Report Type	Deliverable D1.1
Report Name	A State-of-the-Art Investigation
Dissemination Level	Public
Status	Final
Version Number	1.0
Date of Preparation	16th October 2019

Executive Summary

This deliverable (D1.1 State-of-the-Art Investigation) presents the results and achievements of sub-work package WP1.1 (State of the art) of the CyReV project. The goal of this deliverable is to provide insight into the state of current research frontiers in security and resiliency for vehicles and vehicular security. It also serves as an introduction to the issues faced in the automotive domain with an extensive reference list for more detailed studies. Finally, it can be used as background on which to base further investigations. The FFI HOLIstic Approach to Improve Data SECurity (HoliSec) project produced the deliverable "D1.2 State of the art", CyReV uses this deliverable as a baseline and extends the current state of the art in regards to resiliency.

The report includes discussions about hardware and operating system level security, internal and external communication security, intrusion detection systems, secure software development, and related research projects. Many topics are discussed very briefly, since an exhaustive treatment of each would certainly be out of the scope of this deliverable.

Contributors

Editor(s)	Affiliation	Email
Thomas Rosenstatter	Chalmers	thomas.rosenstatter@chalmers.se
Katja Tuma	Gothenburg University | Chalmers	katjat@chalmers.se
Contributor(s)	**Affiliation**	**Email**
Mafijul Islam	ATR, GTT, Volvo AB	mafijul.islam@volvo.com
Pierre Kleberger	Chalmers	pierre.kleberger@chalmers.se
Aljoscha Lautenbach	Chalmers	aljoscha@chalmers.se
Boel Nelson	Chalmers	boeln@chalmers.se
Nasser Nowdehi	Volvo Car Corporation	nasser.nowdehi@volvocars.com
Tomas Olovsson	Chalmers	tomas.olovsson@chalmers.se
Riccardo Scandariato	Gothenburg University | Chalmers	riccardo.scandariato@chalmers.se

CyReV Consortium

Cyber Resilience for Vehicles – CyReV (phase 1)
Dnr 2018-05013

FORDONSSTRATEGISK
FORSKNING OCH INNOVATION

Revision Chart and History Log

Version	Date	Contributor	Description
0.5	2019-08-28	Thomas Rosenstatter	Initial version.
0.9	2019-09-18	Thomas Rosenstatter	Version for consortium feedback.
1.0	2019-10-16	K. Tuma, T. Rosenstatter	Incorporated consortium feedback.

Contents

List of Figures

List of Tables

Acronyms

AMP Arbitration on Message Priority.

ASLR Address Space Layout Randomization.

AVB Audio Video Bridging.

CAN Controller Area Network.

CAN-FD Controller Area Network flexible data-rate.

CARONTE Creating an Agenda for Research on Transportation sEcurity.

CD Collision Detection.

CPS Cyber Physical System.

CPU Central Processing Unit.

CRC Cyclic Redundancy Check.

CRL Certificate Revocation List.

CSMA Carrier Sense Multiple Access.

DoS Denial of Service.

DSRC Dedicated Short Range Communication.

E/E Electrical and/or Electronic.

ECU Electronic Control Unit.

ETSI European Telecommunications Standards Institute.

GPS Global Positioning System.

HEAVENS HEAling Vulnerabilities to ENhance Software Security and Safety.

HoliSec HOLIstic Approach to Improve Data SECurity.

HSM Hardware Security Module.

IDS Intrusion Detection System.

IEEE Institute of Electrical and Electronics Engineers.

IPS Intrusion Prevention System.

ITS Intelligent Transportation System.

LIN Local Interconnect Network.

MOST Media Oriented Systems Transport.

NFC Near Field Communication.

NHTSA National Highway Traffic Safety Administration.

OBD-II On-Board Diagnostics II.

OS Operating System.

RDS Radio Data System.

RFID Radio-frequency Identification.

RSU Road-side Unit.

SoC System on a Chip.

TDMA Time Division Multiple Access.

TTE Time-Triggered Ethernet.

V2I Vehicle-to-Infrastructure.

V2M Vehicle-to-Mobile.

V2V Vehicle-to-Vehicle.

V2X Vehicle-to-X.

VANET Vehicle ad-hoc Network.

WLAN wireless LAN.

WSN Wireless Sensor Network.

Chapter 1

Introduction

The internet has already reached our vehicles, new services, e.g., Android Auto[1] and BMW ConnectedDrive[2] have been introduced and many more will come in the coming years. Some services target drivers and passengers such as navigation and driver assistance systems, and other focus on the vehicle itself such as remote diagnostics and remote software updates. Most vehicle manufacturers have plans to offer a fairly large number of services and we now face the challenge to implement new functionality without sacrificing traffic safety. The vehicle is a complex safety-critical system with components that must function at all times, and security problems should never result in safety problems or in an immediate halt of all systems. It is from utmost importance that the vehicle is resilient – it is able to keep the system in a stable state even when a malicious attack or behaviour was detected.

Vehicles currently have an internal network consisting of more than 100 microcontrollers or Electronic Control Units (ECUs). The internal network is of the size of a small company and internal security is currently largely missing. The software in modern vehicles contains tens of millions of lines of code with a total size of more than 100 MBytes [1]. This vehicle is now gradually going to be connected to the infrastructure around it, i.e. to Road-side Units (RSUs), to other vehicles and to the internet. Therefore, it is imperative that security is properly addressed before these new services are introduced. Moreover, it might be necessary to redesign current architectures in order to cope with the demands on safety, security and resilience.

1.1 The connected Vehicle

Communication between vehicles and the outside world will in almost all cases be wireless. Exceptions may be found in repair shops and when vehicles are parked. It is possible to access the internal network by connecting a device directly to the internal busses of a vehicle, for example in a repair shop to diagnose problems and to update the

[1] https://www.android.com/auto/
[2] https://www.bmwusa.com/explore/connecteddrive.html

software, but also by vehicle owners and other people in order to “enhance” or change the vehicle’s functionality. With a sound security design, it should not matter whether the communication is wired or wireless. However, physical modification of ECUs and the possibility to physically attach devices to the internal network must be paid special attention to, as it can not be ruled out that the vehicle owner modifies or adds equipment to the vehicle that interferes with its normal functionality.

Vehicular Communication is divided into two or sometimes three categories, collectively called Vehicle-to-X (V2X):

- *Vehicle-to-Infrastructure (V2I)* communication: New services will be implemented that communicate with roadside equipment (i.e. new technical infrastructure). Most of those serices are related to safety, for example to alert drivers about traffic lights, speed limits or to inform about road works ahead.

- *Vehicle-to-Vehicle (V2V)* communication: This is the area most researchers and application developers focus on. Typical services are anti-collision systems such as early break warnings from other vehicles, information about emergency vehicles approaching, synchronized lane change support, traffic jam ahead warnings, and services facilitating the driving experience such as vehicle platooning.

- *Vehicle-to-Mobile (V2M)* communication: Communication with a mobile device, for example via Bluetooth or Near Field Communication (NFC). In the rest of this document, we will not include V2M communication in the V2X concept unless explicitly stated.

Different communication technologies are used for vehicular communication: WiFi (IEEE 802.11a,b,g or n) can be used to connect vehicles to conventional access points, for example to download multimedia contents when parked at home. Mobile phones offering Bluetooth connectivity for hands-free operations are already in place, often without considering that telephones also have a 4G/5G data connection to the internet and therefore may bridge the vehicle with the internet.

Currently the new telecommunication technology 5G is also, next to IEEE 801.11p, considered as a candidate for V2I and V2V communication. 5G may also provide an advantage in regards to security, as it has already some security measures in place. There are also many other devices communicating with the vehicle, e.g., wireless keys, Radio-frequency Identification (RFID) cards identifying drivers, radio communication for traffic information (RDS) and navigation (GPS).

Communication patterns and the actual technology used depend highly on the application, some are classical client-server applications where the vehicle connects to a server or a portal for provisioning of a specific service. Other services communicate with RSUs or are based on ad-hoc communication between different parties where short-lived Vehicle ad-hoc Networks (VANETs) are formed.

For some of the services it is essential that the parties are fully authenticated and their identities are known by all participants, and for other services privacy can be more important than being able to identify each other. A compromise may sometimes be acceptable, for example where vehicles can be anonymous to each other but RSUs are able to do proper authentication. Protection against non-repudiation and Sybil attacks (multiple identities) is important for some services. Even if the parties do not want to reveal their identities to each other, there should be some limitations to what damage they may cause, and it should always be possible for an authorized party or authority to reveal their identities.

1.2 New Applications and Services

There is a high demand for applications and services around the connected vehicle, and vehicle manufacturers have long lists of applications they would like to offer to their customers. There will likely be something similar to an "AppStore" in the vehicle where the owner, driver and passengers can choose to install both free applications and subscribe to different services. In addition, mobile phones and other hand-held devices will be seamlessly integrated into the vehicles, enhancing the driving experience.

Applications will be offered by many parties such as the vehicle manufacturer, government organizations, trusted third parties (cooperating with the manufacturers), and independent third-party application developers. Some applications may be mandatory and offer services from legal authorities, others are safety improving services (e.g. driver assistance/autopilot and accident reporting systems) and yet other are services the owner, driver and passengers would like to install.

It yet remains to define an architecture that allows third party applications to run in the vehicle environment in a safe and secure way. Many solutions such as certification, sandboxing and isolation have been proposed, but it will take many years before vehicle manufacturers can allow third parties to freely develop software for the vehicles and still be able to guarantee the safety of the vehicle.

The services can roughly be categorized as:

- *Services improving the driving experience and safety*: They give advice and notify drivers about events. Examples include vehicle platooning (cooperative driving) and collision avoidance systems that give early warnings and notifications to drivers. Vehicles talk to each other and are fully aware of other vehicles' plans and act accordingly, for example by notifying drivers to give way to emergency vehicles approaching at high speed.

- *Critical safety services*: For example emergency actions from the vehicle when a crash is impossible to avoid and to call for help when sensors detect that an accident has occurred [2]. Many of these services are active and the vehicle will take action and help the driver in difficult situations.

- *Traffic optimization*: Information is spread among vehicles about road conditions, congestion problems, accidents and overall traffic throughput.
- *Commercial services*: For instance automatic payments of parking, road tolls and taxes based on when and where a vehicle has been driven.
- *Subscriptions to improved vehicle functionality*: The vehicle may have more functionality than the owner has initially purchased; it may be possible to use the internet to purchase or for a limited time rent functionality such as *automatic parking support* without having to visit the vehicle dealership. Vehicles may also be remotely diagnosed and the vehicle manufacturers can offer updated services and patch software in the field, not only in the repair shops.
- *Services unique to the driver, not to the vehicle*: Insurances may in the future follow the driver and not necessarily be tied to a particular vehicle, as is already the case in the U.K. Drivers may subscribe to services that are available to them regardless of what vehicle they are driving.

The proposal of the U.S. Department of Transportation (DoT) highlights that national departments also see this promising technology (V2V communication) as an enhancement for traffic safety. On the 13^{th} December 2016 the DoT proposed a rule that all light-duty vehicles are required to have an active V2V and V2I communication module in order to increase traffic safety and prevent accidents due to the exchange of information between the vehicles and infrastructure [3].

1.3 Summary

The use of many different communication technologies, different types of communication requirements, real-time requirements, authentication services mixed with anonymity, etc., makes security work very challenging. In addition, we will soon have a large number of applications in our vehicles and several of them will be third party applications. All these changes and trends require a new design for vehicles that is considering resilience and robustness from the beginning of development.

It is obvious that all communicating systems must be protected against external threats (attackers), but there are also many different parties involved with the vehicle that do not necessarily trust each other. The driver may not always have the same interests as the owner of the vehicle, for example with respect to sharing information about where and how aggressively the vehicle has been driven. Furthermore, owners and drivers may not always be trusted by the vehicle manufacturers, since there may be optional services offered by the manufacturers that the owner must purchase or subscribe to, to be able to use. If there is an easy way to obtain such services for free, e.g. to patch the software in the vehicle, many owners will probably use this opportunity.

The remaining structure of the report is as follows. Chapter 2 provides an overview about the complexity of modern and future vehicles including the challenges in securing

them, which are discussed in more detail in later chapters. Chapter 3 discusses the various threats to a vehicle and its infrastructure: (1) common network security threats due to vulnerabilities in protocols; (2) automotive specific threats. This chapter further highlights the necessity for securing the vehicle. Several cases are shown where the vehicle could be hacked from a wide range (the internet), or a closer range. An overview of the different communication technologies internal as well as external is provided in Chapter 4. There are more than a billion vehicles on the roads and there are also many different configurations and versions of a single vehicle model. Given this variety and considering past attacks, it is evident that the deployment of intrusion detection systems (IDSs) is necessary for the automotive domain in order to react in a reasonable time to threats. Chapter 5 gives a brief overview of the two types of IDSs and discusses the use of so-called honeypots. Chapter 6 describes the different security features as well as threats on hardware and operating system level. Chapter 7 starts with a definition of resilience and resilient architectures and continues to explore proposed resilient architectures and ways to measure resilience. Chapter 8 is about the certification and evaluation of security functionality. Chapter 9 presents relevant research projects to the issues raised in this report. Finally, Chapter 10 closes with some final remarks.

Chapter 2

Complexity and Security Challenges

This chapter gives a brief overview of the specific challenges we face when working with security in the vehicular domain.

Security researchers and analysts often face the problem to reach the market in time, not only in the vehicular domain. This often leads to service implementations which solve security problems one at a time, while waiting for standards and established methods to emerge. For obvious reasons this approach will not be successful in the long run.

More general methods that can be applied to a large number of applications and use cases must be considered. One such example is the framework developed by the European EVITA project [4]. Standardization work to specify security frameworks for V2X services is also in progress. The European Telecommunications Standards Institute (ETSI) *Intelligent Transportation System (ITS) station* is an attempt to standardize V2X communication nodes [5] and the Institute of Electrical and Electronics Engineers (IEEE) standardized protocols for wireless communications (IEEE 1609 and IEEE 802.11p).

The HoliSec project[1] funded by VINNOVA was focussing on providing holistic security for automotive systems. One focus of this project was to review and analyse current threat analysis techniques [6, 7] and proposing a novel way of approaching threat analysis [8]. Moreover, this project was concentrating on the classification of security needs for the automotive domain [9] and how to perform a mapping from these security levels to specific security mechanisms [10].

Since vehicle manufacturers act on multinational markets where they have to comply with different legal requirements, there may also be conflicting requirements, e.g. the trade-off between privacy and traceability. General security solutions are needed which provide the possibility to adapt the functionality of a vehicle model depending on the target country.

[1] https://autosec.se/holisec-results/

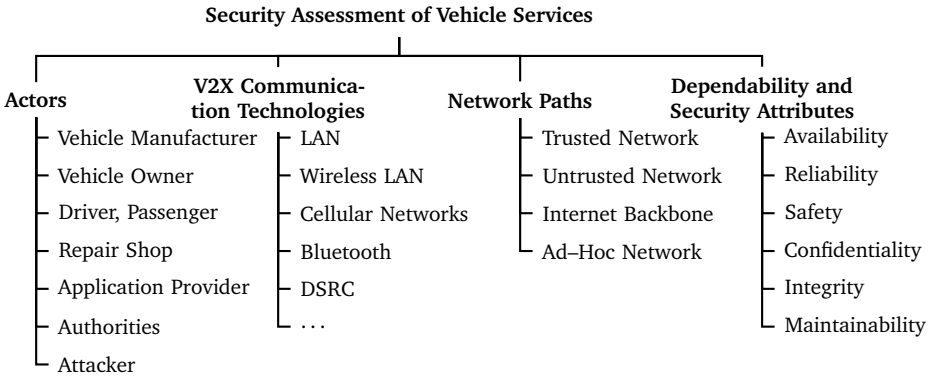

Figure 2.1: Security Assessment Tree

Further security complications arise from the vast number of services, communication technologies and protocols that must be supported. There are also real-time requirements and safety aspects that affect security and the functionality of the systems. In the following, we will show the complexity of the problem and highlight the challenges we face in each area.

2.1 Involved Parties

Many parties are involved during the lifetime of a vehicle, which leads to several trust and privacy issues. It is obvious that people with no legitimate access to a vehicle (i.e traditional attackers) must be considered in a threat model. However, serious security problems may also be caused by authorized people with access to the vehicle.

Trust is generally problematic in the vehicular environment. For example, the owner of a vehicle is not necessarily trusted by the vehicle manufacturer, the driver may not be fully trusted by the owner, and repair shops may not be fully authorized or trusted by the vehicle manufacturer or the vehicle owner (i.e. they should not have full access to all data and programs in the vehicle). There are also third party program developers who want to offer services which none of these parties can fully trust. All involved parties have different notions of what is important to address, complicating the security work even further.

The complexity of security problems can be seen in Figure 2.1 in which different parties, communication technologies, and security attributes that must be considered are listed [11]. A security assessment of the vehicle must consider all possible combinations of actors, communication technologies, paths and security attributes, a quite complex task. The interests of the identified parties are listed in the following paragraphs.

The manufacturer of the vehicle. They will continue to offer services to the vehicle during its full lifetime, not only in repair shops. Critical software updates have to be

applied more or less immediately, not two or three years after a vulnerability has been discovered and the vehicle eventually visits an authorized repair shop.

The owner of the vehicle. He/she should have access to most, but not all, functions in the vehicle. It should also be possible to delegate privileges to other users to configure some settings in the vehicle. The owner may change over time and transfers of ownership must be handled.

The driver(s) of the vehicle. Many services will in the future be based on who is driving the vehicle.

The passengers. Passengers may need to be authenticated to access various applications and (remote) services, for example from e-commerce servers. Some services may be available in the vehicle based on a passenger's identity, for example multimedia contents or navigational software.

Authorized technicians/repair shops. Technicians should have access to most of the vehicle's data, but not necessarily to private information related to the owner, driver or passengers. The technicians should not be able to use their identities in, for example commercial activities, nor to access personal information such as driver's behaviour, Global Positioning System (GPS) logs containing vehicle location history, or to install non-authorized software.

Third party application suppliers. Software developers may be granted access to certain parts of the vehicle network, but not to all. This may be a rather complex issue to solve, similar to privilege levels offered for example for Android devices.

Trusted authorities. Authorities may need access to certain data, for example after an accident or a crash or in real-time to track vehicles for various reasons; road tolls, parking fees, etc..

Privacy issues are also important. Most vehicle owners and drivers do not want to reveal their identities to everyone. However, being completely anonymous opens up for attacks against many services. Sybil attacks, i.e. to be able to use multiple identities could be useful for example by an attacker to spread false information about congestion in order to have the road for him/herself. To mitigate this problem, the use of pseudonyms has been suggested. When needed, for example for legal reasons, a trusted party will be able to reveal the real identity behind a pseudonym.

2.2 External Communication

Communication with the outside world can be done using many different technologies. Examples include Dedicated Short Range Communication (DSRC) (e.g. IEEE 802.11p), normal wireless LAN (WLAN) communication (802.11a,b,g,n), and cellular communication (GSM, GPRS, 3G, 4G, 5G). These technologies are often used by traditional client-server applications which possibly connect to the internet, Bluetooth devices, or NFC devices. The large number of technologies complicates security work since several communication stacks necessarily lead to a bigger attack surface. It may also be hard to know which traffic should be allowed on what interface.

Through gateway ECUs which connect several busses, external devices are also able to send messages to the internal network, for that reason gateway ECUs need to be especially well secured against intrusions.

Section 4.2 provides a more detailed description including a brief description of the different communication technologies and related protocols for external communication.

2.3 The In-vehicle Network

The in-vehicle network spans the whole vehicle and consists of networks of different bus-system technologies, depending on brand and vehicle type: *Controller Area Network (CAN)*, *Local Interconnect Network (LIN)*, *Media Oriented Systems Transport (MOST)*, *FlexRay* and *Automotive Ethernet*. These networks are connected to each other through special *gateway ECUs*, although the internal architecture and number of gateways also vary depending on brand and vehicle type. The Evita project [4] has defined a “use-case” architecture model that describes a possible configuration of a vehicular network.

