARM TZ
4	25	UC3	Compromise of external connectivity	ECUs(2,3)	Elevation of privilege	Corrupted applications, or those with poor software security, used as a method to attack vehicle systems	3	0	1	2		root access/ root of trust	ARM TZ

5	28	UC3	Target of an attack on a vehicle	ECUs(2,3)	Information disclosure	Product piracy / stolen software	3	1	1	3		IP/	ARM TZ
6	44	UC3	Communication channels used to attack a vehicle	ECU Network / CANBUS (18-21)	Repudiation	Accepting information from an unreliable or untrusted source	3	1	1	3		IP/Packege autentication	CAN
7	55	UC3	Compromise of external connectivity	ECU Network / CANBUS (18-21)	Elevation of privilege	External interfaces such as USB or other ports may be used as a point of attack, for example through code injection …	3	1	1	3			CAN
8	56	UC3	Compromise of external connectivity	ECU Network / CANBUS (18-21)	Elevation of privilege	Utilise diagnostic access (e.g. dongles in OBD port) to facilitate an attack, e.g. manipulate vehicle parameters (directly or indirectly)	3	1	1	3		root access/ authenticated OBD access	CAN
9	68	UC3	Compromise of back-end server	Gateway to ECU network(2,3,18-21)	Elevation of privilege	Unauthorised internet access to the server (enabled for example by backdoors, unpatched system software vulnerabilities, SQL attacks or other means)	3	1	0	2	Safety relevant	Autorisation/ usage of TLS	CAN

10	69	UC3	Compromise of back-end server	Gateway to ECU network(2,3,18-21)	Elevation of privilege	Unauthorised physical access to the server (conducted by for example USB sticks or other media connecting to the server)	3	1	0	2		secure storage of Keys and IP / usage of SE	CAN
11	76	UC3	Communication channels used to attack a vehicle	Gateway to ECU network(2,3,18-21)	Repudiation	Accepting information from an unreliable or untrusted source	3	0	1	2		IP/Packege autentication	CAN
12	85	UC3	Human factor and social engineering	Gateway to ECU network(2,3,18-21)	Social engineering (several)	Erroneous use or administration of devices and systems (inc. OTA updates)	3	1	0	2		root access/4 eyes principle	CAN
13	105	UC3	Compromise of back-end server	Wider vehicle network(+)	Elevation of privilege	Unauthorised physical access to the server (conducted by for example USB sticks or other media connecting to the server)	3	1	0	2		secure storage of Keys and IP / usage of SE	CAN
14	121	UC3	Human factor and social engineering	Wider vehicle network(+)	Social engineering (several)	Misconfiguration of equipment by maintenance community or owner during installation/repair/use causing unintended consequence	3	1	1	3		root access/4 eyes principle	CAN

15	122	UC3	Human factor and social engineering	Wider vehicle network(+)	Social engineering (several)	Erroneous use or administration of devices and systems (inc. OTA updates)	3	1	1	3		root access/4 eyes principle	CAN
16	123	UC3	Human factor and social engineering	Wider vehicle network(+)	Social engineering (several)	Innocent victim (e.g. owner, operator or maintenance engineer) being tricked into taking an action to unintentionally load malware or enable an attack	3	1	1	3		root access/4 eyes principle	CAN
17	198	UC3	Compromise of back-end server	External server(1)	Elevation of privilege	Abuse of privileges by staff (insider attack)	4	0	0	3		Authentication /usage of SE	Ext. server
18	199	UC3	Compromise of back-end server	External server(1)	Elevation of privilege	Unauthorised internet access to the server (enabled for example by backdoors, unpatched system software vulnerabilities, SQL attacks or other means)	4	0	0	3		Autorisation/ usage of TLS	Ext. server
19	200	UC3	Compromise of back-end server	External server(1)	Elevation of privilege	Unauthorised physical access to the server (conducted by for example USB sticks or other media connecting to the server)	4	1	0	3		secure storage of Keys and IP / usage of SE	Ext. server

20	202	UC3	Compromise of back-end server	External server(1)	Tampering	Abuse of privileges by staff (insider attack)	4	0	0	3		Authentication /usage of SE	Ext. server
21	203	UC3	Compromise of back-end server	External server(1)	Tampering	Loss of information in the cloud. Sensitive data may be lost due to attacks or accidents when stored by third-party cloud service providers	3	1	0	2		IP/ Data Backup	Ext. server
22	204	UC3	Compromise of back-end server	External server(1)	Tampering	Unauthorised internet access to the server (enabled for example by backdoors, unpatched system software vulnerabilities, SQL attacks or other means)	4	1	0	3		Autorisation/ usage of TLS	Ext. server
23	205	UC3	Compromise of back-end server	External server(1)	Tampering	Unauthorised physical access to the server (conducted for example by USB sticks or other media connecting to the server)	4	1	0	3		secure storage of Keys and IP / usage of SE	Ext. server
24	206	UC3	Compromise of back-end server	External server(1)	Tampering	Information leakage or sharing (e.g. admin errors, storing data in servers in garages)	3	1	0	2		IP/ Secure Data Storage	Ext. server

25	220	UC3	Communicatio n channels used to attack a vehicle	External server(1)	Elevation of privilege	An unprivileged user gains privileged access, for example root access	4	0	0	3		Keys,IP,root access/ SE	Ext. server
26	222	UC3	Update process used to attack a vehicle	External server(1)	Tampering	Compromise of over the air software update procedures, This includes fabricating system update program or firmware	3	1	1	3		IP/Packege autentication	Ext. server
27	225	UC3	Human factor and social engineering	External server(1)	Social engineering (several)	Erroneous use or administration of devices and systems (inc. OTA updates)	3	1	0	2		root access/4 eyes principle	Ext. server
28	226	UC3	Human factor and social engineering	External server(1)	Social engineering (several)	Innocent victim (e.g. owner, operator or maintenance engineer) being tricked into taking an action to unintentionally load malware or enable an attack	3	1	0	2		root access/4 eyes principle	Ext. server
29	227	UC3	Compromise of external connectivity	External server(1)	Tampering	Manipulation of telematics (e.g. manipulate temperature measurement of sensitive goods, remotely unlock cargo doors)	3	0	2	3		Keys,IP,root access/ SE, root of trust	Ext. server

30	234	UC3	Communication channels used to attack a vehicle	V2X communications(10-13)	Repudiation	Accepting information from an unreliable or untrusted source	3	0	2	3		IP/Packege autentication	TLS
31	246	UC3	Human factor and social engineering	Extra	Social engineering (several)	Defined security procedures are not followed	4	0	0	2		root access/4 eyes principle	Soc. Eng.
32	247	UC3	Compromise of external connectivity	Extra	Elevation of privilege	Virus from infected media connected to system	3	0	2	3		IP/Packege autentication	Soc. Eng.

Developed Solution and Evaluation of Improvement

We used authenticated and encrypted communication between the Update infrastructure <-> gateway (GW) and between GW <-> ECU which provided integrity and authenticity. The update package is E2E (Update infrastructure <-> ECU) encrypted, which produces the pre-requirement of a shared key between the Update infrastructure and ECU. In DP19, we consider only one ECU which receives its own update package and process it is flashing. Confidentiality, availability and non-repudiation are achieved through correct configuration and use of TLS and encrypted CAN messaging. To ensure availability in the face of DoS-attacks in CAN is a difficult problem in general. However, there are well known patterns to avoid CAN issues, see [1].

TLS

TLS covers several security features: confidentiality or privacy (by encryption), integrity (by message authentication codes), and (one-party or mutual) authenticity (by verification of pre-shared keys or via a public key infrastructure). Integrity is a mandatory feature, privacy and authenticity are optional at least in older versions of the protocol. Within DP19, TLS can address several problems, related particularly to the delivery phase of Over-the-Air updates (OTAs):

- **Encrypted delivery of updates to the recipient:**
  to avoid exposure of partial or complete firmware dumps to malicious eavesdroppers, who might analyze and reverse engineer the images to steal intellectual property like configuration or trained model data, or to find vulnerabilities and exploits to prepare attacks.
- **Integrity of the updates:**
  to mitigate the risk of corrupted updates that might leave the updating asset in an unusable state, potentially even unrecoverable or unsafe.
- **Authenticity of the parties:**
  - **Authenticity of the recipient device:**
    to ensure that the correct firmware update is delivered and matches the exact device; e.g., in case a device within a vehicle has been swapped with one from a different production run due to a repair.
  - **Authenticity of the server delivering the update:**
    to ensure that an adversary cannot spoof the device and deliver tampered or outdated versions.
- **Non-repudiation of the connection:**

to prove that information about available updates has been propagated to and acknowledged by the vehicle; e.g., to ensure that mandatory update cycles are kept for legal and warranty reasons.

We remark that some of these goals can be achieved by other means:

- Encryption is not strictly necessary if firmware on the device is stored encrypted, and each update is already pre-encrypted with a device-specific key.
- Integrity can be ensured by signing the updates, assuming the receiving end has the possibility to verify the signature against a local trust anchor.
- Authenticity of the recipient is not required if device-specific images are produced and the device has means to do pre-flight checks whether the delivered update is applicable (e.g., if images must include the serial number of the device and updates are only applied if the serial number matches and the manufacturer's or OEM's signature is valid).

In contrast, authenticity of the update provider cannot easily be achieved without a handshake; as in key revocation, the crucial property that is ensured here, is "up-to-date" information. However, there are valid reasons to not make this a strict requirement or allow bypassing checking the provider authenticity: for example, if the device needs to be updated in a remote area without direct link to the OEM, or if the connectivity unit itself is damaged or outdated. In such situations, the update could be delivered on a different channel (e.g., with physical access to the vehicle) by a man-in-the-middle caching the update, and no party could verify the authenticity of the communication partner or the freshness of the payload end-to-end. In our DP TLS used only between Update infrastructure and GW and not between Update infrastructure and ECU.

TLS provides authenticated and confidential communication channel. By the usage of TLS in our Secure delivery pattern, the following threats will be mitigated: Spoofing, Tampering, Repudiation.

Secure on-board communication

Here we assume the usage of symmetric vehicle specific keys for secure on-board communication (pre requirement in DP19). Also, SHE (Secure Hardware Extension) module or HSM (Hardware Security Module) is recommended as secure storage for symmetric encryption key and HW acceleration of AES calculation. Here we need bus system with high data rate (e. g. CAN FD) to cope with additional payload of authentication data. The use of MAC (Message Authentication Code) will provide the authenticity and

integrity of message exchange (GW<->ECU), cause this cryptographic checksums or keyed hash functions can be reproduced only with correct key.

The first step is that both ends of the system (GW<->ECU) must agree on the format and content of messages. That is why most of the time the used technology to secure the communication is defined by the vehicle integrator. The key management questions are not in scope of this document. We assume that communication partners have already shared the secret for symmetric-key based communication in a secure way. Furthermore, we assume a secure key protected storage in isolated area not susceptible to software attacks.
To provide the non-repudiation of communication (GW<->ECU), the usage of counter to the message and calculation of the MAC over the payload and the entire counter are needed. Each communication partner needs have a mechanism to sync his respective counter.

This security mitigations caused certain amount of overhead in terms of:

- Message size
- Latency
- Computation time
- Memory allocation

The communication between ECUs within the same vehicle requires low-speed, asymmetric cryptography and high-speed symmetric encryption. The Secure Onboard Communication (SecOC) is Secure transmission of sensitive data to protect against malicious manipulation and from intruders sending unauthorized information. To provide the authentication of provided update binary the package needs to be checked with the corresponding key (E2E (Update infrastructure <-> ECU)). Using a group key for update packages is possible but prevents individual download packages for each ECU.

Secure Boot-Chain of trust(usage of ARM TrustZone)

One of the main assets is Secure Boot-Chain of trust, which detects SW manipulation in boot process. For that, we need cryptographic integrity checks added to each stage of the boot sequence. These checks should prevent running of any unauthorised or maliciously modified SW. In this case, we approach symmetric encryption with ECU individual keys and use of CMAC to provide integrity and authenticity of the SW.

As part of the secure boot process, the HW should first verify the first SW part (bootloader) of the firmware to be executed. Afterwards, each part verifies and executes the subsequent software part – using functions of the HW. The use of Arm TrustZone is motivated by its provided security features and the possibility to reduce the security costs. Combined with AUTOSAR technology, this simplifies the aggregation of several applications on a single ECU, allows keeping costs of the environment low, and spending more time on improving application performance. The selected ECU with ARM Trust Zone provides us with secure key stores to guarantee device validity. TrustZone technology can isolate partitions within existing NVM and RAM resources, so that they can only be accessed by software running in the secure mode of the CPU.

Described isolation allows hardware supported separation of secure and non-secure functions. This hardware security technology makes it possible to execute the processor instructions in one of two possible security models. The first normal-world model executes untrusted code, where the second secure-world runs secured services. These processor modes have independent memory address spaces and different privileges.

Code running in the normal world cannot access the secure world address space, while code in secure world can access the normal world address space. For the current executing world indication, there is a special processor bit (the NS bit). This bit accompanies transactions over the memory bus and certain I/O buses for peripherals, which enables to allocate memory solely to the secure world, and to control which peripherals are accessible from the different worlds. This feature ensures the communication of subsystems, components or even individual applications only through strictly controlled communication channels, which provide strong defenses against illegally manipulation of application resources.

The storing of Security functions and protecting their functionality enables enforcement of the cryptographic boundary and realizes secure boot. Secure boot enables authentication during loading, but before execution. The attestation of the used execution environment and attestation of any software component that could influenced the security of used execution environment, needs to be established. Only an isolated execution environment equipped with a root of trust is a real “trusted" execution environment (TEE).

One each next step in the chain concerns the authenticity check of code and integrity of next link needs to be performed before execution. Any weakness of the chain breaks the whole security of the ECU. The process of code integrity confirmation (called: measurement) needs to be completed before the loaded software module is executed, because the system is in known state and will not have started to generate

any divergent local state. The measurement can be realized with previously loaded trusted code on the platform, which performs the “measurement” of the next loaded software module before its execution. The resulted chain of trust guarantees the measures of each loaded software module, before transferring control.

The first code executed on the platform becomes trusted by using a security anchor (small immutable piece of verifier code, isolated within ROM). Its execution during every boot process initializes the root of trust of the ECU. In case of non-authenticated code (i.e. the authentication fails), the system should retain the maximum level of functionality, which is still secure and safe (as an alternative solution the previous version need to be available).
TrustZone itself is not a complete solution. By itself, the capabilities include separating CPU state, memory and peripherals into two separate worlds. With the help of this hardware enforced isolation, multiple security concepts can be implemented: In other words: ARM TrustZone is a hardware-based isolation technology, where software can be executed in one of two possible worlds. However, it relies on additional hardware in the SoC to enforce proper isolation of memory and peripherals.

While the dependence on SoC vendor hardware implementations is a potential weakness of TrustZone, it simultaneously entails one of its great strengths: flexibility. Another key advantage of TrustZone is that the isolation happens directly on the application processor, which offers relatively high performance. Additionally, since the technology is enabled directly on the CPU itself, there are no additional costs for external co-processors or flash. Every code added to the Trusted Execution Environment (TEE) naturally increases the TEE’s attack surface. Therefore, special care should be taken with the implementation of such secure services. The trusted code base should be kept minimal to reduce the risk of security vulnerabilities.

The TrustZone based Trusted Execution Environment provides a “Trusted World“ where the security boundary is small enough to offer a route to certification and provable security. It is typically used for securing cryptographic keys, credentials and other secure assets. The Trusted World is used with its own dedicated secure operating system, called the Trusted Execution Environment (TEE) that works together with the conventional operating system, such as Linux or Android, to provide secure services. The interface from client applications to access the TEE, and for the use of trusted applications executing within the TEE, are standardised by GlobalPlatform [4,5]. OP-TEE (Open Portable Trusted Execution Environment) is an

open-source implementation supporting GlobalPlatform APIs. It is meant to execute at the TrustZone S-EL1 level and is one of the most wide-spread TEE implementations using Arm TrustZone (it has been ported to over 20 platforms).

The biggest threat to the security of TrustZone-based secure services remains the software running in the trusted execution environment. The entire TEE may be compromised from a vulnerability inside a trusted application. The state-of-the-art for attestation is to compute a signature with an attestation key over the software's measurement, with requirement, that the software's measurement and the attestation key are securely stored by the root of trust. Only an isolated execution environment equipped with a root of trust is a real “trusted" execution environment (TEE). The software component for TPM service (secure functions provided TPM functionality) is implemented in the secure word of Trust Zone and uses the devise key as endorsement key of the TPM service. This isolation protects the root of trust from software attacks. Trust Zone does not protect against physical tampering of device main memory. In our realisation, the TPM service provides secure storage, measurement and attestation functionality. That makes possible to run the normal world application to run in trusted state.

Related Work

For more information about DPs and used CTEs see deliverables of WP3.1 [16] and WP3.2 [17].

Conclusion

The aim of this AP is to analyse DP19 against security threats that may be imposed by the used CTEs and technologies. According to identified threats, we suggest appropriate countermeasures.

References

[1] T. Hoppe, S. Kiltz, and J. Dittmann Security Threats to Automotive CAN Networks – Practical Examples and Selected Short-Term Countermeasures

[2] F. Ahmed-Zaid, F. Bai, S. Bai, C. Basnayake, B. Bellur, S. Brovold, G. Brown, L. Caminiti, D. Cunningham, H. Elzein, et al. Vehicle Safety Communications – Applications (VSC-A) Final Report: Appendix Volume 3 Security. Tech. rep. U.S. Department of Transportation, National Highway Traffic Safety Administration, Sept. 2011, p. 21. url: https://www.nhtsa.gov/sites/nhtsa.dot.gov/files/811492d.pdf.

[3] Dirk Fox. “Hardware Security Module (HSM)”. de. In: Datenschutz und Sicherheit (DuD) 33 (Sept. 2009), p. 564. doi: 10.1007/s11623- 009- 0145-9.

[4] GlobalPlatform. Annex C: TLS Specification of TEE Sockets API Specification v1.0.1. Jan. 2017. url: https://globalplatform.org/specs- library/tee-sockets-api-specification-v1-0-1/ .

[5] GlobalPlatform. Introducing IoTOPIA™. Tech. rep. Oct. 2019. url: https://globalplatform.org/resource-publication/ebook-introducing- iotopia/.

[6] Sebanjila Kevin Bukasa et al. “How TrustZone Could Be Bypassed: Side-Channel Attacks on a Modern System-on-Chip”. In: *Information Security Theory and Practice*. Ed. by Gerhard P. Hancke and Ernesto Damiani. Cham: Springer International Publishing, 2018, pp. 93–109. isbn: 978-3- 319-93524-9.

[7] SECREDAS, D1.7
[8] SECREDAS, UC3 SAHARA
[9] IEEE, "RFC 8446: The Transport Layer Security (TLS) Protocol Version 1.3," 2018.
[10] IEEE, "RFC 5246: The Transport Layer Security (TLS) Protocol Version 1.2," 2008.
[11] Bundesamt für Sicherheit in der Informationstechnik BSI, "Technische Richtlinie TR-02102-2: Kryptografische Verfahren: Empfehlungen und Schlüssellängen," 2019.
[12] E. Ronen, R. Gillham, D. Genkin, A. Shamir, D. Wong and Y. Yarom, "The 9 Lives of Bleichenbacher's CAT: New Cache ATtacks on TLS Implementations," in *Symposium on Security and Privacy*, San Fransisco, CA, US, 2019.
[13] IEEE Transport Layer Security working group, "TLS 1.3 Implementations," 2019. [Online]. Available: ^ https://github.com/tlswg/tlswg- wiki/blob/master/IMPLEMENTATIONS.md. [Accessed 18 02 2020].
[14] H. C. Rudolph, "TLS Cipher Suite Search," [Online]. Available: https://ciphersuite.info/. [Accessed 18 02 2020].
[15] TPM https://www.infineon.com/cms/en/product/security-smart-card-solutions/optiga-embedded-security-solutions/optiga-tpm/sli- 9670/?redirId=68293
[16] SECREDAS, WP3.1 Deliverables (link)
[17] SECREDAS, WP3.2 Deliverables (link)

2.2.3 AP04: CCPP – Common Criteria Profile Protection

2.2.1.1 Brief Overview

Analysis Aim:	The aim of this Analysis Pattern is to analyse the security level of the Hardware platform called Secure Element (HSM). Within SECREDAS, the security level of end-to-end security shall be sufficiently high to resist to the identified threats. This shall be reached by integrating an HSM. CCPP analysis patterns shall specify the minimum requirements of the CTE Secure Element (HSM) to reach this minimum level. CTE shall be resistant to: • Malfunction • Abuse of functionality • Physical Manipulation and Probing • Leakage • Random number
Abstract:	PP0084 is a protection profile used by the certification body of common criteria evaluation. PP0084 is in particular related to Hardware Security chips used for example within smart card for banking. Important number of attacks are presented in the literature and used in the real life to extract the secret of the Secure Element; we can for example listed: • tamper-resistance attacks, • side channel attacks, • laser attacks (disturbing of the computing…) This analysis pattern will specify security requirement, which have to respect in regards of threat on the system (tamper-resistance, side channel, disturbing of the computing…). Goal is to take care of all these security requirements during design in order to allow to evaluate that the secure element will have a sufficient secure level in the identified domain.
Addressed Problem:	The security element shall be sufficiently resistant with respect to attacks known in the automotive domain. Analyses pattern Common Criteria Profile proposes the security level to be reached. This shall guarantee that the secure element within the end-to-end system will allow to secure the automotive

	infrastructure and potentially communication between different vehicle/servers.
Stakeholder:	Common Criteria Profile has to be used as an input to the design or as an output of the realization to know, if the required security level can be guaranteed. For systems with End-to-End security requirements, only hardware secure element (CTE Secure Element) shall be used, which conform these requirements.
Categories and Scope:	
Scope:	Methodology-AP
Safety:	CCPP Analysis pattern shall evaluate if the CTE Secure Element (used in of End-to-End security) respects part of PP0084 is sufficiently robust to attacks in this domain.
Security:	
Privacy:	
Input:	
Description:	The design of security features of the secure elements shall take care of the requirements given in PP0084
Example(s):	Secure Element shall be designed in order to mitigate side channel attacks
Addressed Elements:	
Related DP:	DP12 End-to-end security (focus on CTE Secure Element)
Contributors:	
Responsible:	• ISSM-THALES DIS Design Services SAS
List of Contributors:	MORIER Franck

2.2.3.2 Detailed description

Introduction

Security Certification Methods exist to evaluate the security level of hardware security platforms. These are applied in particular in security chips for banking cards, as well as in the identity domain (i.e. in the smart card area). One certification method used systematically is the common criteria evaluation method. It takes care of the security level of hardware circuit design but also considers known attacks realized on this circuit. In the smart card domain, PP0084 is the protection profile proposed in this context, in particular related to Hardware Security chips.

We propose to also use Common Criteria Profile as analysis pattern within the automotive domain. This will allow to:

- Ensure that a CTE Secure element is present in the automotive system.
- Guarantees that the security level for CTE Secure Element is well adapted to automotive context.
- Propose several security functions to be implemented or used by the SoC (System-on-a-Chip) of the automotive platform.
- Have sufficient resistance again threat (tamper-resistance, side channel, disturbing of computing…).

Description of the Common Criteria Protection Profile Analysis method

The CCPP method identifies pertinent threats on the hardware secure element. With the knowledge of these, according counter measures must be implemented into the secure element to mitigate these threats. Below we list part of the threats based on PP0084 which are relevant to the realization of secure elements (TOE – Target of Evaluation) in general:

Threat ID	Threat Description
T.Leak-Inherent	**Inherent Information Leakage** An attacker may exploit information, which is leaked from the TOE during usage of the Security IC in order to disclose confidential user data as part of the assets.
T.Phys-Probing	**Physical Probing** An attacker may perform physical probing of the TOE in order (i) to disclose user data while stored in protected memory areas, (ii) to disclose/reconstruct the user data while processed or (iii) to disclose other critical information about the operation of the TOE to enable attacks disclosing or manipulating the user data of the Composite TOE or the Security IC Embedded Software.
T.Malfunction	**Malfunction due to Environmental Stress** An attacker may cause a malfunction of TSF or of the Security IC Embedded Software by applying environmental stress in order to (i) modify security services of the TOE or (ii) modify functions of the Security IC Embedded Software (iii) deactivate or affect security mechanisms of the TOE to enable attacks disclosing or manipulating the user data of the Composite TOE or the Security IC Embedded Software. This may be achieved by operating the Security IC outside the normal operating conditions.
T.Phys-Manipulation	**Physical Manipulation** An attacker may physically modify the Security IC in order to (i) modify user data of the Composite TOE, (ii) modify the Security IC Embedded Software, (iii) modify or deactivate security services of the TOE, or (iv) modify security mechanisms of the TOE to enable attacks disclosing or manipulating the user data of the Composite TOE or the Security IC Embedded Software.
T.Leak-Forced	**Forced Information Leakage** An attacker may exploit information, which is leaked from the TOE during usage of the Security IC in order to disclose confidential user data of the Composite TOE as part of the assets even if the information leakage is not inherent but caused by the attacker.
T.Abuse-Func	**Abuse of Functionality** An attacker may use functions of the TOE which may not be used after TOE Delivery in order to (i) disclose or manipulate user data of the Composite TOE, (ii) manipulate (explore, bypass, deactivate or change) security services of the TOE or (iii) manipulate (explore, bypass, deactivate or change) functions of the Security IC Embedded Software or (iv) enable an attack disclosing or manipulating the user data of the Composite TOE or the Security IC Embedded Software.
T.RND	**Deficiency of Random Numbers**

	An attacker may predict or obtain information about random numbers generated by the TOE security service for instance because of a lack of entropy of the random numbers provided.

Analysis of DP12: End-to-End Security with a Secure Element

To analyze the security level of the secure element and to verify if it is robust against previously listed threats, PP0084 proposes to check if specific security features are implemented in the system. Since we do not have a concrete implementation as reference, we will give a check list on the most critical security features to verified.

Relevant security features related to PP084 are listed below:

- Limited fault tolerance (malfunction).
- Failure with preservation of secure state (malfunction).
- Limited capabilities (abuse of functionality).
- Limited availability (abuse of functionality).
- Audit storage (abuse of functionality).
- Stored data confidentiality (physical Manipulation and Probing).
- Stored data integrity monitoring and action (physical Manipulation and Probing).
- Resistance to physical attacks (physical Manipulation and Probing).
- Basic internal transfer protection (leakage).
- Subset information control (leakage).
- Random number generation (random number).

Other security features related to authentication and loader may be added in a specific context and are scope of future work.

Developed solution

The proposed solution is to use the PP0084 security feature and to apply while realizing secure element for the usage in the automotive domain. This ensures, that the developed secure element will have a robust design and will be resistant to known attacks such as side channel attacks, tampering attacks and disturbing of computing.

Evaluation of Improvement

If the common criteria profile constraints are respected, the security level of the CTE secure element is qualified for the automotive domain. The initial threats will be restricted to a low level of apparition and the security services will be sufficiently robust.

Evaluation of Analysis method

Also other security analysis methods can be applied. But using the CCPP methods allows to have a common referential compared to other domains than automotive. In particular for Secure Element, using CCPP allows to apply P0084 and thereby transfers knowledge for realizing secure elements from the smart card domain.

Related Work

References to existing work:

- Common Criteria Chip Evaluation for smart card chip. PP0084 and any evaluated chip referring to this specification.[1]
- JIL attacks.[2]

Conclusion

The CCPP Analysis Pattern formulates the idea to use common criteria evaluation method within automotive context to evaluate the hardware security element and guide a secure implementation. This will allow having a minimum-security level respecting requirement to be implemented. In consequence, the End-to-End security embedding and using the secure element has a sufficient security level as well as a solid the security foundation to reach the overall targeted security level.

References

[1] PP0084 - BSI-CC-PP-0084-2014 for Security IC Platform Protection Profile with Augmentation Packages Version 1.0.
[2] Joint Interpretation Library – Application of Attack Potential to Smartcards and Similar Devices, Version 3, 2019.

