DEGREE PROJECT IN COMPUTER SCIENCE AND COMPUTER ENGINEERING,
SECOND LEVEL
STOCKHOLM, SWEDEN 2017

Security Analysis of Ethernet in Cars

AMMAR TALIC

KTH ROYAL INSTITUTE OF TECHNOLOGY
INFORMATION AND COMMUNICATION TECHNOLOGY

Security Analysis of Ethernet in Cars

Ammar Talic

2017-10-07

Master's Thesis

Examiner
Gerald Q. Maguire Jr.

Academic adviser
Anders Västberg

Company Supervisor:
Roland Hocke, Vector Informatik GmbH Stuttgart, Germany

KTH Royal Institute of Technology
School of Information and Communication Technology (ICT)
Department of Communication Systems
SE-100 44 Stockholm, Sweden

Abstract

With the development of advanced driving assistance systems, the amount of data that needs to be transmitted within a car has increased tremendously. Traditional communication bus based systems are unable to meet today's requirements; hence automotive Ethernet is being developed and standardized.

Ethernet has for many years been the *de facto* standard in interconnecting computers. In that time several vulnerabilities of the networking protocol stack implementations and even the protocols themselves have been discovered. The knowledge from exploiting computer networks can be applied to the automotive domain. Additionally, vehicle manufacturers tend to implement their own stacks, due to copyleft reasons; hence the chances of implementation faults increases as opposed to using well-tested open source solutions. Since the line between security and safety in cars is almost nonexistent, security has to be properly addressed.

This thesis investigates the security of automotive Ethernet and its accompanying protocols. It starts with an introduction to computer and automotive networking and protocols. After a solid foundation is laid, it investigates what makes up automotive Ethernet, its application in the field, and the automotive specific components relying on it. After looking at related work, a data network security audit and analysis as defined by the open-source security testing methodology is performed. The system is graded with risk assessment values. Weak points are identified and improvements suggested. The impact of the proposed improvements is shown by reevaluating the system and recalculating the risk assessment values.

These efforts further the ultimate goal of achieving increased safety of all traffic participants.

Keywords

Automotive security, penetration testing, automotive Ethernet, TCP/IP, DoIP, BroadR-Reach, OSSTMM, flash bootloader

Sammanfattning

Med utvecklingen av avancerade körningsassisterande system har mängden data som behöver sändas inom en bil ökat enormt. Traditionella kommunikationsbussbaserade system kan inte uppfylla dagens krav. Därmed utvecklas och standardiseras Ethernet för fordon.

Ethernet har i många år varit de facto-standarden i sammankopplandet mellan datorer. Under den tiden har flera sårbarheter hos nätverksprotokolls implementeringar och protokoll själva upptäckts. Det finns anledning att tro att kunskapen från att utnyttja datanätverk kan tillämpas på fordonsdomänen. Att tillägga är att fordonstillverkare tenderar att genomföra sina egna staplar. På grund av copyleft skäl, ökar chanserna för implementeringsfel i motsats till att använda testade open source-lösningar. Eftersom människors säkerhet hos bilar är extremt viktigt, måste även dess system hanteras ordentligt.

Denna avhandling undersöker säkerheten för Ethernet och kompletterande protokoll hos bilar. Den börjar med en introduktion till datorers och bilars nätverk och protokoll. Efter en stabil grund fastställts, undersöker den vad som utgör Ethernet hos bilar, dess tillämpning inom fältet, och de bilspecifika komponenterna den beror av. Efter att ha tittat på relaterat arbete utförs en säkerhetsgranskning och analys av datanätverk som definieras av säkerhetsmetoden för open-source. Systemet värderas med riskbedömningsvärden. Svaga punkter identifieras och förbättringar föreslås. Effekten av de föreslagna förbättringarna framgår utav omvärdering av systemet och omräkning av riskbedömningsvärdena.

Dessa bedömningar leder till det yttersta målet för ökad säkerhet för alla trafikanter.

Nyckelord

Ethernet för fordon, TCP/IP, Ethernet säkerhet, DoIP, BroadR-Reach, OSSTMM, flash bootloader

Acknowledgments

This thesis was created as result of research performed at Vector Informatik GmbH in Stuttgart, Germany.

There are many people who helped in the creation of this thesis, to whom I owe a great debt of gratitude. I would like to acknowledge:

- Professor Gerald Q. Maguire Jr. for the continuous support, and invaluable feedback throughout the whole course of the thesis.
- My industrial supervisor, Roland Hocke who was there to provide all technical and nontechnical support I needed at Vector
- My manager, Simon Friesch and all other colleagues for their support and illuminating discussions.
- Slađan Veselinović, Bilal Parvez, Mayank Joneja and Abraham Martin for their feedback and insightful discussions on various topics related to the thesis
- Aldin Burdžović for help with the Swedish translation of the abstract

Lastly, I would like to thank my parents, sister and friends for their endless support and encouragement.