Traditional security mechanisms cannot directly be used to secure in-vehicle networks due to limitations and constraints specific to vehicles and the vehicle industry:

- Resource constraints of the ECUs, i.e. limited memory and processing power. Complex cryptographic operations or storage of large amounts of data are not possible.
- Cost efficiency. The cost of a typical ECU is in the order of €1 and even a marginal increase of costs is problematic. If the cost of each ECU in a vehicle with 100 ECUs increases by €1, the additional costs for a vehicle manufacturer producing 1,000,000 vehicles per year would be €100,000,000.
- Vehicle Lifetime. It may be in use for at least 10-15 years, preferably even longer. A complicating factor is that the design phase of vehicles is usually very long; vehicles to be released in 10 years are already on the drawing board and many design decisions are already made. Security designs must therefore be modular and sound to survive for such a long time.

Some specialized ECUs, like *gateway ECUs*, may be equipped with additional functionality such as hardware support for encryption, but due to cost constraints most ECUs should preferably consist of standard hardware. This places many constraints on the security solutions that can be implemented in a vehicle.

Some vendors are also considering to replace some of the internal wired networks with wireless networks, more specifically Wireless Sensor Networks (WSNs). The main reasons for this consideration are cost and weight savings, which can also translate to less fuel consumption. WSNs present a whole new range of potential security problems. In 2011, SysSec published a state of the art report on sensor network security, which provides a good overview of the topic [12]. Lee, Tsai and Tonguz also provide a good overview of WSNs in vehicle networks [13].

2.4 Hardware and OS Security

Due to cost and power constraints, the ECUs in a vehicle are limited in terms of processing power and memory capacity, and all security measures must take this into account. Usually cheap "off-the-shelf" hardware is preferred for this reason, which may not have hardware support for cryptographic functions. As security is becoming even more important in the embedded domain, units with cryptographic hardware support will probably be deployed increasingly.

Modern Central Processing Units (CPUs) have the ability to distinguish between executable and non-executable memory spaces, which makes certain kinds of attacks harder to perform (e.g. the traditional buffer overflow attack).

Another CPU feature that is generally seen to be beneficial for security is hardware virtualization support. Virtualization can be used to create process instances which are isolated from one another, and which also have no direct access to real hardware resources. This is also an upcoming trend in embedded systems, but it is currently not used.

Physically securing ECUs against tampering is another challenge. This is called tamper resistance and is further discussed in Chapter 6. An attacker might for instance try to directly read the memory of an ECU to retrieve encryption keys. One approach could be to erase the keys permanently if the physical case is opened, which presents both technical and practical problems.

Also related to physical attributes are side channel attacks, which aim to break a cryptosystem based on information related to its physical implementation, e.g. timing information or power consumption.

Much ECU software currently runs directly on the microprocessors, without an Operating System (OS). This means that security mechanisms commonly found in many operating systems are missing. However, it is anticipated that AUTOSAR will be extensively used in the automotive industry in the future. AUTOSAR is essentially a specification for a OS, but the concrete implementations of different vendors can vary greatly. It's

security mechanisms need to be examined and tested, and it is worth to investigate if there are security mechanisms from more powerful OSs that should also be implemented in AUTOSAR.

Chapter 6 contains a more detailed discussion of all the issues mentioned above.

2.5 Authentication, Interoperability and Mobility

Vehicles are highly mobile and can travel long distances in a short period of time. In Europe commercial vehicles can easily cross the borders of several countries in one day. The high mobility of vehicles requires unified key management for authentication to infrastructures, even beyond country borders. For instance, vehicles should at any time be able to stop for maintenance at a repair shop and connect to their network using wireless communications.

2.6 Product Development and Life Cycle Issues

In this Section issues are highlighted which are related to the product development phase or the relatively long life cycle of a vehicle.

2.6.1 Interplay of Security and Safety

Vehicles should be both safe and secure. A vehicle which is not secure can not be safe, since security vulnerabilities can have a higly detrimental effect on safety. However, this relationship is not necessarily clear cut, standard mechanisms which increase the safety of a vehicle such as redundancy can have a negative impact on security. For instance, a specific hot-standby ECU might increase the attack surface for hackers, since the extra communication and synchronization with the rest of the system can open new security holes. The more complex the vehicle is and the more services are implemented, the more vulnerable the system becomes.

Security functionality must focus on preventing accidents. The functions must be designed to support all safety-critical functionality at the expense of other functions if needed. The system must continue to operate even after a security problem, possibly with some limitations or with degraded service for the driver and passengers.

If drivers start to rely on collision avoidance systems, a silent failure of that system may result in an accident. At the same time, a system that gives too many warnings or behaves erroneously may give bad reputation to the vehicle brand.

Mechanisms implemented for safety, e.g. fault detection mechanisms must be implemented in such a way that they cannot be used by an attacker and result in security problems or evade protection mechanisms. Intrusion Detection Systems (IDSs) and Intrusion Prevention Systems (IPSs) may be used to detect malicious activities and dangerous scenarios. IDSs are discussed more in detail in Section 5. But since IPSs also react to what they see, they may be used by attackers to spawn other types of attacks. IPSs may

also pose legal problems if they are able to take over the control of a vehicle and bypass the driver's instructions.

2.6.2 Real-time Requirements

Real-time requirements define boundaries on security functionality, both for internal and external communication. In addition, many applications require hard real-time, or near real-time responses to work. Applications, such as collision avoidance systems need to make decisions in a fairly short time to be useful (real-time but not hard real-time requirement), and to make a decision they have to communicate with other vehicles and also be able to verify the correctness of all the messages they receive.

Security functionality must be designed in such a way that real-time requirements can be fulfilled and that it does not disturb other real-time functions in the vehicle. In addition, it is important to protect against misbehaving nodes which try Denial of Service (DoS) attacks, for example by flooding the internal network with traffic so that essential functions fail.

2.6.3 ECU Origins

Vehicle manufacturers often buy finished ECUs from third parties, which comprises both hardware and software. This can make it hard for the vehicle manufacturer to influence the security architecture of specific ECUs. Since a security architecture is only as good as its weakest link, a badly implemented ECU from a third party poses a serious security risk, so vehicle manufacturers need to make sure that the security architecture is well defined and that the security requirements are clearly communicated to the suppliers.

2.6.4 Software Updates and Maintenance

Vehicles have a comparatively long expected lifetime, it is not unreasonable to assume that many vehicles will be in use for 15 to 20 years. The lifetime of security solutions and implemented mechanisms must be equally long. It is worthy to note that this is about the same amount of time that cryptographic solutions have been in use to secure internet communication. Within that time frame, many of those security mechanisms have been broken (e.g., secure hash functions [14]). Since it is not possible to foresee future threats and risks, security mechanisms must be dynamic and adaptable.

The vehicle architecture must allow security functionalities such as cryptographic algorithms, keys, firewalls, etc., to be updated without major software and hardware changes. The SeVeCOM project has therefore suggested an architecture, where security functionality is isolated from the rest of the software and implemented as plug-ins into the network stack to allow easy modification.

Since it will be necessary to patch vulnerabilities as soon as they are discovered, support for secure remote software updates is needed. This is already common practice in the computer domain (e.g. Microsoft's patch Tuesday), but in the vehicular domain this is still lacking. Vehicles have to be patched not only when they return for scheduled

service once a year, but an internet-based secure software update mechanism to close the vulnerability is needed.

Different vendors will also have different implementations of similar services, even though they need to communicate with the same devices. Coordination, design, testing and compatibility of such systems will be a major challenge in the future.

Chapter 3

Security Threats

There are many possible ways to intrude into a vehicle or the infrastructure it's connected to. This chapter focusses on the various attack vectors that can potentially lead to an attacker taking over a vehicle or steal information. First, we provide a general overview of the different attack types, discuss common network security threats and lastly focus on demonstrated security threats targetting road vehicles and their infrastructure.

3.1 Common Network Security Threats

All protocols, at all levels, must ensure they have proper protection against several types of attacks.

- *Eavesdropping:* attacker reads (copies) data from the network. For wireless communications, sensitive and confidential data needs to be encrypted. An example of confidential data can be the software (firmware) for the ECUs.

- *Deletion and modification:* data is dropped or modified during transmission. Application-level protocols must either have their own detection mechanisms or ensure that the underlying network protects the data. Modification of traffic can be done by a man-in-the-middle attack, for example by abusing link-level protocols and changing the traffic routing.

- *Injection and data origin spoofing*: new traffic is injected into an ongoing session either by a man-in-the-middle or by a remote attacker. In a repair shop, an attacker may insert additional diagnostics messages into an existing session in order to steal data, change settings or the configuration of a vehicle, or possibly even update the firmware with new "enhanced" functionality.

- *Impersonation*, also called identity spoofing. The identity of other trusted users or devices may be spoofed with similar results as for traffic injection.

- *Recording, replay and delay:* data from older sessions are reused. It could contain old authentication information or the attacker could resend information that was intended to be used during other circumstances, at other locations or reuse data that should have been valid only during a short time period. Even authentic data which is immediately relayed over the internet to another location and replayed, could cause confusion and problems.
- *Denial of Service attacks:* often done by flooding networks or by using known vulnerabilities that cause nodes to crash or exhaust their resources.
- *Malicious traffic:* Malformed packets that should not normally be seen on networks but can cause systems to malfunction, crash or execute arbitrary code.

3.1.1 Protocol Vulnerabilities

There are many protocol stacks implementing link, network, and transport services in a modern vehicle which can potentially be exploited. In addition, not only attacks against the low-level network stack are possible, but all application level protocols that are introduced into the vehicle are potential targets for attacks.

Manipulation of data from application layer protocols may also result in severe security problems, for example enable modification of the functionality of an ECU.

There will be many application-level protocols, some based on TCP/IP and communicating in traditional client-server manner, and other use broadcast communicating to reach only objects around the car (V2X). Some applications will take care of security themselves, others will rely on security offered by the communication technology itself, for example for encryption and authentication. ETSI has been mandated to standardize V2V communications within the European Union [15]. Future standards such as their ITS Station architecture [5, 16, 17] will therefore likely dictate security requirements for nodes participating in V2X communications, but it will not help developers with how security functionality should be handled within the vehicle and how it may be implemented in various applications.

Unauthorized manipulation of traffic to vehicles may also result in unwanted behaviour, for example to enforce non-existing speed limits, to avoid empty but seemingly congested roads, announce lane-changes without the driver having any such plans, cause problems with car platooning and in general disturb functions in and around the car. A more dangerous scenario is an attack that sends spoofed messages to ECUs, for example that orders full speed ahead and disabling the break ECUs when the sensors in the car detect pedestrians in the way.

3.1.2 Attacks against Core Protocols

All protocols in the network stack can be attacked. The internet-connected car uses the IP protocol and therefore all Core protocols (DHCP, ARP, ICMP, DNS, TCP, UDP) as well as link-level protocols (802.11p, WiFi, cellular 3G/4G/5G, etc.) face similar threats as other

systems do. Application level protocols are unique for this domain and will be obvious targets for attacks and must be designed to be both secure and robust.

The V2X communication protocols (WAVE, IEEE 802.11p, WSMP) can also be targets for attacks, as well as Bluetooth, DSRC and RFID communications.

Attacks against core protocols have existed for decades and many tools exist that can be used by attackers but also by developers to perform their own penetration tests to make sure that at least all old and well-known vulnerabilities are handled properly. Historically, there are many well-known core protocol attacks, for example the Land attack (victim's address present in both source and destination fields), Ping-of-death (oversized IP datagrams), and header exploits (e.g. the length specified in protocol headers differ from the actual length) which may cause various problems from crashes to execution of arbitrary code.

This list of known problems is short enough to allow developers to test their implementations against almost all of them. However, history has shown that many new implementations do not take these known attacks into consideration, thus they re-appear in new implementations and cause systems to fail. Examples include Microsoft's network stack in the beta version of Windows Vista where they failed to perform such tests [18], and in Smartphone operating systems where a single link-level datagram can make the device freeze completely and only a removal of the battery makes it functional again [19]. These attacks mainly target the communications unit of the vehicle, although a failure in it may result in a denial of service, or worse, that malware is installed that can send messages fabricated remotely by attackers, on the internal busses.

Many of the threats to vehicles can be analyzed using the same methods and tools as are used to secure other internet-connected systems, and they should be used also in this environment to test the robustness of the implementations. Lang et al. [20] provide an interesting discussion of the security implications when the vehicle is connected using an IP-based network. Nine "hypothetical attack scenarios" are suggested based on attacks known from ordinary IT systems, i.e. attacks on the communication protocols, malicious code, and social engineering. Each scenario was analyzed with respect to confidentiality, integrity, availability, authenticity, and non-repudiation. Also, an attempt to quantitatively estimate the impact on safety was conducted and for each of the scenarios a SIL value was proposed.

3.2 Automotive Security Threats

The lack of security in today's vehicles has recently been demonstrated by several research groups. By connecting a device to the in-vehicle network, for example through the On-Board Diagnostics II (OBD-II) port, it is possible to send arbitrary commands to the vehicle. Many of the practically demonstrated attacks are performed through this diagnostic interface, which until today has required physical access to the vehicle. However, in the near future, the same attacks will be possible to perform through a wireless connection, with similar results.

3.2.1 GPS Spoofing

Attacks based on GPS spoofing have been widely demonstrated in all the major transportation sectors. One of the main reasons is the lack of any form of signal authentication. For instance, today it is possible to change the course of a ship [21], force a drone to land in an hostile area [22] or fake the current location in a road navigation system [23] by spoofing GPS signals.

3.2.2 Compromised ECUs can send arbitrary Messages

A team of researchers from University of Washington and University of California have practically shown that infiltrated ECUs in a vehicle can be used to send arbitrary messages to the CAN bus [1]. They have demonstrated this functionality both in the lab and in road tests.

They started by listening (sniffing) the CAN bus to determine how units communicated and what messages were sent. Replay attacks were then trivial to perform. To enhance the functionality in an ECU, they performed reverse engineering by dumping the ECU code over the bus using a third-party debugger. This enabled an attacker to silently "enhance" the functionality of ECUs but still keep its original functionality. They have demonstrated several different attacks: taking total control of the radio (user could not turn it off), produce various sounds in the vehicle (the audio component is used for warning sounds), display arbitrary messages on the instrument panel, open and close doors, honk the horn, disturb engine functions, lock individual brakes, control the A/C, etc.

They also demonstrated that the breaks could be released while driving, making the driver unable to break. These attacks were done locally, i.e. with physical access to the vehicle, but with the correct software, remote attacks with similar results are possible to perform (see below). The team was also able to get access to the high-speed CAN bus by only having a physical connection to the low-speed bus. They identified a system that is connected to both busses, in their case the telematics unit, and reprogrammed this unit so that it acted as a bridge.

The results are interesting but maybe not surprising since the internal networks and protocols lack all kinds of security, but they clearly show what the outcome could be if a node in a vehicle becomes compromised. Based on these results, Charly Miller and Chris Valasek did similar work for different car brands (see Section 3.2.5), and unlike the previous group, published all engineering details of their hacks.

3.2.3 Attacks via the Media Player

The same team of researchers from University of Washington and University of California recently also showed that serious attacks can be performed without physical access to the vehicle [24].

A weakness in the media player was used to gain access to the local CAN bus. The media player they tested came from a large third party supplier and it plays CDs and also accepts MP3 and WMA files. Media players in vehicles normally have access to the CAN bus, for example to be able to change the sound level when the vehicle's speed changes. Using reverse engineering, they discovered that it was possible to do a buffer overflow attack when playing WMA music. The weakness allowed the attackers to create a CD which contained specially crafted music that the player plays perfectly, but also silently performed a buffer overflow attack which can send arbitrary commands on the CAN bus. It is not hard to imagine that music containing such viruses would be popular to distribute on the internet. The telematics unit also contained vulnerabilities in its cellular communication protocols opening it up for remote internet attacks.

The attacks show the necessity to have good security practices in place when designing software for ECUs and other equipment connected to the internal networks. In this case, the media player and the telematics unit in the vehicle had traditional buffer overrun bugs, allowing arbitrary messages to be sent on the internal CAN bus. In the future, we can expect that many ECUs and subsystems like these contain software implemented by third parties, about which the vehicle manufacturer have limited knowledge and no possibility to know all design details or to be able to evaluate the implementation.

3.2.4 Attacks via Wireless Tire Pressure System

All vehicles manufactured in the U.S. after 2007 are required to have a Tire Pressure Monitoring System (TPMS) installed. Rouf et al. [25] have demonstrated an attack where the wireless communication (RF transmitter) between the vehicle and the sensors in the tires are compromised.

Each tire sensor has a unique 32 bit identifier to prevent vehicles from displaying messages from other vehicles. The radio communication between the tires and the vehicle turned out to be unprotected and reached around 40 meters with low-cost antennas and amplifiers. They also showed that remote spoofing of messages was possible, and all messages with correct IDs were unconditionally accepted, triggering warning messages for the driver. The equipment used was available on the market for around $1,500.

During their tests with spoofed messages, they also managed to completely crash the ECU receiving the tire-pressure messages, and the only recovery possible was to return the vehicle to the repair shop and have the ECU replaced.

They also concluded that the 32 bit identifiers used to uniquely identify each tire pressure sensor, can be used to track vehicles and therefore creating privacy problems for the owners.

3.2.5 The Car as part of the Internet of Things

Since 2009 many vehicles have a built-in sim card module to offer customers new services, such as real time traffic information, automated emergency call in case of an accident, remote diagnostics, or control via smartphone application (unlocking the vehicle, or

turning on the pre-heating). The vehicle as part of the internet introduces new security flaws, as it provides remote accessibility.

100 Cars disabled Remotely. In March 2010 media, including *wired.com*, reported that more than 100 cars were disabled remotely. Until the cars were fully paid for, the cars were controlled by the car dealer and contained functionality to be remotely disabled if the customer slipped with the monthly payments. 1,100 cars were reported to have this functionality.

This day, a former disgruntled 20 year old employee used a fellow employee's account to log in to the dealership's computer system and disabled more than 100 vehicles for their customers. The ignition system was disabled and he managed to honk the horns in the middle of the night. The only way the customer could turn off the horn was to remove the battery.

This security problem again shows the risk of installing third party applications in vehicles. There may have been limitations to what the system could have done, but we have also seen that the vehicles lack protection if an ECU or a connected system wants to send arbitrary messages.

Unlock of a vehicle remotely. In 2015 a vulnerability has been found that allowed an attacker to unlock a vehicle remotely. The car manufacturer offered a remote service to perform certain control and infotainment commands via a smartphone. An analysis has shown that a man-in-the-middle attack can be used to gather the unencrypted *unlock* message from the mobile phone by mimicking a cellular base station and replaying this message. In this particular case, the car manufacturer was able to patch 2.2 million vehicles having this security flaw with an over the air software update, as this vulnerability didn't involve the hardware modules [26].

Remote Compromise of a Jeep Cherokee. In 2012 Miller and Valasek started their research, which was funded by the Defense Advanced Research Projects Agency (DARPA), to root out vulnerabilities in vehicles. Their first hack required a physical connection and allowed them to control the vehicle and even disable the brakes. The demonstration was performed on a Toyota Prius and a Ford Escape. Miller and Valasek focussed on attacking the system via a physical connection, because of the already published hacks to access a vehicle remotely. Their findings have been published in 2013 [27].

In 2015 Miller and Valasek presented their results in remotely hacking a Jeep Cherokee at blackhat USA 2015. They started their investigation on possible vulnerabilities with a 2014 Jeep Cherokee. Their first discovery was that the radio was connected to both of the vehicles CAN busses. The head unit (radio) has not been developed by FIAT Chrysler Automotive, it was produced by one of the company's supplier [28].

Further investigation showed that the WiFi password of the vehicle was generated automatically based on the time when the head-unit of the vehicle was started the very first time. The knowledge of the production year and the month can narrow down the number of combinations to 15 Millions. By taking also the fact into account that the vehicle was likely started the first time during the day, one is able to reduce the possible combinations to about 7 millions. Nevertheless, the researchers found out, that the WiFi password is set before the system has set the correct time. In their case it was the 1^{st} January 2013 at 00:00:32. This fact reduces the possible number of combinations dramatically and allows a nearly instantaneous access to the vehicle's WiFi. The implemented password generator would have forced the attacker to follow the target for about one hour to have a steady WiFi connection when performing the brute force attack. However, knowing that the generated password uses an incorrect system time reduces the time for the attack tremendously. This vulnerability emphasizes the importance of security tests in third-party devices [28].

The access to the vehicle's WLAN network allowed the researchers to further investigate the D-BUS interface, which has shown that no authentication was necessary. Changing the volume, fans and even the display was possible with lua scripts due to the open D-BUS interface of the Uconnect system [28].