2.2.4 AP05: Privacy-preserving Multi-party Access Control

2.2.4.1 Brief Overview

Analysis Aim:	Evaluation of multi-party access control policies under encryption.
Abstract:	Multi-party access control has been proposed to enable collaborative decision making for the protection of co-owned resources. In particular, multi-party access control aims to reconcile conflicts arising from the evaluation of policies authored by different stakeholders. However, existing solutions do not address the protection of policies themselves, whose disclosure can leak sensitive information about, e.g., the relationships of co-owners with other parties. To address this issue, we propose a privacy preserving mechanism for the evaluation of multi-party access control policies, which preserves the confidentiality of users' policies.
Addressed Problem:	Preserving the confidentiality of users' access policies regulating co-owned resources.
Stakeholder:	Data holders (e.g. Users), Access requester (e.g. Service Provider), Data Server, Semi Trusted Party
Categories and Scope:	
Scope:	Improvement-AP (Implementation of the multi-party policy evaluation in private setting)
Safety:	Privacy is the main focus of the proposed architecture.
Security:	
Privacy:	
Input:	
Description:	1. Users' policies in encryption format. 2. A multi-party access control policy specifying how users' policies are combined. 3. An access request for a co-owned resource. Example: Location-based application -> protecting the drivers' privacy policies, which determine who can access their location information.
Example(s):	
Addressed Elements:	
Related DP:	DP26: Privacy Preserving Multi-party Access Control
Contributors:	
Responsible:	Mina Sheikhalishahi, Nicola Zannone
List of Contributors:	N/A

2.2.4.2 Detailed description

Introduction

Over the last decade, collaborative systems (e.g., social networks, content sharing platforms) have gained momentum. These systems provide platforms and tools to support the interplay among their users and facilitate the sharing of information and resources. Users can establish online relationships that resemble real-life interpersonal relationships and, based on them, cooperate to create, manage and protect their

resources. As collaborative systems have emerged and their popularity is significantly increased, the need of proper mechanisms for protecting sensitive resources shared in these systems has become of paramount importance.

Multi-party access control has been proposed as a new paradigm to overcome the inherent limitations of traditional access control models in dealing with the demands of collaborative systems. Specifically, multi-party access control aims to support the collaborative protection of co-owned resources. This is achieved by providing a means for collaborative decision-making, in which the access requirements provided by all entities involved in the protection of a jointly managed resource are accounted for. Possible conflicts arising from conflicting access requirements are reconciled to determine whether access to the resource should be granted or not.

We have developed three set of protocols, one based on homomorphic encryption (HE) [3], one based on secure function evaluation (SFE) [4] using a direct encoding of three-valued logic operators and one based on SFE using a Boolean logic encoding of the operators. The proposed protocols can be used as building blocks for the evaluation of arbitrary collaborative policies, guaranteeing that policy evaluation does not leak any unintended information. We have implemented the proposed protocols and evaluated their computation and communication costs through experiments.

Analysis of shortcomings of traditional access control models

Several solutions for collaborative decision-making have been proposed over the years, ranging from authoritative approaches, in which the access requirements provided by the various entities are combined in a predefined manner, to approaches aiming at a mutual agreement among the entities involved in the protection of co-owned resources [1]. Regardless of how collaborative decisions are made, to the best of our knowledge no prior work has addressed the protection of the policies themselves within collaborative systems. Policies might contain sensitive personal information, e.g., about the relationships among entities, or confidential corporate information and, thus, their disclosure could raise concerns for users and organizations.

At the same time, users might also want to minimize the amount of information to be disclosed to the collaborative platform itself. Several cases have been reported where online platforms and services used and/or disclosed their personal data (including their privacy preferences) without user consent or were impacted by data breaches. These concerns are becoming particularly critical with the advancement in data science and the increasing number of available data analysis tools that can be used to correlate user policies with other pieces of personal information and therefore reveal much more insights about users.

For instance, knowledge of the interpersonal relations between users along with their privacy settings and other metadata can be used to profile users (e.g., identifying individuals alienated from the community), and this information exploited for targeted advertising.

We argue that minimizing the information a user must disclose, allows reducing the trust that he/she has to pose on the platform. In this light, users should be able to specify their policies in a private form while the collaborative platform should still remain able to make access decisions based on those policies. In this work, we address this challenge and propose a privacy-preserving framework for multi-party access control in which users provide their policies in a private form and policy evaluation is performed on the provided private policies, thus preserving the confidentiality of user policies. Following the multi-party policy model in [2], we build a `collaborative' policy. It specifies positive and negative authorizations for a given co-owned resource by combining user policies using policy combining operators. Intuitively, policy combining operators define strategies specifying how policy conflicts should be resolved (e.g. permit overrides, deny overrides, first applicable). For the evaluation of collaborative policies while protecting the confidentiality of users' policies, we employ secure computation protocols that implement policy combining operators on private inputs.

To achieve a practical mechanism for multi-party access control supporting policy evaluation on private inputs, we investigate the use of two well-known privacy-preserving techniques, namely HE [3] and SFE [4] for the design of secure computation protocols tailored to operate on three-valued logic. We also investigate different encodings for these operators. Based on these privacy-preserving technologies and encodings, we designed three variants for each policy combining operator: one based on homomorphic encryption, one based on SFE using a direct encoding of three-valued logic operators and one based on SFE using a Boolean logic encoding of the operators. The proposed protocols can be used as building blocks for the evaluation of arbitrary collaborative policies.

We have demonstrated the security of the proposed protocols as well as of their composition in the presence of a semi-honest adversary, thus guaranteeing that policy evaluation does not leak any unintended information. We have implemented the proposed protocols and evaluated their computation and communication costs through experiments. The results show that the SFE-based protocols defined over a Boolean logic encoding of the operators outperform the protocols defined over the other approaches and provide a foundation for the practical realization of privacy-preserving mechanisms for multi-party access control.

Developed solution

To enable the evaluation of multi-party policies while protecting the confidentiality of user policies, we design an access control mechanism that supports policy evaluation on private inputs. An overview of our mechanism is presented in Figure [1]. Our framework comprises four main entities:

- *Data holders* share their resources and data along with policies for their protection and want their policies to remain private. The data holders provide their policies in a private form to a Data Server (DS).
- *Access requester* requests access to resources and data.
- *DS* stores the data holders' resources and is responsible for their protection. The DS evaluates data holders' policies to determine whether access to their resources should be granted.
- Semi Trusted Party (STP) is a semi-honest entity that assists DS in the secure evaluation of data holders' policies.

Data holders are not involved in the evaluation of either their policies or the multi-party policy and, thus, they are not required to be online for a decision to be made. This task is performed by DS together with STP. The ’private form' in which policies are provided and the protocols used for policy evaluation are dictated by the underlying privacy-preserving technique. In HE, STP generates public (p_k) and private (s_k) keys and sends the public key p_k to the data holders and to DS. Data holders encrypt their policies using the public key received by STP and send the encrypted policies to DS. On the other hand, in SFE users generate two secret shares of their policies and provide one share to DS and one to STP. Upon receiving an access request, DS's policy evaluation point evaluates the request against each user policy in private form with the assistance of STP, resulting in an access decision (in private form) for each policy (represented by (d_1),...,(d_n) in Figure 7).

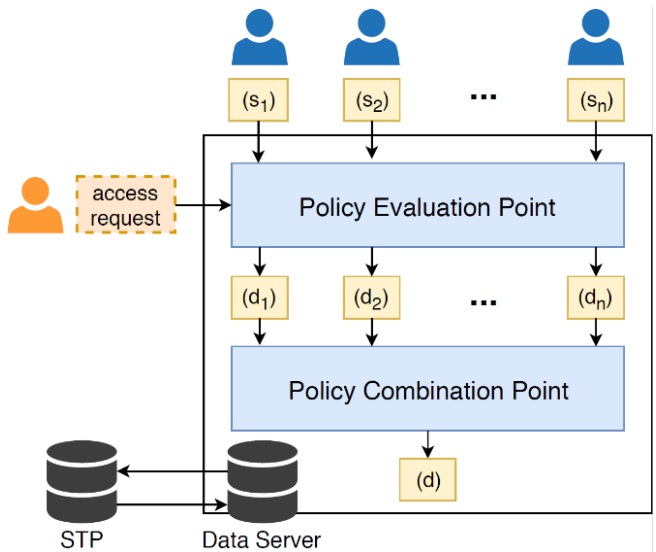


Figure 7: The architecture of privacy-preserving multi-party policy evaluation

Below, we present the designed protocols in this framework in terms of three different private encodings:

1) Homomorphic Encryption:

Operator	**Protocol**	*Definition*
$\neg a$	$not^H([a])$	$([a \stackrel{?}{=} 1] \otimes [a]) \oplus (([1] \ominus [a \stackrel{?}{=} 1]) \otimes ([2] \ominus [a]))$
$\sim a$	$wea^H([a])$	$[a \stackrel{?}{=} 2] \otimes [a]$
$a \widetilde{\sqcup} b$	$smax^H([a],[b])$	$(([1] \ominus [a \stackrel{?}{\leq} b]) \otimes [a]) \oplus ([a \stackrel{?}{\leq} b] \otimes [b])$
$a \widetilde{\sqcap} b$	$smin^H([a],[b])$	$([a \stackrel{?}{\leq} b] \otimes [a]) \oplus (([1] \ominus [a \stackrel{?}{\leq} b]) \otimes [b])$
$a \sqcup b$	$wmax^H([a],[b])$	$([a \stackrel{?}{=} 1] \ \texttt{XOR} \ [b \stackrel{?}{=} 1]) \oplus \Big(([1] \ominus ([a \stackrel{?}{=} 1] \ \texttt{XOR} \ [b \stackrel{?}{=} 1])) \otimes smax^H([a],[b])\Big)$
$a \sqcap b$	$wmin^H([a],[b])$	$([a \stackrel{?}{=} 1] \ \texttt{XOR} \ [b \stackrel{?}{=} 1]) \oplus \Big(([1] \ominus ([a \stackrel{?}{=} 1] \ \texttt{XOR} \ [b \stackrel{?}{=} 1])) \otimes smin^H([a],[b])\Big)$
$a \nabla b$	$po^H([a],[b])$	$([a \stackrel{?}{=} 1] \otimes [b \stackrel{?}{=} 1]) \oplus \Big(([a \stackrel{?}{=} 2] \otimes [b \stackrel{?}{=} 2]) \oplus ([a \stackrel{?}{=} 2] \ \texttt{XOR} \ [b \stackrel{?}{=} 2])) \otimes [2]\Big)$
$a \triangle b$	$do^H([a],[b])$	$([a \stackrel{?}{=} 1] \otimes [b \stackrel{?}{=} 1]) \oplus \Big(([a \stackrel{?}{=} 2] \otimes [b \stackrel{?}{=} 2]) \oplus ([a \stackrel{?}{=} 2] \otimes [b \stackrel{?}{=} 1]) \oplus ([a \stackrel{?}{=} 1] \otimes [b \stackrel{?}{=} 2])) \otimes [2]\Big)$
$a \rhd b$	$fa^H([a],[b])$	$(([1] \ominus [a \stackrel{?}{=} 1]) \otimes [a]) \oplus ([a \stackrel{?}{=} 1] \otimes [b])$

In this table addition, subtraction, multiplication, comparison, and equality protocols (already defined in the literature [3], [5], [6]) serve as building blocks for creating our protocols representing the main operators for evaluating a policy.

2) Secure Functional Evaluation using three valued logic:

Operator	Protocol	*Definition*
$\neg a$	$not^T(\langle a\rangle)$	$\langle a \stackrel{?}{=} 1\rangle \times \langle a\rangle + (\langle 1\rangle - \langle a \stackrel{?}{=} 1\rangle) \times (\langle 2\rangle - \langle a\rangle)$
$\sim a$	$wea^T(\langle a\rangle)$	$\langle a \stackrel{?}{=} 2\rangle \times \langle a\rangle$
$a \tilde{\sqcup} b$	$smax^T(\langle a\rangle, \langle b\rangle)$	$\langle a\rangle \vee \langle b\rangle$
$a \tilde{\sqcap} b$	$smin^T(\langle a\rangle, \langle b\rangle)$	$\langle a\rangle \wedge \langle b\rangle$
$a \sqcup b$	$wmax^T(\langle a\rangle, \langle b\rangle)$	$(\langle a \stackrel{?}{=} 1\rangle \vee \langle a \stackrel{?}{=} 1\rangle) + (\langle 1\rangle - (\langle a \stackrel{?}{=} 1\rangle \vee \langle a \stackrel{?}{=} 1\rangle)) \times smax^T(\langle a\rangle, \langle b\rangle)$
$a \sqcap b$	$wmin^T(\langle a\rangle, \langle b\rangle)$	$(\langle a \stackrel{?}{=} 1\rangle \vee \langle a \stackrel{?}{=} 1\rangle) + (\langle 1\rangle - (\langle a \stackrel{?}{=} 1\rangle \vee \langle a \stackrel{?}{=} 1\rangle)) \times smin^T(\langle a\rangle, \langle b\rangle)$
$a \nabla b$	$po^T(\langle a\rangle, \langle b\rangle)$	$(\langle a \stackrel{?}{=} 2\rangle \vee \langle b \stackrel{?}{=} 2\rangle) \times \langle 2\rangle + (\langle 1\rangle - (\langle a \stackrel{?}{=} 2\rangle \vee \langle b \stackrel{?}{=} 2\rangle)) \times smin^T(a, b)$
$a \vartriangle b$	$do^T(\langle a\rangle, \langle b\rangle)$	$(\langle 1\rangle - (\langle a \stackrel{?}{=} 0\rangle \vee \langle b \stackrel{?}{=} 0\rangle)) \times smax^T(a, b)$
$a \triangleright b$	$fa^T(\langle a\rangle, \langle b\rangle)$	$\langle a \stackrel{?}{=} 1\rangle \times \langle b\rangle + (\langle 1\rangle - \langle a \stackrel{?}{=} 1\rangle) \times \langle a\rangle$

In this table addition, subtraction, AND, OR, equality, and comparison gates serve as building blocks.

3) Secure Functional Evaluation using Boolean Circuits

Given a user policy s = (X,Y) , where X identifies the set of users who are permitted to access the content and Y represents the ones whose access is denied, and the ID of the requester r, we define:

$$\begin{aligned} \pi_P(s) &= (r \in X) \wedge \neg(r \in Y) \\ \pi_D(s) &= r \in Y \\ \pi_{NA}(s) &= \neg(r \in X) \wedge \neg(r \in Y) \end{aligned}$$

This formula helps to express the main operations of policies in terms of AND and OR operations. These two operations require low computational cost in Boolean Circuits. The following Table shows how the policy operators have been shaped using AND and OR gates.

Operator	π_P	π_D	π_{NA}
$\neg a$	$\pi_D(a)$	$\pi_P(a)$	$\pi_{NA}(a)$
$\sim a$	$\pi_P(a)$	$\pi_D(a) \vee \pi_{NA}(a)$	$false$
$a \tilde{\sqcup} b$	$\pi_P(a) \vee \pi_P(b)$	$\pi_D(a) \wedge \pi_D(b)$	$(\pi_{NA}(a) \wedge \neg\pi_P(b)) \vee (\pi_{NA}(b) \wedge \neg\pi_P(a))$
$a \tilde{\sqcap} b$	$\pi_P(a) \wedge \pi_P(b)$	$\pi_D(a) \vee \pi_D(b)$	$(\pi_{NA}(a) \wedge \neg\pi_D(b)) \vee (\pi_{NA}(b) \wedge \neg\pi_D(a))$
$a \sqcup b$	$(\pi_P(a) \wedge \neg\pi_{NA}(b)) \vee (\pi_P(b) \wedge \pi_{NA}(a))$	$\pi_D(a) \wedge \pi_D(b)$	$\pi_{NA}(a) \vee \pi_{NA}(b)$
$a \sqcap b$	$\pi_P(a) \wedge \pi_P(b)$	$(\pi_D(a) \wedge \neg\pi_{NA}(b)) \vee (\pi_D(b) \wedge \neg\pi_{NA}(a))$	$\pi_{NA}(a) \vee \pi_{NA}(b)$
$a \nabla b$	$\pi_P(a) \vee \pi_P(b)$	$(\pi_D(a) \wedge \neg\pi_P(b)) \vee (\pi_D(b) \wedge \neg\pi_P(a))$	$\pi_{NA}(a) \wedge \pi_{NA}(b)$
$a \vartriangle b$	$(\pi_P(a) \wedge \neg\pi_D(b)) \vee (\pi_P(b) \wedge \neg\pi_D(a))$	$\pi_D(a) \vee \pi_D(b)$	$\pi_{NA}(a) \wedge \pi_{NA}(b)$
$a \triangleright b$	$\pi_P(a) \vee (\pi_{NA}(a) \wedge \pi_P(b))$	$\pi_D(a) \vee (\pi_{NA}(a) \wedge \pi_D(b))$	$\pi_{NA}(a) \wedge \pi_{NA}(a)$

Evaluation

This is the first time that the *privacy policies* have been considered as private information by themselves. We implemented the proposed framework in three different private setting, namely: 1) homomorphic

encryption, 2) Boolean circuits based on a three-valued encoding, and 3) Boolean circuits based on a Boolean encoding. Tables 1 and 2 show the computation and communication costs.

Table 1:

Computation time (in ms) for single operator policies.

The best results for each protocol are reported in bold.

Protocol	*HE*	*SFE-T*	*SFE-B*
not	262.710	3.106	**0.314**
wea	182.827	2.903	**0.787**
smax	216.687	**0.782**	0.803
smin	215.137	**0.768**	0.788
wmax	576.736	3.290	**0.835**
wmin	578.821	3.208	**0.821**
po	697.489	3.229	**0.800**
do	746.158	3.065	**0.783**
fa	236.091	3.292	**0.851**

Table 2:

Bandwidth usage (in bytes) for single operator policies.

The best results for each protocol are reported in bold.

Protocol	*HE*	*SFE-T*	*SFE-B*
not	7681	67326	**42**
wea	6144	62866	**4122**
smax	7169	4230	**4072**
smin	7169	4242	**4125**
wmax	19459	66394	**4078**
wmin	19459	67122	**4090**
po	24580	71210	**4130**
do	26117	61369	**4071**
fa	7681	65428	**4124**

Related Work

To the best of our knowledge, our work is the first that addresses the problem of secure evaluation of composite policies encompassing both positive and negative authorizations.

Conclusion

Multi-party access control has emerged as a paradigm to support collaborative decision-making for co-owned resources. Several solutions have been proposed with the purpose of combining the access requirements of the users involved in the protection of co-owned resource. However, the problem of protecting the confidentiality of users' policies in multi-party access control has not been addressed. In this work, we have presented the first mechanism for the secure evaluation of multi-party access control policies, which preserves the confidentiality of user policies. To this end, we have proposed secure computation protocols based on three different privacy-preserving approaches: 1) homomorphic encryption, 2) Boolean circuits based on a three-valued encoding, and 3) Boolean circuits based on a Boolean encoding. An experimental evaluation of the proposed protocols shows that the latter approach outperforms the other two approaches in terms of both computation and communication costs. In

particular, the results obtained using SFE-based protocols relying on the Boolean encoding demonstrate the practical feasibility of privacy-preserving mechanisms for multi-party access control.

References

[1] Ricard. L. Fogues, Pradeep K. Murukannaiah, Jose M. Such, Munindar P. Singh, Sharing Policies in Multiuser Privacy Scenarios: Incorporating Context, Preferences, and Arguments in Decision Making, ACM Trans. Comput.-Hum. Interact., pages 5:1--5:29, 2017.

[2] Rauf Mahmudlu, Jerry den Hartog, Nicola Zannone, Data Governance and Transparency for Collaborative Systems, Data and Applications Security and Privacy, pages 199—216, 2016.

[3] Pascal Paillier, Public-key Cryptosystems Based on Composite Degree Residuosity Classes, Proceedings of International Conference on Theory and Application of Cryptographic Techniques, pages 223—238, 1999.

[4] Daniel Demmler, Thomas Schneider, Michael Zohner, ABY - A Framework for Efficient Mixed-Protocol Secure Two-Party Computation, Proceedings of Annual Network and Distributed System Security Symposium, 2015.

[5] Zekeriya Erkin, Thijs Veugen, Tomas Toft, and Reginald L. Lagendijk. 2012. Generating Private Recommendations Efficiently Using Homomorphic Encryption and Data Packing. IEEE Transactions on Information Forensics and Security 7, 3 (2012), 1053–1066.

[6] Majid Nateghizad, Zekeriya Erkin, and Reginald L. Lagendijk. 2016. An efficient privacy-preserving comparison protocol in smart metering systems. EURASIP Journal on Information Security 2016, 1 (2016), 11.

2.2.5 AP06: Safety Contents Management (SaCoMa)

2.2.5.1 Brief Overview

<table>
<tr><td>Name:</td><td>Safety Contents Management (SaCoMa)</td></tr>
<tr><td>Analysis Aim:</td><td>Safety-based classification of different common technological safety related solutions.</td></tr>
<tr><td>Abstract:</td><td>Classification of different main HW and SW components or parameters which allow to develop safety related architectures in combination with a specific DP for Sensor Data Fusion applications (DP24).</td></tr>
<tr><td>Addressed Problem:</td><td>Definition of an approach to classify different common technological safety related solutions to be used in safety related architectures. The following table lists safety contents (Application Note, Assumption of Use, Datasheet, etc.) that shall be used for safety-based classification of components/solutions. For each type of component/solution (HW Component, SW Component, Peripheral, SW Tool) is indicated (with the symbol “X”), which are the safety contents required for a safety-based classification.
<table>
<tr><th>Safety Content</th><th>HW Component</th><th>SW Component</th><th>Peripheral</th><th>SW Tool</th></tr>
<tr><td>Application Note</td><td>X</td><td>X</td><td>-</td><td>-</td></tr>
<tr><td>Assumption of Use</td><td>X</td><td>X</td><td>-</td><td>-</td></tr>
<tr><td>Datasheet</td><td>X</td><td>X</td><td>-</td><td>-</td></tr>
<tr><td>Errata Sheet</td><td>X</td><td>X</td><td>-</td><td>-</td></tr>
<tr><td>FIT</td><td>X</td><td>-</td><td>-</td><td>-</td></tr>
<tr><td>FMEDA</td><td>X</td><td>-</td><td>-</td><td>-</td></tr>
<tr><td>ISO 26262 compliance evidence</td><td>X</td><td>X</td><td>-</td><td>-</td></tr>
<tr><td>Safety Analysis Report</td><td>X</td><td>X</td><td>-</td><td>-</td></tr>
</table></td></tr>
</table>

	Safety Case	X	X	-	-
	Safety Functions	X	-	-	-
	Safety Manual	X	X	-	-
	Safety Mechanisms	X	X	-	-
	SW Tool Criteria Evaluation Report	-	-	-	X
	SW Tool Qualification Report	-	-	-	X
	User Manual (non safety-related)	X	X	-	-

Stakeholder:	Tier1, Tier2
Categories and Scope:	
Scope	Methodology-AP
Safety:	SaCoMa is focused on the Functional Safety of the HW and SW component architectures that are part of the Vehicle Architecture. • HW Components: contains HW components table and its related tables (HW component peripherals, HW component type, ASIL, documentation, etc.) • SW Components: contains SW components table and its related tables (SW component type, SW release type, ASIL, documentation etc.)
Security:	Cybersecurity consideration
Privacy:	Out of scope of this AP
Input:	
Description:	Management of safety related components available on the shelf according to Sensor/Data fusion application.
Example(s):	N/A
Output:	
Description:	Evaluation of best fitting safety related components to be used in order to develop safety related architectures according to the safety requirements constraints.
Addressed Elements:	
Related DP:	Sensor Data Fusion architecture (DP24) (CTE-20 Safety Element, CTE-21 Watchdog, CTE-22 Memory Protection Unit)
Contributors:	
Responsible:	• Calogero Calandra (calogero.calandra@marelli.com) • Salvatore Petroni (salvatore.petroni@marelli.com) • Sara Potenza (sara.potenza@marelli.com)
List of Contributors/Users:	• Calogero Calandra (calogero.calandra@marelli.com) • Salvatore Petroni (salvatore.petroni@marelli.com) • Sara Potenza (sara.potenza@marelli.com)

2.2.5.2 Detailed description

Introduction

For the development of real-time applications (e.g. in the automotive field) a proper level of safety integrity must be ensured for the functionalities that are labelled as safety critical. For example, in the automotive field the safety critical functionalities of an application shall be developed according the

proper automotive safety integrity level (ASIL) by following the guidelines provided by the Functional Safety standard for road vehicles (ISO 26262:2018). An example of application whose functionalities are labelled as safety critical, is the Data Fusion unit. The proper elements (HW component, SW component, peripheral and tool) must be chosen and used so that they comply with the integrity level required by each functionality of the Data Fusion unit. The required ASIL can for example be determined using Hazard Analysis and Risk Assessment (HARA) or equivalent methods and then decomposed to the single components according to the rules of ISO 26262. For this reason, the Safety Contents Management AP addresses the safety need of the Sensor Data Fusion architecture DP (DP24) by providing a safety-based classification of HW components, SW components, peripherals and tools.

The general idea of Safety Contents Management is to classify safety elements (HW, SW, peripherals and tools) in order to choose the proper element according to the safety requirements needed by the application (or Design Pattern) to be developed.

Description of the Safety Contents Management Analysis method

Each element (HW component, SW component, peripheral or tool) has a list of safety contents that shall be evaluated during the development of each functionality of the Data Fusion unit. According the integrity level required by each functionality, the proper element is selected by evaluating its safety contents. Safety contents for the different components (involved in the development process based on ISO26262) that have to be considered during the for safety-based classification of components can be found in the following table ():

Safety Content	HW Component	SW Component	Peripheral	SW Tool
Application Note	X	X	-	-
Assumption of Use	X	X	-	-
Datasheet	X	X	-	-
Errata Sheet	X	X	-	-
FIT	X	-	-	-
FMEDA	X	-	-	-
ISO 26262 compliance evidence	X	X	-	-
Safety Analysis Report	X	X	-	-
Safety Case	X	X	-	-
Safety Functions	X	-	-	-
Safety Manual	X	X	-	-
Safety Mechanisms	X	X	-	-
SW Tool Criteria Evaluation Report	-	-	-	X
SW Tool Qualification Report	-	-	-	X
User Manual (non-safety related)	X	X	-	-

Table 4: Safety contents for safety-classification of components

To reach a particular integrity level (e.g.: ASIL C) of a functionality, a microcontroller which has a safety measure (or safety mechanism) is used to avoid memory corruption. The microcontroller enables memory

partitioning with a MPU (Memory Protection Unit) safety function. Additionally, this function must be declared ASIL C by the supplier. To be certain that the microcontroller is used correctly for the MPU to consider ASIL C, the microcontroller must be used according its Assumption of Use.

MM developed a tool to realize the SaCoMa approach based on Microsoft Access. The scope of our tool is to collect different components/solutions and, for a specific component/solution, there is a list of attributes and in the corresponding cell is reported the value of the considered attribute (or safety content). The attributes can be:

- safety contents (e.g.: ASIL achieved by the considered component/solution, attachment such as Safety Manual, FMEDA, Safety Analysis Report. etc.).
- informative (e.g.: ID, name, type, vendor of the component solution, etc.).

Below there is a screenshot of our tool.

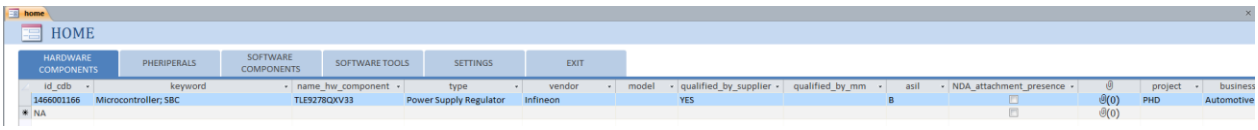


Figure 8: Screenshot of the developed tool for SaCoMa

Each tab refers to a different type of component/solution:

- HW Components.
- Peripherals.
- Software Components.
- Software Tools.

The tabs “Settings” and “Exit” are used to give access permissions to each user (e.g.: read access, read and write access, and so on) respectively quit the application. Considering the HW Component in the image (name: TLE9278QXV33) as an example, it is possible to see the following information:

- Id_cdb: indicates the ID of the component/solution related to a specific database;
- keyword: words associated with the component/solution; e.g.: ADAS, ADC tester, safety mechanism, uC, etc.
- name_hw_component: indicates the name of the component/solution;

- type: indicated the type of the component/solution (e.g.: Power Supply Regulator or ASIC or GPU, etc.);
- vendor: indicates the supplier of the component/solution;
- Model: indicated the model of the component/solution;
- qualified_by_supplier: indicates if the component/solution has been qualified (according ISO 26262) by the supplier or not;
- qualified_by_mm: indicates if the component/solution has been qualified internally;
- asil: indicates the ASIL achieved by the current component/solution;
- NDA_attachment_presence: indicates if there are any attachments, which can not be shared;
- annexes: is indicated by the corresponding symbol and collects different types of documents, such as: Application Note, Assumption of Use, Datasheet, Errata sheet, etc.;
- project: indicates the project in which the component/solution has been used;
- business_line: indicates the line of production, which has used and inserted this component/solution.