Stockholm, October 2017
Ammar Talic

Table of contents

List of Figures

* Figures are or contain copyrighted material from Vector Informatik GmbH. They appear in this thesis with the permission of the company.

List of Tables

List of acronyms and abbreviations

ABS	Automatic Braking System
ADAS	Advanced Driver Assistance Systems
ARP	Address resolution protocol
ARPANET	Advanced Research Project Agency NETwork
AVB/TSN	Audio Video Bridge / Time Sensitive Network
AVTP	Audio Video Transport Protocol
CAN	controller area network
CCP	CAN Calibration Protocol
CRC	cyclic redundancy check
CSMA/CD	Carrier Sense Multiple Access/Collision Detection
DoIP	Diagnostics over IP
DoS	Denial of service
DSP	Digital Signal Processor
ECU	electronic control unit
FCS	Frame Check Sequence
4B5B	4 bits to 5 bits (coding)
GPL	General Public License
GUI	Graphical User Interface
HTTP	Hyper Text Transfer Protocol
ICMP	Internet Control Message Protocol
ICT	Information and Communication Technology
IETF	Internet Engineering Task Force
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ISO	International Organization for Standardization
KWP2000	Key Word Protocol 2000
LAN	Local Area Network
LIN	Local Interconnect Network
LLC	Logical Link Control
LTE	Long Term Evolution
MAC	Media Access Control
MITM	man in the middle
MOST	Media Oriented Systems Transport
MLT-3	Multi Level Transmit with 3 states
NDP	Neighbor discovery protocol
NRZI	Non Return to Zero Inverted
OBD-II	On-Board Diagnostics
OS	operating system
OSI	Open Systems Interconnection
OSSTMM	Open Source Security Testing Methodology Manual
OWASP	Open Web Application Security Project
PAM3	Pulse Amplitude Modulation with 3 states

PAM5	Pulse Amplitude Modulation with 5 levels
PDU	protocol data unit
PHY	physical
PTP	Precision Time Protocol
Pub/Sub	Publish/Subscribe
PWM	Pulse Width Modulation
QoS	Quality of Service
R&D	research and development
RAM	Random Access Memory
RAV	Risk Assessment Values
RFC	Requests for Comments
RPC	Remote Procedure Call
SCTP	Stream Control Transport Protocol
SD	Service Discovery
SIG	Special Interest Group
SOME/IP	Scalable service-Oriented MiddlewarE over Internet Protocol
SMTP	Simple Mail Transfer Protocol
TCB	Transmission control block
TCP	transport layer protocol
TCP/IP	Transport Control Protocol/Internet Protocol
TTL	Time to live
UDP	user datagram protocol
UDS	Unified Diagnostic Services
USB	Universal Serial Bus
UTP	unshielded twisted pair
VLAN	Virtual Local Area Network
V2I	Vehicle to Infrastructure
V2V	Vehicle to Vehicle
WLAN	Wireless Local Area Network)
XCP	eXtended Calibration Protocol

1 Introduction

This section will introduce the reader to the topic, the structure of the thesis, and background information necessary for understanding the rest of the thesis. It lists the thesis project's goals and methods through which they will be achieved and describes the delimitations of the work.

1.1 Background

For more than a century cars have been an integral part of our lives. As time passes these vehicles have become increasingly complex. The ability of unauthorized parties to access the car's internal infrastructure and control its components or read data stored in the car are major concerns facing car manufacturers nowadays.

Starting in the 1980s car manufacturers realized that they needed to re-examine communications within the vehicle. On one side the cabling was becoming increasingly complex, the cables long, and weighed more than 100 kg [1]. On the other side, the car had an increasing number of features that each needed to be controlled. As a consequence of the fact that making point-to-point connections between all electronic control units (ECUs) which needed to communicate was too complex, bus protocols were created and introduced. Examples of such are Controller Area Network (CAN), Local Interconnect Network (LIN), Media Oriented Systems Transport (MOST), and FlexRay.