Miller and Valasek were also able to jailbreak the Uconnect system by switching the pendrive while the vehicle reboots. Moreover, the authors highlight, that the jailbreak of the system was not required to access the vehicle's D-BUS. The use of a femtocell allowed the researchers to access the vehicle from even farther away. Thus, vehicles that do not have enabled the vehicle's WiFi service are vulnerable as well. Surprisingly it turned out, that the D-Bus port was not only opened to the WiFi and the closest cell, it was opened to the entire Sprint network. This has the result, that all vehicles were accessible by any device connected to the Sprint network. A scan of certain IP ranges, that are used by Sprint for vehicles, with an open D-Bus port (in this case port 6667) and a certain reaction via telnet revealed that 16 car models across the United States of America have been effected by this vulnerability [28].

To perform cyber physical actions remotely, it was necessary to flash the firmware of the V850 chip that communicates with the ECUs via CAN bus. Reverse engineering the firmware of the Uconnect system's chip and the V850 chip in order to find a way to remotely upgrade the firmware of the V850 chip took the researchers several weeks [28].

Nevertheless, Miller and Valasek were not able to perform steering or braking while the vehicle was driving more than 10 mph. This restriction was caused by the ABS and Parking Assist Module (PAM) ECU that turn off some of their functionality in case they receive contradicting messages (the valid/correct and malicious messages). For example, the ABS ECU disables the collision prevention entirely when receiving conflicting messages. The researchers had to put the ECUs which are sending the correct messages into the diagnostic mode, what is only possible below 10 mph, to be the only one sending a certain message type. A discussion about the correctness in terms of safety and security of this mechanism encountered in the Jeep Cherokee is left open. This model is not a

self-driving vehicle and thus the driver has to be alert at any time, such as in the case when the collision prevention is suddenly disabled. However, the higher the level of automation of the vehicle, the more robust and fail-safe in terms of safety and security the vehicles have to be.

This research has caused a recall of 1.4 million vehicles for updating the software of the vehicle. Moreover, it is shown that the vehicle as part of the internet of things involves also network providers. The security of the vehicle depends also on the devices/modules from the suppliers. In this case, the vulnerability of the head unit (Uconnect system) allowed the attacker to access the vehicle's system from all across the US [28].

In 2016 Miller and Valasek have revealed a vulnerability of the in-vehicle network of a Jeep Cherokee that was patched against the vulnerability they have published the year before. The focus of this hack was the full control over the vehicle, thus they connected a notebook directly on the ODB-II port. The researchers were able to bypass safeguards that prevented the full control over the vehicle. Their approach was to not only compromise one ECU in order to send control commands, they attacked those ECUs that were sending the correct messages the researchers were sending. This causes the system to only receive their "wrong" messages sent over the CAN bus. Disabling the safeguards was achieved by paralyzing these ECUs by putting them into "bootrom" mode. Miller and Valasek claim that this attack is still possible and that it is only a matter of time that attackers find a new vulnerability to access the vehicle over the internet [29].

Web Portal as Target. In the mid of 2016 the security researcher Benjamin Kunz Mejri of Vulnerability Laboratory has found two vulnerabilities in the web interface of a car OEM that allowed the attacker to access the configuration of other vehicles. Accessing and manipulating the configuration of another vehicle was possible due to a vulnerability in the session management that allowed to bypass the secure validation [30]. This attack allows the attacker to access and manipulate the playlists, e-mail accounts, travel routes, and to unlock or lock the vehicle. The second security flaw is a client-side cross site scripting web vulnerability, that allowed the attackers to inject malicious code [31].

Accessing the CAN bus via WiFi. Researchers from Tencent's KeenLab security team have successfully hacked a Tesla car in 2016. The researchers created a WiFi network that is typical for Tesla stores and exploited a vulnerability of the web browser, which is based on the WebKit framework. This bug allowed the researchers to run malicious code, reprogram the firmware of the gateway that has access to the CAN bus, and in the end to control the vehicle. Tesla not only patched the WebKit framework, the company also introduced *code signing* in order to avoid the firmware installation of unsigned code [32]. This example also demonstrates that third-party software or libraries might lead to new vulnerabilities that need to be tested beforehand or patched in a timely fashion.

In a recent report [33], KeenLab further investigated the Tesla autopilot ECU software module and managed to gain remote control of the steering system in a contact-less way.

The team was able to generate adversarial examples of the features (autowipers and lane recognition) which make decisions purely based on camera data, and successfully achieved the adversarial example attack in the physical world.

Smartphone vulnerabilities. Promon, a security company based in Norway, has shown that it is possible to hack a Tesla car with a smartphone as entry point. They created a WiFi hotspot of a popular burger restaurant and trick the user into installing an application in order to get discount. The installed application is used to control the Tesla application, so that it sends the attacker the user credentials when logging in. With the username and password of the Tesla owner, one was able to track the car, unlock it and drive it [34].

The researchers highlight, that Tesla didn't consider best practice examples for security in their smartphone application, which lead to an easy and fast extraction of the user credentials. Furthermore, it is important to clarify that it is also possible to trick the user into installing a custom keyboard to get the necessary information. The users need to realize that the smartphone contains important information, such as information from banking apps, and that they have to use it more carefully.

Chapter 4

Communication Technologies

Due to the different requirements of certain vehicle functions, a variety of network technologies is currently present in vehicles. Control messages with strong realtime requirements may be exchanged via CAN bus whereas a video stream of the front facing camera demands a high bandwidth, e.g., Automotive Ethernet. This chapter presents the most common in-vehicle networks as well as communication technologies used to communicate with other vehicles and infrastructure.

4.1 Internal Communication

Vehicular systems use a number of different communication technologies with very specific characteristics. In the following, we discuss the bus technologies that are most prevalent in current automotive systems.

In-vehicle networks usually consist of several domains connected through a backbone, and each domain has one or more busses for different purposes. The most prevalent busses are CAN, LIN, MOST and FlexRay. There are also new technologies which are making their way into automotive systems: Ethernet has already seen its debut in the industry and is currently being standardized for use in automotive systems, and Controller Area Network flexible data-rate (CAN-FD) is also a prime candidate for future adoption. Each communication technology has particular speed and timing characteristics which makes it well-suited for a specific domain.

CAN. The Controller Area Network bus is the most common automotive bus: it is well established and well understood, and provides sufficient performance for most applications. CAN is a Carrier Sense Multiple Access/Collision Detection bus with Arbitration on Message Priority (CSMA/CD+AMP). A node on a CAN bus can only transmit when it senses the transmission channel to be free. When two or more CAN frames are transmitted simultaneously, *bit arbitration* takes effect. As a result, the highest priority frame will be transmitted successfully while all other frames will be dropped. Thanks to

this prioritization, CAN can be analyzed for its worst-case real-time properties, which is important for safety-critical systems. The typical speed of a CAN bus is 500 Kbit/s, with a *maximum speed of 1 Mbit/s*. A single CAN frame can carry a *maximum of 8 data bytes*. CAN busses are used for standard operational functions, but also for many safety-critical functions. In many vehicles, CAN is still used for the backbone network, although new backbones are gradually being implemented using FlexRay or Ethernet busses. A typical CAN frame is depicted in Figure 4.1. The arbitration field consists of a 11 bit ID, which is used for prioritizing the messages in decreasing order and to filter the messages. The data field has a variable length between 0 and 8 Bytes, where the length of the data field is transmitted in the control field. The Cyclic Redundancy Check (CRC) field is used to detect errors during a transmission. A valid CAN packet is confirmed with the acknowledgement field [35].

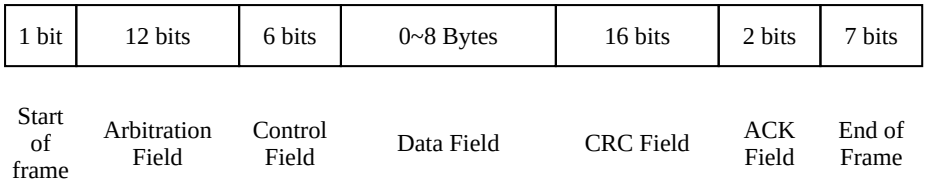

Figure 4.1: CAN frame format

LIN. The Local Interconnect Network bus was invented as a cheaper alternative to the CAN bus. LIN follows a star-topology with one master node and at least one slave node. Communication on the bus is controlled exclusively by the master node, so Collision Detection (CD) is unnecessary, and latencies are guaranteed. The master can itself acts as a slave. The maximum speed of the LIN bus is 19.2 Kbit/s, and a single frame can carry a maximum of 8 data bytes. It is mostly used to control vehicle body electronics, such as windows or windshield wipers.

MOST. MOST[1] is a network that is specially designed for transmitting multimedia and infotainment content. This synchronous network is realized with a ring topology and offers a total bandwidth of approximately 23 MBaud, which is divided into 60 channels. The maximum number of nodes in a MOST network is 64. Synchronization is achieved with a *TimingMaster* that transmits the system clock with a continuous signal. All other devices/nodes are named *TimingSlaves*. The network is synchronous, but it allows both, asynchronous and synchronous data transfer. Moreover, MOST defines all layers of the OSI Reference Model. Advantages of MOST are its efficiency in transmitting multimedia content and that it offers electrical and optical solutions [36].

[1] http://www.mostcooperation.com/technology/most-network/ (2019-07-22)

FlexRay. FlexRay was designed to be faster and more predictable than CAN. It achieves the higher predictability by using Time Division Multiple Access (TDMA) instead of Carrier Sense Multiple Access (CSMA). FlexRay works in cycles, and there are two main types of cycle segments: static and dynamic. Static segments are divided into a number of fixed-length slots, each with a specific ID, and only nodes with the corresponding ID are allowed to transmit in that time slot. This allows predictable communication and supports applications with hard real-time requirements. Dynamic segments allow for event-triggered communication by following the same time slot arrangement as static segments, but if a node needs to transmit more data than fits into its time slot, it can "take over" time slots of nodes with lower IDs, forcing the nodes with lower IDs to wait for a free slot. In order to work correctly, FlexRay requires synchronized clocks, and clock synchronization is built into the protocol to guarantee the required accuracy. The maximum bus speed is 10 Mbit/s, and a single frame can carry a maximum of 254 bytes of data. FlexRay allows for a fault-tolerant dual channel approach, and supports different types of topologies. Its main drawbacks are that it is more expensive and more complex than CAN.

CAN-FD. CAN's bandwidth limitations have given rise to a new CAN-based technology called Controller Area Network flexible data-rate, which is also backward compatible. As the name suggests, CAN-FD is CAN with a flexible and higher data rate: CAN-FD has a *maximum speed of 2 - 5 MBit/s*, depending on the bus topology. A single CAN-FD frame can carry a *maximum of 64 data bytes*. It achieves its backward compatibility with CAN by differentiating between the arbitration phase and the data transmission phase. During the arbitration phase, it uses the same bit rate as classical CAN, but during the data phase CAN-FD can switch to higher bit rates. The stand out advantage of CAN-FD over FlexRay or Ethernet is that it is backward compatible to CAN and thus cheaper to implement.

SAE J1709. The SAE J1709 allows the serial communication between ECUs and is specially designed for the use in heavy duty vehicles. This standard recommends twisted pair cables to achieve a bandwidth is 9600 bits per second. Simplicity and low cost were the key benefits for using this standard. Messages consist of three fields: the *Message Identification Character (MID)*, *Data Characters*, and the *Checksum*. The MID ranges from 0 to 255, where 0 to 68 is defined by the standard, 69 to 111 and 128 to 255 is assigned by other committees, and 112 to 127 is available for general use [37].

SAE J1939. The SAE J1939 [38] is a common communication architecture that has been defined by SAE due to the need of a communication standard between ECUs from different manufacturers for heavy duty vehicles. It defines the message timeouts, the network speed, and vehicle body control. The physical layer in SAE J1939 is the CAN bus. Junger lists in [39] the following as particular characteristics of J1939:

- Extended CAN identifier (29 bit)

- Bit rate 250 kbit/s
- Peer-to-peer and broadcast communication
- Transport protocols for up to 1785 data types
- Network management
- Definition of parameter groups for commercial vehicles and others
- Manufacturer specific parameter groups are supported
- Diagnostics features

SAE J1939 splits the CAN identifier into several fields that indicate among others the priority, parameter group number, and source address. Parameter group numbers (PGNs) define the content of the messages. For example, PGN 65262 *Engine Temperature* contains 8 Bytes of data representing the Engine Coolant Temperature, Fuel Temperature, Engine Oil Temperature, Turbocharger Oil Temperature, Engine Intercooler Temperature, and the Engine Intercooler Thermostat Opening. Additionally to the defined content of this PGN, J1939 also defines the transmission rate which is in this example 1 second [38, 40].

For the reason that this standard defines the structure of the transmitted frames, it is not possible to add security measures, such as Message Authentication Codes, in the same frame. A possible method can be sending this information as an additional frame for specific data frames belonging to the general use group, which has the disadvantage that the ECU would have to wait until the second frame arrives.

Automotive Ethernet. Ethernet can be the solution for the need of a higher bandwidth. It causes a paradigm shift in the design of the vehicle's internal network, for instance subnets can be used for segmentation. As Hank et al. state in their paper [41], the TCP/IP stack is already in use when charging the vehicle to allow a communication with the vehicle's charge control module. This charging process is defined by the ISO 15118. Moreover, DoIP takes also advantage of the existing IP stack.

Important aspects of using Ethernet in the vehicle are the costs for the wiring and the chipset, the bandwidth, and the Electromagnetic compatibility (EMC). The electromagnetic emission profile is critical for the reason that the FM radio band must not be disturbed by cabling. The early work on the BroadR-Reach[2] technology offered a bi-directional communication that operates on an unshielded twisted-pair cable, has a symbol rate of up to 66.6 MBaud and a bandwidth of 100 Mbit/s [41]. BroadR-Reach has been further adapted and standardised as part of the IEEE automotive Ethernet standards *IEEE 100BASE-T1,1000BASE-T1* and *1000BASE-RH*.

[2] http://www.opensig.org/ (2019-07-22)

Audio Video Bridging (AVB). Audio Video Bridging (AVB) is designed for the standard ethernet network technology and has the benefit of providing well synchronised streams in realtime by using a simple cabling. AVB is a solution for the need of synchronised video and audio streams due to the increased presence of camera devices to provide a surround view. This technology is also intend for the use as a backbone for control data. The real-time requirements are achieved with a time synchronization and quality of service.

The following IEEE specifications are part of the IEEE AVB (taken from [42]):

- **IEEE 802.1BA [43]** - Audio Video Bridging Systems Overview
  Introduction and Overview
- **IEEE 802.1AS [44]** - Timing and Synchronization for Time-Sensitive Applications
  Time synchronization of the nodes by using a reference time with an accuracy that is better than 1 micro second.
- **IEEE 802.1Qav [45]** - Forwarding and Queueing Enhancements for Time-Sensitive Streams (FQTSS)
  FQTSS - Traffic Shaping, separation of time critical and not time critical traffic.
- **IEEE 802.1Qat [46]** - Stream reservation Protocol
  Reservation of resources done by using buffers and queues within the bridges.

Jesse concludes in his presentation [47] that AVB can be used in parallel with IP since the frames are already separated on driver level and this ensures the high availability of AVB. This technology is also designed to allow a maximum hardware support, e.g. use of FPGAs [47].

Time-Triggered Ethernet (TTE) The TTTech Computertechnik AG[3] offers a solution called Time-Triggered Ethernet (TTE). This technology is based on Ethernet and provides a "*fully deterministic communication, fault-tolerant synchronisation services, guaranteed constant latency for multi-hop communication routes in the network, and partitioning/protection of network traffic*" [48, p.1].

TTE categorizes traffic in three different groups: time-triggered (TT), rate-constrained (RC) traffic, and best-effort (BE) traffic. The sender and the TTE switches have a transmit or forward schedule, when sending TT traffic. This traffic class requires a clock synchronization among all devices. RC traffic is ensured to provide lossless communication with a bounded latency and jitter. Moreover, the sending node gets a reserved bandwidth. The BE traffic class is compatible with the IEEE 802.3 standard and provides no timing guarantees [48].

[3] https://www.tttech.com/ (2019-07-22)

4.2 External Communication

Many different communication technologies are likely to be used for various communication services. Some technologies, like DSRC and Bluetooth are very local and the objects need to be located closely to the vehicles. Other techniques, such as 4G and similar technologies are used for accessing the internet to and from the vehicles.

DSRC. Dedicated Short Range Communications is a general term often seen in V2X documents but the meaning of the term DSRC has changed over time and may today mean RFID communication, WLAN communication or any other of several different short range communication techniques. In the vehicular domain, DSRC is synonymous with the use of the 5.9 GHz frequency band which is dedicated for V2X communications. Standardization is going on in this domain and in United States the IEEE 1609 WAVE (Wireless Access in Vehicular Environments) protocol which uses the IEEE 802.11p link-layer protocol will be used. In Europe the ITS-G5 which is also on the basis of 802.11p, but with some amendments towards European requirements is being standardized to be used for vehicular ad-hoc communications.

IEEE 802.11p. This technology is based on 802.11a but has some special functions making it more suitable to short-lived ad-hoc communication. Higher layers are expected to follow the IEEE 1609 WAVE standard, which covers layer 3 to 7, where both short broadcast communication as well as TCP/IP is supported. These protocols and their security features are further discussed in Section 4.2.1. The 802.11p standard supports up to 27 Mbps using 7 dedicated channels in half duplex mode and communication range is in the order of 300 meters. 802.11p does not explicitly address security and encryption (like WPA and WPA2 which is present in 802.11a) and it must therefore be addressed by upper layer protocols. Security was omitted because of the short-lived communication patterns and problems with authentication of users on lower protocol layers. Instead of spending valuable time computing crypto-keys and exchanging messages with a base station, vehicles should be able to quickly create ad-hoc networks and exchange signed messages directly with as little overhead as possible.

ITS-G5. This is a collection of European standards for communication in the 5 GHz band, based on IEEE 802.11p. They specify the functionality of the physical and data-link layer used in the ETSI ITS communication (ITSC) architecture to ensure cross-border interoperability. More details are provided in section 4.2.1.

WLAN. Traditional IEEE 802.11a,b,g,n communication can also be used to offer internet access to traditional client-server applications, for example by car owners who want to connect the car's multimedia system to their home network. The car manufacturers

and third party software vendors may also want to use WLAN technology for remote diagnostics and software updates while the vehicle is parked.

Cellular networks. 4G/5G will primarily be used for client-server based applications and used where no WLAN networks are present. It will also be used for safety-related services, such as calling for help if internal sensors have detected a crash and by many other types of applications when road-side devices cannot be used. Cellular communication can also be the communications method used to check the validity of certificates by consulting on-line revocation (CRL) lists.

Bluetooth. This technology is used today to connect mobile phones to the multimedia system, primarily for hands-free operation. This may seem harmless, but the multimedia system is connected to the internal CAN bus, and if compromised, arbitrary messages may be sent. Most mobile phones are at the same time connected to the internet and may function as bridges between the internet and the internal vehicle network. In the future, Smartphones can also use Bluetooth communication to offer many other services to vehicles.

NFC. Near Field Communication, such as RFID cards, is today used for driver identification and can offer more functionality in the future. Vehicles communicating with other devices using Bluetooth, NFC, etc., are often called *V2M* (vehicle to mobile communication) and are sometimes, but not always, included in the *V2X* concept.

GPS and RDS radio communication. RDS information is used to receive broadcast messages about traffic conditions and road work. GPS and RDS technology is in place today, but security problems may arise if ECUs within the car always trust the transmissions and services offered to vehicles depend on the correctness of the data. A driver may, for example, change his GPS position in order to avoid road tolls and other fees.

4.2.1 Broadcast V2V and V2I Communication

Many V2V and V2I services will use broadcast DSRC communication. Messages can contain information such as a vehicle's location, speed, directions, maneuvers such as intention to brake, change lane or pass an intersection, etc. These broadcasted messages are used to spread awareness between vehicles close to each other.