Furthermore, by opening the detail view of a specific component/solution, it is possible to add or see the relationship with other components/solutions:

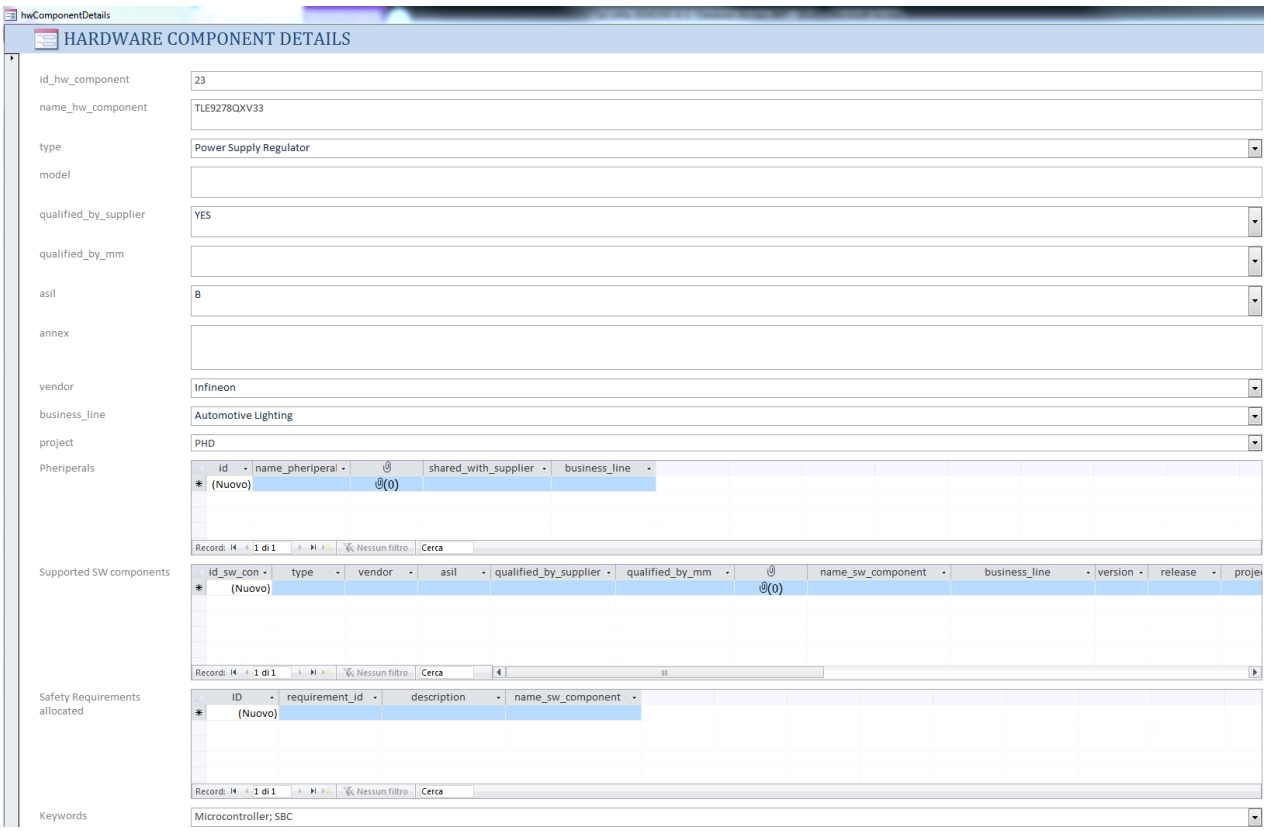


Figure 9: Detail view for hardware component TLE9278QXV33

In the figure above the details of the HW component TLE9278QXV33 can be seen. In addition to the previously mentioned information, it is allowed to link peripherals, supported SW components and allocate safety requirements to the HW Component:

- Peripherals: indicates peripherals connected to the HW component;
- Supported SW components: indicates SW components, that can be hosted by the HW component;
- Safety Requirements allocated: indicates safety requirements allocated to the specified HW component.

The screenshots and explanations illustrate our implementation for supporting the SaCoMa method. Our methodology allows to gather all safety-related information needed to classify each component/solution in such a way, that the proper component/solution can be selected for the application in the Sensor Data Fusion DP.

Analysis of DP26 Data Fusion architecture

For the realization of the Data Fusion process (by following DP26) an incorrect choice of the constituent elements (HW components, SW components, peripherals and tools) may result in a realization of safety critical functionalities not covered by ISO 26262. To ensure safe operation, the inputs and outputs of the Data Fusion architecture (as well as computational steps) must fulfill a certain safety level of integrity. Therefore, the selected safety related elements must ensure this. In this regard, AP06 enables the designer to compare and select appropriate HW components, SW components, peripherals and tools for realizing the "Data Fusion" approach. Thereby the designer/architect can use the information in the SaCoMa database to exploit possible implementation alternatives and the associated pros, cons and trade-offs. The CTEs that have been identified for the DP26 could be a starting point to extend the database of SaCoMa with additional safety elements (and the respective safety content).

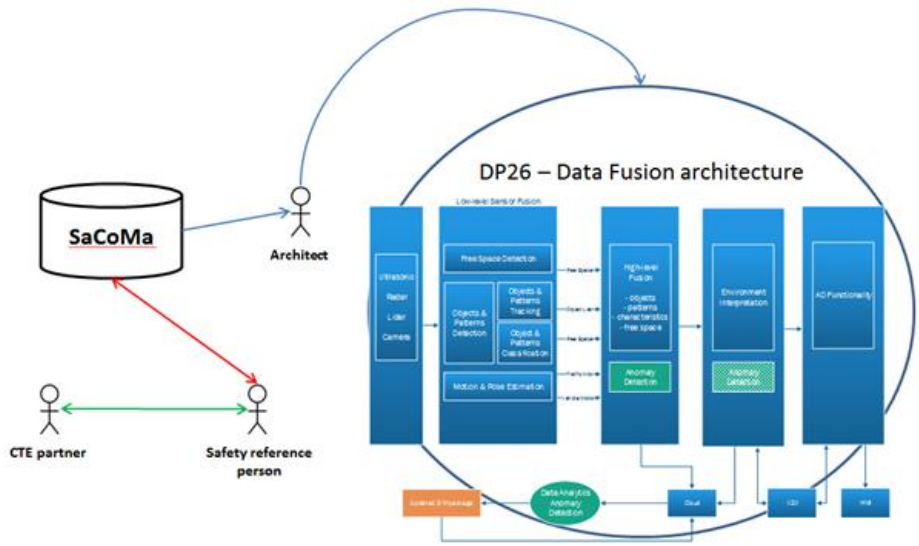


Figure 10: SaCoMa applied to DP26

Developed Solution

The proposed Analysis Pattern guides the identification and choice of proper elements (HW components, SW components, peripherals and tools) with sufficient safety classification, that can be used to realize the Data Fusion unit and their related safety goals by fulfilling the required safety integrity of each functionality.

Evaluation

By realizing the Data Fusion unit with the elements (HW components, SW components, peripherals and tools) selected following the proposed Analysis Pattern, the safety integrity levels of the functions implemented by such elements are met and there are not unreasonable failures leading to risks for human health.

Related Work

The presented SaCoMa method is based on ISO 26262:2018.

Conclusion

AP06 introduces a systematical approach to collect safety relevant information for HW components, SW components, peripherals and tools. This information allows an informed choice of the components and additional provide a documentation and argument for the safety case (i.e. all selected components fulfill the required safety integrity level).

References

ISO 26262. (2018). Road vehicles – Functional safety. (2nd edition).

2.2.6 AP07: Combined Safety and Security Analysis

2.2.6.1 Brief Overview

Analysis Aim:	The goal is to perform a combined safety and security analysis at concept and system level. The aim of this co-analysis is to address safety and cyber-physical security problems through a unified and integrated method.
Abstract:	Safety & security co-analysis refers to methods and techniques that can be used to identify safety hazards and security threats. The Functional Safety standard for road vehicles (ISO 26262) introduces the Hazard Analysis and Risk Assessment (HARA) during the concept phase. The standard proposes to support the functional safety concept by safety analysis (e.g. FMEA, FTA, HAZOP) in order to develop a complete set of effective functional safety requirements. Due to the lack of a safety analysis based on systems theory, a method called System-Theoretic Process Analysis (STPA) is proposed. To address security, a Threat Assessment and Remediation Analysis (TARA) can be used. TARA is an engineering methodology used to identify and assess cyber vulnerabilities and select effective countermeasures to mitigate those vulnerabilities. As security can have immense impact on safety, it is necessary to have a method that covers both, safety and security. STPA sec presents an approach to integrate safety and security analyses. Co-analysis includes activities in the early stage of the development lifecycle, e.g. in the requirements engineering as well as the design phase.

Addressed Problem:	Security for safety The rise of systems which are very dependent on SW has generated benefits in our society but, at the same time, it has increased the services provided by these systems and their infrastructures. The complexity of such systems can increase potential safety and security vulnerabilities. To address such problems, an integrated safety and security analysis approach is presented to encompass complex systems and to prevent losses of cyber-physical systems.
Stakeholder:	Developers, OEM´s, Safety and cybersecurity engineers.
Categories and Scope:	
Scope:	Methodology-AP
Safety:	Safety analyses: HARA for concept level & STPA for concept & system level Security analyses: TARA for concept level & STPA-Sec for concept & system level
Security:	
Privacy:	
Input:	
Description:	We analyse the system at functional and system level with the combined safety and security analysis method to analyse possible threats and malfunctions and develop appropriate safety/security measures. Input: Functions, item definition
Example(s):	
Addressed Elements:	
Related DP:	Related Design pattern: DP20 – Secure remote communication, but in general applicable to all design patterns
Contributors:	
Responsible:	• Joaquim Castella Triginer (Joaquim.CastellaTriginer@v2c2.at)
List of Contributors:	• Nadja Marko (Nadja.Marko@v2c2.at) • Bernhard Winkler (Bernhard.Winkler@v2c2.at)

2.2.6.2 Detailed description

Introduction

Critical systems such as automated driving systems, are highly interconnected. In addition to typical safety problems, security threats from malicious agents must be considered. Traditional techniques predate modern levels of interconnectivity, and thus they do not explicitly address security concerns. Furthermore, fully autonomous driving will need to master connectivity to deliver the V2X (vehicle-to-everything) communication between vehicles and smart devices needed to configure an intelligent transport system in the areas of safety, mobility, and environment (Alnasser, Sun, & Jiang, 2019). To achieve such a demanding level of safety and cybersecurity, a set of analysis methods is necessary to guarantee fail-operational E/E and secure cyber-physical system architectures.

The reliability theory has developed different methodologies in the E/E engineering area, such as FTA (Fault Tree Analysis), FMEA (Failure Mode and Effect Analysis), HAZOP (Hazard and Operability study), reliability block diagrams and Markov modelling, among others. Their representative techniques range

from upper level failure analysis, such as FTA, to lower level failure analysis, such as FMEA (Ericsson, 2005). The reliability of complex systems is not easy to quantify, multiple factors influence on such systems, such as their interactions with other systems, human factors, operating modes and environmental factors. Therefore, a different perspective of analysis is needed to complement reliability methods.

Systems theory introduces an additional perspective of analysis in which safety must be treated as a control problem where the goal is to enforce the safety constraints. Thus, accidents are not just a consequence of a failure; they come from a dynamic process rather than a sequence of events. For systems theory, accidents result from inadequate control or enforcement of safety-related constraints on the development, design, and operation of the system. The traditional causality models define accidents as a chain of failure events, where each failure directly causes the next one in the chain, while with this systems theory, an accident is defined as an unplanned and undesired loss event.

STPA (Systems-Theoretic Process Analysis) is a hazard analysis technique based on the accident causation model called STAMP (Systems-Theoretic Accident Model and Processes). STAMP is an accident model based on systems theory where in the conception of safety, accidents occur when external disturbances, component failures or non-functional interactions between system components are not adequately handled by the control system (Leveson N. G., 2011). One of the purposes of STPA as a systematic approach is to understand that modern systems involve complex interactions between many components (software, hardware, human operators, environment, management, maintenance, etc.). The main goal is to understand why failures were not detected and mitigated or what are the human-computer interaction issues, among others, rather than just identify important failures.

The automotive industry works with functional safety based on the ISO 26262 standard to guarantee the absence of unreasonable risk due to hazards caused by malfunctioning behavior of E/E systems (ISO 26262, 2018). Published in 2011, and updated in 2018, the standard provides a framework that enables the identification of potential risks of software and hardware failure and to ensure functional safety of automotive E/E systems. Furthermore, the automotive industry is aware of new E/E systems and their high level of interconnectivity. There are intentional and malicious acts that must be avoided since they can reduce availability, degrade the system or allow unauthorized access. To provide a cybersecurity process framework and guidance to help organizations to identify and assess cybersecurity threats, SAE introduced J3061 (new ISO/SAE 21434 available) for cybersecurity engineering in cyber-physical vehicle systems (SAE, Cybersecurity Guidebook for Cyber-Physical Vehicle Systems (J3061 Ground Vehicle Standard), 2016).

Safety and cybersecurity engineering systems are interconnected and share a set of common elements. The SAE J3061 defines analogies and unique aspects between system safety and cybersecurity. For

example, objectives and process elements during the concept phase and product development of ISO 26262 and SAE J3061 define analogous processes such as HARA and TARA, safety and cybersecurity goals and safety and cybersecurity requirements. However, some unique aspects differ from each other as safety focuses on analyzing the system for potential hazards and cybersecurity considers potential threats posed by an attacker.

The automotive industry requires a combined analysis to optimize resources. Therefore, common properties have to be found to integrate functional safety and cybersecurity analysis. Thus, the application of systems theory to security contributes with an integrated approach like STPA-sec (STPA for Security) that addresses the cyber-physical security, reframing system design as a general loss prevention problem. According to STPA-sec, safety prevents losses due to unintentional actions by benevolent actors, while security prevents losses due to intentional actions by malevolent actors. Hence, STPA-sec focuses on preventing losses regardless of actor intent using a top-down approach with a common goal to complete a mission. The overall role of the entire socio-technical system is considered, by taking a common top-down system engineering approach, optimizing resources and the potential for resolving conflicts between safety and security early in the development process (Young & Leveson, 2014-02-01).

Description of the AP7: Combined Safety and Security Analysis

The presented approach combines systems theory methods STPA and STPA-sec with the HARA and TARA reliability theory methods (see Figure 11). From the early beginning, the method assumes the STPA strategic point of view, using a common upper level evaluation system engineering approach to safety and cybersecurity that provides the overall role of the entire socio-technical system. Adopting this philosophy, the goal is to ensure the functionality of critical functions and services, identifying and controlling system vulnerabilities rather than defending from possible threats and environmental disruptions. However, the automotive industry has been using ISO 26262 for years being a well-structured and well-known standard that provides a clear and defined framework for all the necessary steps to be taken during the complete lifecycle of the system. Thus, this approach uses ISO 26262 as a reference structure throughout the concept phase and product development at the system level.

Item definition

The method starts with the item definition of ISO 26262 that defines the item and describes its dependencies and interactions with the environment and other items. Actually, ISO 26262 lists requirements and recommendations for establishing the definition of the item with the purpose of providing sufficient information to the persons who conduct the initiation of safety lifecycle, the hazard

analysis and risk assessment and the functional safety concept. The item definition includes a description of the functionality, the operational modes and different states, the operational and environmental constraints, the legal requirements, the assumptions and the potential consequences of behavior shortfalls. Furthermore, it contains the definition for its boundaries, interfaces and assumptions regarding its interactions with other systems.

The STPA method does not comprise a specific definition for the item. STPA centers the attention on the definition of a system in two parts. First, defining the boundaries of the system and its interactions between the system and the environment, and second, identifying and enforcing safety constraints in design and operations of the system.

SAE J3061 cybersecurity standard defines a feature definition and it is described as “it identifies the physical boundaries, cybersecurity perimeter, and trust boundaries of the feature, including the network perimeter of the feature”. Cybersecurity differs from safety in the definition of “boundaries”. For cybersecurity, it is called trust boundaries and is the limit where program execution or data protection change their levels of ‘‘trust’". STPA addresses safety together with cybersecurity (known as STPA-sec) and does not contain an explicit modification from the original analysis on the definition level.

The combination of ISO 26262 and STPA provides a more comprehensive description of the item. ISO 26262 proposes clear steps to define the item and the interfaces that interact with the item, whereas STPA introduces an entire social-technical system´s framework that provides a full definition of the environment. In addition, SAE J3061 define the system and a feature definition and enrich the item definition to address cybersecurity.

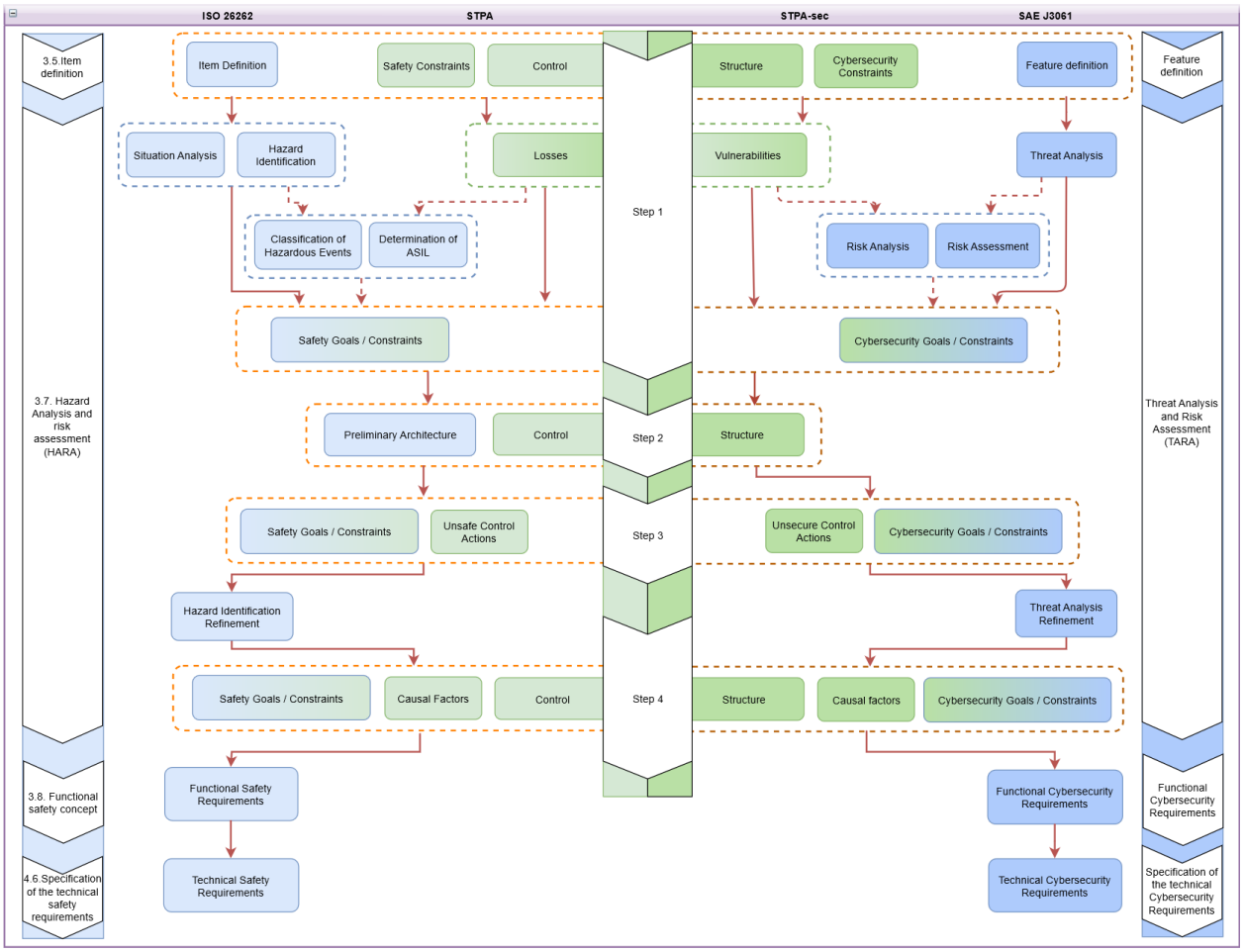


Figure 11 Combined safety and cybersecurity analysis process

Hazards and threats analysis

Based on the item definition, the situation analysis and the hazard identification are the next steps for the safety analysis. The situation analysis generates a list of operational situations and operating modes, which are the basis to determinate hazardous events. The operational situation addresses the limits within which the item is expected to behave in a safe manner. The operating modes describe the different working modes of the item in all range of circumstances. After the situation analysis, the hazard analysis identifies potential sources of harm caused by malfunction behavior of the item and its consequences. The ISO 26262 proposes that hazards shall be determined systematically by using adequate techniques such as brainstorming, checklist, quality history, FMEA and field studies, at the vehicle level.

STPA starts determining hazards at the accident level, analyzing undesired or unplanned events that result on a loss, including loss of human life or human injury, property damage, environmental pollution or mission loss, among others. During the composition of the list of accidents, the identified losses should be ordered by priorities or assigned levels of severity. Once the list of accidents is defined, the system hazards

can be derived. Hazard is defined in STPA as a system state or set of events or conditions that, together with a particular set of worst-case environmental conditions, will lead to an accident. The STPA method recommends finding a core list of the high-level system hazards and afterwards refining them one by one. On the cybersecurity side, SAE J3061 defines the threat analysis as identification of the potential threats to a system or organization. The standard provides a sampling of the most representative security analysis techniques and methods such as the E-Safety Vehicle Intrusion Protected Applications (EVITA) program, the Threat, Vulnerabilities, and implementation Risks Analysis (TVRA) method, the Operationally Critical Threat, Asset, and Vulnerability Evaluation (OCTAVE) method, and the HEAling Vulnerabilities to ENhance Software Security and Safety (HEAVENS) method and attack tree information. Although each of the security analysis techniques uses its particular methods, the threats identification generally contains two main parts, the generation of worst-case scenarios and the identification of generic threats.

To address cybersecurity, STPA reformulates the concept of accident for loss, reframing the problem as a general loss prevention problem. For STPA, safety prevents losses due to unintentional actions by benevolent actors, while security prevents losses due to intentional actions by malevolent actors. In the end, the main difference resides in the intentionality of the actor that produce the loss event. Therefore, STPA defines losses as loss of human life or human injury, property damage, environmental pollution, loss of mission, loss of reputation, loss or leak of sensitive information, or any other loss that is unacceptable to the stakeholders. The next step is to determine the vulnerabilities that can lead to losses following the same approach as safety.

Risk assessment

ISO 26262 presents a technique to obtain ASIL determination, based on the estimation values for each hazard in terms of severity of potential harm, the probability of exposure of each operational situation and controllability of each hazardous event. SAE J3061 introduces a sampling of different techniques eligible according to the needs of each organization and does not provide its own technique such as the safety standard. STPA classifies and prioritizes losses from the early safety and cybersecurity analysis but does not provide a standard risk assessment. Therefore, the presented co-analysis proposes to follow the risk assessment established in the standards and postpone their integration for further future steps.

Safety and cybersecurity goals/constraints

The definition of safety and cybersecurity goals is the next step. Safety and cybersecurity goals have to be defined based on the identified hazards and threats with the possibility that multiple potential hazards or threats may have the same safety or cybersecurity goal. STPA uses the terms safety and cybersecurity constraints, but basically it follows the same process as the standards. Safety and cybersecurity goals or

constraints are top-level requirements without the need to specify a specific technological solution or implementation.

Preliminary architecture and control structure

The preliminary architecture (ISO 26262) and the modelling of the control structure (STPA) are different. The objective is to integrate both designs. The preliminary architecture is part of the functional safety concept, which allocates functional safety requirements to the architecture. The standard defines some properties that functional safety concept must address such as fault detection and failure mitigation, safe state transitions, fault tolerance mechanisms, fault detection and driver warning and arbitration logic to select in all the circumstances the most appropriate control request.

The control structure is a hierarchical system model as a set of feedback control loops that captures functional relationships and interactions of the system components. It can be created at various levels of abstraction and decomposed into multiples sub-models. STPA presents a generic control structure based on a loop that contains controllers, control actions, feedback, other inputs to and outputs from components, controlled processes, actuators and sensors (G.LEVESON & P.THOMAS, 2018, p. 25). The proposed combination of the system theory and reliability theory methods requires to use a common design to integrate the control structure into the preliminary architecture. Since ISO 26262 does not provide specific rules or structure to design the preliminary architecture, it is considered beneficial to integrate a structured and proved model. Furthermore, both methods have to allocate the safety and cybersecurity goals/constraints to the item's architecture.

Unsafe and unsecure control actions and causal factor analysis

The safety and security requirements are derived and allocated to the preliminary architecture/control structure. The result is a description of the high-level strategy for obtaining safety and cybersecurity of the item. However, STPA includes further activities before refining safety and cybersecurity goals to identify potential inadequate control actions. STPA introduces two more steps: (1) the identification of potential inadequate control actions and (2) the identification of causal scenarios for each inadequate control action. Step (1) leads to a hazard or threat. Step (2) identifies scenarios that could lead to unsafe and unsecure control actions in order to examine them one by one in the control loop and to understand which are the potential causes. As a result, STPA contributes to the robustness of the architecture and to the refinement of the safety and cybersecurity constraints.

Functional safety and cybersecurity requirements

Functional safety and cybersecurity requirements is derived from the safety and cybersecurity goals, and takes into account the preliminary architecture and control structure. ISO 26262 specifies that "at least one functional safety requirement shall be specified for each safety goal" and should consider operation modes, fault-tolerant time interval, safe states, emergency operation interval and functional redundancies. ISO 26262 includes warning and degradation concepts, especially for transitions to and from a safe state. Finally, it proposes two lasts steps for validation and verification of the functional safety concept (for more details see (ISO 26262, 2018) [part 3, p. 14-16]). SAE J3061 does not provide any other further specification regarding cybersecurity. STPA does not provide any specific step for this part.

Technical safety and cybersecurity requirements

The technical safety and cybersecurity requirements shall be specified in accordance with the functional safety and cybersecurity requirements together with its preliminary architectural and control structure assumptions of the item and its respective properties. The standards put special effort on the safety and security mechanisms, the proper ASIL decomposition and the methods to avoid latent faults.

Analysis of DP20: Secure remote communication

The safety and cybersecurity analysis pattern provides a framework for analyzing several design patterns. This analysis uses design pattern DP20: Secure remote communication. The selected CTEs of this design pattern are validated and verified by the analysis pattern to guarantee its correct application. This section presents the different steps necessary to apply the safety and cybersecurity co-analysis framework for the secure remote communication design pattern.

The design pattern is considered as the item definition. From the design pattern, the description of the secure communication, its dependencies on and interaction with the environment and other items provides the basis so that the activities in subsequent phases can be performed.

Identify losses, vulnerabilities and safety and security constraints

The presented losses (see Figure 12) are harmonized results for safety and cybersecurity, ranked and prioritized by severity. The vulnerabilities extracted from the losses represent generic vulnerabilities applicable to both safety and cybersecurity, however, cases V-4 and V-5 address only cybersecurity (see Figure 12).

Losses

ID	Title
L-1	Human serious injury or loss of life
L-2	Loss of or damage to vehicle
L-3	Loss of or damage to objects outside/inside the vehicle
L-4	Loss of customer satisfaction or company reputation
L-5	Loss of restricted information

Vulnerabilities

ID	Title	Links
V-1	Remote communication is not sending or receiving signals	L-1, L-2, L-3, L-4
V-2	Remote communication is sending or receiving signals incompletely	L-1, L-2, L-3, L-4
V-3	Remote communication is sending or receiving corrupted signals	L-1, L-2, L-3, L-4
V-4	Remote communication is sending or receiving signals from unauthorized operators	L-1, L-2, L-3, L-4, L-5
V-5	A third agent takes information from the remote communication without authorization	L-4, L-5

Figure 12 Losses and Vulnerabilities

The integration of safety and cybersecurity analysis provides an integrated list of safety and cybersecurity constraints as shown in Figure 13. The integration of the analysis methods does not implicate the integration of all safety and cybersecurity constraints but also includes independent safety or security constraints.

Safety and Security Constraints

ID	Title
SC0.1	Remote communication must prevent the lack of availability of the signal
SC0.2	Remote communication must prevent the lack of the completeness of the signal.
SC0.3	Remote communication must prevent the lack of the integrity of the signal
SC0.4	Remote communication must verify that the signal is authorized.
SC0.5	Remote communication must be between the vehicle and the operator without a third unauthorized agent

Figure 13 Safety and Cybersecurity Goals/Constraints

Preliminary architecture and control structure

Figure 14 shows the control structure as a preliminary architecture. The control structure provides a control loop model where allocates the control actions of remote communication. At the same represents the preliminary architecture used to allocate the functional safety and security requirements.