A premium car from 2012 has about 100 ECUs [2]. Moreover, infotainment (a combination of *info*rmation and enter*tainment*) and assisted driving with the help of cameras are both becoming more and more important. As a result, the data rates of the existing buses are insufficient to support the needs of modern vehicles. Automotive Ethernet, due to its much higher data rates, is becoming the new main player in this field. Moreover, new advances in technology have allowed automotive Ethernet to operate over a single unshielded twisted pair (UTP) cable. Additionally, it is also possible to use this single pair of wires for power distribution, hence cabling is significantly reduced as compared to the existing bus standards. For these reasons, automotive Ethernet is increasingly being implemented by various manufacturers.

Due to the fact that Ethernet has for many years been the *de facto* standard for interconnecting computers, years of experience in hacking computers might be used for hacking cars. *Only* the physical layer of the automotive communications system is undergoing a major change with new encoding schemes and modulation techniques being used (see Section 2.3.1). The higher layers are mostly going to be kept intact in automotive Ethernet, hence security has become a major concern. This thesis project investigates whether conventional network exploitation techniques can be utilized to attack automotive Ethernet networks. It will also evaluate whether conventional means to avoid or mitigate such attacks can be applied in the automotive Ethernet domain.

Figure 1-1 shows a typical automotive network topology with optimal security in place. Unfortunately, all communication indicated as secure is not always

implemented correctly. As can be seen, the network consists of multiple networks. Ethernet is used for advanced driver assistance systems (ADAS) (which mostly consis of cameras), diagnostics via the on-board-diagnostics (OBD) port, a connection with the telematics unit, the head unit (on the dashboard as a centrally located infotainment control system), and between the central gateway and the vehicle's access point which connects it to the external world.

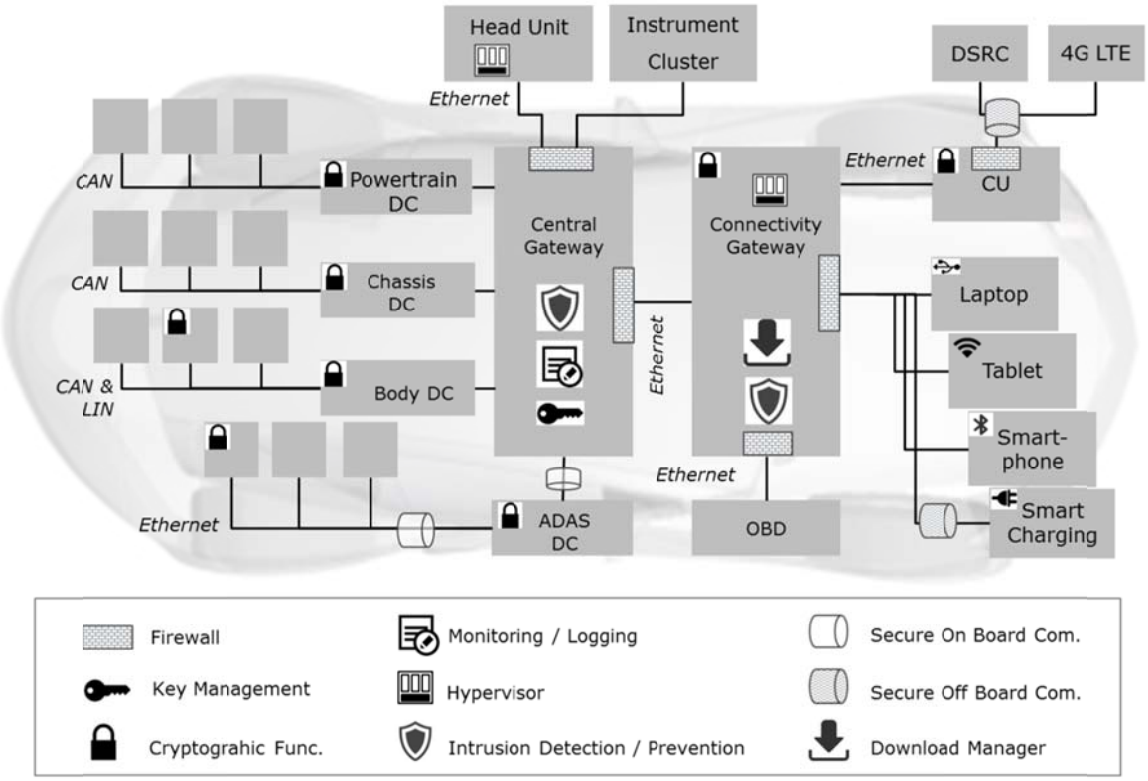

Figure 1-1: Typical topology of a modern vehicle with optimal security*

Communication over the Ethernet is **not** directly related to safety critical operations as it is **not** responsible for any powertrain, chassis, or body related operations. However, the fact that (1) the data provided by ADAS triggers powertrain actions, (2) the software running the car is flashed onto the various ECUs through the OBD port, and (3) external access to the central gateway is possible via the Ethernet, the tight relation between Ethernet and the safety of the vehicle and its driver is obvious.