Wireless access in vehicular environments has recently been standardized by both the IEEE to be used in United States and the ETSI to be used in Europe. In the U.S., the IEEE 1609 WAVE standard supports V2V and V2I communications using the DSRC 5.9 GHz band dedicated for the intelligent transport system (ITS). WAVE standardizes both V2V and V2I communications and describe data exchange, security,and service advertisement between communicating parties and is intended to be a framework for application developers when implementing services. The IEEE 1609.2 standard which

describes the security for the WAVE communications is an active standard now. Figure 4.2 shows an overview of the protocol hierarchy and the different protocols used at the different communication layers.

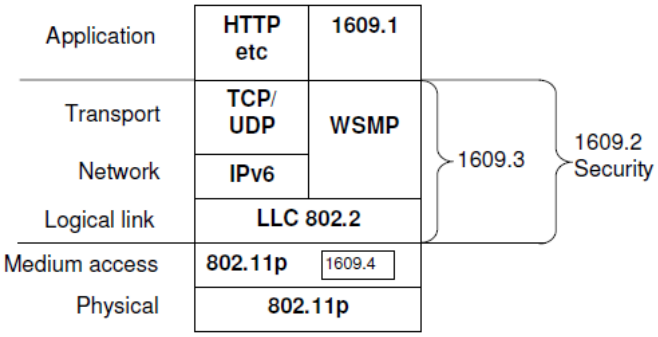

Figure 4.2: WAVE = IEEE 1609 + IEEE 802.11p

The WAVE standard relies on the **IEEE 802.11p** protocol for the physical (PHY) and link (MAC) layers and covers layer 3 to 7. The WAVE standard governs modes of operation in the DSRC band including architecture and resource management (1609.1), management, security services and message formats (1609.2), and network services (1609.3). A new protocol, the *WAVE short message protocol* (WSMP) has also been specially designed for broadcast V2X communications. In addition, two special service channels allocated for safety-critical applications also support IPv6.

The ETSI ITSC architecture standardizes the functionality contained in ITS stations and it follows the principles of the OSI layered architecture extended by amendments towards European ITS applications. Figure 4.3 shows the ETSI ITSC reference architecture.

The ETSI ITSC architecture is specified in ETSI EN 302 665. The Table 4.1 shows some details about the ETSI ITSC architecture and the related publications.

4.2.2 Vehicular ad-hoc Networks, VANETs

VANETs are often considered to be a subset of MANETs, Mobile ad-hoc networks. The most important differences are the short-lived communication patterns in VANETs and that it is unlikely that the same nodes forming a VANET will ever meet again. Thus, vehicles will frequently participate in new dynamic ad-hoc networks with participants they have never seen before, and it is therefore not that meaningful to store or cache any credentials or information to speed up reconnection to the same VANET again. Many VANET applications such as collision avoidance systems also have hard real-time requirements for the communication.

The lifetime of a VANET can be very short. Consider a vehicle approaching a road-side object in 100 km/h. If we assume that communication can take place in a 100m radius from the object, they can only communicate during 7 seconds which includes

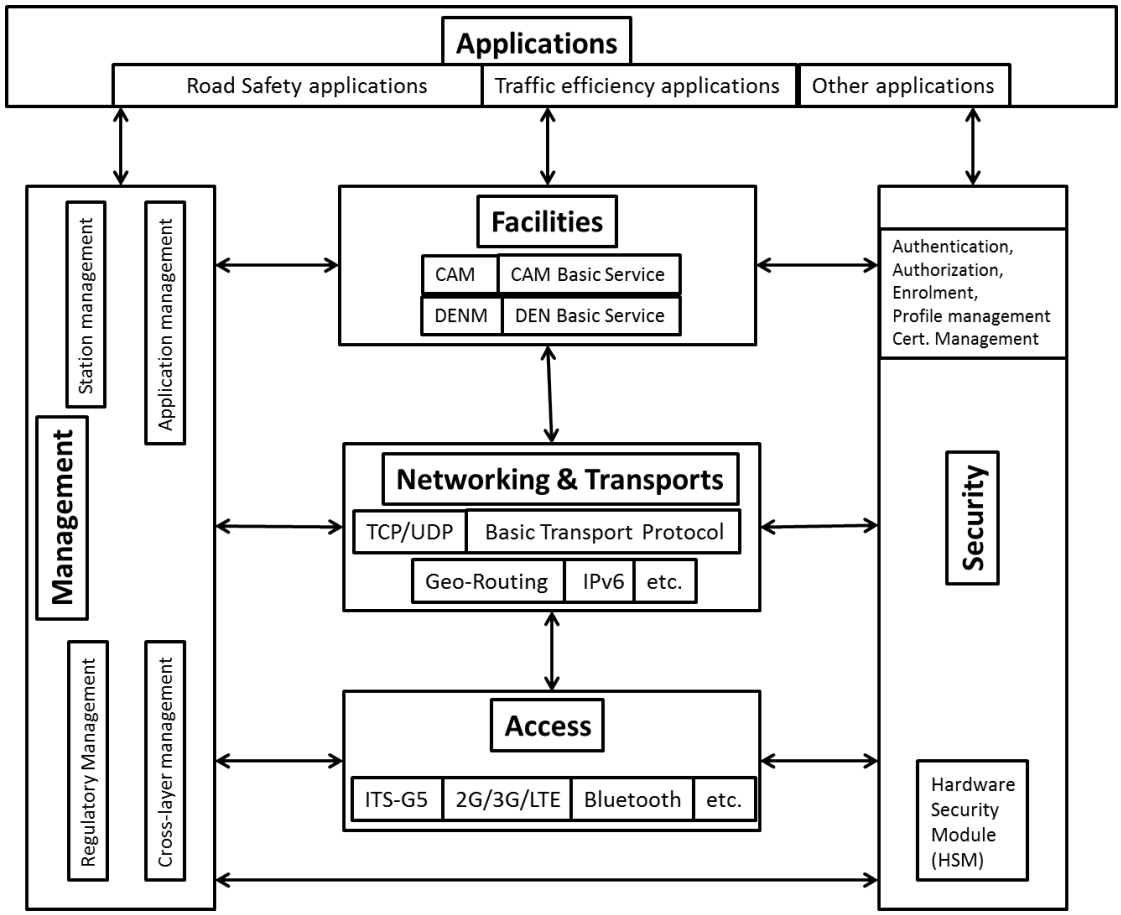

Figure 4.3: ETSI ITSC reference architecture

time for network discovery and possible authentication procedures take. The lifetime of communications between cars driving in opposite directions may be even shorter, and it may therefore be necessary for other vehicles to forward messages to extend the communication range.

Some messages need to be repeated or forwarded by the nodes in the network to reach all participants, possibly with a delay to allow vehicles to move to increase the communication range. Some messages like emergency messages should be spread rapidly and reach as many recipients as possible in a short time, but this can result in network flooding if lots of nodes begin to repeat the same message. Therefore, new routing protocols are needed that are adopted to this situation.

One interesting technique is to allow vehicles (nodes in the communication network) to store information and at a later time and position, retransmit the data, i.e. data is carried from one location to another by vehicles. This will increase communication range for messages, but it also results in new routing problems. Several methods for how this

Layer	Description	Related ETSI Publication
Access	Corresponds to the Physical and Data Link layers of the OSI model	EN 302 663 | ES 202 663 TR 102 960 | TS 102 917 TS 102 916 | TS 102 862 TS 102 861 | TS 102 792 TS 102 724 | TS 102 708-2
Networking and Transport	Corresponds to the Network and Transport layers of the OSI model	TS 102 636 | EN 636-2 TS 102 985 | TS 102 870 TS 102 871 | TS 102 859 TR 103 061
Facilities	Corresponds to the Session, Presentation and Application layers in the OSI model. It provides Application support, information support and communication/session support	TS 102 894 | 102 637 TS 102 869 | TS 102 868 TS 102 863 | TR 103 061
Management	Manages the installation and configuration of ITS-S applications, responsible for regulatory management, cross-layer management, etc.	TS 102 723 | EN 302 665
Security	Providing security services, such as certificate management authentication/authorization, cryptographic functionalities, etc.	TS 102 731 | TS 102 943 TS 102 942 | TS 102 941 TS 102 940 | TS 103 097
Application	Road safety, Traffic efficiency and other applications	TR 102 638

Table 4.1: ETSI ITSC architecture layers and related publications

can be done have been suggested, for example using group communication techniques. All these requirements and restrictions make many algorithms developed for MANETs less useful and new methods for communication are therefore under development.

Chapter 5

Intrusion Detection and Prevention Systems

IDSs have been suggested for all types of systems and environments, including inside vehicles and the cloud, for some time but have for various reasons not yet really been that successful. The main problem is how to handle both false positives (false alarms) and false negatives (failure to detect a problem). There is usually a strong relationship between the two: a high detection rate also results in a high rate of false alarms. In the vehicular domain, it is not clear what should be done when a potential problem is detected. False alarms may cause more problems than the problem itself would have done.

As this topic gained more popularity in the recent years we plan to review proposed IDS solutions for vehicles in the course of this project and also investigate how a cloud-based IDS for detecting intrusions on vehicles can be used. IDSs can be categorised in two groups: specification-based; and anomaly-based detection. Their fundamental concepts are described in the following paragraphs.

5.1 Specification-based Detection

A specification-based IDS triggers an alert when a rule or a pattern that has been described earlier is matched to network traffic or the ECU's behaviour. Due to the fact that this type of IDS is raising alerts when it detects a match to a signature in its database it has a low false positive rate. However, as the abnormal behaviour has to be specified before hand it is likely to have a high rate of false negatives.

Larson et al. [49] propose and evaluate a specification-based IDS for the CAN 2.0 and CANopen 3.01 protocols. They conclude that, since these protocols lack information about the producer and consumer of messages, there is not enough information available for using network-based intrusion detection. Instead, they propose host-based detection, i.e. one detector is placed in each ECU. Incoming and outgoing traffic can then be

investigated based on information from the protocol stack and the object directory of the CAN-protocol at the specific ECU. For the detector in the ECU, security specifications for the communication protocol and the ECU behavior can be developed.

The authors conclude that the gateway ECU is the most important ECU to protect. If the gateway ECU is compromised, all types of attacks can be performed. Unfortunately, performing detection in the gateway ECU is harder than in ordinary ECUs since the detectors for the different interfaces at the gateway have to cooperate to detect certain attacks, e.g. to detect lost or modified messages.

5.2 Anomaly-based Detection

Anomaly-based detection has been studied in ordinary desktop environments, but a problem faced in this environment is how to define a normal network behaviour; The desktop network is provided for arbitrary network communication. In contrast, for in-vehicle networks researchers argue that the expected payload is more well defined as the messages transmitted in the is specified during the design and anomaly-based detection should be easier to implement.

Hoppe et al. [50] demonstrate an anomaly-based IDS for the CAN protocol. In contrast to the specification-based approach by Larson et al. [49], where the IDS is placed in the ECU, they listen to the network traffic on the CAN-bus. By looking at the rate of how often specific messages are transmitted on the bus, and comparing that to what is considered to be normal, deviations from the expected number of transmitted messages can be detected. This was exemplified by investigating the system that detects physical vehicle break-ins. When the anti-theft alarm is activated, the system sends messages to the lights of the vehicle to turn them on and off, so that they are flashing. An attacker does not want these lights to be activated, but since the CAN-bus is a broadcast network, messages sent by the alarm system can not be deleted (except possibly in gateways). Instead the attacker have to create new messages to turn the light off as soon as it is lit. These new messages will be a deviation from the normal messages sent, and detected by the anomaly-based IDS.

Another method proposed by Hoppe et al. in [51] is the identification of misuse of messages. ECUs send their messages with a message ID on the CAN bus. Thus, the sending ECU is able to check if messages with its exclusive message IDs are sent from another source on the same CAN bus. This approach can be applied to gateway ECUs as well, as they know from which subnet which message IDs are sent. Matsumoto et al. present in [52] a similar approach with the extension, that the ECU detecting anomaly is sending an error frame immediately in order to override the malicious frame.

Core building blocks for implementing an anomaly-based in-vehicle IDS are described by Müter, Groll and Freiling [53]. They define eight types of sensor to detect different anomalies in the in-vehicle network and six criteria on sensor operations. For each of the criterion they evaluate how the different sensors will perform. The eight sensors are defined with respect to:

1. *formality* and correctness of the messages transmitted, i.e., the correctness of packet headers, checksums, etc.,
2. the *location* of the message, i.e., in which sub-network the message is transmitted,
3. the *range* of values in the payload of the message, i.e., within what boundaries a value need to be in,
4. the *frequency* of the given message,
5. *correlation* of message occurrence between different sub-networks,
6. different aspects of the inspected *protocol*, e.g., challenge-response behaviour,
7. *plausibility* of the transmitted payload, e.g., vehicle speed is increased instantly from 0 km/h to 100 km/h, and
8. the *consistency* of the over all system, by help of other sources in the vehicle.

The six criteria describes how the different sensors can detect anomalies: (1) based only on deviation from the message specification such as given by the CAN-Matrix, (2) the number of messages needed to detect an anomaly, (3) the number of sub-networks needed to perform detection, (4) how many different types of messages are needed to identify the anomaly, (5) wherever the payload needs to be inspected, and (6) wherever the semantics of the message needs to be considered. A classification of the sensors are also presented, based on how many messages are needed for detection and whether the payload needs to be inspected. Finally, they present a model of how the severity of the sensor alerts are mapped to the three levels of alert (to get the driver's attention) proposed by Hoppe, Kiltz and Dittmann [54].

An information-theory approach by analysing the entropy of transmitted messages are proposed by [55]. They were able to detect a set of anomalies in their test-setup by identifying a deviation in the entropy of the transmitted messages. Among these were the detection of increased frequency of messages and message flooding.

The SeVeCOM project [56] recommends vehicles to use an anomaly-based IDS system internally. However, they do not address in detail how and what the IDS system should do except that "appropriate reactions should be taken to get the system back to a secure and safe state". As we see from the proposals above, some approaches have been suggested.

Ling and Feng propose in [57] an IDS that is listening to the CAN messages, and creates and updates a table of counters that indicate whether the message ID is from a known input set or unknown to the system. This approach requires the specification of all known message IDs that are sent via CAN and thresholds for known and unknown messages. With the use of flags and the threshold values, the system performs an alarm operation in case the counter exceeds the threshold. The algorithm was evaluated in a simulation environment using CANoe [57].

Cho and Shin propose in [58] an anomaly-based IDS called Clock-based IDS (CIDS) that measures the intervals of periodic in-vehicle messages and creates a fingerprint of the ECU. With this method, one is able to distinguish between several source ECUs due to each ECU's individual offset and skew. Thus, this approach is able to identify the ECU that is sending the malicious messages.

Kang et al. provide in [35] a novel approach for an IDS using a Deep Neural Network (DNN) to secure the in-vehicle network. The learning was achieved with an unsupervised pre-training method and following stochastic gradient descent method. The authors evaluated their proposed technique within a simulation environment (OCTANE [59]). The results show an accuracy of 98% on average [35].

5.3 Handling Intrusion Alerts

One crucial issue with intrusion detection is to decide what to do with an alarm. It may be possible to send the alarm to a central portal where a security officer takes care of it. However, it may not be realistic to assume that the portal should have such resources for the large number of cars connected. Further, the car may not be continuously connected to the portal. Thus, it seems more realistic to inform the driver of the alarms. Such an approach is proposed by Hoppe et al. [54] where various security-related events can be presented to the driver. Depending on the severity of the event, different methods are used: visual for non-critical events, acoustic for critical events, and haptic for severe events.

They also propose an "adaptive dynamic decision model". By using the sensors in the vehicle, the environment of the vehicle can be evaluated at the time of the alert. If the currently used ways of alerting the driver is not considered to be enough, the alert-level can be increased.

Since the communication within the vehicle is safety-critical, discarding the wrong message may have catastrophic effects. An IPS system is essentially an IDS system that can take some action, for example to stop a certain behavior. However, an attacker use the fact that an IPS system is present and force it to make incorrect decisions. Hoppe et al. discuss the problem of intrusion response and point out that an active response system might not be allowed to actively make decisions in the vehicle due to legal requirements for safety-critical systems [50].

5.4 Honeypots

A honeypot is a faked target that is intended to attract attacks in order to collect information and to analyze the attacks. To our knowledge, only one such approach has been described so far, by Verendel et al. [60]. It is suggested that the honeypot is attached to the gateway node in the vehicle and simulates the in-vehicle network. The data collected from the honeypot can then be sent to a common portal and analyzed in detail. The purpose of this is to learn about new attacks and distribute solutions as early as possible.

A very important property of the honeypot is how realistic the simulation of the target is. If the simulation is not realistic enough, the attacker will realize that he is not attacking a real vehicle. However, making a realistic honeypot may be very hard since it is not likely that a real vehicle is used for this purpose.

Chapter 6

Hardware and OS-level Security

This chapter provides an overview of the security features in modern microcontrollers suitable for the automotive domain, known side-channel attacks, and a summary of the fundamental security concepts in modern operating systems.

6.1 Hardware Security

Hardware for embedded systems traditionally deals with many more restrictions than general-purpose hardware. Therefore, advances in embedded hardware are often the result of reducing power consumption or costs of already known solutions. Since even in general-purpose hardware has been a shift towards more power efficient solutions, this trend will probably continue or accelerate.

There are a number of issues that are interesting with regard to hardware security, such as supported security features, known attacks that need to be mitigated and the current state-of-the-art in automotive microcontrollers. We will discuss these issues in the following Subsections.

6.1.1 Security related Microcontroller Features

As shortly discussed in Section 2.4, there are a number of security features that are supported by modern CPUs and microcontrollers.

One very obvious feature are hardware implemented cryptographic modules, e.g. hardware support for common algorithms such as AES, RSA or SHA-1, which can greatly speed up encryption or decryption routines.

Moreover, many microcontrollers can distinguish between executable and non-executable memory spaces, which makes certain kinds of attacks harder to perform (e.g. the traditional buffer overflow attack).

Virtualisation techniques known from the server and computer environment have moved to embedded systems as well. Embedded hypervisors, however, have, compared to traditional hypervisors, different attributes and requirements. Efficient memory manage-

ment and usage is important for embedded systems, as the memory resources are limited. The memory limitation requires small hypervisors, which have the advantages that their code is easier to validate and the code can be proven to be bugfree. In the automotive domain it is also highly important that the hypervisor provides a strong separation of applications, since a safety critical application may be executed on the same System on a Chip (SoC) as the infotainment application. Furthermore, an embedded microcontroller must be capable of handling the real time requirements [61].

6.1.2 Side Channel Attacks

Side channel attacks are one of the main types of attacks that hardware needs to guard against. They are generally very time consuming and costly, and an attacker needs to be quite skilled to pull it off. Common side channel attacks are differential power analysis or differential electromagnetic analysis (DEMA) [62]. Attacks such as rowhammer and speculative execution vulnerabilities, however, have shown that side-channel attacks would be possible on a large scale as well.

Rowhammer and Speculative Execution. More recent side-channel attacks are rowhammer [63], Spectre [64, 65, 66], and Meltdown [67]. Rowhammer allows attackers to cause bit flips in adjacent rows of specific DRAM devices, which can be further used for kernel privilege escalation. Spectre has gained wide attention as microprocessors from Intel, AMD, and ARM are vulnerable to this attack. The basic idea behind Spectre is to exploit a feature called *branch prediction and speculative execution*. This feature allows microcontrollers to maximise their performance by guessing the destination of a memory value and trying to execute in advance while this memory value is still being read. When the memory value is read, the CPU compares the results and either commits or discards the speculative computation. Attacks described by Kocher et. al [64] exploit this feature to an extend to be even able to read confidential information of any of the running programs of the victim through cache-based side-channels. Meltdown is also taking advantage of speculative execution and is even able to read kernel memory from user space. The principle behind meltdown is to measure the time an execution takes: a very fast computation infers that the attacker's code has been already executed by speculative execution, which can infer that a certain IF-statement in the attacker's code is true, whereas a longer processing time infers that the result is false, as it has not been executed by the speculative execution beforehand.

6.1.3 Tamper Detection Modules, TDMs

Tamper detection modules need to detect attempts to directly tamper with the hardware and trigger appropriate counter measures. For instance when the lid of an ECU is removed by unauthorized people, it may be appropriate to delete stored keys in order to prevent that they are stolen. However, it this is far from trivial.

6.1.4 Hardware Security Modules, HSMs

A hardware security module (or a Trusted platform module, TPM) contains security-critical functionality needed by other components of the vehicle, such as to protect private keys (used in asymmetric encryption), to distribute session keys, and to sign messages. It contains memory, a processor and software capable of performing basic cryptographic operations and preferably also a good random number generator. It should ideally be a tamper-proof device and be protected against physical access, i.e. that all attempts to access its content should make it useless.