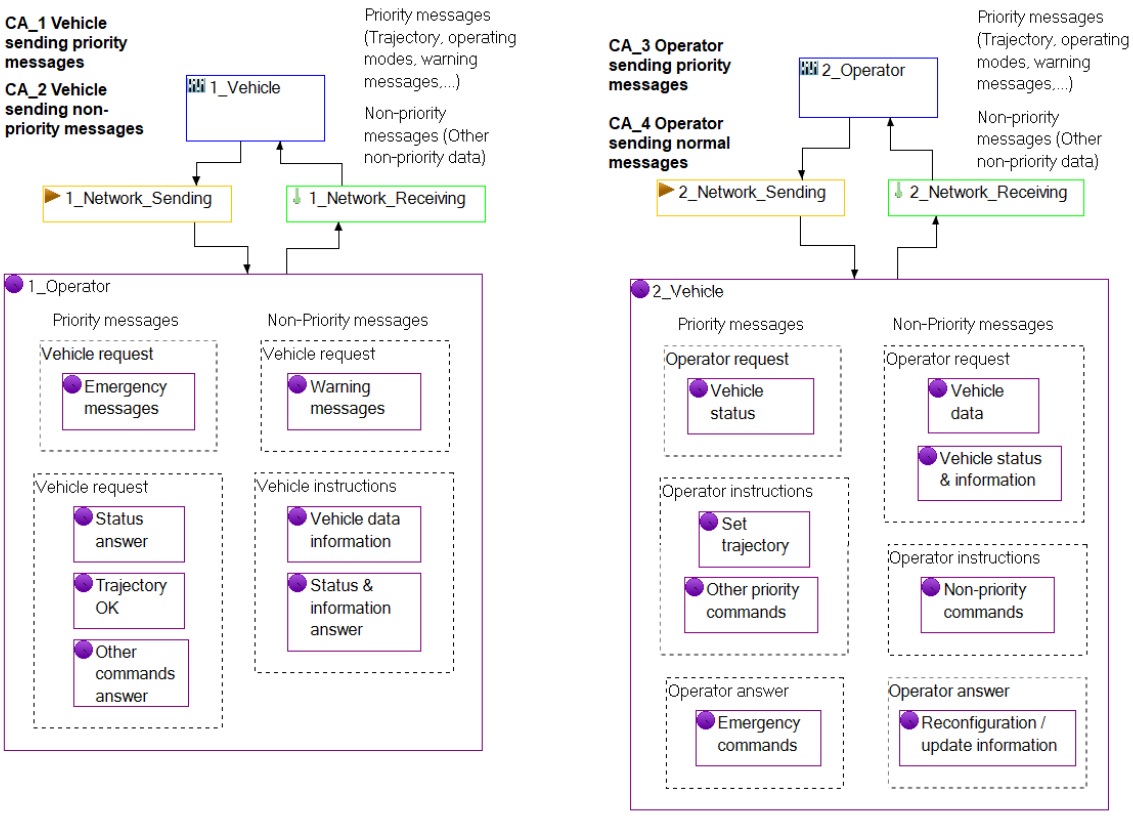


Figure 14 Preliminary Architecture and Control Structure

Unsafe and unsecure control actions

There are two basic control actions: “Send priority messages” and “send non-priority messages”. Both cover the vehicle´s remote communication functionalities (See Figure 14). The next STPA step identifies the potential of inadequate control actions of the system. Figure 15 shows an extraction of the results which build the basis for new and refined safety and cybersecurity constraints. It can be observed that unsafe and unsecure control actions results obtained from the ”Vehicle sending priority messages” control action, are the same. While for the "vehicle sending non-priority messages" control action, only unsecure-related control actions are found.

Control Action	Not providing causes vulnerability	Providing causes vulnerability	Wrong timing or order causes vulnerability	Stopped too soon or Applied too long
CA_1 Vehicle sending priority messages				
Unsafe Control Actions Table Unsecure Control Actions Table	UCA_1 Vehicle is not riority messages mergency occurs. (V-1) and (V-5)	UCA_3 Vehicle is sending incorrect priority messages when an emergency occurs. (V-2), (V-3) and (V-4)	UCA_5 Vehicle is sending priority messages too late when an emergency occurs. (V-1) and (V-2)	UCA_7 Vehicle is sending priority messages too long when an emergency occurs. (V-1) and (V-4)
	Not Vulnerable	Not Vulnerable	Not Vulnerable	Not Vulnerable
	UCA_2 Vehicle is not sending priority messages when an answer is required. (V-1) and (V-5)	UCA_4 Vehicle is sending incorrect priority messages when an answer is required. (V-2), (V-3) and (V-4)	UCA_6 Vehicle is sending priority messages too late when an answer is required. (V-1) and (V-2)	UCA_8 Vehicle is sending priority messages too long when an answer is required. (V-1) and (V-4)
	Not Vulnerable	Not Vulnerable	Not Vulnerable	Not Vulnerable
	Add not given UCA	Add given incorrectly UCA	Add wrong timing UCA	Add stopped too soon UCA
CA_2 Vehicle sending non-priority messages				
Unsecure Control Actions Table	UCA_9 Vehicle is not on-priority messages when a request is required (V-1) and (V-5)	UCA_11 Vehicle is sending incorrect non-priority messages when a request is required. (V-2), (V-3) and (V-4)	Add wrong timing UCA	Add stopped too soon UCA
	Not Vulnerable	Not Vulnerable		
	UCA_10 Vehicle is not sending non-priority messages when an instruction is required. (V-1) and (V-5)	UCA_12 Vehicle is sending incorrect non-priority messages when an instruction is required. (V-2), (V-3) and (V-4)		

Figure 15 Unsafe and unsecure control actions

Causal factor analysis and safety and cybersecurity requirements

The identification of causal scenarios is the last refinement. Causal scenarios provide a set of scenarios which can lead to unsafe and unsecure control actions and hence to hazards or threats. Two types of loss scenarios must be considered. The first type must consider reasons why unsafe and unsecure control actions occur and the second type must consider scenarios why control actions could be improperly executed. This step is the most detailed and time consuming in STPA and the results can be comprehensive. As a final result, ”Vehicle sending priority messages” unsafe and unsecure control actions generate 22 safety constraints and more than 32 cybersecurity constraints for the priority messages of the remote communication. Table 5 and Table 6 show the transformed safety and cybersecurity constraints into requirements.

Table 5: Remote communication network requirements (e.g. vehicle V2X infrastructure)

Safety Requirement **(remote communication network)**	Security Requirement **(remote communication network)**
The network must contain enough redundancy to guarantee the operation of the network (at least more than one routing path).	The network must contain packet filtering to prevent spoofing (ARP spoofing, IP spoofing, DNS spoofing…)
The network must have a priority management to guarantee that priority commands are sent in case of network saturation.	The network must use cryptographic network protocols (e.g. Transport Layer Security (TLS), Secure Shell (SSH), HTTP Secure (HTTPS))
The network must monitor the signal strength and warn if signal is weak.	The network must use tamper-resistant protocols across communication links.

The network must use the same channel and frequency for all the system. Furthermore, must use the same protocol and technology.	The network must use protocols that provide message integrity and variable integrity checking.
The network must be exclusive for V2X communication.	The network must have vulnerability scanner
The network must ensure that the content of the message is delivered and not modified.	The network must have secure audit trails (audit logs)
The time for routing the operator priority messages answer must be below 10ms (time response 10ms). Therefore, the communication should be a time-triggered network protocol or at least a protocol that guarantees the time restriction.	The network must use resource and bandwidth throttling techniques.
The network must control that response message is only received by the sender.	The network must validate the messages of the secure communication and filter the invalid ones.
	If the network shares resources, it must be subject to role-based access-control and separation mechanisms.
	The network should have Denial of service protection systems to be scalable in depth, breadth and height.
	The network should automatically detect, mitigate, report and learn from DDoS attacks. This is especially true in the event of a multi-vector attack (using multiple techniques and methodologies simultaneously).
	The network must have secure communication links layer (OSI layer) with protocols that provide message confidentiality.

Table 6: Remote communication system module requirements (e.g. vehicle V2X module)

Safety Requirement (remote communication module)	**Security Requirement** (remote communication module)
The communication module must contain error management methods to detect faults at sending priority messages.	The communication module must use strong authentication (e.g. Stick to two-factor authentication)
The communication module must control the signal strength and warn if signal is weak.	The communication module should ensure the integrity of data (e.g. with hash functions).
The communication module must control the operational limit range of the signal	The communication module must use public-private encryption scheme.
The communication module must contain a redundant power supply system to guarantee communication basic functionalities.	The communication module must have hardware and software integrity checks
The remote communication module must contain methods to validate priority messages and preserve the content without modification.	The communication module must use highly resistant encryption.
The communication module must manage the queue of messages to send and receive priority messages always before non-priority messages.	The communication module must follow the principle of least privilege and use least privileged service accounts to run processes and access resources.
The communication module must have sufficient computing power and bandwidth to guarantee time restrictions for priority messages.	The communication module should have Denial of service protection systems to be scalable in depth, breadth and height.
The communication module must send priority messages only once.	The communication module should automatically detect, mitigate, report and learn from DDoS attacks. This is especially true in the event of a multi-vector attack (using multiple techniques and methodologies simultaneously).
The communication module must send a priority message in less than 10ms.	

Safety Requirement (remote communication module)	**Security Requirement** (remote communication module)
The vehicle must have a no remote communication operational mode in case the communication module is not available.	
The communication module must contain controls to detect the faults on its components.	
The communication module must manage priority messages to send and receive them without delays.	
Priority messages must be forwarded just in case of receiving a notification error.	
	The communication module must define a permissions policy to specify the privileges of the users
	The communication module must not store secrets (e.g. passwords) in plaintext.
	The communication module must contain packet filtering to prevent spoofing (ARP spoofing, IP spoofing, DNS spoofing…)
	The communication module must use cryptographic network protocols (Transport Layer Security (TLS), Secure Shell (SSH), HTTP Secure (HTTPS) and other secure communications protocols)
	The communication module must have an application for vulnerability scanning.
	The communication module must provide message confidentiality.
	The communication module must validate the messages of the secure communication and filter the invalid ones.
	The communication module does not pass credentials in plaintext over the network

Results of the analysis of DP20: Secure remote communication

The combined safety and security analysis applied to the secure remote communication provides argumentation for verifying the safety and security measures proposed in the design pattern. The safety and security co-analysis is based on the automotive engineering standards (ISO 26262, SAE J3061) and has a well-known proven analysis method (STPA) supported by the STAMP safety model, that presents a substantial improvement of the overall safety and security engineering process. Therefore, all the steps implemented in the analysis patterns provide an argument, conformity checks for the tasks required to perform a safety and security analysis, while at the same time an evident proof of the compliance list of required safety and security activities is done.

The analysis pattern applies the safety and security analysis on the concept phase. The results of such activities are a list of hazards and threats, safety and security goals, preliminary architecture design and safety and security requirements. The safety and security requirements are the main output of the concept phase and represent a basis for product development phases. For the secure remote communication case study, such requirements are presented in Table 5 and Table 6 (in the previous section).

Secure remote communication presents its specific technological solution along with a list of common technology elements. Such common technology elements can be compared with the results of the analysis pattern and be used to provide safety and security technological improvements. The enhancement of the design pattern through the analysis pattern derives from the safety and security requirements. As the design pattern focuses on provisioning security measures, the security requirements will be the requirements applied to the security remote communication design pattern. The first step is to compare the actual common technology elements that are compatible with security requirements (see Table 7 and Table 8).

Table 7: Comparison between requirements and CTEs (remote communication network)

Security Requirement (remote communication network)	**Common technology elements**
The network must use cryptographic network protocols (e.g. Transport Layer Security (TLS), Secure Shell (SSH), HTTP Secure (HTTPS))	CTE-16 Transport layer security: The communication between vehicle and the "other system" is encrypted and hence secured.
The network must use protocols that provide message integrity and variable integrity checking.	
The network must have secure communication links layer (OSI layer) with protocols that provide message confidentiality.	
The network must use tamper-resistant protocols across communication links.	

Table 8: Comparison between requirements and CTEs (remote communication module)

Security Requirement (remote communication module)	**Common technology elements**
The communication module must use highly resistant encryption.	CTE-02 Cryptography: after authentication a secrete key is generated for the session, the whole communication is encrypted.
The communication module must use public-private encryption scheme.	
The communication module must not store secrets (e.g. passwords) in plaintext.	
The communication module must provide message confidentiality.	
The communication module does not pass credentials in plaintext over the network	
The communication module must use strong authentication (e.g. Stick to two-factor authentication)	CTE-08 Authentication & Authorization: vehicle authenticates with a certificate, remote controller authenticates with "public key authentication" and has a private key that is secured with a password, the public key is on the vehicle. An alternative could be a password authentication.
The communication module must follow the principle of least privilege and use least privileged service accounts to run processes and access resources.	
The communication module must define a permissions policy to specify the privileges of the users	
The communication module should ensure the integrity of data (e.g. with hash functions).	CTE-16 Transport layer security: The communication between vehicle and the "other system" is encrypted and hence secured.

The communication module must have hardware and software integrity checks	
The communication module must use cryptographic network protocols (Transport Layer Security (TLS), Secure Shell (SSH), HTTP Secure (HTTPS) and other secure communications protocols)	

Design pattern improvement

The security requirements derived during the applied co-analysis provide an enhancement of the design pattern by the analysis pattern. Table 7 and Table 8 show that requirements are partially connected to the CTEs. This observation provides the possibility to iterate again on such connections and introduce new relations between CTEs and requirements. The Table 9 and

Table 10 show the proposed improvement to use new CTEs for the secure remote communication.

Table 9: Improvement of the design pattern (remote communication network)

Security Requirement (remote communication network)	**Common technology elements**
If the network shares resources, it must be subject to role-based access-control and separation mechanisms.	CTE-7 V2X Communication
The network must use resource and bandwidth throttling techniques.	
The network must contain packet filtering to prevent spoofing (ARP spoofing, IP spoofing, DNS spoofing…)	CTE-12 Firewalls
The network must validate the messages of the secure communication and filter the invalid ones.	
The network should have Denial of service protection systems to be scalable in depth, breadth and height.	
The network should automatically detect, mitigate, report and learn from DDoS attacks. This is especially true in the event of a multi-vector attack (using multiple techniques and methodologies simultaneously).	
The network must have vulnerability scanner	Multiple related CTEs
The network must have secure audit trails (audit logs)	

Table 10: Improvement of the design pattern (remote communication module)

Security Requirement (remote communication module)	**Common technology elements**
The communication module must contain packet filtering to prevent spoofing (ARP spoofing, IP spoofing, DNS spoofing…)	CTE-12 Firewalls
The communication module must validate the messages of the secure communication and filter the invalid ones.	
The communication module should have Denial of service protection systems to be scalable in depth, breadth, and height.	

The communication module should automatically detect, mitigate, report and learn from DDoS attacks. This is especially true in the event of a multi-vector attack (using multiple techniques and methodologies simultaneously).	
The communication module must have an application for vulnerability scanning.	Multiple related CTEs

There are three possible improvements of the design pattern:

1. The first improvement is the possibility to connect secure remote communication with V2X communication. The secure remote communication is based on standard WIFI protocol (IEEE 802.11) and is included on the V2X communication protocols. However, it requires special network requirements which is not within the scope of this design pattern. The improvement remarks the ability to include the V2X CTE for further secure remote communication DP improvements.
2. The second improvement is provided by a group of requirements that require the need to introduce firewalls for the use of filters during the communication of messages. Applying CTE firewalls could provide security enhancements to secure remote communication and help protect the vehicle from attacks such as denial of service attacks, multi-level spoofing, etc.
3. Finally, there are a group of requirements that address two important activities for security coverage. The first suggests the need to keep the system secure by introducing regular vulnerability scanning. The second suggests the need to audit security activities by an external authority to validate the secure remote communication. This second improvement may be out of the scope but it is recommended to consider it once the security design pattern is applied on commercial purposes.

Evaluation of Analysis method: Combined Safety and Security Analysis

The combination of system theory and reliability theory methods implies several benefits. The STPA model argues that losses involve complex dynamic processes including the entire sociotechnical system and cannot simply be described as a chain of events. Therefore, STPA focuses on how to control vulnerabilities rather than how to avoid hazards or threats. However, although STPA argues that highly reliable software is not necessarily safe, it is beneficial to include reliability methods. ISO 26262 and SAE J3061 present clearly defined steps with a diversity of methods to provide a complete hazard and threat analysis.

The safety and cybersecurity integration require connection needs and common properties to establish links for both analyses. SAE J3061 provides a complete life cycle process framework analogous to the process framework described in ISO 26262 in order to maintain consistency and integrity between the results of an organization's cybersecurity and security process. In addition, STPA contributes to integrate safety and cybersecurity proposing an abstraction on the intentionality of the actions and reframing it as

a general loss prevention problem. Furthermore, STPA applies a strategic viewpoint by identifying and controlling systems vulnerabilities rather than reacting continually from changing threats and other environmental disruptions.

Finally, we can evaluate the analysis method by checking the advantages of applying the analysis pattern to secure remote communication. The results provide a list of threats and hazards that can lead to losses which then can be used to generate test scenarios. In addition, a list of safety and security requirements is generated that can be transformed as an input for CTE selection. In the end, the results provide requirements that could improve the design pattern and could enhance the CTEs used to improve the security in the remote communication design pattern.

Related Work

The integration of STPA into the ISO 26262 is something that has already been studied in recent years. The thesis "Functional Safety in Co-operative Driving using Systems-Theoretic Process Analysis" presents a development process which utilises the benefits of STPA, while complying to the ISO 26262 standard to analyze safety on co-operative driving (Oscarsson, 2016). One of the relevant aspects of this thesis is that its method focuses on the concept phase and the product development phase at the system level, obtaining a broad and comprehensive approach.

Another thesis called "Using STPA in an ISO 26262 compliant process" introduces the STPA to overcome some of the limitations of the hazard analysis techniques used in ISO 26262 (Mallya, 2015). The project also provides an approach that applies STPA according to the concept phase of ISO 26262 in an automotive subsystem.

The next paper presented "Using STPA in Compliance with ISO 26262 for Developing a Safe Architecture for Fully Automated Vehicles" shows a concept on how to use STPA to extend the safety scope of ISO 26262 and support the hazard analysis and risk assessments process (Abdulkhaleq, Wagner, Lammering, Boehmert, & Blueher, 2017). The report presents the same arguments as this project in relation to the methods used by ISO 26262 that are based on reliability theory and are not adequate to address new hazards caused by dysfunctional component interactions, software failures or human errors.

The integration of safety and security comes from the STPA process. "An Integrated approach to safety and security based on systems Theory" presents a powerful safety methodology that includes security (Young & Leveson, 2014-02-01). STPA-sec (STPA for Security) addresses the cyber-physical security, reframing system design as a general loss prevention problem. According to STPA-sec, safety prevents losses due to unintentional actions by benevolent actors, while security prevents losses due to intentional

actions by malevolent actors. Hence, STPA-sec focuses on preventing losses regardless of actor intent using a top-down approach with a common goal to complete a mission.

However, “Limitation and Improvement of STPA-Sec for Safety and Security Co-analysis” focus on improving existing approach STPA-Sec (Schmittner, Ma, & Pus, 2016). They have identified the guidance for the identification of intentional casual scenarios not being clear enough and proposed some modifications.

Conclusion

The analysis pattern presents an approach in order to integrate safety and cybersecurity and combine systems and reliability theory methods. For this purpose, an existing design pattern is selected to carry out the analysis and provide results to validate the analysis pattern and the design pattern. The quality in the refinement of the secure remote communication requirements suggests the benefit of combining STPA with the safety and security standards (ISO 26262, SAE J3061). Furthermore, the unification of safety and cybersecurity constraints helps to interconnect both properties and to address integrated analysis on the design patterns. Finally, the result provides new security improvements to secure remote communication and propose the use of the CTE firewall.

References

- Abdulkhaleq, A., Wagner, S., Lammering, D., Boehmert, H., & Blueher, P. (2017). Using STPA in Compliance with ISO 26262 for Developing a Safe Architecture for Fully Automated Vehicles. University of Stuttgart; Continental.
- Abuelsamid, S., Alexander, D., & Jerram, L. (2018, 1Q). Navigant Research. Retrieved from https://www.navigantresearch.com/research/navigant-research-leaderboard-automated-driving-vehicles
- Alnasser, A., Sun, H., & Jiang, J. (2019). Cyber Security Challenges and Solutions for V2XCommunications: A Survey. United Kingdom.
- Christian Herberet, et al. (2013). E3.4.3.1 Dokumentation von Konzept und Evaluierung HW Unterstützung Virtualisierung I/O.
- Ericsson, C. A. (2005). Hazard Analysis Techniques for System Safety. John Wiley & Sons.
- G.LEVESON, N., & P.THOMAS, J. (2018). STPA handbook.
- ISO 26262. (2018). Road vehicles – Functional safety. (2nd edition).
- ISO/PAS 21448. (2019). Road vehicles — Safety of the intended functionality. (ISO/PAS).
- Leveson, N. (2005). A new accident model for engineering safer systems. Aeronautics and Astronautics Department, Massachusetts Institute of Technology.
- Leveson, N. G. (2011). Engineering a Safer World, Systems Thinking Applied to Safety.
- Mallya, A. (2015). Using STPA in an ISO 26262 compliant process. Hamilton, Ontario: McMaster University.
- Marko, N. (2019). D3.5 Initial design patterns for common technology elements. SECREDAS.
- Marko, N. (2020). D3.6 - Design pattern descriptions. SECREDAS.
- Oscarsson, J. (2016). Functional Safety in Co-operative Driving using Systems-Theoretic Process Analysis. STOCKHOLM: KTH ROYAL INSTITUTE OF TECHNOLOGY.
- SAE. (2016). Cybersecurity Guidebook for Cyber-Physical Vehicle Systems (J3061 Ground Vehicle Standard). SAE International.
- SAE. (2016). Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles (J3016 Ground Vehicle Standard). SAE International.
- Sanden, B. v., & Vasenev, A. (2019). D2.7: “Reference Architecture 1st Version”. SECREDAS.

- Sassan Nakhdjavani, Jens Harnisch, et al. (2014 satus: draft). E 4.4.6.1 Implementation of HW-Access Interfaces for Hypervisor and Drivers.
- Schmittner, C., Ma, Z., & Pus, P. (2016). Limitation and Improvement of STPA-Secfor Safety and Security Co-analysis. In Computer Safety, Reliability, and Security . Springer International Publishing.
- Schnellbach, A. (2016). Fail-operational automotive systems. Graz: Graz University of Technology.
- Smulders, A. (2019). D2.6 Security Model. SECREDAS.
- Young, W., & Leveson, N. (2014-02-01). An integrated approach to safety and security based on systems theory. Communications of the ACM, 57(2), 31-35.

2.2.7 AP08: Pre-injection analysis

2.2.7.1 Brief Overview

Analysis Aim:	The aim of the pre-injection analysis pattern is to reduce the fault/attack space when using fault and attack injection experiments. Reduction of the fault/attack space minimizes the time and cost of the testing process, contributing to a more effective verification and validation of systems in respect to safety and security requirements. The analysis could be conducted in all the project sectors, i.e., automotive, health and railway; as fault/attack injection is a useful and recommended verification and validation technique in all these domains.
Abstract:	Fault injection has been used in the past to test system safety. It has been (highly) recommended as an assessment technique by different standards, such as, ISO 26262 [1]. According to recent studies, fault injection concepts and methodologies could also be used when it comes to security testing [2]. In this case, the technique is referred to as attack injection. Attack injection could be directly mapped to different cybersecurity verification and validation methodologies, that are discussed in different domains, such as, the automotive domain. In other words, depending on the type of attacks that is modelled and injected into the system under test, common techniques such as fuzzing, vulnerability scanning and penetration testing could be exploited, using attack injection. The pre-injection analysis pattern could be used to evaluate the Input Validation design pattern (DP07). DP07 focuses on handling of inappropriate inputs, in order to mitigate the threats, by filtering the inputs before processing. Fault/attack injection technique could be used to generate these inputs in order to evaluate the mitigation techniques.
Addressed Problem:	Fault/attack injection may be costly, since the fault/attack space can be enormous. In order to be able to effectively evaluate a system, one needs to conduct an exhaustive fault and attack injection targeting all possible faults/attacks of the error space. However, conducting such an experiment could take so much time that it makes the evaluation infeasible. This is why the aim of this analysis pattern is to investigate the possibility of reducing the fault/attack space, e.g., by only focusing on faults and attacks that are effective, i.e., the ones that reveal the weaknesses of the system under test (SUT). These are in general faults that would eventually cause a system failure or result in a compromised system. This way, the non-effective faults/errors (the ones that do not reveal the weaknesses of the SUT such as

	the ones whose effects are masked by different safety and security detection/correction mechanisms) could be removed from the fault/attack space, reducing the time and cost of the fault/attack injections. Another way to conduct a pre-injection analysis is to group certain attacks that affect the system in the same way and only conduct experiments on one of these attacks. As there may be multiple ways of reducing the time and cost of fault/attack injection experiments, we conduct, in this analysis pattern, a literature review of the previous analyses and investigate their usability in DP07 as well as their applicability to our fault and attack injection tool, MODIFI [4], which is a model-implemented fault and attack injection tool capable of injecting faults/attacks into Simulink models.
Stakeholder:	Stakeholders of this analysis pattern are individuals and organizations who are interested in: • Verifying and validating systems with respect to safety and security requirements, • using fault and attack injection as a testing technique, • conducting fault and attack injections at Simulink model level, and • reducing the time and cost of conducting fault and attack injection campaigns.
Categories and Scope:	
Scope:	Methodology-AP
Safety:	Using pre-injection analysis allows us to reduce the fault space size when evaluating system safety and to reduce the attack space size when evaluating system security. Privacy could also be evaluated when conducting attack injection, since specific attacks that violate system privacy could be modelled. Therefore, we can use pre-injection analysis to reduce the attack space size when modelling privacy-related attacks.
Security:	
Privacy:	
Input:	
Description:	The main inputs required in this analysis pattern are fault and attack injection tools and models. This is an important input to this analysis; as the tools under investigation provide initial lists of faults and attacks, which can be further optimized. The fault/attack injection tool that we are developing for this project, targets Simulink models. Thus, in order to be able to evaluate the results, we would need to acquire representative Simulink models. As the systems, that the DP07 is dealing with, are not implemented as Simulink models in this project, the evaluation conducted in the pre-injection analysis pattern is mostly on the methodology side.
Example(s):	
Addressed Elements:	
Related DP:	Fault/attack injection could be used to evaluate any of the design patterns defined in the project. However, in this task, we focus specifically on DP07 as there is a direct connection between this design pattern and both the interface testing and fuzzing, which are commonly used to evaluate system safety and security requirements. DP07 focuses on handling of inappropriate

	inputs in order to mitigate the threats by filtering the inputs before processing. Fault/attack injection techniques could be used to generate the inappropriate inputs which are then used to evaluate the mitigation techniques. In this case, the faults/attacks could be inserted into the inputs and interfaces of the system.
Contributors:	
Responsible:	Behrooz Sangchoolie, RISE (behrooz.sangchoolie@ri.se)
List of Contributors:	Peter Folkesson, RISE (peter.folkesson@ri.se) and Behrooz Sangchoolie, RISE (behrooz.sangchoolie@ri.se).

2.2.7.2 Detailed description

Introduction

Fault injection and attack injection have been studied and used in the past to test system safety and cybersecurity. Performing these experiments is, in general, time consuming due to the big size of the fault and attack space. This is why analysis techniques such as pre-injection analysis are used to reduce the fault/attack space contributing to a more cost-effective verification and validation of systems, with respect to safety and security requirements. The aim of this analysis pattern is to identify different types of pre-injection analysis and apply at least one of them to a fault and attack injector, developed in house at RISE, called MODIFI [4], which is a model-level fault and attack injector.

Pre-injection analysis could allow us to evaluate the effectiveness of different design patterns defined and detailed as part of the work done in WP3, especially the ones that focus on improving the system safety and cybersecurity. An example of these design patterns is Input Validation (DP07), which focuses on handling of inappropriate inputs to mitigate the threats by filtering the inputs before processing. Fault/attack injection technique could be used to generate these inputs to evaluate the mitigation techniques. Moreover, by using pre-injection analysis techniques, inputs with the most impact on the system's safety and security could be identified.

Throughout the rest of this section, we conduct a literature review of current pre-injection analysis techniques to determine the applicability of pre-injection analysist to MODIFI. Moreover, we propose and discuss two pre-injection analysis techniques adapted for the MODIFI, called inject-on-read and inject-on-write.