1.2 Problem

A major part of automotive Ethernet relies on the Transport Control Protocol/Internet Protocol (TCP/IP). Despite the United States Department of Defense funding the creation of the TCP/IP suite, security was not high on the list of

* Figure courtesy of Vector Informatik GmbH.

priorities in the late 1970s when it was developed, as the emphasis was primarily on communication for sharing of resources. This has led to multiple vulnerabilities being discovered and exploited in the TCP/IP protocol stack implementations. Some of these vulnerabilities were documented as early as 1989 [3].

Another potential problem is that open source is not very valued in the automotive industry due to common copyleft* requirements which would oblige the companies to share their code modifications and additions. Due to this no tested open source stack implementations of any protocol are used, but rather in-house stacks are developed. This leads to new potential implementation faults which can introduce major vulnerabilities. How consequential such implementation faults can be is shown in the analysis of one of the first computer worms, the "Morris worm" [4].

With all this said, the main question this thesis investigates is: "*Is automotive Ethernet together with its accompanying protocols secure and safe enough for the automotive industry?*"

1.3 Purpose

This work will offer a security analysis of Ethernet communication channels in a typical car topology. It will assess the risks of known protocol and implementation vulnerabilities and weaknesses in the vehicle environment and look at scenarios in which they could be exploited. It will create tangible exploitation scenarios and offer countermeasures which can benefit all car manufacturers, and/or their software/component suppliers for testing systems against common faults and vulnerabilities, and thereby increase the safety of the vehicle.

1.4 Goals

The goal of this project is to test the susceptibility of automotive Ethernet against well-known conventional Ethernet vulnerabilities and weaknesses, and investigate the potential consequences of their existence inside the car´s ecosystem. This will be achieved through the following five sub-goals:

1. Do a security analysis of a typical car network topology
2. Evaluate the results
3. Investigate consequences
4. Propose countermeasures
5. Re-evaluate with the countermeasures in place

The final deliverables will consist of the security analysis results and conducted test details for some tests, proposed countermeasures, and a showcase of their impact on the system's security.

* Type of copyright licenses which enable the use of copyrighted materials, but requires any derivatives to be obliged to utilize the same license.

1.5 Research Methodology

Security analyses and penetration testing have been around for quite some time now. As a result, there are multiple methodologies and frameworks which can be followed. Two of the most utilized ones are the Open Web Application Security Program (OWASP) [5] and the Open Source Security Testing Methodology Manual (OSSTMM) [6]. While both of them are full-fledged methodologies, OSSTMM has been selected for this research, because of its wide applicability and adaptability to the specific field. OWASP, on the other hand, is more related to web application testing.

Another great benefit of OSSTMM is the way in which it handles risk assessment. It does not follow the traditional approach of saying that Risk = Likelihood × Impact, where both factors are of relatively subjective nature. In contrast, OSSTMM employs risk assessment values (RAVs), which represent a quantitative balance between operations (lack of security necessary for the system's operation), controls (security measures in place to secure operations), and limitations (vulnerabilities, weaknesses, and such). With those they define the attack surface which is an objective measure. RAVs are basically the metric unit for security. More details about the terms and the methodology itself will be provided in Chapter 3.

Through the research, data network security testing of a typical car topology as defined by OSSTMM will be performed in order to get a complete audit*. The test results will be used to define the attack surface and form attack vectors. Known limitations found to be relevant will be tested, and their potential consequences for the car's safety and integrity will be evaluated. These consequences will be investigated regardless of the test results on the available hardware. The absence of limitations in the given test environment does *not* guarantee their absence in all implementations and configurations.

As a result of every attack, a RAV score will be calculated. Possible controls and/or fixes for identified limitations will be proposed. Afterwards, the RAV score will be recalculated to make the impact of the added controls/fixes visible.

1.6 Delimitations

This thesis project is limited to testing and evaluating the TCP/IP stack and protocols built on top of it. Stacks specially developed for the automotive industry (such as AVB/TSN – described later in Section 2.3.3) will *not* be evaluated, hence are left for future work. The reasons for this are the limited duration of this project, the breadth of the field, and the author's lack of prior experience in the security domain.