A valid and reliable clock is needed by many services to avoid replays of messages and to detect reuse of older messages in other environments. The Hardware Security Module (HSM) module is a good place for such functionality and a timestamp can be applied to the message at the same time as it is digitally signed. This prevents individual ECUs which are compromised from sending and reusing old messages.

HSM functionality is implemented in special hardware similar to the well-known IBM 4578 cryptographic processor with tamper-resistant protection. It should have a well-defined API that can be used by ECUs to perform crypto-related operations such as to sign and verify signatures. Smart-cards have been proposed for some situations, but one problem with them is that they lack functionality such as offering a trusted time source. In situations where this is not needed, smart cards and RFID cards offering crypto-operations can be used, for example when identifying owners, drivers and service technicians.

HSMs have also been suggested to be used by internal ECUs in order to guarantee execution of authentic code. The functionality can be similar to Windows notebooks, where a TPM chip together with the first boot-loader (BIOS) verifies the integrity of the software before storing it and only authentic (genuine) software will be executed. This can be done in steps and even include signed applications from third party developers. All non-trusted and modified software, including malware, will be rejected by this system. There are many advantages with this solution, although the drawback is the increased complexity of the ECUs.

EVITA HSMs

Three different types of hardware security modules have been defined within the EVITA project: full, medium, and light in order to offer different levels of security functionality and performance.

- The full module is deployed in one or two high-performance communication ECUs in the vehicle, and has hardware for asymmetric cryptographic operations needed by more demanding external communications such as V2X communication. It is proposed to be used only in central communication gateways.

- The medium module is used in two to four central multi-purpose ECUs, such as Gateway ECUs isolating traffic between internal networks. It supports asymmetric cryptographic operations, but lacks hardware support and is less powerful than the full module.

- The light module is used in less powerful but still security-critical ECUs. It only has a hardware accelerated symmetric cryptographic engine, a hardware random number generator and a UTC clock. Its typical use is in sensors and actuators.

6.1.5 Event Data Recorders

Event data recorders (EDR) are devices that record important events and stores them in tamper-proof storage, similar to the black boxes used in aircrafts. It is reasonable to assume that government agencies and vehicle manufacturers will require devices like this to be present in all vehicles, when more advanced applications are introduced. The data recorded in the EDR makes it possible to investigate reasons behind crashes and other safety-critical events. Recording ECU states and communication traces when alerts from an IDS have been triggered is vital for post-attack analysis in order to (1) improve the security of the vehicles, and (2) finding the individual or organisation responsible.

6.1.6 Trusted Execution Environments, TEEs

Arm TrustZone[1] and Intel Software Guard Extensions (SGX)[2] are solutions for trusted execution for their respective computer architecture. Intel SGX allows developers to create so-called enclaves which contain small program fragments that are located in private memory regions and protected such that it is not possible for the OS and other running applications to access or interfere with its execution. Example scenarios for using enclaves are manifold, e.g., storing secrets, encryption/decryption of files, and signing/verifying authenticated data. Arm TrustZone (Arm TZ) provides a similar functionality, yet, with a slightly different approach. Arm TZ provides two worlds, an untrusted and a trusted world. Applications running in the trusted world, also TEE, are executed by a trusted OS which is the interface between the two worlds through Arm TZ.

6.1.7 State-of-the-art Automotive Microcontrollers

An example of a microcontroller for automotive applications is the Freescale MPC5646C. Other microcontrollers generally used are from the MPC55XX or MPC56XX series. The Freescale MPC5646 microcontroller provides next to Fast Ethernet, FlexRay and CAN controllers also a Cyrptographic Service Engine (CSE) which is needed in order to perform cryptographic operations efficiently.

More powerful ECUs used for central computations require hardware virtualisation support, security extensions, e.g., a secure storage and hardware acceleration for crypto-

[1] https://developer.arm.com/ip-products/security-ip/trustzone
[2] https://software.intel.com/en-us/sgx

graphic operation, and a trusted execution environment. Candidates for such ECUs are microcontrollers from the ARMv7 architecture family, i.e., ARM Cortex-A15[3].

6.2 OS-level Security

Security measures on OS-level are well studied in IT systems, however, their applicability in the automotive domain, resource constrained systems, has not been fully explored yet. A standard approach to security in operating systems is to use hierarchical protection domains, also known as ring protection. The idea is that outer layers of the ring have a lower privilege level than inner layers. A traditional example of this is the splitting into kernel mode and user mode as in UNIX systems.

Also highly intrinsic to modern operating systems is the separation of process memory. Each process is allocated a specific region in memory, and no process is allowed to access any other region in memory by default.

A more recent approach is to have executable space protection, i.e. to split programs into data and executable portions such that only the executable space is allowed to execute. This is usually accomplished together with hardware support.

Address Space Layout Randomization (ASLR) is another technique that was developed to prevent or at least hinder buffer overflow attacks. This is achieved by randomly arranging the positions of the stack, heap and libraries in the process address space. Furthermore, Lautenbach et al. [68] discuss and highlight how widely studied mitigation techniques from IT systems can be applied to resource constrained systems, such as ECUs.

A more efficient way of gaining isolation between applications compared to virtualisation is container-based virtualisation. This technique differs from virtualisation in the way, that the containers access a single kernel, which results in a more efficient resource usage. Nevertheless this method is not as strong as virtualisation with a hypervisor in terms of security for the reason that it is easier to access the entire system. A method to deploy hot updates in a container-based environment on an embedded system is described in [69].

Finally, OS loaders are often signed in order to prevent the execution of untrusted code (see Section 6.1.4). This is known as “secure boot”, and is highly pervasive in game consoles and computer operating systems.

6.3 Current Research

Research in the area of securing hardware and applications can be split in four major groups: (1) Hardware security, (2) System security / inter-process security, (3) intra-process security, and (4) virtualisation / inter-os security. Figure 6.1 shows a more detailed, yet not complete, view on the topics research is focussing on.

Trusted Execution Environments (TEEs) in combination with virtualisation and container-based solutions are compelling techniques that allow a more centralised exe-

[3] https://developer.arm.com/ip-products/processors/cortex-a/cortex-a15

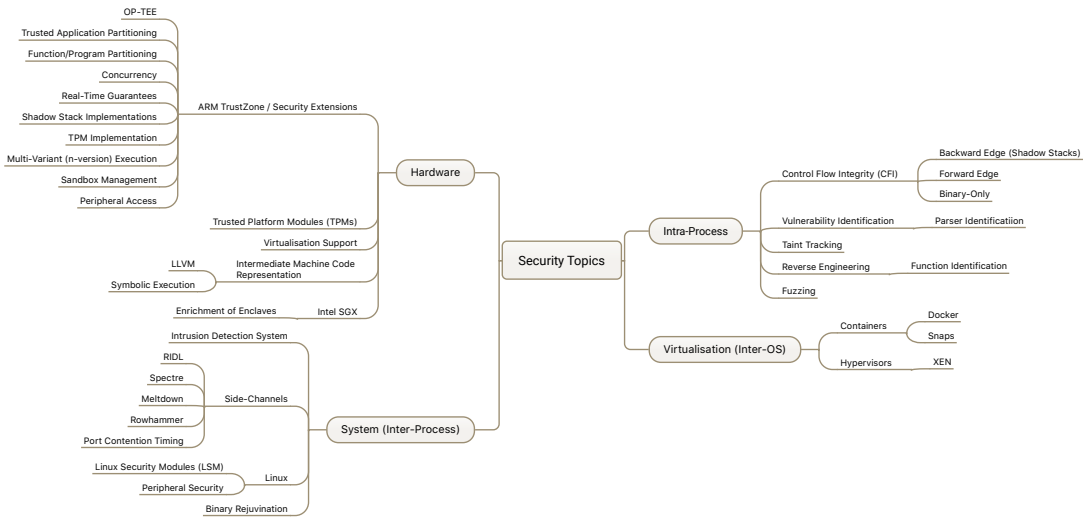

Figure 6.1: Research topics in the area of OS security.

cution of vehicular functions on a more powerful ECU while maintaining the required process separation. In regards to resiliency, however, we see the need for further investigation, as one needs to be able to guarantee the functioning of the vehicle even when an intrusion or other malicious behaviour has been detected.

6.4 Summary

This chapter listed several security mechanisms on hardware and software/OS level. HSMs securely store the private keys, offer cryptographic operations, and have a good random number generator. Additionally, HSMs have to be tamper-proof to ensure that the private keys cannot be stolen or accessed. The three different types of HSMs of the EVITA architecture describe how HSMs can be classified according to their functionality and purpose in the vehicle's system - do they provide hardware accelerated symmetric and asymmetric encryption? Microcontroller that have a built-in HSM are available and some manufacturers even rank their products according to the EVITA project. The use of signature verification of new firmware increases the complexity of the ECUs, but it is necessary for preventing attackers to upgrade an ECU to a modified firmware version.

More complex operating systems have multiple security mechanisms such as a ring layer around the kernel functions, separation of process memory, separation of executable and non-executable storage, booting of only signed firmware, and ASLR. The last mentioned technique hinders the attacker to predict target addresses. However, this method involves also the re-design of diagnostic devices, since they are reading system states from fixed address spaces of an ECU. virtualisation, which is common in the server

environment, is approaching also the embedded systems. The OS PikeOS[4], is specially designed for virtualisation on microcontroller and supports AUTOSAR. Prospective microcontrollers used in the automotive domain may also include multi-core support to allow running several OSs.

[4] http://www.opensynergy.com/en/products/pikeos/

Chapter 7

Resilient Architectures

7.1 Introduction

Resilience is literally defined as the act or action of "springing back". The definition of resilience has been revisited and elaborated in many fields (psychology, ecology, business, industrial safety, etc.). In contrast to largely agreed-upon terms in both the academic and industrial community, such as *vulnerability*, *intent*, *attack*, *capability*, and *risk*, the meaning of resilience in software systems is often still debated. The literature defines resilience in many different ways.

"Resilience is the ability of a system to absorb external stresses."

- ecological systems [70]

"Resilience is a system capability to create foresight, to recognize, to anticipate, and to defend against the changing shape of risk before adverse consequences occur."

- resilience engineering [71]

"Resilience refers to the inherent ability and adaptive responses of systems that enable them to avoid potential losses."

- economy [72]

"Resilience engineering is a paradigm for safety management that focuses on how to help people cope with complexity under pressure to achieve success."

- resilience engineering [73]

A common trait for all these definitions is the ability to accommodate unforeseen environmental perturbations or disturbances. Yet, none of the definition above mention the important notion of *time* as a factor of resilience. Haimes [74] uses the following definition of system resilience:

> *"Resilience is defined as the ability of the system to withstand a major disruption within acceptable degradation parameters and to recover within an acceptable time and composite costs and risks."*

The complexity of pinning down the meaning of resilience in systems is reflected in that resilience can not be measured using a single metric [75]. It is a multidimensional property of a system that changes over time. For instance, risk associated with an cyber attack depends not only of the resilience of the system but also on the sophistication and type of attack. Therefore, resilience can be only measured in the context of a specific threat, the system's recovery time, associated composite costs and risks. Analogically, Haimes [74] compares the resilience of a human body to system resilience. The analogy is useful for understanding that the resilience of an entire system is broken down to the so called "resilience hierarchy" of subsystems. It also ties in well with the fact that resilience depends on the system state at a given time and the type, as well as strength of an attack (e.g., resilience of a human body to a virus depends on the state of the body upon exposure, type of virus, and amount of virus the body is exposed to).

Resilience is related to other non-functional properties of the system. Below is a list of the relationships between resilience and other system properties, based on Haimes [74], Zhang and Lin [76], and Liu et al. [77].

Vulnerability. Vulnerability addresses only a system's protection, while the focus of resilience is also a system's recovery following an adverse event.

Readiness. If the primary objective of readiness is reducing the vulnerability of a system to specific threats, it may also (not necessarily) improve the resilience of the system to the same threats.

Risk. Improving a system's resilience offers significant advantages in managing risk and is an integral part of risk management processes. The fundamental benefit of building resilient systems is an acceptable residual risk. The benefits and trade-offs are subject to risk management and can be calculated in different ways.

Scalability. The main difference between scalability and resilience is that scalability deals with increasing the workload by adding resources to the system, whereas resilience is focused on system's response to partial failures due to unexpected workloads. Adding resources to the system may in some cases (not necessarily) contribute to system resilience. For instance, if heavy workload causes partial failures, then approaches for scalability will also benefit resilience.

Reliability and robustness. Reliability of a system refers to the system's ability to reliably perform it's functions under a nominal loading and predictable disturbance.

The impact of reliability on the system stops at the point of time that the system is damaged. Robustness refers to the the system's ability to perform its functions in an acceptable performance range under unpredictable disturbance. Therefore, robustness has impact on the system prior to its damage and concerns the strength of the system or in a more general sense. In comparison to reliability and robustness, system resilience is a 'post-damage' kind of property.

Survivability. Medvedovic et al. [78] define survivability as an ability to resist, recognize, recover from, and adapt to mission-compromising threats. Apart from the agent-based flavor, this definition of survivability is similar to resilience.

Dependability, fault tolerance. Dependability is defined as the ability to deliver service that can justifiably be trusted. Fault tolerance is defined as the ability to deliver service in the presence of faults. It is part of the means to achieve the dependability, and is achieved by means of: i) error detection, and ii) diagnosis and recovery. The system's ability to function after a software component has been damaged is a kind of fault tolerance and related to the resilience of that software component.

7.2 Design Principles for Resilience

This section given a brief overview of the design principles for resilience. In line with the definition of resilience [74], the software architecture design is resilient, when it accommodates the realization of mechanisms for achieving system resilience. Practice has shown that it is impossible to develop components that are completely resilient, therefore previous work has been done [77, 78] to compile a list of best practices for engineers to consider when designing for resilience. Software architectural styles that are related to the principles described below are Service Oriented Architectures, Microservice architectures, REST, distributed system architectures.

Redundancy. Redundancy refers to deploying into a system more component than are required for a particular functionality. This makes it possible to substitute a failed component with a new one to (partially) recover the lost functionality. It is the most fundamental approach for achieving resilience and reliability. In particular, *data redundancy* (in terms of form and location of data) is the most common form of redundancy in information systems. Similarly, *process* elements can also be redundant. Yet, process redundancy design requires dealing with the problem of process state (stateless processes vs restarting process from initial state upon failure). Finally, redundancy can be achieved by creating replicas of elements (operating in active and passive modes), or serendipitous redundancy (the substitute element can partially recover the lost functionality).

Partitioning. Partitioning refers to increasing the reliability of databases by splitting the data into small pieces and storing them in distributed databases. The reliability of each database remains unchanged, yet the reliability and availability of data as a whole can improve (failure of one database will not effect the availability of data on other partitions). In addition, partitioning provides an opportunity for partial failure isolation.

Virtualization. The functionalities of processing elements or data elements can be abstracted and virtualized into a service. A service is an autonomous technical functionality retrieving and processing information. With the service-orientation paradigm come Software as a Service (SaaS) and Infrastructure as a Service (IaaS) platforms, where entire software (infrastructures) is virtualized into a service. Loosely couples services lend themselves useful for runtime substitution and adaptation, in case of failures. There are existing frameworks for engineering resilience in service oriented architectures, e.g., [79].

Decentralized control. It is difficult to build a centralized control mechanism for a system with heavy redundancy due to the communication and processing load on the central control component. Therefore, decentralizing control is a commonly used strategy for such systems (e.g., peer-to-peer) to achieve resilience.

Explicit messaging. In a distributed system, implicit massage passing may lead to system flaws in a distributed environment (due to network latency and partial failure). Therefore, a common good practice is the use of explicit message passing, where the message sender, receiver, and intermediaries are loosely coupled with each other. If a processing element fails while the message is on the route, the computation task can be recovered based on the available messages.

Uniform interface. Designing uniform interfaces (either system-specific or using standard interfaces) has shown to bring better scalability, reusability and reliability to network-based systems. Uniform interfaces in a system refers to (i) having data, processes identified by one mechanisms, and data, processes are accessed via their identifiers, (ii) unified semantics of operations.

Self-management. A promising architectural strategy for achieving resilience is self-management, a concept borrowed from autonomic computing. Autonomic computing is to develop self-managing systems that can self-configure, self-heal, self-optimize, and self-protect. Autonomic computing components consist of the managed element and the manager, which uses knowledge to monitor, analyze, plan, and execute effects on the element (see also MAPE-K for designing self-adaptive systems). Recent approaches also include architecture-based self-adaptation solutions [80].

7.3 Resilience Design Patterns

Hukerikar et al. [81] have recently proposed a catalog of resilience design patterns which may be used by system architects when designing and deploying resilient solutions. These design patterns do not serve as a complete implemented solution, but rather describe a general repeatable solution to a commonly occurring problem. Figure 7.1 shows the proposed classification of the resilience design patterns. First, the authors separate behavioral (forward progress of the system) and state related patterns (data consistency). For a detailed description of the design strategies and patterns, we refer the reader to the original work.

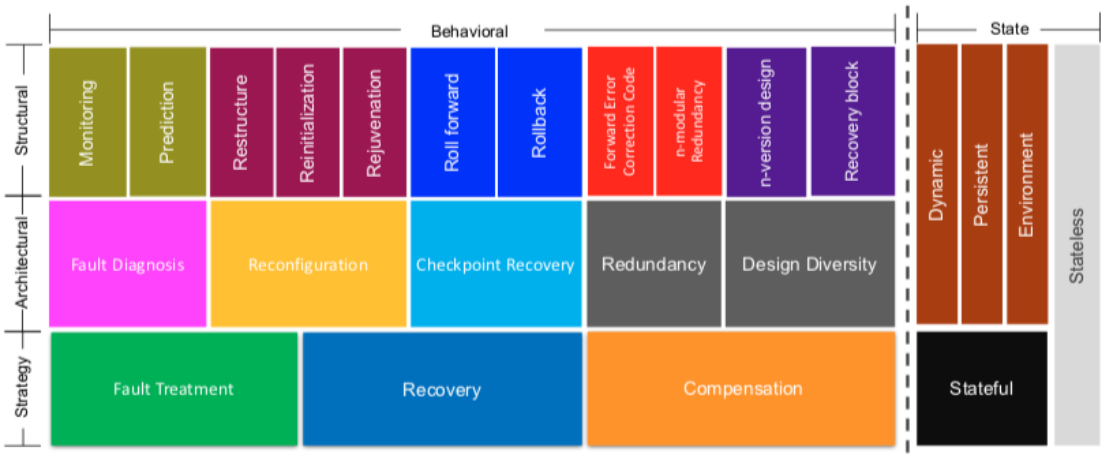

Figure 7.1: Classification of resilience design patterns, as proposed by Hukerikar et al. [81].

7.4 Resilient Architecture Solutions

First, we present existing frameworks, designs, and solutions for resilient architectures in the domain of Cyber-Physical Systems (CPS). Then we provide an overview of the contributions in the automotive domain. Finally, we mention several works from other domains, relevant for application in the automotive domain (real-time systems, distributed systems, and the like).

CPS. Andalam et al. [82] propose CLAIR, a contract-based framework for developing resilient cyber physical systems, with the focus on industrial control systems. Their approach is to introduce a resilience manager that oversees that the defined contracts are fulfilled and reacts in case they are violated. An example given is the object detection component on a conveyor belt. If it is unable to meet the deadline and, thus violates the time restriction in the contract, the resilience manager may switch to a less time-consuming detection algorithm. The implementation of the test bed is compliant with IEC 61499[1], which again highlights the focus of this work on industrial control systems.

Pradhan et al. [83] introduce a set of design-time analysis tools to perform timing, network QoS, and resilience analysis, and a runtime infrastructure that governs the self-reconfiguration mechanisms. In their efforts, they aim to improve the runtime support for resilience in mobile Cyber-Physical Systems, with a special focus on our runtime infrastructure that provides autonomous resilience via self-reconfiguration.