Description of the pre-injection analysis pattern

Scope and goals of pre-injection analysis

When evaluating systems SSP (safety, security and privacy), the fault and attack space is so big that the effectiveness of the injected faults and attacks becomes very important. These experiments, in general,

come with significant evaluation time and cost (such as computational and resource costs) related issues. The goal is, therefore, to reduce the fault and attack space for different SSP evaluation techniques that are using fault and attack injection. This is done by focusing on faults and attacks that are potentially known to be effective, i.e., the ones that reveals the weaknesses of the System Under Test (SUT).

Description and background

Fault injection has been used in the past to test system safety. According to recent studies, the fault injection concept and methodology could also be used when it comes to security testing [2]. In this case, the technique is referred to as attack injection. Attack injection could be effectively mapped to different cybersecurity verification and validation methodologies, such as fuzzing, vulnerability analysis and penetration testing discussed in different domains such as the automotive domain. In other words, depending on the type of attacks that are modelled an injected into the system under test, common techniques such as fuzzing, vulnerability scanning, and penetration testing could be exploited.

Relevant pre-injection analysis techniques

Inject-on-read

The inject-on-read technique follows the rule that faults and attacks should only target resources, such as CPU registers and memory cells immediately before they are read. This way, faults and attacks that have identical activation are grouped into the same equivalence class, thereby reducing the fault and attack space required for evaluating systems SSP. Some examples are given in [3], [5] and [6]. The inject-on-read technique is compared with inject-on-write technique (described below) in [7].

Inject-on-write

The inject-on-write technique is similar to the inject-on-read technique. However, for inject-on-write the faults and attacks are only injected on a resource immediately after it has been produced. This way, faults and attacks injected into the resource any time after it has been produced, but before it is read the first time, have identical activation and may be grouped into the same equivalence class, thereby reducing the possible fault and attack space. Some examples are given in [6] and [7].

Code-slicing

Code-slicing may be performed at the source code level to determine the statements that a certain criterion, e.g., output signal, depends on. The intersection between these statements and the statements executed at a certain fault/attack injection time are the statements to target for fault/attack injection when the aim is to affect the criterion. The technique is used at the HDL code level [8].

Post-injection analysis

Post-injection analysis aims at predicting dependability measures using the results from previous fault and attack injection experiments. Some examples, where post-injection has been performed to predict the results of fault injection experiments, are given in [9] and [11]. Post-injection may also be used for determining relevant fault injection experiments to be performed. For example, post-injection may be used to determine workload input sequences with the lowest error coverage, which then can be used for fault injection to estimate the real coverage [10].

Fault list collapsing

Faults and attacks that are determined to be equivalent, may be collapsed into the same equivalence class. Apart from the inject-on-read and inject-on-write techniques, described above, other criteria may be used for determining equivalence. In [13], pre-injection analysis is performed by constructing fault trees for each node in a data flow diagram derived from the target system assembler code. Faults injected at leaf nodes ending up at the same node along a path are considered equal and the corresponding subtree may be collapsed. Both static (pre-injection) and dynamic (post-injection) gate-level fault-list collapsing is used in [12], identifying states with only one bit-flip difference from the fault-free state, i.e., equivalent single bit-flip faults.

Error space pruning

In error space pruning, pre-injection analysis is performed to prune faults that have a known outcome or are equivalent to other faults, where the equivalence may be determined using either static (pre-injection) or dynamic (post-injection) analyses [17]. Experimental results show that single bit-flip errors result in a higher percentage of Silent Data Corruptions (SDCs) than multiple-bit flip errors [14], [15]. Thus, the error space may be pruned by only focusing on the injection of single bit-flips. In [18], signals and locations are identified that if targeted would result in a severe case of SDC. This way, injections in locations that would result in benign failures could also be pruned.

The MODIFI tool

The MOdel-Implemented Fault Injection tool (MODIFI) is a model level fault and attack injection tool for dependability assessment of software developed as Simulink models [4], [2]. The tool enables functional testing (e.g. stress testing) as well as non-functional testing (such as, fault and attack injection in order to evaluate and exercise safety and security mechanisms) to be performed in early phases of the software development life cycle within the same environment that is typically used for model-based development.

MODIFI injects faults and attacks by adding separate components, modelling the faults and attacks between the connected blocks of the model, see Figure 16. The tool supports a wide range of fault and attack models such as bit-flip, stuck-at and gain faults, and can easily be extended with new models.

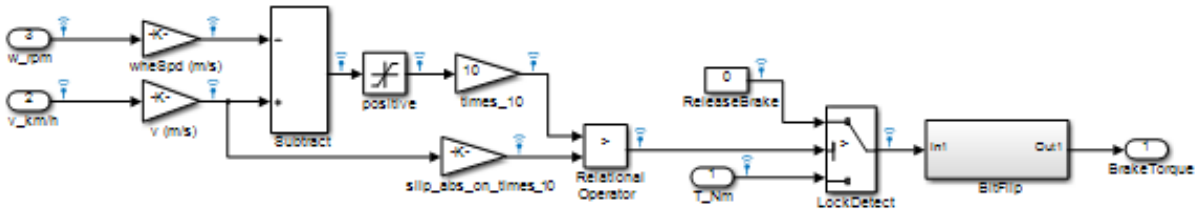


Figure 16. Simulink model with a bit-flip fault model on the output signal.

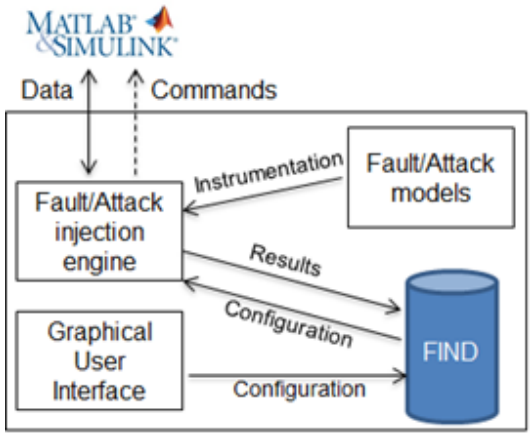


Figure 17. Overview of the MODIFI fault- and attack injection tool.

Figure 17 shows an overview of MODIFI. The GUI (shown in Figure 18) is implemented in Java, and the injection engine is developed in MATLAB m-code. The tool supports visualization of error propagation as well as highlighting of sensitive parts of models within the MATLAB/Simulink environment and is also scriptable to support batch processing of experiments to facilitate inclusion in common software development workflows. FIND, which is an acronym for Fault/attack INjection Database, is a generic fault and attack injection MySQL database for storing configuration data and results of injection experiments. FIND has been developed for storing fault and attack injection data in a generic, tool-independent format to facilitate comparison of experiment results from multiple tools and injection techniques. Generic analysis and visualization tools working on the database have also been developed, some of which are shown in Figure 18.

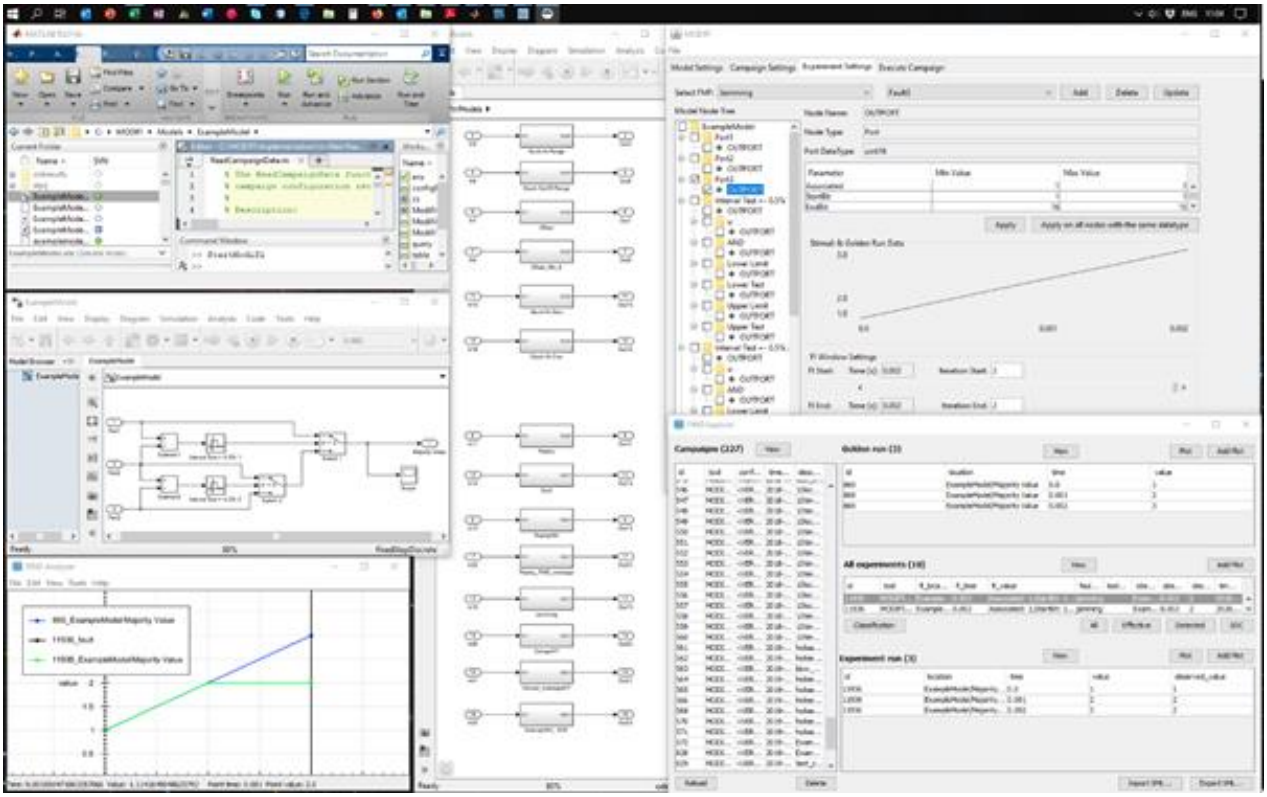


Figure 18. The MODIFI graphical user interface.

Connection between the Pre-injection analysis and the Input Validation design pattern

Input validation (DP07 presented in D3.7) defines the proper handling of the input values and could be either taken into consideration for lower levels of technology or to coding procedures. Input validation occurs before examining further the way that the system under test responds to erroneous inputs. Therefore, it is perceived as the initial process to identify and prohibit inconsistent values or input values to enter the system, especially with the knowledge we have, that is, the system under test cannot handle these inputs at all. The pre-injection analysis pattern constitutes the appliance of the input validation design pattern and extends the pattern further by adding functionalities that facilitate (i) generation of extensive lists of erroneous inputs, (ii) monitoring the behavior/response of the system under test to the generated erroneous inputs, and (iii) exploration of techniques that allow the reduction of the fault and attack space.

The input validation facilitates the evaluation of system with respect to the inputs they receive, especially the ones that potentially cause errors in the system (such as the ones raising exceptions). Some examples of the threats that do not tak into consideration the input validation are buffer overflow, SQL injection, XSS attack, overstimulation of the input values of physical systems and denial of service, among others. Fault and attack injection techniques could be used to further extend the input validation by generating

arrays of erroneous inputs. These inputs could then be used to evaluate the systems response and identifying parts of the system that are sensitive to certain inputs. These parts could then be equipped with mitigation mechanisms, which could also be evaluated with the help of fault and attack injection. As generating an extensive and yet complete list of fault and attack vectors could be quite time consuming, this analysis pattern could facilitate the reduction of the fault and attack space, allowing the evaluation of systems to be done in a relatively shorter time.

Implementation and evaluation of the inject-on-write pre-injection analysis technique

We have implemented the inject-on-write technique for the MODIFI tool. Figure 19 and Figure 20 show examples of the effectiveness of our implementations through reduction of the error space.

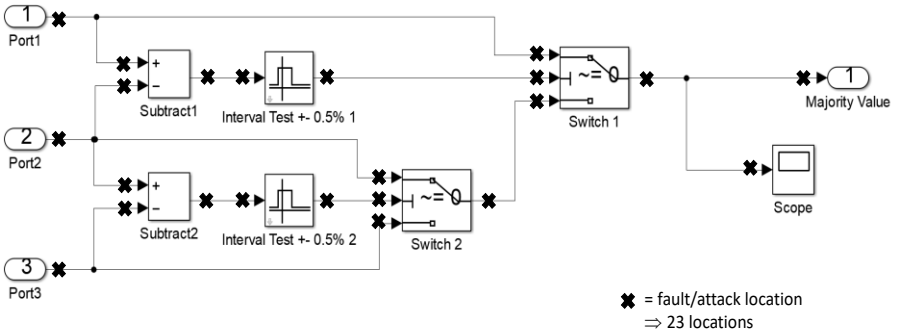


Figure 19. Fault/attack locations without pre-injection analysis.

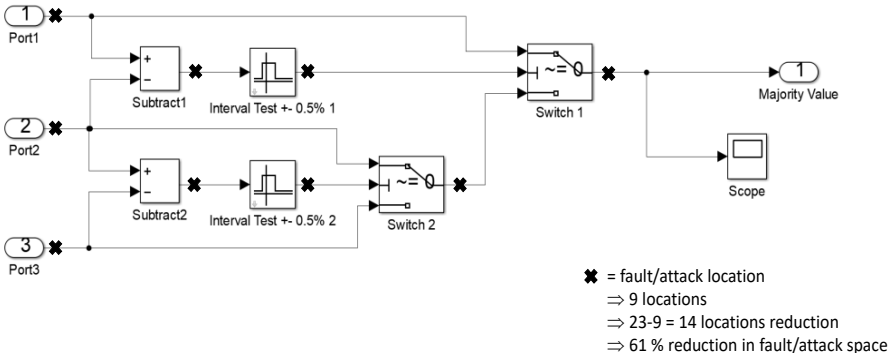


Figure 20. Fault/attack locations with inject-on-write pre-injection analysis.

Figure 19 shows an example of Matlab/Simulink model implementing a simplified TMR (triple modular redundancy) voter where a voter is used (in the form of 2 switches) in order to ignore the result produced by one faulty module. The Simulink signals selectable for fault and attack injection are marked with an **X**.

MODIFI supports the injection of faults and attacks on all signals between Matlab/Simulink model blocks[3] and inside components, but not inside blocks or S-Functions. In the model illustrated in Figure 19, the error space consists of 23 injectable locations (Simulink signals).

Figure 20 shows the effects of applying the inject-on-write technique for the MODIFI tool on the TMR voter model. By using inject-on-write pre-injection analysis, the number of fault/attack locations are reduced to 9, resulting in 61% reduction of the fault/attack space. By selecting proper fault/attack models on the output signals, it would also be possible to simulate several faults and attacks injected *inside* the blocks using this technique.

However, some of the faults/attacks in Figure 19 are not injectable using this approach. For example, faults/attacks on the first input signal to the **Subtract1** block are achieved by injecting the output signal of the **Port1** block in Figure 20, which always affect the first input signal to the **Switch 1** block. Thus, single faults/attacks on the first input signal to the **Subtract1** block are not possible to inject using this technique.

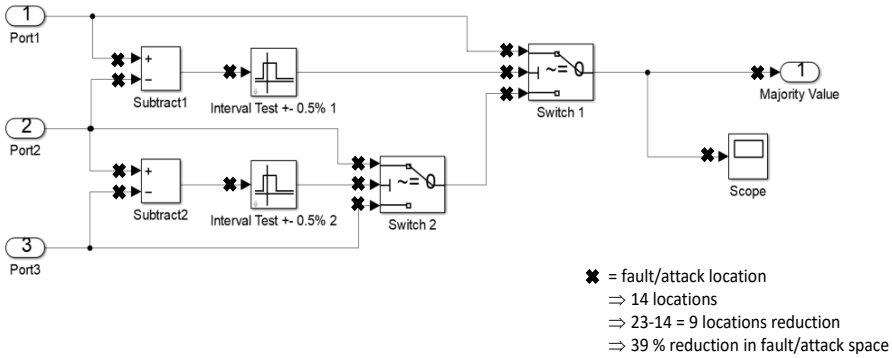


Figure 21. Fault/attack locations with inject-on-read pre-injection analysis.

For comparison, Figure 21 shows the effects of applying the inject-on-read technique for the MODIFI tool on the TMR voter model. Here, the number of fault/attack locations are reduced to 14, resulting in 39% reduction of the fault/attack space. By injecting multiple locations, all faults/attacks in Figure 19 may be injected. For example, injection on the output signal of **Port1** in Figure 19 is equivalent to injection on the first input signal to both the **Subtract1** and **Switch 1** blocks in Figure 21.

One could argue that faults and attacks injected inside blocks may be harder to simulate for inject-on-read than for the inject-on-write technique, since the behavior of the block output signals may not be adequately affected through faults/attacks injected on the block input signals. However, this is not the case for the TMR voter model presented above, since all the output signals are accessible through the input signals in this model.

[3] Blocks are the main elements used to build models in Simulink. In general, Blocks represent equations and modeling components.

Overview of pre-injection analysis techniques

In an earlier section, we have already indicated the pre-injection analysis techniques that are of interest for us and our fault and attack injection setup. These techniques have been picked by reviewing related literature (from the past 30 years). In this section, we provide the reader with an overview (see Table 11) of the pre-injection analysis techniques that have been defined and detailed in the past. The aim of this presentation is to allow the reader to identify the ones that are useful in their setup. In fact, for each identified related work, we provide: (i) a short summary of the work, (ii) the type of pre-injection analysis proposed as well as (iii) the abstraction level where the pre-injection analysis is applied to.

Paper title	Short abstract	Type of pre-injection analysis (error space pruning) investigated	Abstraction level
Assembly-Level Pre-Injection Analysis for Improving Fault Injection Efficiency [3]	A fully automated pre-injection analysis technique is presented that uses the assembly code, read location and PC trace. Experimental results using two different workloads on the MPC565 show a reduction of bit-flip fault space between two and five orders of magnitude, depending on target locations.	inject-on-read	Assembly-level (ISA registers and memory). But this could be done on other levels as well.
Efficient Fault Injection based on Dynamic HDL Slicing Technique [8]	Fault list collapsing is performed by first deriving a static slice of the HDL code, i.e. the statements that a certain criterion (in this case, output signal) depends on. Following that, the statements executed for each clock cycle are derived during a golden run and the intersection of these statements, at each clock cycle with the static slice create the dynamic slice, representing the statements at each clock cycle for which a fault may have an effect on the chosen criterion. Experimental results show a speed-up of 11 times using the proposed technique.	static and dynamic HDL code slicing	register transfer level (slicing techniques should be possible on source code level)

Fault-List Collapsing for Fault-Injection Experiments [22]	A set of rules implementing inject-on-read analysis on assembly code instruction, data and registers for collapsing bit-flip fault lists, prior to fault injection campaigns are presented. The rules also consider analyses determining known effects of faults such as CPU exceptions. Experimental results using three different workloads on Motorola 68040, show between 36.3% and 54.5% reduction of the fault space when all rules are applied.	inject-on-read, fault list collapsing by determining equivalent faults and known effects of faults	assembly code
New Techniques for Speeding-up Fault-injection Campaigns [12]	Static workload independent gate level fault-list collapse and inject-on-read pre-injection analyses are presented. Experimental results with single bit-flips injected show a speed-up of at least an order of magnitude. Fault injection algorithms using checkpoints and observations after exponentially increasing points in time suitable for optimizing VHDL simulations are also presented. Dynamic (post-injection) fault collapsing identifying states with only one bit-flip difference from fault-free state, i.e., equivalent bit-flip faults, is also presented.	static workload independent gate level fault-list collapsing as well as static workload dependent inject-on-read, dynamic (post-injection) fault-list collapsing during simulation	gate level, register transfer level
Observations on the effects of fault manifestation as a function of workload [9]	Post-injection analysis is used to predict the results of fault injection experiments for different workloads depending on the results from previous fault injection campaigns. Experimental results from injecting transient single cycle stuck-at faults using gate-level simulations with three workloads show the fault prediction coverage for various instruction groupings used for the predictions for the different locations the faults were injected into.	prediction using post-injection analysis of workload attributes (instruction types and counts) and fault locations	gate level

Considering Workload Input Variations in Error Coverage Estimation [10]	Post-injection analysis is used to predict the error coverage for different input sequences depending on previous fault injection results using execution profile (i.e., the error coverage for each basic block of the program) and data usage analyses (i.e., the error coverage for different classes of data used by the program such as application data, pointers etc). Experimental results using scan-chain implemented fault injection on three different workload programs executing on the Thor microprocessor show that the methodology is able to identify input sequences with high, medium or low error coverage.	execution-profile and data-usage based prediction using post-injection analysis	assembly level
Path-Based Error Coverage Prediction [11]	Path-based execution profile injection extends the execution profile-based technique [10] by considering fault activation (i.e., when the fault is first propagated) rather than injection. Experimental result using simulation-based fault injection on the Thor microprocessor show that the technique is able to identify input sequences with high or low error coverage provided that the differences are significant.	path-based prediction using post-injection analysis	assembly level using register transfer level simulations
Improving Fault Injection of Soft Errors using Program Dependencies [5]	Inject-on-read pre-injection analyses of a brake-by-wire application running on the MPC565 CPU using scan-chain implemented fault injection is used showing a significant reduction of the fault space.	inject-on-read	assembly level
System dependability evaluation via a fault list generation algorithm [13]	Pre-injection analysis is performed by constructing fault trees for each node in a data flow diagram derived from assembler code. Faults injected at leaf nodes ending up at the same node along a path are considered equal and the corresponding subtree may be collapsed. Experimental results from simulation-based fault injection on a VHDL model of an Interlocking Control System shows that 1879 experiments covered 5780 fault nodes using the pre-injection technique.	assembly level fault list collapsing using fault trees	assembly level

On the Use of Formal Techniques for Validation [23]	Pre-injection analysis using formal verification with inference trees and dependency trees for V&V of real-time protocols is presented. Experimental results show that a fault case in a clock synchronization algorithm was discovered by just 310 experiments that was missed with 2.7 million random test cases. In a 2/3 majority voter, the number of test cases was reduced from 3827 to 24.	formal verification using interference trees and dependency trees	algorithmic protocol level
Efficient Fault Emulation using Automatic Pre-Injection Memory Access Analysis [6]	Pre-injection technique analyzing the memory usage of security relevant code during golden run (nominal behavior). Experimental results with fault injection tool using saboteurs injected into VHDL model of LEON3 executing on a Virtex 5 FPGA show a speed-up of approximately 1000.	inject-on-read, inject-on-write	assembly level
A Comparison of Inject-on-Read and Inject-on-Write in ISA-Level Fault Injection [7]	The paper compares two techniques for ISA-level fault injection (inject-on-read and inject-on-write). In addition, the paper compares two variants of inject-on-read, one where all faults are given the same weight and one where weight factors are used to reflect the time a data-item spends in a register or memory word. The weighted injected-on-read aims to accurately model soft errors that occur when an ionizing particle perturbs a data-item while it resides in an ISA register or a memory word. This is in contrast to inject-on-write, which emulates errors that propagate into an ISA register or a memory word. The experimental results show significant differences in the results obtained with the three techniques.	inject-on-read, inject-on-write	Assembly-level (ISA registers and memory). But this could be done on other levels as well.

A Study of the Impact of Bit-flip Errors on Programs Compiled with Different Optimization Levels [24]	The paper studies the impact of compiler optimizations on the error sensitivity of programs. To do so, ISA-level fault injection is performed. The results show that the percentage of SDCs in the output of the optimized programs is only marginally higher compare to that observed for the non-optimized programs. This suggests that compiler optimizations can be used in safety- and mission-critical systems without increasing the risk that the system produces undetected erroneous outputs. Using compiler optimizations would allow having a much smaller error space which would allow the testing time to be reduced.	Using compiler optimizations to reduce the program size which would result in the reduction of the error space and speeding up the testing time.	Compiler-level
A Study of the Impact of Single Bit-Flip and Double Bit-Flip Errors on Program Execution [14]	The study provides insights on whether it is necessary to consider double bit-flip errors in dependability benchmarking experiments. The results show that the proportion of SDC in the program output is almost the same for single- and double-bit errors. Moreover, the paper shows that the error sensitivity varies significantly between different bit positions and registers. An important observation is that injections in certain bit positions always have the same impact regardless of when the error is injected. This could be used for pre-injection analyses allowing removal of certain locations from the error space.	If it can be shown that the number of SDCs resulted when injecting single bit-flips is higher than when double bit-flips are injected, the error space may be pruned by only focusing on single bit-flip injections.	Assembly-level (ISA registers and memory). But this could be done on other levels as well.

Light-Weight Techniques for Improving the Controllability and Efficiency of ISA-Level Fault Injection Tools [16]	This paper presents two pre-injection analysis techniques that improve controllability and efficiency attributes of fault injection technique using object code analysis. To improve controllability, the paper proposes a technique for identifying the type of data that is stored in a potential target location. This allows the user to selectively direct fault injections to addresses, data and/or control information. The second technique improves efficiency by fault pruning, i.e., by avoiding injection of faults that is known a priori to be detected by the tested system. This technique leverages the fact that faults in certain bits in the program counter and the stack pointer are always detected by machine exceptions hence could be removed from the error space.	Making use of the information we obtained from fault injection experiments to identify areas/signals that could be removed from the error space.	Assembly-level (ISA registers and memory). But this could be done on other levels as well.
One Bit is (Not) Enough: An Empirical Study of the Impact of Single and Multiple Bit-Flip Errors [15]	This paper investigates whether multiple-bit errors could cause a higher percentage of silent data corruptions (SDCs) compared to single-bit errors. The results show that single-bit errors in most cases yields a higher percentage of SDCs compared to multiple-bit errors. The paper also proposes three ways of pruning the multiple bit-flip error space based on the results.	If it is shown that the number of SDCs resulted when injecting x number of bit-flips is higher than when injecting more than x number of bit-flips, the error space can be pruned by only focusing on the injection of x number of bit-flips. Moreover, using post injection data, the error space could be pruned.	Compiler-level, but this could be done on other levels as well.

Relyzer: Exploiting Application-Level Fault Equivalence to Analyze Application Resiliency to Transient Faults [17]	The paper presents Relyzer, an approach that systematically analyses all application fault sites and picks a small subset to perform selective fault injections for transient faults. Relyzer prunes faults that need detailed study by either predicting their outcomes or showing them equivalent to other faults. Relyzer prunes about 99.78% of the total faults across twelve applications studied. Fault injection simulations on the remaining faults can identify SDC causing faults in the entire application. Some of Relyzer's techniques rely on heuristics to determine fault equivalence.	- Known-outcome pruning technique (post injection analysis) - Equivalence-based pruning techniques - Pruning techniques making use of program implementation	Assembly-level (ISA registers and memory) and high-level programs. But this could be done on other levels as well.
Approxilyzer: Towards A Systematic Framework for Instruction-Level Approximate Computing and its Application to Hardware Resiliency [18]	This paper presents a framework for approximate computing, called Approxilyzer, that quantifies the quality impact of a single-bit error in all dynamic instructions of an execution. The paper shows that Approxilyzer determines that a very small loss in output quality (1%) can yield large resiliency overhead reduction. The Relyzer methodology presented by [17] is also used here.	Identification of signals and locations that if targeted would result in a severe case of SDC. This way, injections in locations that would result in a benign failure could also be pruned.	Assembly-level (ISA registers and memory) and high-level programs. But this could be done on other levels as well.
Avoiding Pitfalls in Fault-Injection Based Comparison of Program Susceptibility to Soft Errors [19]	The paper identifies several pitfalls of fault injection experiments and discusses the necessity to use weight factors when using pre-injection analysis techniques such as inject-on-read. This is due to the fact that in inject-on-read, candidate faults will be selected from fault classes, where the result of the injected candidate fault needs to be propagated in the whole class. They also propose the use of extrapolated absolute failure counts over the fault coverage metric.	The analysis and results presented in this paper need to be considered in many pre-injection analysis techniques as their main focus is on the representativeness of the results obtained after pruning the error space.	Could be applied to any layer where the pre-injection analysis is conducted at.

Table 11. An overview of the pre-injection analysis techniques that have been defined and detailed in the past.

Conclusions

Fault injection is a (highly) recommended assessment technique for system safety. Fault injection concepts and methodologies may also be useful for security testing, which is referred to as attack injection, and thus applicable for fuzzing, vulnerability scanning and penetration testing techniques. The aim of the pre-injection analysis pattern is to reduce the fault/attack space of fault/attack injection testing, in order to

minimize the time and cost (computational and resource related) of safety and security verification and validation.