The tests described in this thesis are performed on hardware and software developed and produced by Vector Informatik GmbH. In contrast, hardware

* In this document, an OSSTMM audit or "audit" is the result of the analysis performed after an OSSTMM test. The person who performs this function of testing and analysis is referred to as a Security Analyst or "Analyst".

limitations (i.e. vulnerabilities and weaknesses of microprocessors, or the BroadR-reach transceiver hardware and drivers) will *not* be considered since this would reduce the generality of results.

1.7 Structure of the thesis

The first section of Chapter 2 provides basic knowledge about networking. It is meant as a reference for a reader who is unfamiliar with basic networking concepts. It offers a brief introduction to networking, starting from what it is and why it is necessary, all the way to explaining standardization efforts and some of the most important protocols. If the reader is already familiar with computer networking and network models, Section 2.1 can be skipped. Chapter 2 continues by focusing on networking in the automotive domain. It begins by listing conventional bus-based communication systems, followed by an overview of automotive Ethernet and the protocols built on top of it. This part of the chapter is also meant as a reference for further description of different forms of attacks. Standardization will also be mentioned due to its importance for the industry. After the foundation has been laid, Chapter 2 finishes by presenting related work concerning car hacking.

In Chapter 3 the methodology, test bed, and utilized hardware/software are presented. Chapter 4 presents the security audit that was performed on the data networks channel. Detailed tests conducted throughout the audit are presented in Chapter 5. The results are analyzed in Chapter 6. Finally, Chapter 7 draws conclusions (while keeping all of the limitations in mind) and suggests topics for future work.

2 Background

This chapter will familiarize the reader with networking in general and networking in the automotive domain. It has 4 major sections which explain networking basics (Section 2.1), automotive networking with its standard bus communication systems (Section 2.2), automotive Ethernet (Section 2.3), and finally related work in the automotive security domain (Section 2.4).

2.1 Networking Basics

This section with all its figures and brief explanations is meant as a quick reference for the reader regarding the underlying background knowledge that is assumed later in this thesis when attacks are presented and analyzed. Further background details about networking are outside the scope of this work, hence the reader is referred to the references given in this chapter for additional details.

2.1.1 What are networks and why are they needed?

In the early days of computers, people figured out that it is necessary to find a way of transferring data from one device to another *without* having to use portable media. Furthermore, there was a need to share expensive peripheral hardware not present on all devices. Today, in the automotive domain there are multiple sensors and actuators within a car, hence there is a need to interconnect them in order to make use of all of this data to automate actions and to provide infotainment to the driver and passengers.

A network interconnects a group of devices together for a specific purpose or for multiple purposes. These networked devices are called hosts. The Internet Protocol version 4 (IPv4) uses the term host for all the devices in a network [7]. In contrast, Internet Protocol version 6 (IPv6) goes a step further and defines nodes as "*a device that implements IPv6*", while a host is "any node that is not a router" [8].

2.1.2 Network models – standardization

In order to be able to communicate with as many devices as possible, regardless of different hardware vendors, and to reduce research and development (R&D) costs standardization is necessary.

2.1.2.1 Open Systems Interconnection (OSI) Model

As a result of the need for standardization, the International Organization for Standardization (ISO) published the OSI reference model in 1979 [9].

The OSI model is basically a framework to help coordinate the development of existing and new standards. This model breaks communication into seven layers, with each layer having a specific purpose. Every layer interfaces with the one below and above itself. The idea is that standards are defined for each layer, thus there is flexibility in the implementation of protocols in a given layer as each layer is

independent of the others (aside from the interfaces between layers) [10]. The ISO/OSI model is rarely used for networking as it is, but using it as a framework when designing new protocols ensures compatibility with existing protocols. This makes it possible to communicate over different physical layers; for example, wired and wireless links while still keeping the upper layers unaltered.

2.1.2.1.1 OSI Layers

Figure 2-1 shows the seven layers of the ISO OSI model.

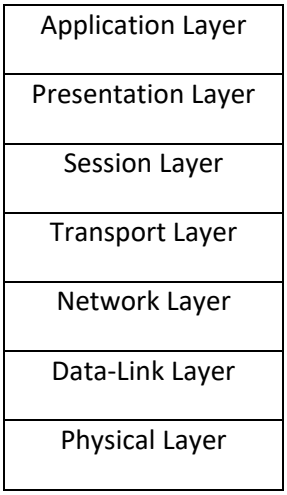

Figure 2-1: ISO OSI Model

To pass data from one application to another, this data has to pass from the application layer to the physical layer and back to the application layer. While passing through the layers the protocol data unit (PDU) generally gets a header and in some cases a trailer added or removed on every layer. A header is added to the beginning of the PDU, while a trailer is added to the end of the PDU (as shown in Figure 2-2). The figure also shows how data travels through all layers from top to bottom on sending host A, then through the communication medium to host B where it again goes through all layers only this time from bottom to top.