Automotive. Klingensmith et al. and Madni et al. [84, 85] look into resilience concepts for autonomous military vehicles. Their findings cannot be directly mapped to ordinary vehicles, nevertheless, it shows also the similarities between these two areas. They identify similar properties that contribute to resilience and continue describing the different types

[1] https://webstore.iec.ch/publication/5506

of resilience. *Network resilience* is challenging to achieve. System of systems require communication as they cannot exist without network communication and at the same time the network can be considered as one of the largest attack surfaces. Sterbez et al. [86] provide an architectural framework for resilience and survivability in communication networks. *Systems Architecture and Resilience* focusses much on vehicles/systems being able to adapt to changes based on new knowledge. The main discussion in *Human-Systems Integration and Resilience* is how soldiers should be able to interact with such highly autonomous vehicles. The authors highlight, for instance, that humans should be kept in the loop when creative solutions and rapid cognition are required. This applies to the automotive industry to a certain extent as well. Drivers are still responsible and need to be aware of the situation in case all safety, security, and resilience measures fail to prevent an attack.

An approach for creating security resilience in AUTOSAR has been presented by Nasser et al. [87]. The authors first focus on safety mechanisms that can be exploited (see also [88]) and thus need to be protected against attackers. Then the authors looked whether safety mechanisms can be used to improve security and resilience. An example of their findings is the use of the so-called AUTOSAR Watchdog Manager (WdgM) as a control flow integrity (CFI) measure. The watchdog manager can be used to define specific code elements that can be monitored in regard to the order of execution.

A roadside architecture to secure wireless sensor networks for intelligent transport systems and to provide resilience is presented by Bohli et al [89]. The architecture aims at supporting accident prevention and post-accident investigation. The authors cover a wide range of topics ranging from key derivation, multicast authentication, concealed data aggregation, access control to premium services, to protocols for post-accident investigation. Resilience in this paper is mostly about data resilience – dealing with nodes transmitting wrong information.

Service-oriented architectures (and microservice architectures) may be good candidates for building resilient architectures. This architectural style focuses on decomposing the system to a set of services that are easier to reconfigure, which is required for achieving resilience in systems. In the automotive domain, service-orientation is novel, but has been postulated as a promising approach in recent studies [90, 91, 92].

Other domains. Alho and Mattila [93] present a service-oriented architecture that improves fault tolerance. The authors discuss the challenges of designing a reliable service-oriented architecture to real time systems. The introduced service manager detects faults with a heartbeat signal, service status updates, resource monitoring and monitoring of exit signals. The fault escalation is split in four levels: (I) soft restart, (II) hard restart, (III) switch to backup service, and (IV) terminate the service. Their approach has been tested on a test bed consisting of two PCs with Pentium 4 processor acting as controller, a virtual reality pc and robot controllers. The authors further highlight the limitations, such as the recovery methods are based on restarts and do not deal with the root cause, and safety considerations.

Werner et al. [94] have recently proposed and approach for increasing resilience of existing system implementations by applying software transformations. The authors present a framework for firmware generation and inclusion of safety measures in the generated code.

Many recent publications introduce design solutions [95] and frameworks [96] for achieving cross-layer resilience in systems. The idea of cross-layer resilience refers to systems where error resilience techniques from various layers of the system stack cooperate to achieve cost-effective resilience. While [96] introduce a systematic approach to explore various resilience techniques (and their cost effectiveness) across the implementation stack, Mitra et al. [97] provide an overview of resilience techniques (e.g., various techniques for Soft Error Resilience).

Finally, we mention the Resilience Engineering Framework (REF), introduced by the Software Engineering Institute (SEI) [98] which brings forward measures for assessing the state of system resilience. SEI also introduced a Resilience Management Model [99] for managing operational resilience in complex, risk-evolving environments.

Chapter 8

Evaluation and Certification of Security Functionality

8.1 A Framework for assessing Security

Although there is a lot of research going on in vehicular systems, we have found very little research referring to models of the connected car and how to assess the security of emerging vehicle services, e.g. remote diagnostics, remote software download, and other internet services brought into future vehicles.

The Car-2-Car Communication Consortium (C2C-CC) have created a reference architecture which is divided into three domains; the *in–vehicle*, the *ad hoc* and the *infrastructure* domains [100]. The in-vehicle domain is represented by the vehicle, its applications and mobile devices directly associated to the vehicle. The ad hoc domain is represented by the vehicles and the RSUs, where the RSUs further can be connected to the infrastructure domain. In their architecture, the access network, the internet, and possible nodes connected to the internet are shown as part of the infrastructure domain.

A more detailed framework for security assessment has been developed in [11] which consists of *a model* for the infrastructure of the connected car and *a security assessment tree*. Such a framework can help to understand and evaluate how secure protocols and applications should be evaluated and designed in different vehicle settings [101, 102, 103]. The proposed model together with the security assessment tree makes it easier to identify the weaknesses of the system and the existence of threats both when designing new services and when assessing security as a whole.

This model is shown in Figure 8.1. *The infrastructure* is divided into two domains, the managed infrastructure and the vehicle communication domain. The managed infrastructure is further divided into five regions: automotive company applications' center, third party applications' center, trusted network, untrusted network, and the internet

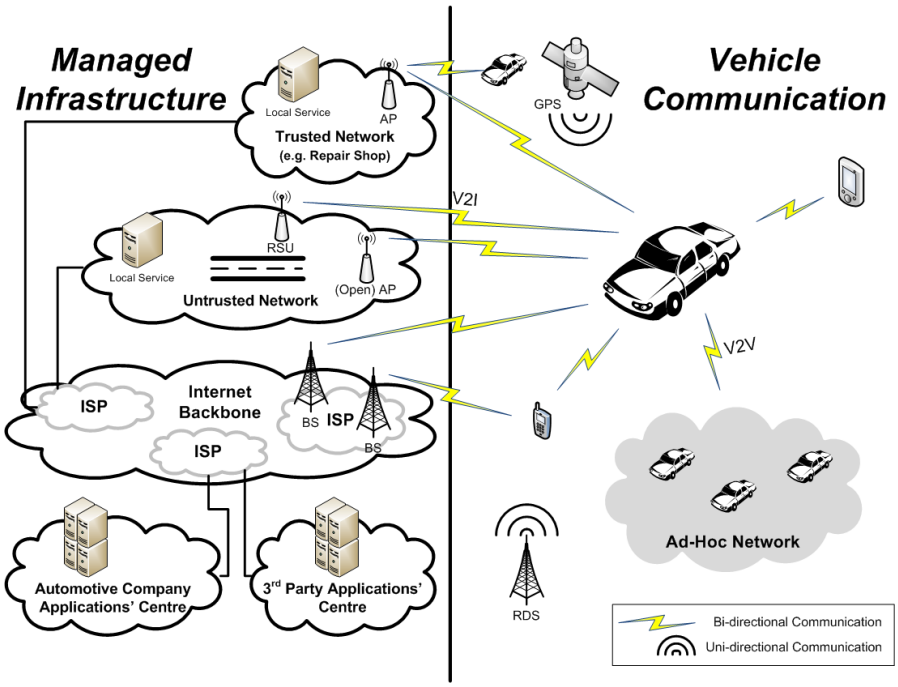

Figure 8.1: Communication scenarios and trust relationships

backbone. *The vehicle communication* describes the possible means of communication with the vehicle. These concepts are further explained in the following paragraphs.

Kiening et al. propose in [104] the use of trust assurance levels. These levels are used to verify the protection level of the sender. The levels are from 0 to 4 and indicate the trust in the information. Thus, a node is able to use the received information combined with the knowledge of the trust assurance level. For instance, safety critical information has to be provided with a certain trust assurance level in order to be further processed. A level of 0 implies that no security mechanisms have been used. 1 entails that software mechanisms for security are implemented. Level 2 involves hardware and software security mechanisms. The security of directly connected components is guaranteed in level 3 and level 4 implies the protection of all components involved in V2X applications [104].

8.1.1 Managed Infrastructure

The five regions of the managed infrastructure show different levels of trust which may require different protection mechanisms for transmitted data.

- *Automotive Company Applications' Center*. In the literature, the automotive company applications' center has different names, e.g. a *portal* or a *remote service center*. To summarize, it consists of a set of servers providing services to their vehicles. It holds necessary information about the vehicle, such as information from previous

services (e.g., diagnostics data), configuration data, cryptographic keys, as well as new software available for the ECUs.

- *Third Party Applications' Center*. Apart from services provided by the automotive company, third party services can be provided to the vehicle. We could imagine that large "application stores" for vehicles will be available in the future. These applications can provide any kind of service to the vehicle.

- *Trusted Network*. Some networks can be considered to be trusted by the applications' centers and the vehicle. For example, a repair shop may be considered to be a trusted network by the automotive company and the vehicle. In delivering a service to this network, it may well be that some requirements in an implementation can be relaxed.

  An example where the security requirements in the implemented service can be relaxed is when performing remote diagnostics over a wireless network in a repair shop. If appropriate link layer encryption is applied, the security of the wireless communication could be considered to be equal to that of a cable. This will not be the case when the communication with the vehicle is performed through the Untrusted Network; here end-to-end application encryption might be the only choice.

- *Untrusted Network*. All networks, except for the trusted networks, are considered to be untrusted. In these networks, the services provided to the vehicle have to be adapted to the hostile environment of the internet. In the same way as for the trusted networks, other local services may also be provided in these networks.

- *Internet Backbone*. The internet backbone, with its ISPs, is the core network for connecting the other four regions together. A backbone network is usually well protected and operated by network specialists in a networks operations center, NOC. Therefore, when network traffic has reached the internet backbone, it is assumed in the model that it is unlikely that the data will be intentionally modified.

8.1.2 Vehicle communication

The vehicle communication domain describes the possible communication means between the vehicle and the managed infrastructure and with other objects. The following communications scenarios exist.

- *Vehicle to Wireless AP*. The vehicle can establish a connection to a wireless AP. All open APs (hotspots) are considered to be part of the untrusted network. Furthermore, a protected AP, where the vehicle needs authentication keys, can be available in both trusted networks and in untrusted networks.

- *Vehicle to RSUs*. RSUs can also be used to establish a connection from the vehicle to other networks in the managed infrastructure.

- *Vehicle to Cellular Base Stations*. A mobile data network, can be used to establish a connection from the vehicle to the internet.
- *Vehicle to Mobile Devices*. Mobile devices can be connected to the vehicle. For example, a connection can be established to a mobile phone, a laptop, or a PDA. Furthermore, the vehicle can also act as a gateway for the mobile device, so that the mobile device can reach the same network as the vehicle.
- *Vehicle to Cellular Base Station via a Mobile Device*. If the vehicle lacks the possibility to connect directly to a cellular base station, another mobile device with a connection to the cellular base station can be used as a gateway. One example is to use the driver's mobile phone.
- *Vehicle to other Vehicles*. Finally, the vehicle can connect to other vehicles and create a VANET. This V2V communication will be critical in future traffic- and safety-related services.

It should be noted that the description of the vehicle communication above is based on just one vehicle; any connected car will have the same communication surroundings. This means that the vehicle may possibly reach the managed infrastructure, via other vehicles or other mobile devices acting as gateways.

8.1.3 Using the framework to assess the security of vehicle services

From the model of the infrastructure of the connected car, there are different aspects that can be discussed regarding the V2V and the V2I communication. One of them is the security of the services delivered to the vehicle. Figure 2.1 presents a brief taxonomy of the security of these services. Four categories are described: the *actors*, the *V2X communication technologies*, *network paths*, and the *dependability and security attributes*. A description of them follows below.

- *Actors*. Six different actors that can be involved in a service have been identified. Common for them all are that they have interests in how the service is being designed and delivered; the automotive company and the application provider can state requirements, the car owner and the driver can have concerns on how the data from a service is processed, the authorities can issue legal requirements, and an attacker can try to manipulate the service in an unwanted way.
- *V2X Communication Technologies*. A number of communication technologies are available for connecting the vehicle to other devices. Examples of these are listed. An extended list, including classifications of the communication technologies, can be found in [105].
- *Network Paths*. The service may be delivered to the vehicle using one of several network paths. The model describes four possible network paths that the service

can be delivered through (see Figure 8.1): the trusted network, the untrusted network, the internet backbone, and an ad-hoc network.

- *Dependability and Security Attributes*. To deliver the service in a secure and safe manner, the six attributes for dependability and security need to be considered [106].

From these four categories, an analysis can be made to further clarify how a service will work in the infrastructure, and also highlight the dependability and security attributes that need to be addressed in providing such a service.

With the security assessment tree, the problem with the vast number of issues that need to be considered in securing a service, is identified. It helps us to state requirements regarding security and provides us with a framework and a template for security assessment by identifying threats and communication patterns and to define countermeasures.

8.2 Certifications

Even if standards for security design are developed and legal requirements are defined, we still face the problem of *guaranteeing* that a particular design fulfils all requirements. Certification of individual components and functions using procedures such as *Common Criteria* (CC) and *FIPS* may be a way to, at least guarantee a sound design. Such certifications may be useful for individual components in a vehicle, but will not be applicable to a whole network in a car since any change of the software or hardware requires a full re-certification of all software in the vehicle, a very time consuming and work-intensive process.

It is reasonable to assume that all V2X traffic is sent by stations similar to ETSI's ITS station that fulfil a wide range of security requirements, and such components used by many car manufacturers are good targets for certification. The use of standardized and certified communication nodes for V2X short range communications have several advantages, not just for interoperability, but cooperative development of common modules can enhance security significantly and enable certification to be done.

Our conclusion is therefore that certifications are to some degree useful but come with several shortcomings: First of all, only smaller components can be certified to higher levels, otherwise the increased complexity makes it impossible to prove any functionality. Therefore, only individual components, such as firewall functionality in a subsystem can be certified, but not a full vehicle with hundreds of communicating ECUs. Second, to require only certified products in vehicles may prevent deployment of new functionality and create unacceptable delays of patches to known problems.

8.2.1 Common Criteria Certification

Common Criteria certification is an ISO standard often used when security devices are evaluated. It uses different *Evaluation Assurance Levels* (EAL1 to EAL7) with different requirements on the *target of evaluation*, TOE. The higher the level is, the higher are

the requirements. The targets are evaluated according to some criteria, *claims*, often specified in a *Security Target* written by the organization that wants to certify a device. It is not possible to compare two devices even if they both have been certified at the same level if the evaluation criteria differ. Therefore, for some types of devices such as firewalls and operating systems, there exist a pre-defined sets of criteria they should be validated against, called *Protection Profiles* to enable such a comparison. It also means that with weak validation criteria, it is possible to achieve a high assurance level but not necessary having a more secure device. A good example of this is that Microsoft Windows XP was certified at Level 4 (EAL 4) which is probably as high as a complex system like an operating system will reach, but it is still not considered to be a secure platform at all. Therefore, by relaxing the the criteria to evaluate against, it is possible to reach a high level in the certification.

The larger and more complex a system is, the harder it becomes to certify. Levels EAL1 to EAL4 are possible to reach with targets like conventional operating systems and products based on (executing in) such systems. Up to this level, it is enough to be able to motivate that it is reasonable to assume that the target of evaluation can fulfill the criteria and have a reasonably sound design, since level EAL4 states that the target should be methodically designed, tested, and reviewed. This is mostly done through documentation which states how and by what modules in the code the functions are implemented. Higher levels (EAL5-EAL6) require semi-formal design and verification which soon becomes too complex for larger targets, and EAL7 requires formally verified design and verification.

Even if it would be possible to certify the functionality of a full vehicle at level EAL4, a problem still exists when new functionality is added or patches are applied. A re-certification is needed as soon as *any* functionality or code is changed. Delta certifications are permitted where only the changes and documentation that shows that the changes do not affect other security-sensitive functions, are re-evaluated, but a complete re-evaluation is needed on regular basis. This makes it is more or less impossible (or at least extremely costly and time consuming) to certify a full vehicle with all its components at any higher level and maintain that certification level over time. Another problem is that the a high certification of a complex target takes time and will significantly affect time to market in a negative way.

Our conclusion is therefore that certification of smaller components or devices within the vehicle make sense. Such devices could be TPM modules or isolated components that perform, for example, V2X communications. Preferably these devices are used by many car manufacturers which motivates the time, effort and cost to perform and maintain the certification. Work is also needed to create proper protection profiles that are useful for all security sensitive devices used in the vehicular domain.

Chapter 9

Research Projects

The European Commission research and innovation programs fund several framework programs with a variety of topics. Horizon 2020 is the currently ongoing program where the majority of the R&D activities within ITS are handled. Projects from the 7^{th} Framework Program (2007–2013) are ongoing or already finished. Some of the more influential research projects within FP6 and FP7 are introduced here. ITS related projects of Horizon 2020 are listed in Section 9.7.

The mentioned projects are almost all geographically located in Europe, and their outcome mainly affect ETSI (European Telecommunications Standards Institute) and therefore also end up in ISO for standardization.

9.1 CARONTE (Creating an Agenda for Research on Transportation sEcurity)

Creating an Agenda for Research on Transportation sEcurity (CARONTE) [107] was a European project from 2014 - 2016. The project's aim is to create a strategic research plan for security in land transport. The results of this project are the identification and classification of research in this particular area. The identified top priorities for land transportation security are [107]:

- staying operational in the event of a cyber-incident,
- timely and efficient threat detection, and
- special security problems of railways as open systems (an integrated approach).

The results highlight the necessity of a secure system even though a cyber-incident occured. The proposed research title for this topic is "An integrated approach to assure cyber resilience in land transportation".

9.2 HEAVENS (HEAling Vulnerabilities to ENhance Software Security and Safety)

The HEAling Vulnerabilities to ENhance Software Security and Safety (HEAVENS) project (2013 - 2016) can be seen as the predecessor of the HoliSec project. The project's findings were the introduction of basic dependability and security concepts, basic concepts of system engineering processes from security perspective, baseline architectures of ITS, in-vehicle network and in-vehicle software to be used in the HEAVENS project, and the identification, explanation of use cases that lead to a better understanding of security needs in the automotive Electrical and/or Electronic (E/E) systems, and the security requirements based on the needs derived from the use cases [108].

The HEAVENS security model has also received much attention from standardization organizations, such as SAE J3061 [109], AUTOSAR WP-X-SEC and National Highway Traffic Safety Administration (NHTSA). This model is a systematic approach that includes methods, processes and tool support to derive security requirements and to perform security testing and evaluation for automotive E/E systems.

9.3 HoliSec (Holistic Approach to Improve Data Security)

HoliSec, 2016-2019, addresses security in the automotive lifecycle during the concept, design, testing and the operational phase. New solutions for threat assessment have been proposed, a framework to move from security levels to mandatory security mechanisms has been introduced and a data-driven solution for an IDS for the in-vehicle network using the CAN bus has been presented. Furthermore, the project has been also looking into the interplay between safety and security and has also provided an overview of the current and future trends regarding bug bounties in the automotive domain.

9.4 SeFram

SeFram, 2012-2015, was a Swedish research project in which Volvo Cars Corporation and Chalmers are the main partners. The goals of the project were:

- securing communication within the car and between the car and the outside infrastructure,
- developing models and define security requirements for vehicle communication, and
- creating a practically useful framework for future security work within the connected car.

The project is a continuation of earlier projects in this area (SIGYN I and II). The project has resulted in an in-depth analysis of how to offer secure remote diagnostics for vehicles, and especially securing the repair shop and its communications. A new protocol using a trusted third party has been developed with the intention to protect vehicles

against unauthorized access from compromised diagnostics units. The protocol has been formally verified to be correct and able to guarantee some characteristics. Work has also been done in protecting the repair shop network from unauthorized access by vehicles, i.e. from access by vehicles not scheduled for service or remote diagnostics.

Another task being performed within this project was to create a test implementation of the ETSI ITS station. The implementation has revealed several problems with the standard and weaknesses in the description of essential parts. The revealed problems are described in [110] and lead to a modification of the ETSI ITS specification.

9.5 SESAMO (Security and Safety Modelling)

SESAMO [111], 2012-2015, is a European project that addresses the problems arising with convergence of safety and security in embedded systems at architectural level, where subtle and poorly understood interactions between functional safety and security mechanisms impede system definition, development, certification, and accreditation procedures and standards.

9.6 EVITA (E-safety Vehicle Intrusion protected Applications)

EVITA [4] was a European project, funded under the 7th Framework Program (FP7) 2008-2011. Its main objective was to design, verify and implement a hardware security module to be used in an architecture for securing on-board networks. In this architecture, security relevant components are protected against tampering and sensitive data are protected against compromise. By focusing on protecting the intra-vehicle communication, EVITA complements other projects which mainly focus on external (V2X) communication.