We have conducted a literature review of current pre-injection analysis techniques to determine the applicability of pre-injection analysis to our model-implemented fault and attack injection tool MODIFI, which is capable of injecting faults/attacks into Simulink models.

Two pre-injection analysis techniques adapted for the MODIFI tool are proposed. The first technique is called inject-on-write, where faults and attacks are only injected on a resource immediately after it has been produced. Results from applying the inject-on-write technique on a simplified TMR voter shows a 61% reduction of the fault/attack space. The second technique is called inject-on-read, where faults and attacks only target resources immediately before they are read. Results from applying the inject-on-read technique on the TMR voter model show that the fault/attack space may be reduced by 39%.

Thus, an advantage of using the inject-on-write technique compared to inject-on-read is that the fault/attack space may be reduced more (61% vs. 39% on the chosen target model). Another advantage is that it may be possible to simulate faults and attacks injected inside model blocks, which are typically not accessible for fault/attack injection using MODIFI. A disadvantage with the inject-on-write technique is that it is not always possible to inject all faults/attacks provided by the inject-on-read technique.

Future work includes implementing additional pre-injection analyses techniques for the MODIFI tool combining the proposed approaches with other techniques, providing further pruning of the fault/attack space.

References

[1] ISO, I., 2011. 26262: Road vehicles-Functional safety. International Standard ISO/FDIS, 26262.

[2] B. Sangchoolie, P. Folkesson and J. Vinter, "A Study of the Interplay Between Safety and Security Using Model-Implemented Fault Injection," 2018 14th European Dependable Computing Conference (EDCC), Iasi, 2018, pp. 41-48.

[3] Barbosa R., Vinter J., Folkesson P., Karlsson J. (2005) Assembly-Level Pre-injection Analysis for Improving Fault Injection Efficiency. In: Dal Cin M., Kaâniche M., Pataricza A. (eds) Dependable Computing - EDCC 5. EDCC 2005. Lecture Notes in Computer Science, vol 3463. Springer, Berlin, Heidelberg.

[4] Svenningsson R., Vinter J., Eriksson H., Törngren M. (2010) MODIFI: A MODel-Implemented Fault Injection Tool. In: Schoitsch E. (eds) Computer Safety, Reliability, and Security. SAFECOMP 2010. Lecture Notes in Computer Science, vol 6351. Springer, Berlin, Heidelberg.

[5] G. Munkby and S. Schupp, "Improving Fault Injection of Soft Errors Using Program Dependencies," Testing: Academic & Industrial Conference - Practice and Research Techniques (taic part 2008), Windsor, 2008, pp. 77-81.

[6] J. Grinschgl, A. Krieg, C. Steger, R. Weiss, H. Bock and J. Haid, "Efficient fault emulation using automatic pre-injection memory access analysis," 2012 IEEE International SOC Conference, Niagara Falls, NY, 2012, pp. 277-282.

[7] B. Sangchoolie, F. Ayatolahi, R. Johansson and J. Karlsson, "A Comparison of Inject-on-Read and Inject-on-Write in ISA-Level Fault Injection," 2015 11th European Dependable Computing Conference (EDCC), Paris, 2015, pp. 178-189.

[8] Bagbaba, Ahmet Cagri et al. “Efficient Fault Injection based on Dynamic HDL Slicing Technique.” 2019 IEEE 25th International Symposium on On-Line Testing and Robust System Design (IOLTS) (2019): 52-53.

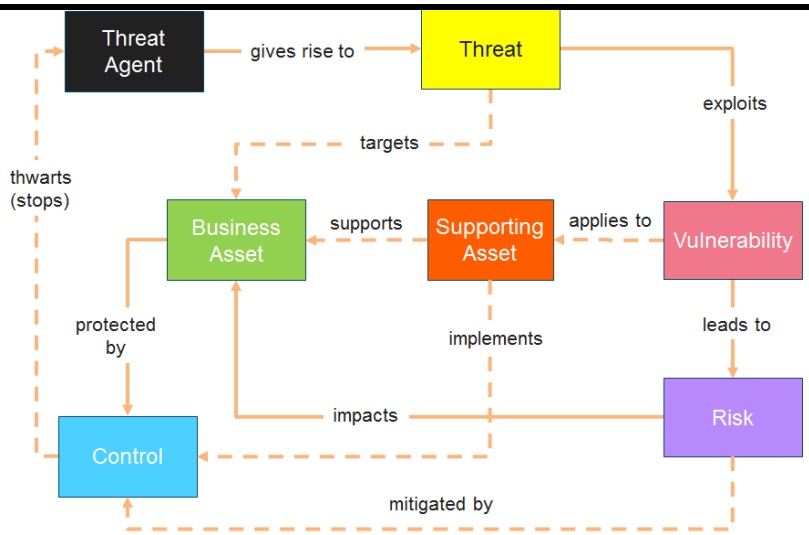
[9] Czeck, Edward W. and Daniel P. Siewiorek. “Observations on the Effects of Fault Manifestation as a Function of Workload.” IEEE Trans. Computers 41 (1992): 559-566.

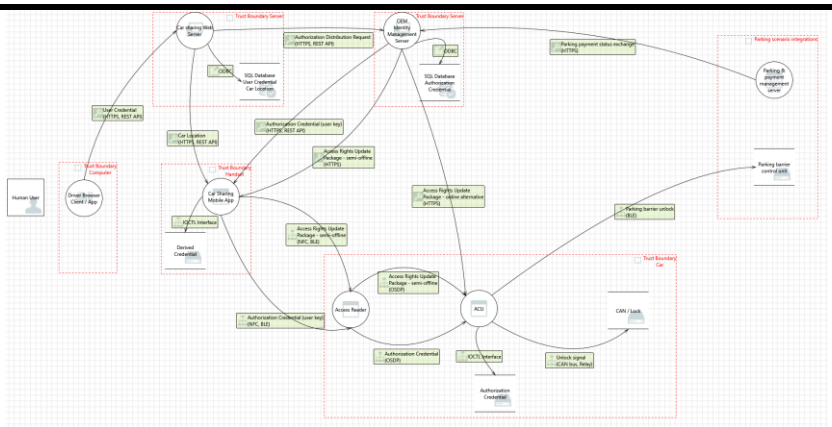
[10] Folkesson P., Karlsson J. (1999) Considering Workload Input Variations in Error Coverage Estimation. In: Hlavička J., Maehle E., Pataricza A. (eds) Dependable Computing — EDCC-3. EDCC 1999. Lecture Notes in Computer Science, vol 1667. Springer, Berlin, Heidelberg
[11] J. Aidemark, P. Folkesson and J. Karlsson, "Path-based error coverage prediction," Proceedings Seventh International On-Line Testing Workshop, Taormina, Italy, 2001, pp. 14-20.
[12] L. Berrojo, I. Gónzález, F. Corno, M. Sonza Reorda, G. Squillero, L. Entrena, and C. López. 2002. New Techniques for Speeding-Up Fault-Injection Campaigns. In Proceedings of the conference on Design, automation and test in Europe (DATE '02). IEEE Computer Society, USA, 847.
[13] D. T. Smith, B. W. Johnson and J. A. Profeta, "System dependability evaluation via a fault list generation algorithm," in IEEE Transactions on Computers, vol. 45, no. 8, pp. 974-979, Aug. 1996.
[14] Fatemeh Ayatolahi, Behrooz Sangchoolie, Roger Johansson, and Johan Karlsson. 2013. A Study of the Impact of Single Bit-Flip and Double Bit-Flip Errors on Program Execution. In Proceedings of the 32nd International Conference on Computer Safety, Reliability, and Security - Volume 8153 (SAFECOMP 2013). Springer-Verlag, Berlin, Heidelberg, 265–276.
[15] Sangchoolie, Behrooz et al. "One Bit is (Not) Enough: An Empirical Study of the Impact of Single and Multiple Bit-Flip Errors." 2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN) (2017): 97-108.
[16] B. Sangchoolie, R. Johansson and J. Karlsson, "Light-Weight Techniques for Improving the Controllability and Efficiency of ISA-Level Fault Injection Tools," 2017 IEEE 22nd Pacific Rim International Symposium on Dependable Computing (PRDC), Christchurch, 2017, pp. 68-77.
[17] Siva Kumar Sastry Hari, Sarita V. Adve, Helia Naeimi, and Pradeep Ramachandran. 2012. Relyzer: exploiting application-level fault equivalence to analyze application resiliency to transient faults. SIGPLAN Not. 47, 4 (March 2012), 123–134.
[18] R. Venkatagiri, A. Mahmoud, S. K. S. Hari and S. V. Adve, "Approxilyzer: Towards a systematic framework for instruction-level approximate computing and its application to hardware resiliency," 2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO), Taipei, 2016, pp. 1-14.
[19] H. Schirmeier, C. Borchert and O. Spinczyk, "Avoiding Pitfalls in Fault-Injection Based Comparison of Program Susceptibility to Soft Errors," 2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, Rio de Janeiro, 2015, pp. 319-330.
[20] Svenningsson R., Vinter J., Eriksson H., Törngren M. (2010) MODIFI: A MODel-Implemented Fault Injection Tool. In: Schoitsch E. (eds) Computer Safety, Reliability, and Security. SAFECOMP 2010. Lecture Notes in Computer Science, vol 6351. Springer, Berlin, Heidelberg.
[21] B. Sangchoolie, P. Folkesson and J. Vinter, "A Study of the Interplay Between Safety and Security Using Model-Implemented Fault Injection," 2018 14th European Dependable Computing Conference (EDCC), Iasi, 2018, pp. 41-48.
[22] Benso, A., Rebaudengo, M., Impagliazzo, L. and Marmo, P., 1998, January. Fault-list collapsing for fault-injection experiments. In Annual Reliability and Maintainability Symposium. 1998 Proceedings. International Symposium on Product Quality and Integrity (pp. 383-388). IEEE.
[23] Suri, N. and Sinha, P., 1998, June. On the use of formal techniques for validation. In Digest of Papers. Twenty-Eighth Annual International Symposium on Fault-Tolerant Computing (Cat. No. 98CB36224) (pp. 390-399). IEEE.
[24] Sangchoolie, B., Ayatolahi, F., Johansson, R. and Karlsson, J., 2014, May. A study of the impact of bit-flip errors on programs compiled with different optimization levels. In 2014 Tenth European Dependable Computing Conference (pp. 146-157). IEEE.

2.2.7 AP09: Common Risk Assessment methodology

2.2.7.1 Brief Overview

Analysis Aim:	Methodology of Risk Assessment, compliant with ISO20005, that identifies, formalizes, explains and assesses the inherent and residual security risk. Privacy assessment based on dataflow description and data collected by a CTE or DP.

	 *Risk Assessment Methodology*
Abstract:	The objectives are to deliver safer and more trustworthy element, and to ensure a higher efficiency and confidence that known classes of security vulnerabilities are understood and effectively addressed. This include privacy aspect if the analysed system possesses personal information.
Addressed Problem:	Even if all identified risks cannot be eliminated, they can be reduced to an acceptable level by the implementation of technical and organizational countermeasures.
Stakeholder:	Driver, Car, Cloud Service, Identity Provider (IdP), Authorization server
Categories and Scope:	
Scope:	Improvement-AP
Safety:	Security & Privacy
Security:	
Privacy:	
Input:	
Description:	Data flow diagram of a system, including hardware, software and communication protocol, optional threat analysis if available as done in WP1. List of collected information handle by CTEs or DP that are in scope of this analyse.
Example(s):	

	 *Example of Data flow diagram of Use Case 4 (D.1.2)*
Addressed Elements:	
Related DP:	Any DP that includes several processes that exchange information among each other. Example: DP32 – Derived Identity
Contributors:	
Responsible:	Cyrille FALCOU, Thales DIS
List of Contributors:	Cyrille FALCOU, Thales DIS

2.2.7.2 Detailed Description

Introduction

There are many methodologies to conduct a Risk Analysis, based on different standards (e.g. ISO 27005, Microsoft SDLC, TARA, SAHARA, etc.). The first objective of this AP is to propose a common methodology in terms of Risk Assessment, including risks that can impact privacy. The second objective is to apply this methodology to a DP as mainly performed on complete system or use case. The purpose of this analysis is to highlight what could be the weak point of a design pattern and propose some controls based on CTEs or other DPs (developed in the context of SECREDAS).

In the next chapters, the methodology will be explained through a concrete example of DP.

Analysis of DP32: Derived Identity

The DP32: “Derived Identity” is a concept that allows a service provider to delegate authentication and identity management to a trusted third Party. The identity provider who manages the generation of derived identities from an official document, such as driving license, and uses it as a mean of authentication and privacy protection. Therefore, as this concept will become the core of the authentication and privacy protection for an automated system involving access to eService in the cloud, we focus the main efforts of our security and privacy analysis on this DP.

Developed solution

We split our analysis into 3 steps:

Step 1, description of the system:

The first step consists of representing a view of the system, including elements, connections between these elements, technologies used for these elements and communication used for data/elements exchange. Among many tools and representations, we chose to use the Data Flow Diagram provided by the Microsoft Modelling Tool 2016, which proposes a standard notation for visualizing system components, data flows and security boundaries, and is suitable for this kind of analyses. Figure 22 describes elements of DP32 Derived Identity and their connections:

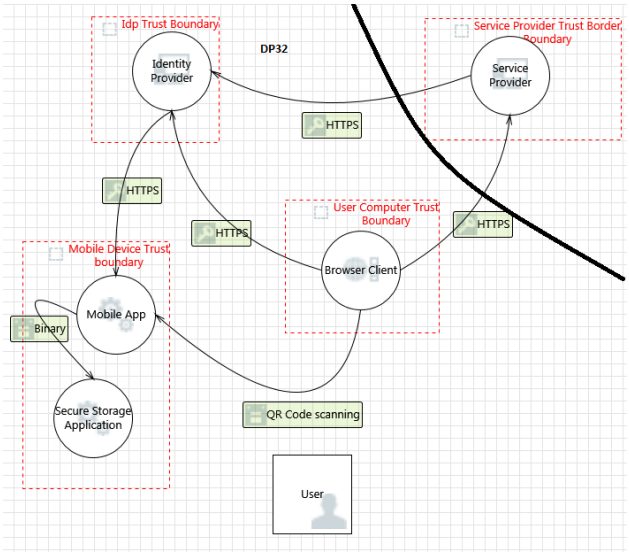


Figure 22: Data Flow Diagram Derived Identity

Step 2, list the assets:

There are 2 asset categories:

A. <u>Business assets:</u>

Represents assets that makes value to the system, it can be information (e.g. sensitive data) or services (e.g. payment service). Each business asset is assigned a security level (CRITICAL/HIGH/MEDIUM/LOW), that helps in measuring the impact for each of the CIA properties, namely:

- **Confidentiality**: Is the property that is concerned in ensuring that information is not made available or disclosed to unauthorized individuals, entities or processes.

- **Integrity**: Is the property that is concerned in protecting the accuracy and completeness of assets, in other words, it is the property that ensures that assets have not been modified by unauthorized entities.
- **Availability**: Is the property that is concerned in ensuring information being accessible and usable upon demand by an authorized entity.

In Table 12, we list the identified business assets for DP32:

Data	Description	Confidentiality	Integrity	Availability
Derived Credential Data	Credential generated by the Idp (Identity provider) during registration	HIGH Can grant access to the service several times if disclosed	HIGH Can grant access to the service several times if modified	LOW
Verified User Attributes	Part of derived credential, protected by user consent	LOW	HIGH Information about the user distributed to the Service Provider used the check if user is allowed to access the service (i.e.: age>18)	LOW
Id of session between the IdP and the Mobile App	Session ID displayed on IdP page in form of QR code and then scanned by the App and used during credential verification	MEDIUM Can grant access to the service only if disclosed	LOW	LOW
IdP Session Id	Standard Web application Session ID	MEDIUM Can grant access to the service only if disclosed	LOW	LOW
Service Provider Session Id	Standard Web application Session ID	MEDIUM Can grant access to the service only if disclosed	LOW	LOW

Authorization Code	Part of OpenID Connect protocol, a random number used as a key to access protected Id Token	MEDIUM Can grant access to the service only if disclosed	LOW	LOW
Id Token	Contains Authentication result and User Id. It is protected by signature and has limited time validity.	MEDIUM Can grant access to the service only if disclosed	HIGH Contain Verified Attributes	LOW

Table 12: List of Business Assets, DP Derived Identity

B. Supporting assets:

Represents the means that support and protect the Business Assets, it can be software, server, component, network or hardware. Below the list of supporting assets are identified for DP32:

Supporting Assets	**Description**	**Type**
Mobile Application	Dedicated application running on a Mobile Device and used for User Authentication	Software application
Identity Provider	Web Application that manages authentication of the User for the Service Provider and provides API for Credential Verification and OpenID connect	Web Service
Browser Client	Browser running on user PC	Computer Application
Service Provider	Provider of a Service as Car Sharing service in SECREDAS context	Web Service

Table 13: List of Supporting Assets, DP Derived Identity

Link Matrix

This matrix describes which business asset are handled by which supporting asset and is the input for step 3:

Supporting Assets	**Business Assets**
Mobile Application	Derived Credential Data
	Verified User Attributes
	ID of session between IdP and Mobile App
Identity Provider	Verified User Attributes
	Authentication Credentials/Transaction
	ID of session between IdP and Mobile App
	IdP Session Id
	Authorization Code
	Id Token
Browser Client	IdP Session Id
	Authorization Code
	Service Provider Session Id

	ID of session between IdP and Mobile App
Service Provider	Authorization Code
	Service Provider Session Id
	Id Token

Table 14: Link Matrix between Business Assets and Supporting Assets, DP Derived Identity

Step 3, Expressing a risk:

This step consists of using the link matrix and a combination of supporting assets, business assets and other elements, listed below, to formulate a risk, starting from the following generic sentence:

Generic Sentence:

"As a THREAT AGENT, I can THREAT these BUSINESS ASSET, compromising this SUPPORTING ASSET, in order to MOTIVATION, exploiting this VULNERABILITY"

With:

THREAT AGENT:

In cyber security risk domain, the list of threat agents depend on the context in which the system is deployed, it could be a nation state, employee, terrorist, activist, user (itself), researcher, etc. But for this example, we consider a unique agent, that is: the criminal.

THREAT:

The list of threats is a standard categorized by Microsoft, well known by the mnemonic STRIDE:

- Spoof (Spoofing).
- Tamper with (Tampering).
- Repudiate (Repudiation).
- Disclose (Information Disclosure).
- Deny access to (Denial of service).
- Gain unauthorized permission/access (Elevation of privilege).

And some additional threats that we consider relevant:

- Flood (brute force to retrieve secret).
- Loose (hardware containing sensitive data or process lost).
- Steal (hardware containing sensitive data or process stolen).

VULNERABILITY:

The list of vulnerabilities is taken from OWASP catalogue, but we retain the following ones for this example:

- Session Management Vulnerability.
- Sensitive Data Protection.

The list of risks provided by this method includes some irrelevant ones, that we can exclude intuitively, and some other generic ones. We consider the relevant ones listed in Table 15:

Risk Id	Risk	Remark
Risk 1	As a Criminal, I can gain access to the Idp session ID compromising the Browser Client in order to have unauthorized access to the service provider, exploiting this Session Management Vulnerability	merge in same risk: Session Theft to have illegal access to the Service provider
	As a Criminal, I can gain access to the Service Provider Session Id compromising the Browser Client in order to have unauthorized access to the service provider, exploiting this Session Management Vulnerability	
	As a Criminal, I can gain access to the Id Session between IdP and Mobile App compromising the Browser Client in order to have unauthorized access to the service provider, exploiting this Session Management Vulnerability	
Risk 2	As a Criminal, I can disclose the Derived Credential Data compromising the Mobile Application in order to have unauthorized access to the service, exploiting Sensitive Data Protection vulnerability	Credential Forging depends on security of the App and the cryptographic scheme
Risk 3	As a Criminal, I can disclose the Verified User Attributes compromising the Service Provider in order to gather personal information from the user, exploiting Sensitive Data Protection vulnerability	Information Leaks depending on the Service Provider implementation and its protection of personal data.
Risk 4	As a Criminal, I can tamper with the Authorization Code compromising the Browser Client in order to deny access to the service provider, exploiting this Sensitive Data Protection vulnerability	Merge in same risk as same data compromise in the same way
	As a Criminal, I can gain access to the Authorization Code compromising the Browser Client in order to have unauthorized access to the service provider, exploiting this Sensitive Data Protection vulnerability	
Risk 5	As a Criminal, I can gain access to the Id Token compromising the Service provider in order to get personal information, exploiting this Sensitive Data Protection vulnerability	Merge in same risk as same data compromise in the same way

	As a Criminal, I can gain access to the Id Token compromising the Identity provider in order to get personal information, exploiting this Sensitive Data Protection vulnerability	
Risk 6	As a Criminal, I can tamper with the Verified Attribute compromising the Mobile Application in order to spoof my identity , exploiting this Sensitive Data Protection vulnerability	

Table 15: Expressed Risks, DP Derived Identity

Once the risks are identified and correctly expressed, mitigation could be applied using relevant control, depending on the harm probability that the risk, if exploited, will cause to the system.

<u>A step further:</u>

To go further than the technical analysis, we could quantify a risk including the impact on the business, according to additional financial aspects (loss of business, loss of reputation, etc.) using the following formula:

With the following elements:

- **Impact**:

  CRITICAL =4: Catastrophic impact on business, could lead to losing customers.

  HIGH = 3: Major impact on business with high financial impact, could lead to loss of reputation.

  MEDIUM = 2: Serious impact on business with direct or indirect significant financial impact, could lead to loss of reputation

  LOW = 1: Significant impact on business with very limited financial impact.

- **Likelihood**: VERY HIGH[4] / HIGH[3] / MEDIUM[2] / LOW[1]

  The likelihood is based on OWASP evaluation method that consider different factors:

  - *Threat Agent Factors*: Skill Level, Motivation, Opportunity, Size.
  - *Vulnerability Factors*: Ease to discover, Ease to exploit, Awareness, Intrusion detection.

- **Vulnerability**: HIGH[7-10] / MEDIUM[4-6.9] / LOW[0-3.9] (CVSS)

  This element is computed according NIST model CVSS2 : https://nvd.nist.gov/CVSS-v2-Calculator

  The result of the equation quantifies the risk:

  CRITICAL: 16 <Risk < 20

  HIGH: 11< Risk < 15

  MEDIUM: 6 < Risk < 10

  LOW: 0< Risk < 5

Evaluation of Improvement

As a DP improvement, according to the identified risk, we can enforce the security of the DP with some control developed in SECREDAS context or eventually highlight the misused usage of the DP in a system, as listed in Table 16:

Risk Id	Control	Relevant CTE
Risk 1	• Sufficient Session ID length and entropy • Enforce protection of communication	CTE-16: Transport-Layer Security
Risk 2 Risk 6	• State of the Art in term of Cryptographic library strength • Secure storage of Derived Credential data	CTE-02: Cryptographic libraries CTE-05: Secure Element CTE-06: Secure OS/Trusted Execution Environment
Risk 3 Risk 4 Risk 5	• Usage Guide to Service Provider with recommendation • Enforce protection of communication	CTE-16: Transport-Layer Security

Table 16: List of controls for improvement of DP Derived Identity

Related Work

This work is a methodology based on combination of different standard:

- ISO 27005 Risk Management
- Microsoft SDL : https://www.microsoft.com/en-us/securityengineering/sdl/threatmodeling
- NIST CVSS : https://nvd.nist.gov/CVSS-v2-Calculator
- OWASP Top 10

References

[1] ISO/IEC 27005:2018 - Information technology – Security techniques – Information security risk management
[2] Microsoft Security Development: https://www.microsoft.com/en-us/securityengineering/sdl/
[3] NIST National Vulnerability Database: https://nvd.nist.gov/
[4] OWASP Risk Rating Methodology: https://owasp.org/www-community/OWASP_Risk_Rating_Methodology

2.2.8 AP10: Security Zones approach

2.2.8.1. Brief Overview

Analysis Aim:	Create network zoning based on security and safety (assurance) requirements
Abstract:	Some security architectures are based on a perimeter defence model. With an increase in networked systems, a method is needed to analyse and support design choices on how to segment these networks in a way that they are more resilient to cyber-attacks within these segments. In addition, these forms of segmentation are difficult to apply due to fading lines between organizational infrastructures, cloud and integration of user devices.
Addressed Problem:	Cyber security incident detection and mitigation while preserving safety objectives
Stakeholder:	OEM
Categories and Scope:	
Scope:	Improvement-AP
Safety:	Functional Safety (Assurance) Requirements
Security:	Cyber Security (Assurance) Requirements
Privacy:	Out of scope
Input:	
Description:	Assurance levels to be met and interdependencies
Example(s):	Functions with different ASIL requirements, are placed in different security zones Function dependencies requirements, are a basis to establish communication security requirements
Addressed Elements:	
Related DP:	DP06
Contributors:	
Responsible:	Andre Smulders, TNO
List of Contributors:	Sterre den Breeijen, TNO Irina Chiscop, TNO

2.2.8.2 Detailed Description

Introduction

Designing secure systems which encompass many functions and components, requires a structured methodology to ensure that the appropriate security mechanisms are identified and applied. This problem

increases in difficulty when not all functions are confined within a single system but need to be distributed over different systems. Moreover, there is a need for a reasoning model that steps away from the traditional approach in which the internal infrastructure is secure and needs to be protected only from the outside intrusions. Such a model must allow a way to continuously monitor how security requirements are being fulfilled, derive new requirements and identify gaps in the current implementation. The Security Zones Approach addresses these problems in a structured way, in order to identify which security requirements should be implemented at what place in a system architecture. As shown in this example of the health data exchange, the approach can also be used to evaluate an existing architecture and identify security gaps.

Description of the Security Zones methodology

The Security Zones Approach yields a functional decomposition of a design pattern into Security Zones while preserving the dependencies of security requirements for each zone[4]. In the context of the current deliverable, the Security Zones Approach method is used to identify whether security or safety requirements are sufficiently addressed in a certain design pattern. By using this structured approach, one can determine if and where in the design pattern additional security mechanisms would be required. The core idea behind the Security Zones Approach is to take the zero-trust principle as a starting point. Access from and to a zone is by default not-granted, unless it has been confirmed that sufficient security mechanisms are in place.

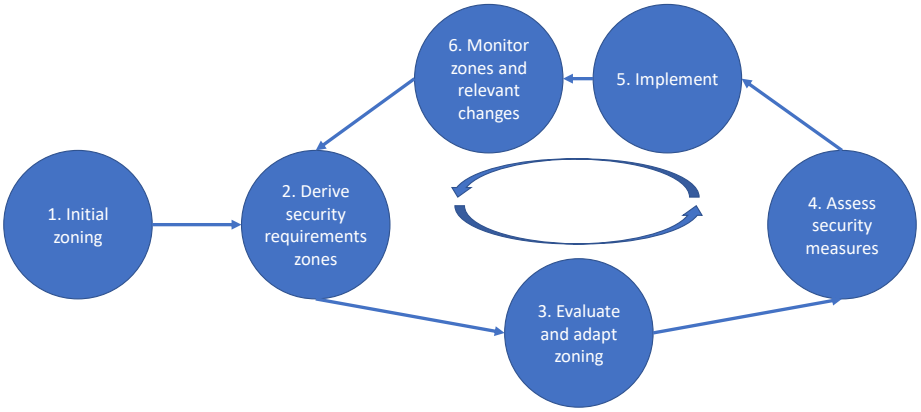


Figure 23. Security Zones Implementation Steps

The methodology is schematically represented in Figure 23 and comprises the following steps:

[4] The security requirements can be the outputs of existing analysis methodologies (Smulders, 2019).

- Split the architecture into 'trusted' security zones.
- Derive security requirements for each zone.
- Analyze the system based on the interaction between these zones.
- Identify eventual additional security mechanisms.
- Implementation of newly proposed mechanisms.
- Monitor consequences and impact of changes onto the system.
- Repeat analysis cycle.

The functional decomposition depends on formulating clear security requirements for each zone. These requirements can be derived in different ways: applying threat identification or risk management models (Smulders, 2019), enforcing security/safety/privacy principles (Sanden & Vasenev, 2019) and additional existing rules/regulations/ organizational policies. In this analysis steps 1-4 are executed. The results have been discussed with the DP owner, who can decide which countermeasures should be implemented in steps 5 and 6.

Applying Security Zones to the DP06 Healthcare data exchange

The Security Zones Approach takes the initial description of Design Pattern DP06 Healthcare Data Exchange (Marko, D3.5 Initial design patterns for common technology elements, 2019) as input. The objective of this design pattern is providing driver's health data towards the in-car fitness-to-drive logic such that it is compliant with privacy and security guidelines, principles and regulations. This concept uses mostly existing technical standards (e.g. OAuth2, XACML, TLS, and Bluetooth LE) and implementations of such standards.