The following paragraphs explain what each layer is responsible for. Since each layer deals with data differently, there are different PDUs for each layer. However, these also have other names that are used to refer to a PDU in a particular layer. These names will be explained along with the description of the layer.

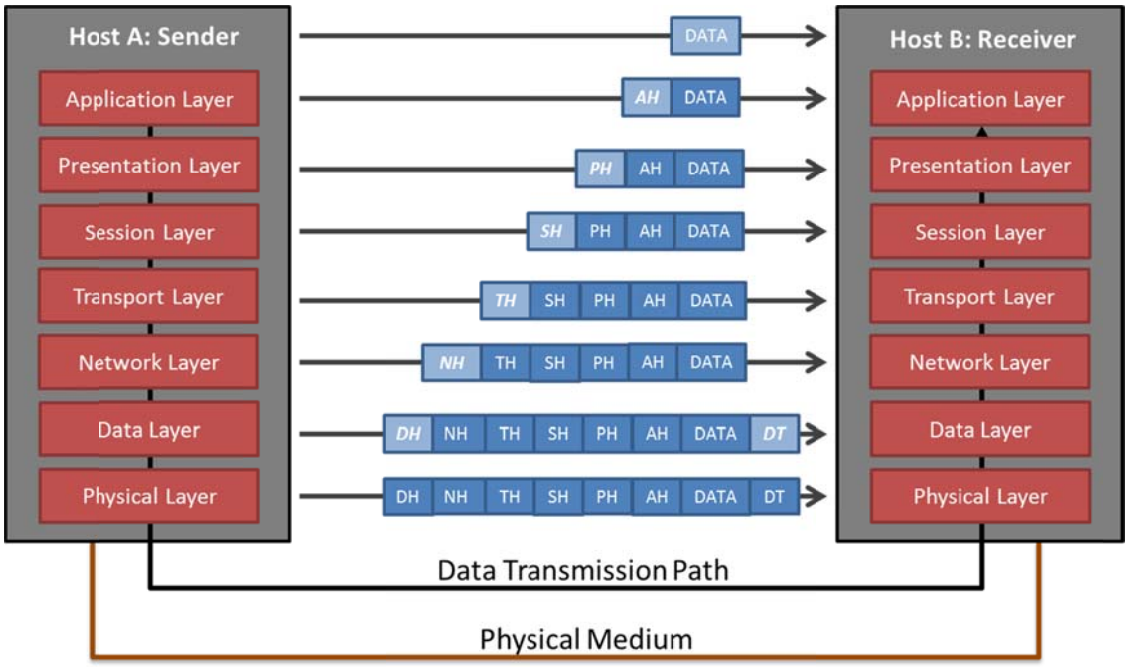

Figure 2-2: Path that data traverses from host to another according to the ISO OSI model*

2.1.2.1.1.1 Physical Layer

The physical (PHY) layer is responsible for generating and receiving signals to/from the physical medium. Some examples of physical media are UTP copper cables, optical fiber, or air in the case of wireless communications. The PDU at the physical layer is typically called a symbol or a bit. The details of the physical layer are described in the relevant standards, often in combination with the media access and control sublayer of the data link layer.

2.1.2.1.1.2 Data-Link Layer

The data-link layer handles PDUs in the form of link layer frames. The size of these frames can be as small as a few bytes to a few thousand bytes. Figure 2-2 shows that the data-link layer adds a header (DH) and trailer (DT). The header information includes control information for the data link layer processing and the trailer typically is used for error detection (such as a Frame Check Sequence (FCS), for example, computed using a cyclic redundancy check (CRC)). A data frame encapsulates the PDU received from the network layer. Frames are transmitted and received via the physical layer. In many implementations, the data-link layer can detect errors (via the FCS) and in some cases, the link layer can fix minor errors in the frames received from the physical layer. The link layer is often divided into two sublayers: the Media Access Control (MAC) and Logical Link Control (LLC) sublayers. The MAC sublayer is frequently specified together with the physical layer. For example, the IEEE 802.3 family of standards specifies the MAC sublayer and physical layers of what is commonly called Ethernet, while IEEE 802.2 specifies an LLC layer that can be used with many different MAC/PHY standards.

---

* Adapted from http://pharoah-net.blogspot.de/2011/12/osi-model.html

2.1.2.1.1.3 Network Layer

The network layer handles PDUs commonly called packets. This layer is responsible for the translation of network addresses to hardware addresses and vice versa. The network layer is responsible for routing a packet between a source & destination node and fragmenting packets that are larger than the maximum PDU of the underlying link layer.