In EVITA, an architecture is created which should enable ECUs to implement cryptography operations in a secure manner. The ECU is equipped with a cryptographic co-processor protected in a HSM. This module is responsible for performing all cryptography applications. More details on the HSM can be found in Section 6.1.4.

9.7 European Projects - Horizon 2020

Several projects funded under Horizon 2020 are about securing Cyber Physical System (CPS). The automotive sector is thereby considered in many projects as a use case or as area of interest. Table 9.1 lists related projects in course of Horizon 2020.

Table 9.1: European Projects in the course of Horizon 2020 related to the automotive sector

Project Name	**Description**
SAFURE [112] 2015 – 2018	*SAFety and secURity by design for interconnected mixed-critical cyber-physical systems:* The automotive sector is mentioned as one of three use cases for the proof-of-concept. The results will be a framework with the capability to detect, prevent and protect from security threats on safety, able to monitor from application level down to the hardware level potential attacks to system integrity from time, energy, temperature and data threats; methodology that supports design of safety and security of embedded systems (inlc. tools and modelling language extensions); proof-of concept through three use cases; specifications to design and develop SAFURE compliant products.
SafeCOP [113] 2016 – 2019	*Safe Cooperating Cyber-Physical Systems using Wireless Communication:* SafeCOP will establish a safety assurance approach, a platform architecture, and tools for cost-efficient and practical certification of cooperating CPS (CO-CPS). SafeCOP defines a platform architecture and develops methods and tools, which will be used to produce safetey assurance evidence needed to certify cooperative functions. SafeCOP will extend current wireless technologies to ensure safe and secure cooperation. The project contributes to new standards and regulations by providing scientifically validated solutions.
SCOUT [114] 2016 – 2019	*Safe and COnnected aUtomation in road Transport:* Identifies pathways for an accelerated spread of safe and connected high-degree automated driving in Europe. The project considers: the user needs and expectations, technical and non-technical gaps and risks, and viable business models. It gathers use cases for connected automation in road transport, assesses and ranks the non-technical and technical gaps, and risks for the implementation of these use cases. Sustainable business models will be also identified. Furthermore, advice for policies and regulatory frameworks for safe and connected automation will then be derived on the basis of the aforementioned analysis with the support of a network of relevant stakeholders, and classified into a common roadmap aiming to implement the formulated vision for "Safe and connected automation in 2030".

SHARCS [115] 2015 – 2018	*Secure Hardware-Software Architectures for Robust Computing Systems:* A framework for designing, building and demonstrating secure-by-design applications and services, that achieve end-to-end security for their users. SHARCS will achieve this by systematically analyzing and extending, as necessary, the hardware and software layers in a computing system. The framework will be assessed by using a diverse set of security-critical, real-world applications (from different domains: medical, cloud and automotive).
CYRail [116] 2016 – 2018	*Cybersecurity in the RAILway sector:* This project is concerned with the security of CPS in the area of railways. It aims to deliver tailored specifications and recommendations for secure modern rail systems design and operation. The challenges are related to the automotive industry: wide and distributed geographical display limit the traditional cyber-protection and cyber-defence tools & practices, heterogeneous nature of rail systems attacks; passenger-centric services may expose rail systems to threats.
IMMORTAL [117] 2015 – 2018	*Integrated Modelling, Fault Management, Verification and Reliable Design Environment for Cyber-Physical Systems:* Development of an integrated, cross-layer modelling based tool framework for fault management, verification and reliable design of dependable CPS.
5GCAR [118] 2017 – 2019	*5G Communication Automotive Research and innovation:* The focus of this project is to use the upcoming 5G technology for V2V and V2X communication. However, they also looked into how privacy and security can benefit from 5G.

Chapter 10

Conclusions

This deliverable consists of a discussion about several demonstrated security threats and their implications. It is shown why security and resilience are important in the automotive domain. How the different security aspects have been addressed by various research projects is described in this document as well. We also commented on the practicability of proposed solutions. The report covers security in the internal (in-vehicle) as well as in the external (V2X) communication.

Securing the in-vehicle communication with traditional mechanisms has been discussed. These mechanisms are among others internal separation of traffic, the use of message authentication codes (MACs) to guarantee message integrity, firewalls for both external traffic and for internal traffic implemented in gateway ECUs, use of IDSs, use of certificates for identification and verification of devices or software (vehicles, road-side objects, ECUs, drivers, and firmware), and the problems with distributing revocation lists (Certificate Revocation Lists (CRLs)). Tamper-proof hardware security modules (HSM modules) to speed-up cryptographic operations and to offer a safe storage for private or symmetric keys have been discussed as well.

The goal of this document was to write an objective and comprehensive summary of this interesting field. We hope that the work will lead to an increased understanding and be an inspiration to future work to make road traffic even safer and more secure than it is today.

New vehicle applications will soon be introduced that communicate with applications in other vehicles and road-side units (V2X). Examples are applications that process information messages from traffic lights, road signs and other vehicles in urban traffic. Other applications will be offered by car manufacturers, such as remote software updates, remote diagnostics and other services for car owners and drivers. Other applications will be offered by third parties, for example applications for automatic road toll payments, navigational systems and automatically transmitted vehicle reports about current road conditions. Security work must address how to implement these applications in a secure and safe way, how to isolate safety critical functions from all “nice to have” applications

and implement protection against attacks, internal and external, that may compromise the vehicle. In short, we must find ways to guarantee the safety of the vehicle.

Standardization work has begun although the work has mainly focused on low-level communications and protocols. Security design and architecture has only been addressed to some degree, for example in the ITS station standardization work done by ETSI. But it still remains to be seen if the proposed standards will be universally accepted by the industry and all countries in the world. We provided an overview over current standardization efforts and what they provide us with respect to security.

Internal security is more or less absent in vehicles. In this report, we also discuss how to prevent and detect if a device with access to the internal bus has sent falsified messages. Furthermore, we provide a summary of state-of-the-art research in the area of resilience and resilient architectures.

Bibliography

[1] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway et al. 'Experimental Security Analysis of a Modern Automobile'. In: *2010 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2010, pp. 447–462. ISBN: 1424468949. DOI: 10.1109/SP.2010.34 (cit. on pp. 1, 18).

[2] QualComm. *eCall Whitepaper, v1.5*. QualComm. Mar. 2009. URL: http://ec.europa.eu/information_society/activities/esafety/doc/ecall/pos_papers_impact_assessm/qualcomm.pdf (cit. on p. 3).

[3] National Highway Traffic Safety Administration (NHTSA). *U.S. DOT advances deployment of Connected Vehicle Technology to prevent hundreds of thousands of crashes*. URL: https://www.nhtsa.gov/press-releases/us-dot-advances-deployment-connected-vehicle-technology-prevent-hundreds-thousands (visited on 19th Dec. 2016) (cit. on p. 4).

[4] EVITA Project. *E-safety Vehicle Intrusion Protected Applications (EVITA)*. URL: http://www.evita-project.org/ (visited on 14th Nov. 2016) (cit. on pp. 7, 10, 65).

[5] ETSI. *Intelligent Transport Systems (ITS); Security; Threat, Vulnerability and Risk Analysis (TVRA)*. Technical Report TR 102 893, v1.1.1. 650 Route des Lucioles, F-06921 Sophia Antipolis Cedex, France: ETSI, Mar. 2010 (cit. on pp. 7, 16).

[6] K. Tuma, G. Çalikli and R. Scandariato. 'Threat analysis of software systems: A systematic literature review'. In: *Journal of Systems and Software* 144 (2018), pp. 275–294 (cit. on p. 7).

[7] K. Tuma and R. Scandariato. 'Two Architectural Threat Analysis Techniques Compared'. In: *European Conference on Software Architecture*. Springer. 2018, pp. 347–363 (cit. on p. 7).

[8] K. Tuma, R. Scandariato, M. Widman and C. Sandberg. 'Towards security threats that matter'. In: *Computer Security*. Springer, 2017, pp. 47–62 (cit. on p. 7).

[9] T. Rosenstatter. and T. Olovsson. 'Open Problems when Mapping Automotive Security Levels to System Requirements'. In: *Proceedings of the 4th International Conference on Vehicle Technology and Intelligent Transport Systems - Volume 1: VEHITS,* INSTICC. SciTePress, 2018, pp. 251–260. ISBN: 978-989-758-293-6. DOI: 10.5220/0006665302510260 (cit. on p. 7).

[10] T. Rosenstatter and T. Olovsson. 'Towards a Standardized Mapping from Automotive Security Levels to Security Mechanisms'. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. Nov. 2018, pp. 1501–1507. DOI: 10.1109/ITSC.2018.8569679 (cit. on p. 7).

[11] P. Kleberger, A. Javaheri, T. Olovsson and E. Jonsson. 'A Framework for Assessing the Security of the Connected Car Infrastructure'. In: *Proceedings of the Sixth International Conference on Systems and Networks Communications (ICSNC 2011)*. Barcelona, Spain, Oct. 2011, pp. 236–241. ISBN: 978-1-61208-166-3. URL: http://www.thinkmind.org/index.php?view=article&articleid=icsnc_2011_10_30_20229 (cit. on pp. 8, 57).

[12] SysSec. *D6.1: Report on the State of the Art of Securit in Sensor Networks*. URL: http://www.syssec-project.eu/m/page-media/3/syssec-d6.1-SoA-SecurityInSensorNetworks.pdf (visited on 30th Nov. 2013) (cit. on p. 11).

[13] H. Lee, H.-M. Tsai and O. Tonguz. 'On the Security of Intra-Car Wireless Sensor Networks'. In: *Vehicular Technology Conference Fall (VTC 2009-Fall), 2009 IEEE 70th*. 2009, pp. 1–5. DOI: 10.1109/VETECF.2009.5378964 (cit. on p. 11).

[14] B. Preneel. 'The First 30 Years of Cryptographic Hash Functions and the NIST SHA-3 Competition'. In: *Topics in Cryptology - CT-RSA 2010*. Ed. by J. Pieprzyk. Lecture Notes in Computer Science 5985. Springer Berlin Heidelberg, Jan. 2010, pp. 1–14. ISBN: 978-3-642-11924-8, 978-3-642-11925-5. URL: http://link.springer.com/chapter/10.1007/978-3-642-11925-5_1 (visited on 28th Nov. 2013) (cit. on p. 13).

[15] European Commission. *M/453 EN: Standardisation mandate addressed to CEN, CENELEC and ETSI in the field of Information and Communication Technologies to support the interoperability of Co-operative systems for Intelligent Transport in the European Community*. European Commission. Oct. 2009 (cit. on p. 16).

[16] ETSI. *Intelligent Transport Systems (ITS); Communications Architecture*. European Standard EN 302 665, v1.1.1. 650 Route des Lucioles, F-06921 Sophia Antipolis Cedex, France: ETSI, Sept. 2010 (cit. on p. 16).

[17] ETSI. *Intelligent Transport Systems (ITS); Security; Security Services and Architecture*. Technical Specification TS 102 731, v1.1.1. 650 Route des Lucioles, F-06921 Sophia Antipolis Cedex, France: ETSI, Sept. 2010 (cit. on p. 16).

[18] J. Hoagland, O. Whitehouse, T. Newsham, M. Conover and O. Friedrichs. 'Vista's Network Attack Surface'. Presented at CanSecWest. Apr. 2007. URL: http://hoagland.org/presentations/CanSecWest07-Vista-Ntw-Attack-Surface.pdf (cit. on p. 17).

[19] S. Habib, C. Jacob and T. Olovsson. 'An Analysis of the Robustness and Stability of the Network Stack in Symbian-based Smartphones'. In: *Journal of Networks* 4.10 (2009), pp. 968–975. DOI: 10.4304/jnw.4.10.968-975. URL: http://ojs.academypublisher.com/index.php/jnw/article/view/0410968975 (cit. on p. 17).

[20] A. Lang, J. Dittmann, S. Kiltz and T. Hoppe. 'Future Perspectives: The Car and Its IP-Address — A Potential Safety and Security Risk Assessment'. In: *Proceedings of the 26th International Conference on Computer Safety, Reliability, and Security (SAFECOMP '07)*. SAFECOMP '07. Nuremberg, Germany, Sept. 2007, pp. 40–53 (cit. on p. 17).

[21] *UT Austin Researchers Successfully Spoof an $80 million Yacht at Sea - UT News - UT News*. https://news.utexas.edu/2013/07/29/ut-austin-researchers-successfully-spoof-an-80-million-yacht-at-sea/. (Accessed on 10/11/2019) (cit. on p. 18).

[22] D. P. Shepard, J. A. Bhatti and T. E. Humphreys. 'Drone hack: Spoofing attack demonstration on a civilian unmanned aerial vehicle'. In: (2012) (cit. on p. 18).

[23] K. C. Zeng, Y. Shu, S. Liu, Y. Dou and Y. Yang. 'A practical GPS location spoofing attack in road navigation scenario'. In: *Proceedings of the 18th International Workshop on Mobile Computing Systems and Applications*. ACM. 2017, pp. 85–90 (cit. on p. 18).

[24] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner and T. Kohno. 'Comprehensive Experimental Analyses of Automotive Attack Surfaces'. In: *Proceedings of the 20th USENIX Security Symposium*. San Francisco, CA, USA, Aug. 2011, pp. 77–92 (cit. on p. 18).

[25] I. Rouf, R. Miller, H. Mustafa, T. Taylor, S. Oh, W. Xu, M. Gruteser, W. Trappe and I. Seskar. 'Security and Privacy Vulnerabilities of In-Car Wireless Networks: A Tire Pressure Monitoring System Case Study'. In: *Proceedings of the 19th USENIX conference on Security*. USENIX Security'10. Washington, DC: USENIX Association, 2010, pp. 21–21. ISBN: 888-7-6666-5555-4. URL: http://dl.acm.org/citation.cfm?id=1929820.1929848 (cit. on p. 19).

[26] T. Fox-Brewster and Forbes. *BMW Update Kills Bug In 2.2 Million Cars That Left Doors Wide Open To Hackers*. Feb. 2015. URL: http://www.forbes.com/sites/thomasbrewster/2015/02/02/bmw-door-hacking (visited on 12th Dec. 2016) (cit. on p. 20).

[27] A. Greenberg and Forbes. *Hackers Reveal Nasty New Car Attacks–With Me Behind The Wheel*. Aug. 2013. URL: http://www.forbes.com/sites/andygreenberg/2013/07/24/hackers-reveal-nasty-new-car-attacks-with-me-behind-the-wheel-video/#1990f3e85bf2 (visited on 8th Dec. 2016) (cit. on p. 20).

[28] C. Miller and C. Valasek. 'Remote Exploitation of an Unaltered Passenger Vehicle'. In: *Blackhat USA* 2015 (2015), pp. 1–91. URL: http://illmatics.com/Remote%20Car%20Hacking.pdf (cit. on pp. 20–22).

[29] wired.com. *The Jeep Hackers Are Back to Prove Car Hacking Can Get Much Worse*. URL: https://www.wired.com/2016/08/jeep-hackers-return-high-speed-steering-acceleration-hacks/ (visited on 8th Dec. 2016) (cit. on p. 22).

[30] Vulnerability Lab. *BMW ConnectedDrive - (Update) VIN Session Vulnerability*. July 2016. URL: https://www.vulnerability-lab.com/get_content.php?id=1736 (visited on 17th Nov. 2016) (cit. on p. 22).

[31] Vulnerability Lab. *BMW - (Token) Client Side Cross Site Scripting Vulnerability*. July 2016. URL: https://www.vulnerability-lab.com/get_content.php?id=1737 (visited on 17th Nov. 2016) (cit. on p. 22).

[32] Keen Security Lab of Tencent. *Car Hacking Research: Remote Attack Tesla Motors*. Sept. 2016. URL: http://keenlab.tencent.com/en/2016/09/19/Keen-Security-Lab-of-Tencent-Car-Hacking-Research-Remote-Attack-to-Tesla-Cars/ (visited on 17th Nov. 2016) (cit. on p. 22).

[33] Keen Security Lab of Tencent. *Experimental Security Research of Tesla Autopilot*. 2019. URL: https://keenlab.tencent.com/en/whitepapers/Experimental_Security_Research_of_Tesla_Autopilot.pdf (visited on 10th Nov. 2019) (cit. on p. 22).

[34] Promon AS. *Updates and precisions on the Tesla hack*. Nov. 2016. URL: https://promon.co/blog/updates-precisions-tesla-hack/ (visited on 25th Nov. 2016) (cit. on p. 23).

[35] M.-j. Kang and J.-w. Kang. 'A Novel Intrusion Detection Method Using Deep Neural Network for In-Vehicle Network Security'. In: *PLoS ONE* 11.6 (2016), pp. 1–17. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0155781 (cit. on pp. 26, 38).

[36] MOST Cooperation. 'MOST Specification Rev. 3.0 E2'. In: (2010), p. 32. URL: http://www.mostcooperation.com/ (cit. on p. 26).

[37] D. J. Gerhardt. 'Serial Data Communications Between Microprocessor Systems'. In: *SAE Technical Paper*. SAE International, Apr. 1986. DOI: 10.4271/860728. URL: http://dx.doi.org/10.4271/860728 (cit. on p. 27).

[38] SAE International. 'Serial Control and Communications Heavy Duty Vehicle Network - Top Level Document Superseding J1939 JUN2012'. In: SAE International, Aug. 2013. URL: https://saemobilus.sae.org/content/J1939_201308 (cit. on pp. 27, 28).

[39] M. Junger and Vector Informatik GmbH. *Introduction to J1939 - Application Note AN-ION-1-3100*. Apr. 2010. URL: https://vector.com/portal/medien/cmc/application_notes/AN-ION-1-3100_Introduction_to_J1939.pdf (visited on 8th Dec. 2016) (cit. on p. 27).

[40] Copperhill technologies. *A Brief Introduction to the SAE J1939 Protocol*. 2016. URL: http://copperhilltech.com/a-brief-introduction-to-the-sae-j1939-protocol/ (visited on 9th Dec. 2016) (cit. on p. 28).

[41] 'Automotive Ethernet: In-vehicle Networking and Smart Mobility'. In: *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2013* (2013), pp. 1735–1739. ISSN: 15301591. DOI: 10.7873/DATE.2013.349. URL: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6513795 (cit. on p. 28).

[42] L. L. Bello. 'Novel trends in automotive networks: A perspective on Ethernet and the IEEE Audio Video Bridging'. In: *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)*. Sept. 2014, pp. 1–8. DOI: 10.1109/ETFA.2014.7005251 (cit. on p. 29).

[43] IEEE-Standards Association. *802.1BA-2011 - IEEE Standard for Local and metropolitan area networks– Audio Video Bridging (AVB) Systems*. Tech. rep. 2011 (cit. on p. 29).

[44] IEEE-Standards Association. *P802.1AS - Standard for Local and Metropolitan Area Networks - Timing and Synchronization for Time-Sensitive Applications*. Tech. rep. 2011 (cit. on p. 29).

[45] IEEE-Standards Association. *802.1Qav-2009 - IEEE Standard for Local and metropolitan area networks– Virtual Bridged Local Area Networks Amendment 12: Forwarding and Queuing Enhancements for Time-Sensitive Streams*. Tech. rep. 2009 (cit. on p. 29).

[46] IEEE-Standards Association. *802.1Qat-2010 - IEEE Standard for Local and metropolitan area networks– Virtual Bridged Local Area Networks Amendment 14: Stream Reservation Protocol (SRP)*. Tech. rep. 2010 (cit. on p. 29).

[47] B. Jesse and Vector Informatik GmbH. *Introduction of Audio/Video Bridging (AVB) over Ethernet in Vehicles – Embedded Software Architecture, Specifics and Use Cases*. 2015. URL: https://vector.com/portal/medien/cmc/events/Webinars/2015/Vector_Webinar_Audio_Video_Bridging_20150612_EN.pdf (visited on 6th Dec. 2016) (cit. on p. 29).

[48] TTTech Computertechnik. 'TTEthernet – A Powerful Network Solution for Multiple Purpose From Ethernet to TTEthernet'. In: *Deterministic Real-Time Ethernet Platform* (2016), pp. 1–14 (cit. on p. 29).