This design pattern is primarily focused on preserving privacy (confidentiality and integrity) of sensitive personal health data which originates from professional health infrastructure (e.g. hospitals, lab, doctor practices). The objective of the design pattern is to provide personal health data to the vehicle so it can be used to incorporate health status in monitoring the driver when accessing and driving the vehicle.

Based on the design pattern and interactive sessions with ROCHE, an analysis of the design pattern was performed. In the first iteration the Design Pattern was broken down into a number of security zones. For the zoning, the following inputs where used as a starting point:

- High level security requirement from the DP06 description (Marko, D3.5 Initial design patterns for common technology elements, 2019).
- Principles identified in the reference architecture (Sanden & Vasenev, 2019).

Based on the description and objectives for the Design Pattern, the most relevant principles for our analysis are:

- Do not trust the infrastructure.
- Authenticate users and processes.
- Authorize after you authenticate.
- Asset protection and resilience.
- Resistant in the garage.

In further analysis, additional principles could be used if required. These inputs form the starting point of our analysis with the objective to preserve the confidentiality of health care data (originating from the cloud environment) throughout the DP, while observing the stated principles. The analysis is done in three iterations.

The **first iteration** resulted in the following division of functions and zoning as is given in Figure 24. The functions derived from the DP06 description are distributed over different zones and for each zone owners (A…C) are assigned. In this case the actor responsible for the cloud environment (A) is also responsible for the app (A') and both the cloud and app development are under control of the same actor. If this is not the case, a check needs to be done if some of the measures should be implemented differently. Each step in the communication process between the resulting zones is identified by a number (1 – 5), where (1) is the use of the Multi-Factor Authentication (MFA) provided by the phone to access the app. In (2) the phone sets up a secure communication with the healthcare cloud. In this case, we assume that this is done using credentials from the certificate store of the phone, establishing an SSL/HTTPS connection. The phone establishes a secure connection with the vehicle (3) using Bluetooth with the vehicle, again using functionality provided by the phone and standard Bluetooth functionality available in the vehicle. Additionally, credentials stored in the app's secure data store can be used to authenticate the phone to the vehicle (4). The same could be done to authenticate the app to the healthcare cloud in the last step (5). These communications are described in more detail in the DP06 (Marko, D3.6 - Design pattern descriptions, 2020).

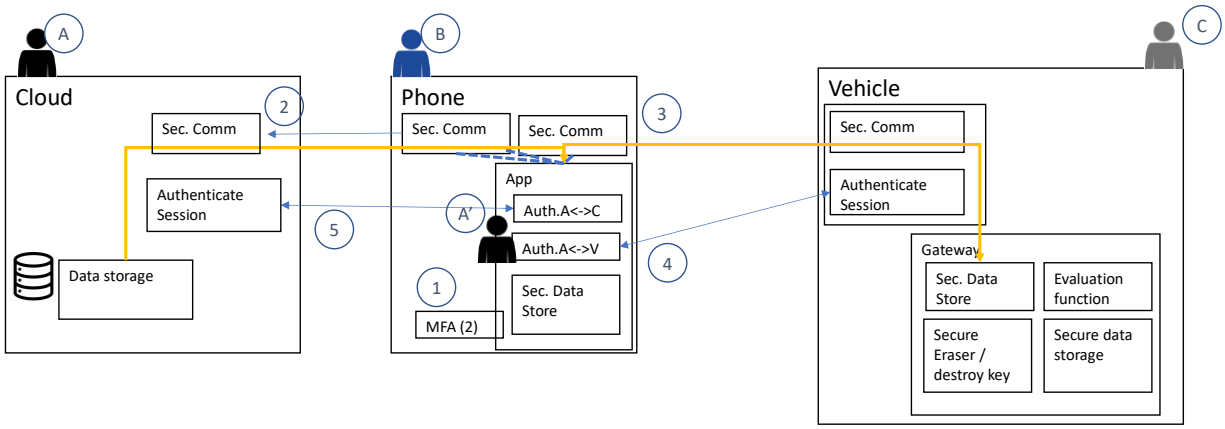


Figure 24. First iteration of Security Zones on DP06

For the **second iteration**, we looked in more detail into the dependencies between these functions and checked the potential additional requirements from the various owners. Decomposition is done by looking at functions where these reside and what entities have specific requirements for that function. Based on this, the next step is to see whether additional mechanism or requirements should be addressed.

Using the DP06 description of the intended use, there is no need for the phone to handle confidential healthcare data. This observation in combination with the principle "Do not trust the infrastructure" leads to the assignment of a low trust level to all components within the zone of the phone including the app. As a consequence, all security related processing of confidential healthcare data should be done either in the healthcare cloud or in the vehicle. As a consequence, the need for access to secure data storage by the app on the phone then only applies to credentials for authentication and authorisation purposes. Also, the security of the data transferred via the phone should be implemented in functions in the cloud and the vehicle and should not depend on further functionality in the phone. Further applying the principle "Do not trust the infrastructure", the analysis focusses on the security mechanisms provided by the infrastructure components and identify if additional mechanisms are needed to protect the health care data while there are differences in the amount of trust in the various zones. In our example we will further look into the secure communication between infrastructure components, and the preservation of confidentiality of data handled by these components. Another assumption is that there is a difference in trust over the components in the vehicles to setup a secure connection with the phone (which has a low trust level) and the functions in the gateway (which has a high trust level assigned). The implication direct hereof is that actual implementations of zones which require a high trust level, must undergo more rigorous analysis and testing and monitoring in order to attain that level of trust. One example are the

functions related to functional safety or personal safety which result in assurance (trust) requirements from the context of the vehicle.

Using this information, a relative trust level (high/low) and basic security requirements are applied to each zone. The function of these trust levels is to identify potential gaps in the design in the next step. Since the Design pattern is focused on health care data, we will focus on the preservation of confidentiality of the healthcare data originating from the healthcare cloud. The confidentiality of this data will serve as a security requirement in this case.

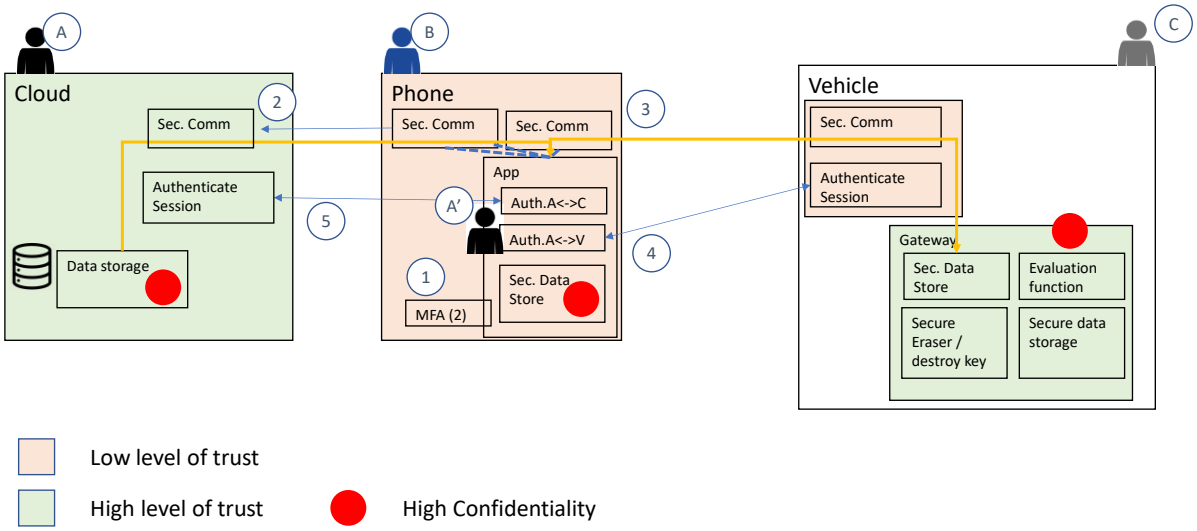


Figure 25. Trust levels and confidentially requirement applied to zones

For the **third iteration**, we elaborated on the assumption that the underlying infrastructure provided by the phone and components in the vehicle is less trusted. Although the phone is able to set up a secure connection, both to the vehicle and to the cloud, these connections (2) and (3) cannot be trusted sufficiently since there is gap in the trust levels of these zones. This needs to be compensated, possibly by additional authentication of the app to the vehicle (4) and the cloud (5).

Therefore, additional mechanisms are needed to assure that the healthcare cloud communicates (securely) with an authorized vehicle. As we can derive from the DP, the phone is used to setup both connections to the healthcare cloud as well as the vehicle. As the phone has been assigned a lower level of trust, additional mechanisms are needed to counter the difference in trust required by the healthcare cloud and gateway while preserving the correct level of confidentiality. Which means that based on the secure connections provided by the phone, another level of secure communication between the cloud and

the vehicle needs to be established. This is an additional mechanism, since the trust level for the secure connection is setup by the phone, is lower than the trust level needed between the cloud and the vehicle. To support this mechanism, DP06 foresees another zone (D) (see Figure 26) providing registration, authentication and authorization of cloud and vehicle-based functionality. Please note that this zone is out of scope for this design pattern. To demonstrate the analysis pattern, we extended the analysis to show how this can be used to generate input for other design patterns, focusing on the functionalities within this zone (D).

There are two arguments for placing this functionality in a new zone (D). The first argument is that it is not yet clear who will be the zone owner, and therefore responsible for the functions in this zone. The second argument relates to the fact that this functionality is not in the scope of the design pattern (D06) and will thus not be implemented in an already defined zone.

Since this zone needs to provide the key functions on which the whole security implementation will be based, a high level of trust is assigned to this zone. Other zones with a high confidentiality level, depend on this zone to handle essential information (such as key material) needed to maintain that confidentially. Therefore, a high level of confidentiality is also assigned to this zone[5]. Since this zone is not in scope of DP06 we added another scope owner as well. This choice was further motivated by the DP06 owner, confirming that the healthcare data exchange is based on the assumption that the registration/credential management can be carried out securely (and possibly by a different party).

[5] Note that here is no processing of the healthcare data itself in this zone. But this zone is the key in providing key material needed to secure the data communication between the cloud and the vehicle (gateway).

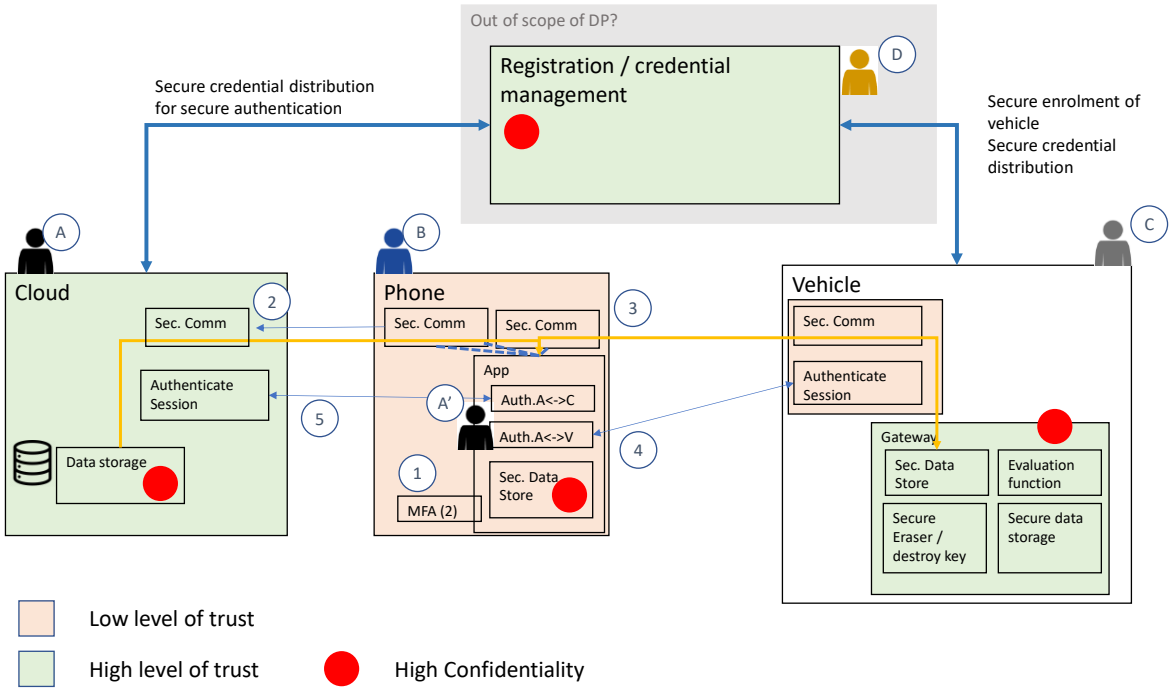


Figure 26. Additional zone added

Taking an objective of the registration function (of which D is the zone owner), which is to register vehicles credentials in such a way that the Healthcare cloud can authenticate an individual vehicle. This forms the basis to establish a secure channel between the cloud and the vehicle, while using the less trusted secure channels setup by the phone. In this case we assume that the registration function generates the key material needed for the authentication (cloud – vehicle). Outside of the scope of this analysis is the follow-up question, via which secure communication channels should this registration and credential distribution be implemented? These channels should at least have the same trust level requirement to preserve the required confidentiality level of the healthcare data.

Evaluation of Security Zones

By decomposing the functionality into different security zones, adding zone owners and trust levels to the different zones, helps to identify potential weaknesses in the security design in an early stage. It reveals if the already identified security mechanisms for a design are sufficient for the variations in ownership and trust between the various functions in a design. Adding a zone owner also supports in an early stage the dialogue with the business that needs to make decisions on how to implement the design pattern such that it fits their business model (i.e. who will provide the registration and credential management function?).

In its current state the analysis pattern mainly supports the dialogue between IT security functionality and the organisational aspects of IT security risk management. Further development will be in incorporating security requirements in more detail for each zone and the secure interaction between the zones for specific technologies in order to aide security engineers in selecting technological solutions fit for the specific technical security requirements.

Related work

The work is related to

- ISO/IEC 62443-3-2- Industrial communication networks- Security for industrial automation and control systems- Part 3-2: Security risk assessment and system design.
- 2nd draft of the NIST Special Publication 800-207, Zero Trust Architecture, February 2020.

Conclusion

Applying the Analysis Pattern to the design pattern, resulted in the following suggested improvements to the initial design pattern:

- In order to assure privacy under the assumption that the phone is less trusted, it is suggested to add an additional authentication / authorization function to counter the limited security functions offered by the Bluetooth facilities offered by the phone. This suggestion was adapted by the DP owner and as a consequence, DP06 will be refined such that the phone will receive only encrypted data which cannot be decrypted by the phone itself but by the vehicle. In other words, the healthcare data payload from the cloud towards the smartphone will be exchanged not only over a TLS channel but also additionally encrypted in a way which does not allow decryption by smartphone.
- Additionally, include requirements on security functions within the vehicle related to integrity and confidentiality protection of health care related user data stored in the vehicle. Although there are already requirements for security functions within the vehicle mentioned in the first version of DP06 (Marko, D3.5 Initial design patterns for common technology elements, 2019) the content relating to the registration phase and in-vehicle systems will be formulated by the DP owner in a more structured and clear way.
- By using the structured approach, these suggestions that can be applied in the design pattern, can be traced to the high-level security requirements and principles set out as a starting point for this design pattern.

References

- Abdulkhaleq, A., Wagner, S., Lammering, D., Boehmert, H., & Blueher, P. (2017). Using STPA in Compliance with ISO 26262 for Developing a Safe Architecture for Fully Automated Vehicles. *University of Stuttgart; Continental*.
- Abuelsamid, S., Alexander, D., & Jerram, L. (2018, 1Q). *Navigant Research*. Retrieved from https://www.navigantresearch.com/research/navigant-research-leaderboard-automated-driving-vehicles
- Alnasser, A., Sun, H., & Jiang, J. (2019). *Cyber Security Challenges and Solutions for V2XCommunications: A Survey.* United Kingdom.
- Christian Herberet, et al. (2013). *E3.4.3.1 Dokumentation von Konzept und Evaluierung HW Unterstützung Virtualisierung I/O.*
- Ericsson, C. A. (2005). *Hazard Analysis Techniques for System Safety.* John Wiley & Sons.
- G.LEVESON, N., & P.THOMAS, J. (2018). *STPA handbook.*
- ISO 26262. (2018). Road vehicles – Functional safety. (2nd edition).
- ISO/PAS 21448. (2019). Road vehicles — Safety of the intended functionality. (ISO/PAS).
- Leveson, N. (2005). A new accident model for engineering safer systems. *Aeronautics and Astronautics Department, Massachusetts Institute of Technology*.
- Leveson, N. G. (2011). *Engineering a Safer World, Systems Thinking Applied to Safety.*
- Mallya, A. (2015). *Using STPA in an ISO 26262 compliant process.* Hamilton, Ontario: McMaster University.
- Marko, N. (2019). *D3.5 Initial design patterns for common technology elements.* SECREDAS.
- Marko, N. (2020). *D3.6 - Design pattern descriptions.* SECREDAS.
- Oscarsson, J. (2016). *Functional Safety in Co-operative Driving using Systems-Theoretic Process Analysis.* STOCKHOLM: KTH ROYAL INSTITUTE OF TECHNOLOGY.
- SAE. (2016). Cybersecurity Guidebook for Cyber-Physical Vehicle Systems (J3061 Ground Vehicle Standard). *SAE International*.
- SAE. (2016). *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles (J3016 Ground Vehicle Standard).* SAE International.
- Sanden, B. v., & Vasenev, A. (2019). *D2.7: "Reference Architecture 1st Version".* SECREDAS.
- Sassan Nakhdjavani, Jens Harnisch, et al. (2014 satus: draft). *E 4.4.6.1 Implementation of HW-Access Interfaces for Hypervisor and Drivers.*
- Schmittner, C., Ma, Z., & Pus, P. (2016). Limitation and Improvement of STPA-Secfor Safety and Security Co-analysis. In *Computer Safety, Reliability, and Security* . Springer International Publishing.
- Schnellbach, A. (2016). *Fail-operational automotive systems.* Graz: Graz University of Technology.
- Smulders, A. (2019). *D2.6 Security Model.* SECREDAS.
- Young, W., & Leveson, N. (2014-02-01). An integrated approach to safety and security based on systems theory. *Communications of the ACM, 57*(2), 31-35.

2.2.9 AP11: Tracing Security and Privacy Principles in Architectures

2.2.9.1 Brief Overview

Analysis Aim:	To explicate security and privacy principles in system architectures
Abstract:	Principles, as high-level statements, can be used to reason how to design and improve systems and products. Architectural views like privacy and security are not directly comparable. This Analysis Pattern makes explicit how privacy and security principles are linked to specific technologies and system components. The outcomes can be used in informed holistic discussions and reasoning on security approaches with stakeholders involved in the system development process.
Addressed Problem:	Complexity of modern high-tech systems pose challenges related to their design and analysis. Security and privacy are system-wide properties. Traceability of privacy and security principles can help to think about maturity of either existing or future system. A clear way to consider principles in high-tech systems is therefore desirable.

Stakeholder:	Integrators (OEMs as integrators of Tier 1 inputs; Tier 1 companies as integrators of Tier 2 inputs)
Categories and Scope:	
Scope:	Improvement-AP/Methodology-AP (select one)
Safety:	Indirect connection. Potentially, can be extended using FMEA.
Security:	Security principles linked to technologies linked to deployment.
Privacy:	Privacy principles linked to technologies linked to deployment.
Input:	
Description:	Lists of technology elements; System elements (e.g., actors from scenarios; nodes from data flow diagrams)
Example(s):	List of SECREDAS CTEs from D3.1; Elements from use case descriptions and use case diagrams from D1.1
Addressed Elements:	
Related DP:	DP06
Contributors:	
Responsible:	Alexandr Vasenev, TNO
List of Contributors:	Behnam Asadi Khashooei, TNO

2.2.9.2 Detailed Description

Introduction

We provide a holistic and systematic reasoning over the security and privacy concerns by addressing traceability of these concerns from stakeholders to the principles and further to the components. Furthermore, we illustrate these reasonings by analysing a specific DP from D3-4. This analysis was carried out in steps through a series of discussion with the developer of the DP. At each step, we provided results of our analysis to the developer of the DP, which helped him to further elaborate the DP and provided us valuable feedback on improvements of the AP.

Analysis of DP06

We based current AP on DP06 (Healthcare Data Exchange (Roche)) in D3-4. The possible improvements described in this report are in the following areas:

1. Identification of the main stakeholders, for example, identifying technician in garage.
2. Clarification of scope of a design pattern by defining system boundaries and prerequisites for the design pattern (registration phase).
3. Identification of Data architecture.
4. Clarification of Data Flow by identifying the data routing and their storage.

Developed solution

The main stakeholders and their relations with each other and also their security and privacy concerns were introduced in this analysis pattern (see Figure 27). This provided the first layer of the traceability from stakeholder concerns to the system parameters. For example, given that driver concerns regarding his personal data, the addition of technician in the garage to the stakeholders emphasized the importance of adding additional layer of protection to the data that will be stored in the car in order to prevent the leak of information when the car is in the garage for instance for a regular check-up.

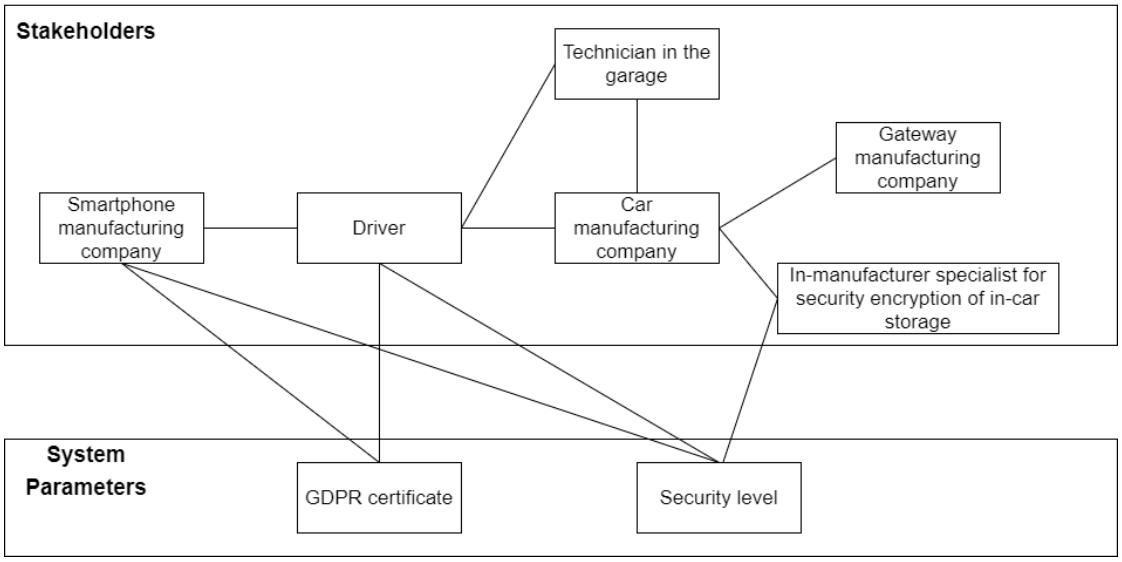


Figure 27. MBSA layers showing the connection in between main stakeholders and with security and privacy concerns

As a follow-up, we carried over the identification of the next layer of traceability from security and privacy principles to CTEs. We noticed that in DP06, the connection of CTEs and the security and privacy principles were not explicit. Therefore, we first identified a list with relevant principles and then made the corresponding mappings of these to relevant CTEs (see Table 17 and Figure 28).

Table 17: Mapping of principles to CTEs in DP06

N°.	Principles	Se	Pr	CTE	Reasoning
1	Don't trust infrastructure	x		2. Cryptographic Libraries	
2	Authenticate users and processes	x		8. Authentication and Authorization	
3	Authorize after you authenticate	x		8. Authentication and Authorization	
4	Always consider the users	x			Using personal phone for A&A to should be easy.
5	Asset protection and resilience	x		2. Cryptographic Libraries	Data are stored in phone and car. Potentially, those assets can be

					stolen. Stolen phone considerations were in the scope.
6	Security should be cost effective	x			Applicable, but not illustrate in the MBSA
7	Resistant in the garage	x		2. Cryptographic Libraries	The principle is relevant for crypto libraries because of double enveloping (logged data are not even accessible in the garage).
8	Don't use keys for different purposes	x		8. Authentication and Authorization	Different key for BLE and phone to cloud. Regular update of access key for cloud.
9	Leveraging existing components	x			Applicable, but not illustrate in the MBSA
10	Data minimization		x		Applicable, but not illustrate in the MBSA
11	Accuracy of personal data		x	2. Cryptographic Libraries	
12	Accountability		x		Applicable, but seen as a prerequisite for the DP. Accountability of practice needs to keep audit logs
13	Purpose Limitation		x	8. Authentication and Authorization	Authorization (specifically xacml pdp) ensures that only the relevant data is transferred and nothing more.
14	Fairness of Data Processing		x		Applicable, but not illustrate in the MBSA
15	Transparency of data processing				Applicable, but seen as a prerequisite for the DP
16	Storage Limitation		x	Secure Erasure	Can be connected to Secure Erasure, if such CTE is included to the DP
17	Integrity and Confidentiality		x	2. Cryptographic Libraries	The stored medical data are encrypted and decrypted on phone hence integrity and confidentiality.
18	Lawfulness of Data Processing		x		Applicable, but not illustrate in the MBSA

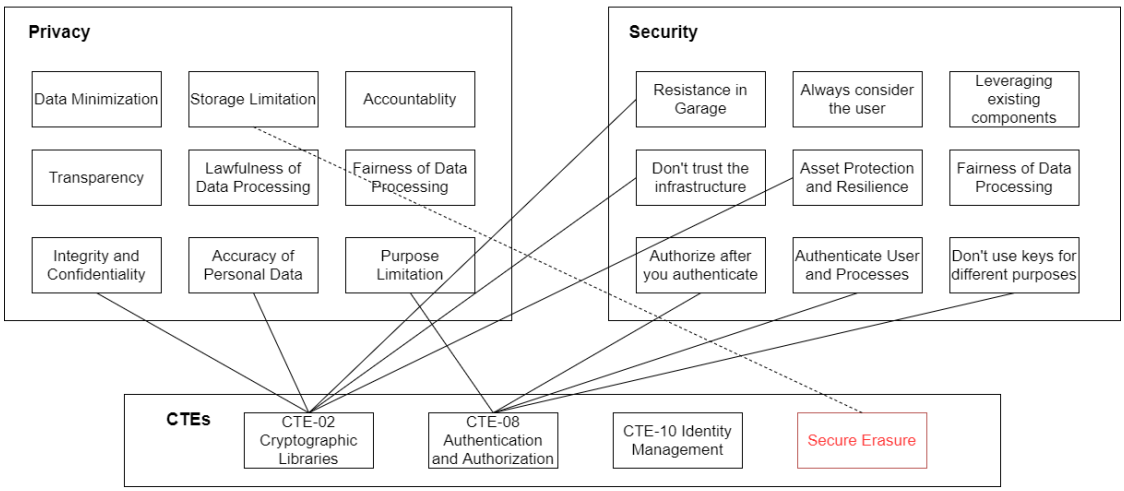


Figure 28. MBSA layers showing relation between principles and CTEs based on Table 17

Following the same line of thought to improve traceability, we noticed that although an initial mapping of required CTEs to the DP06 process can be inferred from Figure 3-9 in D3.4, it does not contain the explicit mapping of the CTEs to the components of DFD. This can lead to misunderstanding and misinterpretation due to hidden assumption and will require rework. By using the MBSA layers we explicate the relations as depicted in Figure 29. As can be observed in Figure 29, some data are temporary stored in the car storage this observation lead to proposition of a need for a new CTE to securely erase the stored data.