2.1.2.1.1.4 Transport Layer

For both connectionless and connection-oriented transport protocols, the transport layer does end-to-end error detection. The transport layer handles connection-oriented communications connections, flow and sequence control, and segmentation. Additionally, for a reliable transport protocol, there can be automated error recovery, for example via retransmission. Moreover, a transport protocol can also perform monitoring of Quality of Service (QoS) parameters. The PDU of this layer is called a segment in the case of the transport layer protocol (TCP), a datagram in the case of the user datagram protocol (UDP), and a data chunk in the case of the Stream Control Transport Protocol (SCTP).

2.1.2.1.1.5 Session Layer

The session layer is used by different host processes for handling (opening, closing, and managing) sessions. It may try to recover dropped connections and close connections which are no longer needed.

2.1.2.1.1.6 Presentation Layer

The presentation layer is commonly merged with the application layer. It is responsible for converting the data from the session layer into a presentable format for the application layer. This includes conversions from different formats, different character sets, encryption, decryption, etc.

2.1.2.1.1.7 Application Layer

Contrary to common belief, the application layer does not include the applications, with which a user interacts, i.e., it does not include the browser or the email client. However, the application layer gives these applications access to the network by providing services that these applications depend on. For example, in the case of a browser this service would be provided by the HyperText Transfer Protocol (HTTP); while for an email client, the service could be provided by the Simple Mail Transfer Protocol (SMTP).

2.1.2.2 Transport Control Protocol/Internet Protocol Model

TCP/IP is actually older than the OSI model. TCP/IP is the successor of the Advanced Research Project Agency NETwork (ARPANET) network control program which implemented the Host to Host protocol. As TCP/IP technology evolved ARPANET shifted to the TCP/IP model leading to what eventually became known as the Internet. Some of the reasons for TCP/IP's wide success were its prior existence (about ten years older than the OSI model), source code availability, independence from specific hardware and software, free and public documentation, and flexibility [11].

This subsection briefly compares the ISO and the TCP/IP models and then goes into more detail about protocols used on each layer. These details are important because, in addition to the automotive specific protocols which will be described in Chapter 3, the TCP/IP protocols are also widely used in the automotive sector.

2.1.2.2.1 Comparison with the OSI model

Unlike the OSI model, the TCP/IP model has only four layers. As can be seen in Figure 2-3 the data-link and physical layers form a single network access layer. The OSI network layer is called the internet layer in TCP/IP, the transport layer keeps its name and position, while the OSI application, presentation, and session layers are merged into TCP/IP's application layer.

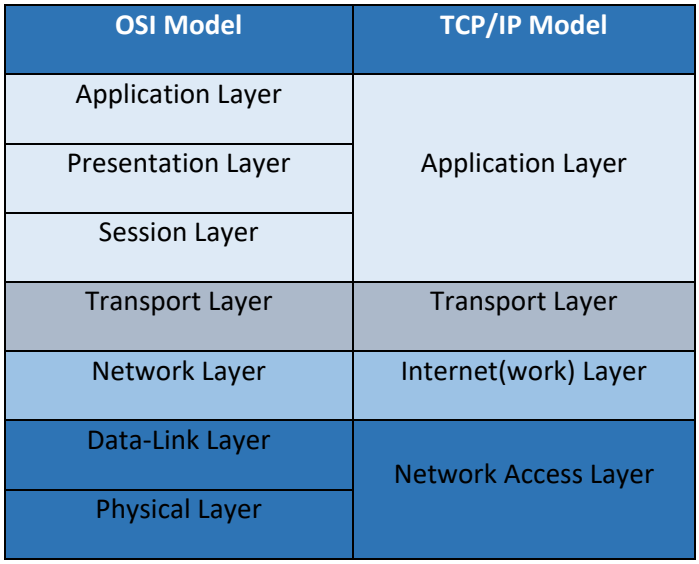

Figure 2-3: Comparison of ISO OSI and TCP/IP Models

2.1.2.2.2 Examples of protocols commonly used together with the TCP/IP stack

On the left side of Table 2-1 are the layers and on the right are some examples of protocols implemented in the given layer. More protocols are listed than will be explained in detail, for further details see either an Internetworking textbook* or the relevant Internet Engineering Task Force (IETF) Requests for Comments (RFCs) for full details. Only some of these protocols are used in the automotive domain; hence there is no need to go into detail about the other protocols (they are only given here for context).