[49] U. E. Larson and D. K. Nilsson. 'Securing Vehicles against Cyber Attacks'. In: *CSIIRW '08: Proceedings of the 4th annual workshop on Cyber security and information intelligence research*. CSIIRW '08. 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. New York, NY, USA: ACM, 2008, 30:1–30:3. ISBN: 978-1-60558-098-2. DOI: 10.1145/1413140.1413174. URL: http://doi.acm.org/10.1145/1413140.1413174 (cit. on pp. 35, 36).

[50] T. Hoppe, S. Kiltz and J. Dittmann. 'Applying Intrusion Detection to Automotive IT — Early Insights and Remaining Challenges'. In: *Journal of Information Assurance and Security* 4.3 (2009), pp. 226–235. URL: http://www.mirlabs.org/jias/hoppe.pdf (cit. on pp. 36, 38).

[51] T. Hoppe, S. Kiltz and J. Dittmann. 'Security Threats to Automotive CAN Networks - Practical Examples and Selected Short-Term Countermeasures'. In: *Proceedings of the 27th International Conference on Computer Safety, Reliability, and Security (SAFECOMP '08)*. SAFECOMP '08. Newcastle upon Tyne, UK, Sept. 2008, pp. 235–248 (cit. on p. 36).

[52] T. Matsumoto, M. Hata, M. Tanabe, K. Yoshioka and K. Oishi. 'A Method of Preventing Unauthorized Data Transmission in Controller Area Network'. In: *Vehicular Technology Conference (VTC Spring), 2012 IEEE 75th*. May 2012, pp. 1–5. DOI: 10.1109/VETECS.2012.6240294 (cit. on p. 36).

[53] M. Müter, A. Groll and F. C. Freiling. 'A Structured Approach to Anomaly Detection for In-Vehicle Networks'. In: *2010 Sixth International Conference on Information Assurance and Security (IAS)*. Atlanta, GA, Aug. 2010, pp. 92–98. DOI: 10.1109/ISIAS.2010.5604050 (cit. on p. 36).

[54] T. Hoppe, S. Kiltz and J. Dittmann. 'Adaptive Dynamic Reaction to Automotive IT Security Incidents Using Multimedia Car Environment'. In: *Proceedings of the 4th International Conference on Information Assurance and Security (ISIAS '08)*. Sept. 2008, pp. 295–298. DOI: 10.1109/IAS.2008.45 (cit. on pp. 37, 38).

[55] M. Müter and N. Asaj. 'Entropy-Based Anomaly Detection for In-Vehicle Networks'. In: *2011 IEEE Intelligent Vehicles Symposium (IV)*. Baden-Baden, Germany, June 2011, pp. 1110–1115. DOI: 10.1109/IVS.2011.5940552 (cit. on p. 37).

[56] F. Kargl, P. Papadimitratos, L. Buttyan, M. Muter, E. Schoch, B. Wiedersheim et al. 'Secure Vehicular Communication Systems: Implementation, Performance, and Research Challenges'. In: *IEEE Communications Magazine* 46.11 (Nov. 2008), pp. 110–118. DOI: 10.1109/MCOM.2008.4689253 (cit. on p. 37).

[57] C. Ling and D. Feng. 'An Algorithm for Detection of Malicious Messages on CAN Buses'. In: *2012 National Conference on Information Technology and Computer Science. Atlantis Press*. 2012 (cit. on p. 37).

[58] K.-T. Cho and K. G. Shin. 'Fingerprinting Electronic Control Units for Vehicle Intrusion Detection'. In: *25th USENIX Security Symposium (USENIX Security 16)*. Austin, TX: USENIX Association, Aug. 2016, pp. 911–927. ISBN: 978-1-931971-32-4 (cit. on p. 38).

[59] P. Borazjani, C. Everett and D. McCoy. 'OCTANE: An Extensible Open Source Car Security Testbed'. In: *Proceedings of the Embedded Security in Cars Conference*. 2014 (cit. on p. 38).

[60] V. Verendel, D. K. Nilsson, U. E. Larson and E. Jonsson. 'An Approach to using Honeypots in In-Vehicle Networks'. In: *Proceedings of the 68th IEEE Vehicular Technology Conference (VTC)*. Sept. 2008, pp. 1–5 (cit. on p. 38).

[61] M. Tim Jones and IBM Corporation. *Virtualization for embedded systems: The how and why of small-device hypervisors*. Apr. 2011. URL: https://www.ibm.com/developerworks/library/l-embedded-virtualization/ (visited on 12th Dec. 2016) (cit. on p. 42).

[62] K. Gandolfi, C. Mourtel and F. Olivier. 'Electromagnetic Analysis: Concrete Results'. English. In: *Cryptographic Hardware and Embedded Systems — CHES 2001*. Ed. by Ç. Koç, D. Naccache and C. Paar. Vol. 2162. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2001, pp. 251–261. ISBN: 978-3-540-42521-2. DOI: 10.1007/3-540-44709-1_21. URL: http://dx.doi.org/10.1007/3-540-44709-1_21 (cit. on p. 42).

[63] M. Seaborn and T. Dullien. *Exploiting the DRAM rowhammer bug to gain kernel privileges*. https://googleprojectzero.blogspot.com/2015/03/exploiting-dram-rowhammer-bug-to-gain.html. Accessed: 2019-07-31. 2015 (cit. on p. 42).

[64] P. Kocher, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz and Y. Yarom. 'Spectre attacks: Exploiting speculative execution'. In: *arXiv preprint arXiv:1801.01203* (2018) (cit. on p. 42).

[65] *CVE-2017-5753*. Available from NIST NVD CVE-2017-5753. 2018. URL: https://nvd.nist.gov/vuln/detail/CVE-2017-5753 (visited on 31st July 2019) (cit. on p. 42).

[66] *CVE-2017-5715*. Available from NIST NVD CVE-2017-5715. 2018. URL: https://nvd.nist.gov/vuln/detail/CVE-2017-5715 (visited on 31st July 2019) (cit. on p. 42).

[67] M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, A. Fogh et al. 'Meltdown: Reading Kernel Memory from User Space'. In: *27th USENIX Security Symposium (USENIX Security 18)*. Baltimore, MD: USENIX Association, 2018, pp. 973–990. ISBN: 978-1-939133-04-5. URL: https://www.usenix.org/conference/usenixsecurity18/presentation/lipp (cit. on p. 42).

[68] A. Lautenbach, M. Almgren and T. Olovsson. 'What the Stack? On Memory Exploitation and Protection in Resource Constrained Automotive Systems'. In: *International Conference on Critical Information Infrastructures Security*. Springer. 2017, pp. 185–193 (cit. on p. 45).

[69] S. Aidanpää and E. Mk Nordmark. 'Flexible Updates of Embedded Systems Using Containers'. MA thesis. KTH Royal Institute of Technology, 2016, p. 112 (cit. on p. 45).

[70] C. S. Holling. 'Resilience and stability of ecological systems'. In: *Annual review of ecology and systematics* 4.1 (1973), pp. 1–23 (cit. on p. 49).

[71] D. Woods and R. Cook. *Incidents–markers of resilience or brittleness. Resilience Engineering. Concepts and Precepts*. 2006 (cit. on p. 49).

[72] A. Rose and S.-Y. Liao. 'Modeling regional economic resilience to disasters: A computable general equilibrium analysis of water service disruptions'. In: *Journal of Regional Science* 45.1 (2005), pp. 75–112 (cit. on p. 49).

[73] E. Hollnagel, D. D. Woods and N. Leveson. *Resilience engineering: Concepts and precepts*. Ashgate Publishing, Ltd., 2006 (cit. on p. 49).

[74] Y. Y. Haimes. 'On the definition of resilience in systems'. In: *Risk Analysis: An International Journal* 29.4 (2009), pp. 498–501 (cit. on pp. 50, 51).

[75] D. Henry and J. E. Ramirez-Marquez. 'Generic metrics and quantitative approaches for system resilience as a function of time'. In: *Reliability Engineering & System Safety* 99 (2012), pp. 114–122 (cit. on p. 50).

[76] W.-J. Zhang and Y. Lin. 'On the principle of design of resilient systems–application to enterprise information systems'. In: *Enterprise Information Systems* 4.2 (2010), pp. 99–110 (cit. on p. 50).

[77] D. Liu, R. Deters and W. J. Zhang. 'Architectural design for resilience'. In: *Enterprise Information Systems* 4.2 (2010), pp. 137–152. DOI: 10.1080/17517570903067751. eprint: https://doi.org/10.1080/17517570903067751. URL: https://doi.org/10.1080/17517570903067751 (cit. on pp. 50, 51).

[78] N. Medvidovic and R. N. Taylor. 'Software architecture: foundations, theory, and practice'. In: *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 2*. ACM. 2010, pp. 471–472 (cit. on p. 51).

[79] O. Erol, M. Mansouri and B. Sauser. 'A framework for enterprise resilience using service oriented Architecture approach'. In: *2009 3rd Annual IEEE Systems Conference*. IEEE. 2009, pp. 127–132 (cit. on p. 52).

[80] J. Cámara, P. Correia, R. de Lemos and M. Vieira. 'Empirical resilience evaluation of an architecture-based self-adaptive software system'. In: *Proceedings of the 10th international ACM Sigsoft conference on Quality of software architectures*. ACM. 2014, pp. 63–72 (cit. on p. 52).

[81] S. Hukerikar and C. Engelmann. 'Resilience design patterns: A structured approach to resilience at extreme scale'. In: *arXiv preprint arXiv:1708.07422* (2017) (cit. on pp. 52, 53).

[82] S. Andalam, D. J. X. Ng, A. Easwaran and K. Thangamariappan. 'CLAIR: A Contract-Based Framework for Developing Resilient CPS Architectures'. In: *2018 IEEE 21st International Symposium on Real-Time Distributed Computing (ISORC)*. May 2018, pp. 33–41. DOI: 10.1109/ISORC.2018.00013 (cit. on p. 53).

[83] S. Pradhan, A. Dubey, T. Levendovszky, P. S. Kumar, W. A. Emfinger, D. Balasubramanian, W. Otte and G. Karsai. 'Achieving resilience in distributed software systems via self-reconfiguration'. In: *Journal of Systems and Software* 122 (2016), pp. 344–363 (cit. on p. 53).

[84] K. Klingensmith and A. M. Madni. 'Resilience Concepts for Architecting an Autonomous Military Vehicle System-of-Systems'. In: *Disciplinary Convergence in Systems Engineering Research*. Ed. by A. M. Madni, B. Boehm, R. G. Ghanem, D. Erwin and M. J. Wheaton. Cham: Springer International Publishing, 2018, pp. 65–81. ISBN: 978-3-319-62217-0 (cit. on p. 53).

[85] A. M. Madni and S. Jackson. 'Towards a Conceptual Framework for Resilience Engineering'. In: *IEEE Systems Journal* 3.2 (June 2009), pp. 181–191. ISSN: 1932-8184. DOI: 10.1109/JSYST.2009.2017397 (cit. on p. 53).

[86] J. P. Sterbenz, D. Hutchison, E. K. Çetinkaya, A. Jabbar, J. P. Rohrer, M. Schöller and P. Smith. 'Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines'. In: *Computer Networks* 54.8 (2010), pp. 1245–1265 (cit. on p. 54).

[87] A. M. Nasser, D. Ma and P. Muralidharan. 'An Approach for Building Security Resilience in AUTOSAR Based Safety Critical Systems'. In: *Journal of Cyber Security and Mobility* 6.3 (2017), pp. 271–304 (cit. on p. 54).

[88] A. M. K. Nasser, D. Ma and S. Lauzon. 'Exploiting AUTOSAR Safety Mechanisms to Launch Security Attacks'. In: *Network and System Security*. Ed. by Z. Yan, R. Molva, W. Mazurczyk and R. Kantola. Cham: Springer International Publishing, 2017, pp. 73–86. ISBN: 978-3-319-64701-2 (cit. on p. 54).

[89] J.-M. Bohli, A. Hessler, O. Ugus and D. Westhoff. 'A Secure and Resilient WSN Roadside Architecture for Intelligent Transport Systems'. In: *Proceedings of the First ACM Conference on Wireless Network Security*. WiSec '08. Alexandria, VA, USA: ACM, 2008, pp. 161–171. ISBN: 978-1-59593-814-5. DOI: 10.1145/1352533.1352562. URL: http://doi.acm.org/10.1145/1352533.1352562 (cit. on p. 54).

[90] A. Iwai and M. Aoyama. 'Automotive cloud service systems based on service-oriented architecture and its evaluation'. In: *2011 IEEE 4th International Conference on Cloud Computing*. IEEE. 2011, pp. 638–645 (cit. on p. 54).

[91] M. Traub, A. Maier and K. L. Barbehön. 'Future automotive architecture and the impact of IT trends'. In: *IEEE Software* 34.3 (2017), pp. 27–32 (cit. on p. 54).

[92] S. Kugele, P. Obergfell, M. Broy, O. Creighton, M. Traub and W. Hopfensitz. 'On service-orientation for automotive software'. In: *2017 IEEE International Conference on Software Architecture (ICSA)*. IEEE. 2017, pp. 193–202 (cit. on p. 54).

[93] P. Alho and J. Mattila. 'Service-oriented approach to fault tolerance in CPSs'. In: *Journal of Systems and Software* 105 (2015), pp. 1–17. ISSN: 0164-1212. DOI: https://doi.org/10.1016/j.jss.2015.03.041. URL: http://www.sciencedirect.com/science/article/pii/S0164121215000643 (cit. on p. 54).

[94] M. Werner, K. Devarajegowda, M. Chaari and W. Ecker. 'Increasing Soft Error Resilience by Software Transformation'. In: *Proceedings of the 56th Annual Design Automation Conference 2019*. ACM. 2019, p. 199 (cit. on p. 55).

[95] S. S. Sahoo, B. Veeravalli and A. Kumar. 'Cross-layer fault-tolerant design of real-time systems'. In: *2016 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*. IEEE. 2016, pp. 63–68 (cit. on p. 55).

[96] E. Cheng, J. Abraham, P. Bose, A. Buyuktosunoglu, D. Chen, H. Cho, Y. Li, U. Sharif, K. Skadron, M. Stan et al. 'Cross-Layer Resilience: Challenges, Insights, and the Road Ahead'. In: *Proceedings of the 56th Annual Design Automation Conference 2019*. ACM. 2019, p. 198 (cit. on p. 55).

[97] S. Mitra, P. Bose, E. Cheng, C.-Y. Cher, H. Cho, R. Joshi, Y. M. Kim, C. R. Lefurgy, Y. Li, K. P. Rodbell et al. 'The resilience wall: Cross-layer solution strategies'. In: *Proceedings of Technical Program-2014 International Symposium on VLSI Technology, Systems and Application (VLSI-TSA)*. IEEE. 2014, pp. 1–11 (cit. on p. 55).

[98] R. A. Caralli, J. F. Stevens, C. M. Wallen, D. W. White, W. R. Wilson and L. R. Young. *Introducing the CERT Resiliency Engineering Framework: Improving the Security and Sustainability Processes*. Tech. rep. Software Engineering Institute (SEI), 2007 (cit. on p. 55).

[99] R. A. Caralli, J. H. Allen, P. D. Curtis, D. W. White and L. R. Young. *CERT Resilience Management Model, Version 1.0*. Tech. rep. Software Engineering Institute (SEI), 2010 (cit. on p. 55).

[100] CAR 2 CAR Communication Consortium. *C2C-CC Manifesto*. v1.1. Aug. 2007. URL: http://www.car-to-car.org/ (visited on 19th Dec. 2016) (cit. on p. 57).

[101] G. Karagiannis, O. Altintas, E. Ekici, G. Heijenk, B. Jarupan, K. Lin and T. Weil. 'Vehicular Networking: A Survey and Tutorial on Requirements, Architectures, Challenges, Standards and Solutions'. In: *IEEE Communications Surveys & Tutorials* 13.4 (2011), pp. 584–616. DOI: 10.1109/SURV.2011.061411.00019 (cit. on p. 57).

[102] T. L. Willke, P. Tientrakool and N. F. Maxemchuk. 'A Survey of Inter-Vehicle Communication Protocols and Their Applications'. In: *IEEE Communications Surveys & Tutorials* 11.2 (2009), pp. 3–20. DOI: 10.1109/SURV.2009.090202 (cit. on p. 57).

[103] M. L. Sichitiu and M. Kihl. 'Inter-Vehicle Communication Systems: A Survey'. In: *IEEE Communications Surveys & Tutorials* 10.2 (2008), pp. 88–105. DOI: 10.1109/COMST.2008.4564481 (cit. on p. 57).

[104] 'Trust assurance levels of cybercars in v2x communication'. In: *Proceedings of the 2013 ACM workshop on Security, privacy & dependability for cyber vehicles* (2013), pp. 49–60. ISSN: 15437221. DOI: 10.1145/2517968.2517974. URL: http://dl.acm.org/citation.cfm?id=2517974 (cit. on p. 58).

[105] K. Dar, M. Bakhouya, J. Gaber, M. Wack and P. Lorenz. 'Wireless Communication Technologies for ITS Applications'. In: *IEEE Communications Magazine* 48.5 (2010), pp. 156–162. DOI: 10.1109/MCOM.2010.5458377 (cit. on p. 60).

[106] A. Avivzienis, J.-C. Laprie, B. Randell and C. Landwehr. 'Basic Concepts and Taxonomy of Dependable and Secure Computing'. In: *IEEE Transactions on Dependable and Secure Computing* 1.1 (2004), pp. 11–33. DOI: 10.1109/TDSC.2004.2 (cit. on p. 61).

[107] CARONTE Project. *Creating an Agenda for Research On Transportation sEcurity*. URL: http://www.caronte-project.eu/ (visited on 14th Nov. 2016) (cit. on p. 63).

[108] *HEAVENS: HEAling Vulnerabilities to ENhance Software Security and Safety*. http://www.vinnova.se/sv/Resultat/Projekt/Effekta/HEAVENS-HEAling-Vulnerabilities-to-ENhance-Software-Security-and-Safety/. Accessed: 2016-11-25 (cit. on p. 64).

[109] *SAE J3061 Cybersecurity Guide-book for Cyber-Physical Automotive Systems*. Standard. Vehicle Electrical System Security Committee, 2016 (cit. on p. 64).

[110] N. Nowdehi and T. Olovsson. 'Experiences from implementing the ETSI ITS SecuredMessage service'. In: *IEEE Intelligent Vehicles Symposium, Proceedings* Iv (2014), pp. 1055–1060. DOI: 10.1109/IVS.2014.6856587 (cit. on p. 65).

[111] SESAMO. *Security and Safety Modeling (SESAMO)*. URL: http://www.sesamo-project.eu (visited on 19th Dec. 2016) (cit. on p. 65).

[112] CORDIS - Community Research and Development Information Service. *SAFURE - SAFety and secURity by design for interconnected mixed-critical cyber-physical systems*. URL: http://cordis.europa.eu/project/rcn/194149_en.html (visited on 8th Dec. 2016) (cit. on p. 66).

[113] CORDIS - Community Research and Development Information Service. *SafeCOP - Safe Cooperating Cyber-Physical Systems using Wireless Communication*. URL: http://cordis.europa.eu/project/rcn/203404_en.html (visited on 8th Dec. 2016) (cit. on p. 66).

[114] CORDIS - Community Research and Development Information Service. *SCOUT - Safe and COnnected aUtomation in road Transport*. URL: http://cordis.europa.eu/project/rcn/204978_en.html (visited on 8th Dec. 2016) (cit. on p. 66).

[115] CORDIS - Community Research and Development Information Service. *SHARCS - Secure Hardware-Software Architectures for Robust Computing Systems*. URL: http://cordis.europa.eu/project/rcn/194217_en.html (visited on 8th Dec. 2016) (cit. on p. 67).

[116] CORDIS - Community Research and Development Information Service. *CYRail - Cybersecurity in the RAILway sector*. URL: http://cordis.europa.eu/project/rcn/206014_en.html (visited on 8th Dec. 2016) (cit. on p. 67).

[117] CORDIS - Community Research and Development Information Service. *IMMORTAL - Integrated Modelling, Fault Management, Verification and Reliable Design Environment for Cyber-Physical Systems*. URL: http://cordis.europa.eu/project/rcn/194260_en.html (visited on 8th Dec. 2016) (cit. on p. 67).

[118] CORDIS - Community Research and Development Information Service. *5GCAR - 5G Communication Automotive Research and innovation*. URL: https://cordis.europa.eu/project/rcn/211068/factsheet/en (visited on 25th Aug. 2019) (cit. on p. 67).