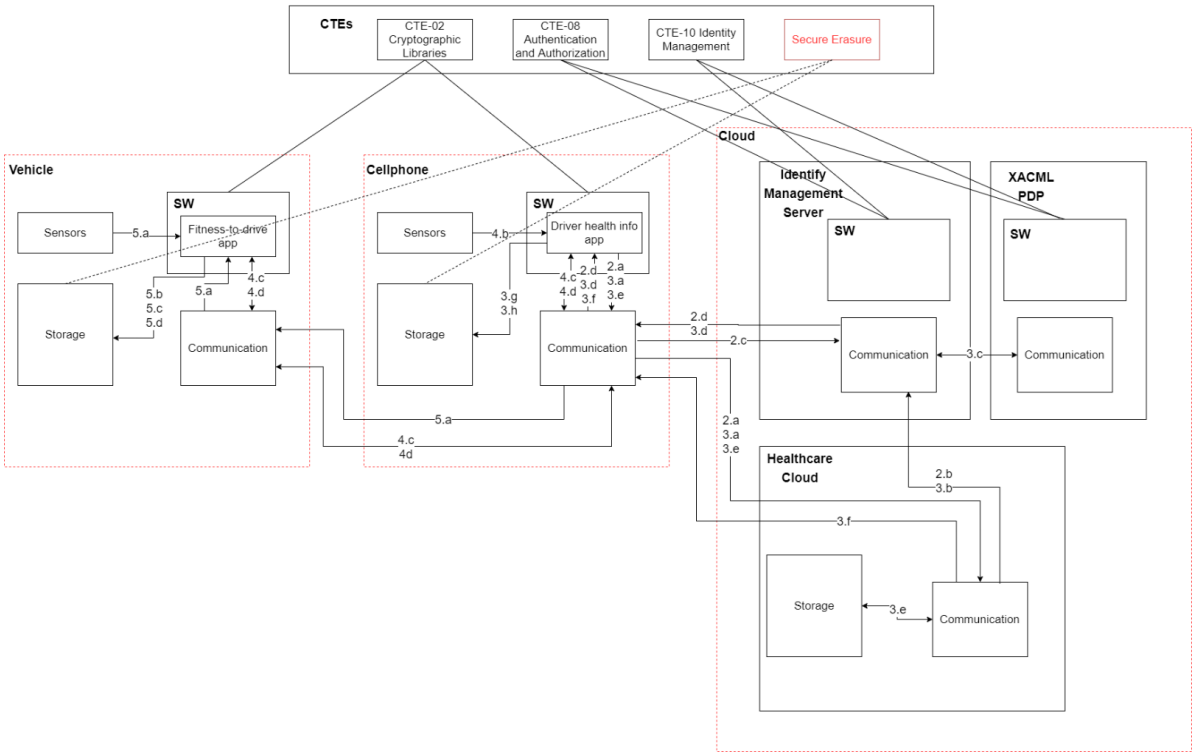


Figure 29. MBSA layer showing the latest DFD development in DP06 and the relation between related CTEs and DFD components

After building the layers of MBSA, we carried out the further discussion with the developer of DP06, using the identified layers of MBSA (Figure 27, Figure 28 and Figure 29) and identified different categories of data employed in the DP06. The first category of data is the personal healthcare data that is obtained from Healthcare cloud. The second category is the driver's healthcare data that is obtained from other sensors e.g. wearables, smartphone, etc. that the driver carries with himself/herself. Finally, the third category of the data is the logging data that is generated during the Authentication and Authorization process and during the transfer of the data to Fit-to-drive app. This identification further clarified the design of DP06 and also provide more insight to the designer for improvement of protection measures for data which for example resulted in the identification of the need for secure erasure of the log data as mentioned before. Furthermore, following our discussion with the designer, we identified that the data sent from phone to vehicle is not encrypted and this can cause exposure of data if the secure Bluetooth channel is compromised. A possible solution is to transmit encrypted data and consider decryption functionality in the vehicle system.

Evaluation of Improvement

Based on this analysis pattern, DP06 obtained a complete view on the main stakeholders, relevant principles, relevant CTEs, data flow path and data storage. This resulted in a holistic view of the design and improved the traceability from stakeholders' concerns to elements of design by bringing out the relation between different layers of MBSA regarding to DP06. Furthermore, this AP assisted in refinement of the scope of DP06 by identifying the boundaries of the system. Moreover, by the identifying the potentially problematic data flows, a simpler / more robust design has been obtained. Furthermore, through the development of this AP, a more specific and elaborated flow diagram was created which not only improved the elaboration of the results of the AP but also reduced the ambiguity of DP06. Finally, with the use of this AP, it is easier to handover the DP06 to the developers for the later stage of software development.

Related Work

T2.3 used the same logic to relate principles to CTEs and DFDs (see D2.7 appendixes). We also made use of DAARIUS (MBSA) [1-3] methodology for this analysis pattern.

Conclusion

In this analysis pattern, principles, as high-level statements, were used to reason about a design pattern. We made explicit how privacy and security principles are linked to specific technologies and system components. Furthermore, we related the concerns of main stakeholders of the design pattern to privacy and security principles. The outcomes can be used as an example of informed holistic discussions and reasoning on security approaches with stakeholders involved in the system development process. For future, work we aim to extend this approach to include other aspects of interest e.g. performance, cost, etc. into consideration.

References

[1] "DAARIUS methodology." [Online]. Available: https://www.esi.nl/solutions/daarius-methodology.dot. [Accessed: 20-Feb-2020].
[2] T. Bijlsma, W. T. Suermondt, and R. Doornbos, "A knowledge domain structure to enable system wide reasoning and decision making," Procedia Computer Science, vol. 153, pp. 285–293, 2019.
[3] T. Bijlsma, B. van der Sanden, Y. Li, R. Janssen, and R. Tinsel, "Decision support methodology for evolutionary embedded system design," in 2019 International Symposium on Systems Engineering (ISSE), 2019, pp. 1–8.

2.2.10 AP13: Embedding Identity and Access Management to the Event Log processes

The aim of this AP is to identify security and privacy threats as well as to present countermeasures which will give the confidence in using Identity and Access Management when handling Event Logs continuously.

2.2.10.1 Brief Overview

Analysis Aim:	In the current state identity management is focused for initialization and to create or revoke access automatically at the beginning and the end of the identity's lifecycle. For example, there is no option to automatically revoke specific access rights if unprohibited actions are maintained using impersonation attacks allowing the malicious user to gain access to the system. For such incidents, the system is required to mitigate the risks by responding automatically. Event management systems could collect data regarding the identity's behaviour and raise warning regarding the actions of the identity during the identity's lifecycle.
Abstract:	Identity and Access Management is used usually for initiating the identity and provide or revoke privileges and access rights to the assets. The assets usually include software/processes, access to files and folders and licence management, among others. Nowadays, most of the cyber security attacks are executed from insiders, meaning that malicious users spoof the identity of an employee or by using various methods such as phishing in order to gain privileges and access to the systems and assets. Furthermore, it is critical to revoke access of a past employee and to handle the identity during the whole identity's lifecycle and for a specific time afterwards as well. Modern devices and processes include a lot of information and it is possible to manage them as identities or sub-identities as well. For example, IoT devices or Cameras include unique identifiable information such as the physical network interface address (MAC address) and other details related to the operating system and version of the running services. Therefore, the management of the assets could be integrated into the management of the individuals' identities and correlate them to specific individuals. This AP focuses on analysing the impact to the security posture when including such processes and to address the potential privacy issues.
Addressed Problem:	In the current state Identity Management systems do not deal with security incidents and usually are not used for incident response processes. Furthermore, log files and events are handled usually inappropriately and do not meet to the current legislation related to privacy, transparency and confidentiality. Such issues are addressed in this AP to define the best practices for handling identities, network devices and to correlate the information used for incident response.
Stakeholder:	Vehicle Owners (i.e. drivers), Other (cooperating) vehicles, Roadside units, Developers of updates, In-manufacturer specialists responsible for Security and Privacy
Categories and Scope:	
Scope:	Improvement-AP
Safety:	This AP is about security and privacy.
Security:	Security measures are directly affected by this AP w.r.t. the incident response and we also pursue to enhance confidentiality.

Privacy:	Privacy issues are addressed and this AP is supposed to maintain the privacy rights of the individuals. Compliance processes, auditing and privacy regulation are taken into consideration.
Input:	
Description:	Specific unauthorised access, brute force attacks and privilege escalation to the assets of the network. Common access to folders and files and also uncommon behaviour in order to raise warning and events.
Example(s):	Individuals (e.g. Employees) usually have access to specific files and folders every day. When an uncommon access is identified, this should be reported. This also applies for granting authorization for processes within the system and network access (e.g. Access to sensors, cameras, IoT devices, printers etc.)
Addressed Elements:	
Related DP:	**DP02 - Automated Threat Detection and Vulnerability Management:** This design pattern supports the identity management to detect applicable threats and to archive past vulnerabilities, which were encountered during past incidents. It is also important to enhance security of newly connected devices which could bring in potential vulnerabilities. Therefore, the taxonomy of the vulnerabilities and threats will identify the potential security issues. **DP05 - Digital Identity Management and Smart Profiling:** This AP is mostly related to this design pattern but extends the approach to identify authorization processes and behaviour from network devices as well. **DP03 – Bastion Host:** Information from Bastion Host could be used to identify easier malicious identities. **DP28 – Anonymity Set:** This DP is important for enhancing privacy to the data collection and to enhance the privacy rights of the individuals.
Contributors:	
Responsible:	Beyond Vision and PDMFC: • BVD - stelios.karagiannis@beyond-vision.pt • PDMFC - secredas.team@pdmfc.com
List of Contributors:	stelios.karagiannis@beyond-vision.pt

2.2.10.2 Brief Overview

Introduction

In task 3.2, we introduced Identity Management and Access Management as a way of monitoring authorization processes, privileges and to identify incidents related to inconsistent behavior actions of the systems. The goals are to enhance the security aspects for interactively respond to incidents and to raise events and warnings. Privacy issues are addressed and the goal of this AP is to provide the details regarding best practices that must be kept in order to enhance privacy. The properties of the individuals are conceptualized from their everyday actions inside a network of devices. The devices themselves might also trigger specific interactions to the system (e.g. sending GPS data or uploading other information

interactively). Since the modern world includes more and more IoT devices the ability to manage all the assets and processes as individual identities is important.

This AP initiates and revokes privileges and licenses as well as monitors the common and uncommon actions of the individual identities. When an unwanted action is considered, the identity is matched with a warning level which if gets a high value then the incident will be reported. The identity management is enabled not only for individuals but for network and IoT devices as well as sensors and other devices having a network connection. For deploying the monitor process an agent is installed on the device; however agentless monitoring will be considered as well. There are many cases where for example a cluster of devices could be handled as a unified group monitoring only the gateway. Using agents to monitor only the gateways could also enhance the privacy attributes (e.g. data minimization, unlikability), since not the actual devices or sensors are monitored. Not all infrastructures and organizations are governed by the same principles and the regulations could differ according their infrastructure and internal compliance rules.

More importantly the interconnected devices are possible to be correlated to specific individuals which will eventually raise a lot of privacy issues. Through this approach these issues are identified, and our proposal includes the integration of the extracted events from the behaviour and lifetime of network devices which match to specific identities and individuals.

Example: Individuals are connected using specific network devices such as smartphones, laptops, personal computers and often include other devices such as smartwatches and other IoT devices as well. It is important for security to identify and authenticate every access of the individuals to keep logs and to prohibit any unprivileged action.

In this AP, we use an existing approach to manage identities and we consider extending the approach to manage other entities as well as to monitoring processes such as authorization to processes and access to the assets. For applying this pattern, the integration of the event management (SIEM) and of the identity management is required. For the SIEM to capture the inconsistencies regarding the entity's behavior other design patterns are required as well (Anomaly Detection). The main challenges on that is the orchestration of the processes and the required interactions between.

A major concern is also the size of data and the resources required for maintaining such process. For being able to identity the impact of this process we decided to analyze the security impact of the proposed analysis pattern and to enable privacy enhancing technologies, not only for protecting personal data but to minimize the total data size for performance purposes as well. Privacy and security requirements include that data must be protected and it is important to enhance privacy, especially when workforce,

clients, or end-users are involved. This includes the data minimization, data purpose for the log files and other analysis.

Analysis of DP05 - Digital Identity Management and Smart Profiling

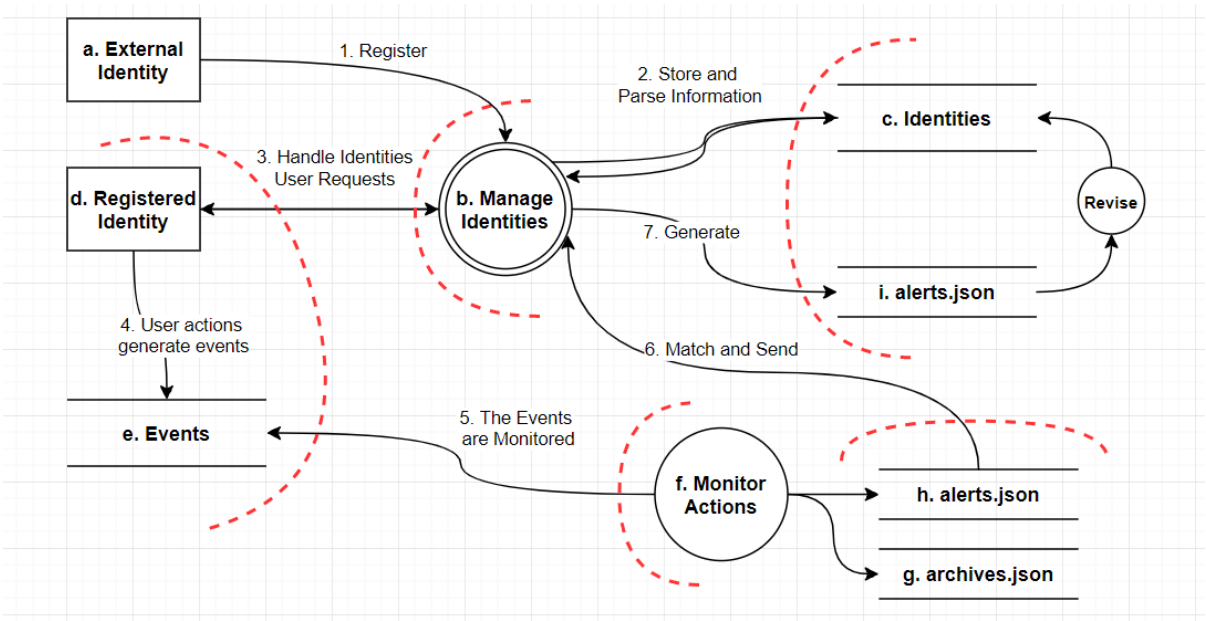


Figure 30. Threat model for Identity and Access Management

The risks for this design pattern are mostly related to privacy issues since a lot of information is collected regarding the behavior of every entity. The log files and the stored events are very important for maintaining security since the events are fingerprints of every action and process that is happening. Specific processes related to privacy are important to consider, such as data minimization and storage limitation. Another important aspect is that this data must be stored for specific time periods and has to comply with the policies of the organization. To be able to identify the impact of this pattern to the security and privacy, we define the scope from 3 different angles, namely: a. impact of the pattern on internal workforce, b. impact of the pattern on the internal infrastructure/organisation and c. impact of the pattern on clients or users which do not fall under internal policies. The methodology we follow, is to include the *STRIDE* threat model. The solution will be applied to 3 different cases:

1. The first one relates to the internal workforce and actions which are established inside the organization/infrastructure.
2. The second one is related to guest identities, meaning the clients that use the system/the network after their registration.

3. The third case is when the AP is applied directly to sensors, IoT devices and identities which maintain their processes automatically.

The generic threats and security issues are provided with the table below. The elements were identified in a simple way to understand the concept and to identify the potential threats from using top to bottom approach. In the later part of this AP the threats will be further identified using threat tracking tables and by identification of threat vectors using existing taxonomies such as that of OWASP[6].

#	Elements	Interactions	S	T	R	I	D	E
1	External Entity	Register	✓	✓		✓		✓
2	Manage Identities	Store and Parse Information	✓	✓	✓	✓	✓	✓
3	Manage Identities	Generate alerts		✓	✓	✓		
4	Registered Identity	Handle Identities and Execute Requests	✓	✓			✓	✓
5	Registered Identity	Execute actions and generate events	✓	✓			✓	✓
6	Monitor Actions	Monitor Events from Identities		✓	✓	✓		
7	Manage Identities	Match and Receive alerts	✓	✓	✓	✓		

Developed Solution

Identity and access management systems [5, 7 ,10] are responsible for handling the identities of an organization, meaning the registered users and their access to specific services and assets. Identity management systems certify and revoke access to multiple systems and services by providing an easy way to handle all the relevant information. As a result, the identities are handled easier and the privileges, licenses and access details are easily maintained. An important aspect is to handle identities after these unregister from the system. For example, when a new employee registers to an organization, a set of licenses, credentials and access to services are provided. This could include licenses related to software components and access to active and shared directories, among others. Furthermore, an email is created, and credentials are provided asking to replace these with new credentials as soon possible.

Identity managements are mostly used during registration; however, the identity management can be used also during the identity's lifecycle, monitoring any unwanted events. Sometimes these are used during the lifetime (e.g. for renewing access credentials for security purposes) and during the de-

[6] https://owasp.org/www-community/attacks/

registration. This AP addresses the importance of using the Identity and Access Management systems during the lifetime for enhancing the security posture and for maintaining privacy rights and policies accordingly.

Evaluation of improvement

We emphasize that the focus of this analysis pattern is to continuously monitor the identities and the access to assets and processes of the system. Not only this, but the privacy policies and rights are maintained during the identity's lifecycle. For enabling such processes, it is important to also include technologies such as Intrusion Detection Systems (IDs), system audit tools and Security Incident and Event Management Systems (SIEMs) [1, 2, 3]. Such systems include security rules which trigger events when any of the specified rule is compromised. The identities which violate any of the specified rules are classified. The whole process could be integrated in one solution, however, in our approach the process above will be divided in 3 different components.

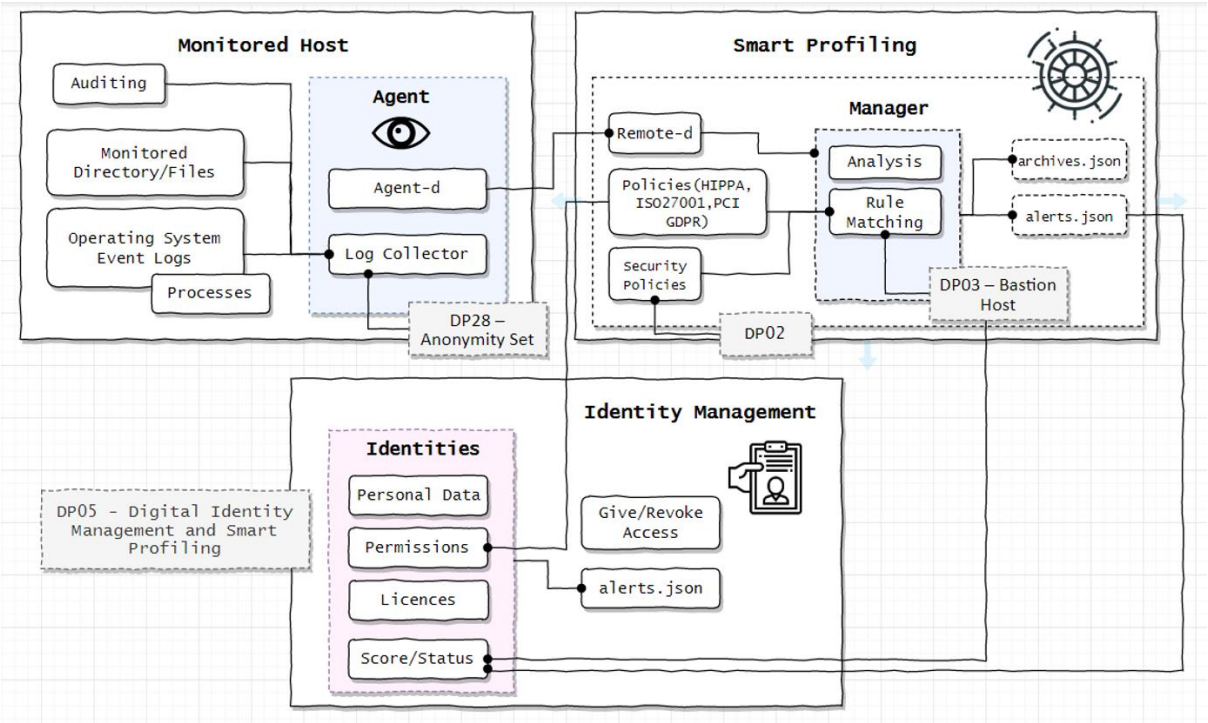


Figure 31. Developed Solution (Block Diagram)

The first component (Monitored Host) includes the deployment of an agent which is a software/service component that will capture every action (by using the log collector), authorization and access to the assets. The auditing takes place on each agent and logs are collected accordingly. The logs are sent to the

manager where the compliance rules are kept. The manager matches the logs and can trigger events for each rule or policy violation. Specific filters on the agent could be applied to focus the monitoring or to customize the collected events. The analysis will match the rules and provide alerts if something suspicious happens. The overall procedure is extracted and the log files can be maintained for a specific time to increase transparency. Procedures with large impact on privacy are maintained mostly on the agent side, so that minimal data is provided to the log collector. Furthermore, if the agent captures a cluster of sensors or systems which provide critical data, privacy mechanisms such as k-anonymity [13] are enabled to enhance the attribute of unlikability when required. The identity management holds every detail and tag the identities accordingly to the provided alerts from the smart profiling process.

Related Work

Related work includes the addressed issues by other researchers [11] who focus mostly on inappropriate ways for managing and handling digital identities. The issues that arise are related to inappropriate control and insufficient protection of digital identities from identity theft. Not only this, but their research [11], discusses the aspects of usability and privacy in the online identity management systems. Therefore, their proposal focuses on using user-side technologies with aim to better assist the management of identities.
Furthermore, in their research [12], Wohlgemuth and Muller propose a generic privacy-preserving protocol for sharing attributes and credentials to others, extending the current identity management systems. Other researchers [4, 5, 6, 7, 8, 9] also address issues related to privacy when handling identities. However, most of the research focus on specific perspectives and mostly on communication protocols for enhancing privacy. Our approach considers these aspects; however, our focus is mostly to provide the basis for implementing a service where privacy will be handled accordingly using some of the proposed privacy-preservation approaches. Therefore, we consider analyzing our proposal regarding the privacy issues that arise and to handle them properly. Finally, our approach defines the procedures for conducting real time monitoring of the identities which might rise a lot of privacy issues and includes great challenges to overcome. Some of these challenges include performance and the ability to handle the privacy properties properly, while conducting the required monitoring for securing the infrastructure posture as well.

Future Work

Future work includes the implementation of the proposal and to further describe the privacy and security rules which must apply and match in each case. The main goal is to provide a service where identities will

be handled securely, the infrastructure will be monitored for any irregular or uncommon access, while the privacy of the identities will be maintained giving the opportunity for the security and privacy policies to be applied directly.

Furthermore, regarding this AP more extended work is required regarding specific application cases such as the analysis pattern applied to entities related to the Workforce, the Analysis Pattern regarding registered users and finally the Analysis Pattern applied to entities related to the Infrastructure such as registered IoT, sensors, internal network and autonomous devices. The analysis for privacy will be developed using privacy impact assessment when this analysis pattern is deployed. Towards this direction specific guidelines will be created for each of the applied cases.

Finally, we intend to provide an/some example/s with threat tracking tables which address specific threats and to use existing threat taxonomies to better address the issues related to security.

Conclusion

This AP provides the baseline proposal for integrating the identity management to log files which will be extracted using agents including information related to the access to folder/files, authorization processes and access to the assets of a system. In our approach we handle every network device, user, and asset as identity, and we try to address the security and privacy aspects for applying the proposed solution. The main goal is to provide real time monitoring of the identities to identity suspicious or uncommon behavior and actions. Using the identity and access management system together with a security monitoring solution for threat detection, integrity monitoring, incident response and compliance, the purpose is to extend and to review the DP of Identity and Access Management in order to address the privacy and security issues together with the benefits which such approach provides.

References

[1] Bedwell, P. (2014). Finding a new approach to SIEM to suit the SME environment. Network Security, 2014(7), 12-16.

[2] Menges, F., Böhm, F., Vielberth, M., Puchta, A., Taubmann, B., Rakotondravony, N., & Latzo, T. (2018). Introducing DINGfest: An architecture for next generation SIEM systems.

[3] Montesino, R., Fenz, S., & Baluja, W. (2012). SIEM-based framework for security controls automation. Information Management & Computer Security, 20(4), 248-263.

[4] Pashalidis, A., & Mitchell, C. J. (2012). Privacy in identity and access management systems. In Digital Identity and Access Management: Technologies and Frameworks (pp. 316-328). IGI Global.

[5] Thakur, M. A., & Gaikwad, R. (2015, January). User identity and Access Management trends in IT infrastructure-an overview. In 2015 International Conference on Pervasive Computing (ICPC) (pp. 1-4). IEEE.

[6] Hansen, M., Berlich, P., Camenisch, J., Clauß, S., Pfitzmann, A., & Waidner, M. (2004). Privacy-enhancing identity management. Information security technical report, 9(1), 35-44.

[7] Gaedke, M., Meinecke, J., & Nussbaumer, M. (2005, May). A modeling approach to federated identity and access management. In Special interest tracks and posters of the 14th international conference on World Wide Web (pp. 1156-1157).

[8] Tsukada, Y., Mano, K., Sakurada, H., & Kawabe, Y. (2009, August). Anonymity, privacy, onymity, and identity: A modal logic approach. In 2009 International Conference on Computational Science and Engineering (Vol. 3, pp. 42-51). IEEE.

[9] Martucci, L. A., Ries, S., & Mühlhäser, M. (2011). Sybil-free pseudonyms, privacy and trust: Identity management in the internet of services. Information and Media Technologies, 6(3), 1001-1015.
[10] Tuecke, S., Ananthakrishnan, R., Chard, K., Lidman, M., McCollam, B., Rosen, S., & Foster, I. (2016, October). Globus Auth: A research identity and access management platform. In 2016 IEEE 12th International Conference on e-Science (e-Science) (pp. 203-212). IEEE.
[11] Jøsang, A., Zomai, M. A., & Suriadi, S. (2007, January). Usability and privacy in identity management architectures. In Proceedings of the fifth Australasian symposium on ACSW frontiers-Volume 68 (pp. 143-152). Australian Computer Society, Inc..
[12] Wohlgemuth, S., & Müller, G. (2006, June). Privacy with delegation of rights by identity management. In International Conference on Emerging Trends in Information and Communication Security (pp. 175-190). Springer, Berlin, Heidelberg.
[13] Sweeney, L. (2002). k-anonymity: A model for protecting privacy. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 10(05), 557-570.

3 Conclusion

In this deliverable, we illustrate the relation between T3.2 and T3.3 and show how the results from T3.3 will be used to improve the DPs of T3.2. In the first iteration of this deliverable (D 3.7: *Security analysis report v1*), we included an initial list of APs provided by the different task partners along with their connection to specific DPs. In this final version of the deliverable, we elaborate on the developed APs. Thereby, we illustrate how new developed analysis methods (from MA) can be applied, using DPs from T3.2 as reference. We also show which benefits they offer by combining complementary aspects such as security, safety and privacy. Based on the analysis results, countermeasures have been identified and realized (IA). These improvements will be utilized to enhance the DPs (and their description) in deliverable D3.6 of T3.2.

Tables

Table 1: Overview of APs and related DPs 11
Table 2: Template for the AP overview. 12
Table 3 Security Analysis of DP25 ConSerts [10] 19
Table 4: Safety contents for safety-classification of components 58
Table 5: Remote communication network requirements (e.g. vehicle V2X infrastructure) 74
Table 6: Remote communication system module requirements (e.g. vehicle V2X module) 75
Table 7: Comparison between requirements and CTEs (remote communication network) 77
Table 8: Comparison between requirements and CTEs (remote communication module) 77
Table 9: Improvement of the design pattern (remote communication network) 78
Table 10: Improvement of the design pattern (remote communication module) 78
Table 11. An overview of the pre-injection analysis techniques that have been defined and detailed in the past 97
Table 12: List of Business Assets, DP Derived Identity 104
Table 13: List of Supporting Assets, DP Derived Identity 104
Table 14: Link Matrix between Business Assets and Supporting Assets, DP Derived Identity 105
Table 15: Expressed Risks, DP Derived Identity 107
Table 16: List of controls for improvement of DP Derived Identity 108
Table 17: Mapping of principles to CTEs in DP06 120

References

- SECREDAS Project Partners. (2020). *D3.2 (Common Technology Change Report v1), SECREDAS Project Deliverable.*
- SECREDAS Project Partners. (2020). *D.3.6 (Design Pattern Description v1), SECREDAS Project Deliverable.*

www.secredas.eu
mail@secredas.eu
Social media @secredas_eu

SECREDAS has received funding from the Electronic Component Systems for European Leadership Joint Undertaking under grant agreement nr.783119. This Joint Undertaking receives support from the European Union's Horizon 2020 research and innovation programme and Austria, Belgium, Czech Republic, Germany, Finland, Hungary, Italy, The Netherlands, Poland, Romania, Sweden and Tunis