* Such as: James F. Kurose and Keith W. Ross, Computer Networking: A Top-Down Approach, Sixth Edition, Pearson Education, 2012, ISBN-13: 9780273768968, ISBN-10 0-27376-896-4.

Table 2-1: Examples of some protocols at the different layers of TCP/IP

TCP/IP Layers	Protocols
Application	HTTP, Telnet, SMTP, DHCP, DNS, SNMP
Transport	TCP, UDP
Internet(work)	IPv4, IPv6, ARP, NDP, ICMP, IGMP
Network Access	Ethernet, Wi-Fi

2.1.2.2.2.1 Ethernet

Ethernet was developed by Xerox Corporation based on earlier research at the University of Hawaii. The university used radio broadcasts to create a network between campuses of the university located on different islands. The name Ethernet came from the medium - "*ether*" and the fact that it was a network. However, in practice, Ethernet used coaxial cable as a media and used Carrier Sense Multiple Access/Collision Detection (CSMA/CD) as a MAC protocol.

There were multiple versions of Ethernet, but the most significant is the one defined in the Ethernet Blue Book 2 specification created by the Ethernet Consortium (consisting of Xerox, Digital Equipment Corporation, and Intel). This specification was adopted by IEEE to create their IEEE 802.3 standard [10]. The IEEE 802.3 standard defines multiple types of supported cable and speeds, but only two will be mentioned here because they are very common and used in the automotive domain.

There were three frame formats defined by IEEE, but only the Ethernet frame II format will be shown here because it is used in the automotive domain. The reason for its use is that it can contain an additional Virtual Local Area Network (VLAN) field. This frame format is shown in Figure 2-4. The SFD field is the start of frame delimiter and indicates the start of the actual data frame's header fields. The other fields will be described later in this thesis.

Preamble	SFD	Receiver MAC address	Sender MAC address	VLAN	Type	Data	PAD	CRC
7 Bytes	1 Byte	6 Bytes	6 Bytes	4 Bytes	2 Bytes	0-1500 Bytes	1 Byte	4 Bytes

Figure 2-4: Ethernet frame II format

Today most home and corporate Ethernet networks use full-duplex mode together with switched Ethernet, hence there is no longer a need for CSMA/CD as there is only one host connected to each port of the switch and there are distinct transmit and receive media (for example, fibers or wire pairs).

2.1.2.2.2.1.1 100Base-TX also known as Fast Ethernet

IEEE 802.3 100Base-TX usually operates with two physical channels and can operate in full duplex* mode. The maximum communication speed is 100 Mbit/s. Only point-to-point communication is possible, hence a switch or router is necessary in order to interconnect multiple devices.

The physical layer is divided into the physical medium *independent* and physical medium *dependent* sublayers. The physical medium independent sublayer is the upper part of the physical layer and specifies 4 bits to 5 bits (4B5B) coding and Non-Return to Zero Inverted (NRZI) coding along with clocking. The combination of these two encodings guarantees transitions within each 5-bit block, which in turn helps ensure synchronization.

The physical medium dependent sublayer operates on two pairs of category 5 or 5e UTP cables using Multi-Level Transmit with 3 states (MLT-3) with the states: -1, 0, and +1 volts. On a transition from low to high or high to low, the level changes to the next state in a circular order (0V, +1V, 0V, -1V, 0V, and so on). This makes the signal resemble a sine wave and leads to a considerable bandwidth reduction as compared to the binary equivalent [10].

2.1.2.2.2.1.2 1000Base-T also known as Gigabit Ethernet

IEEE 802.3 1000Base-T uses all 4 channels of a category 5e UTP cable. Similarly, to 100Base-TX it supports full duplex mode but operates at a maximum speed of 1000 Mbit/s (i.e., 1 Gbit/s).

The same encoding scheme that was developed for 100Base-T2 is used, but in order to gain ten times the speed 4 channels instead of 2 are used, and the clock frequency has been increased 5 times. The data is modulated with Pulse Amplitude Modulation with 5 levels (PAM5) using 5 different voltage levels, which makes it more prone to noise, but digital signal processors (DSPs) are used to recover the transmitted bits [10]

2.1.2.2.2.2 IPv4

The internet protocol is connectionless, meaning that no connections are established prior to transferring data and no acknowledgments are sent. This means that there is no way to ensure the successful delivery of data. IP leaves these matters to higher layer protocols, such as TCP. Hence IP is often considered to offer an unreliable best-effort delivery service. The IPv4 frame header is shown in Figure 2-5.

<table>
<tr><th>4 Bits</th><th>4 Bits</th><th>4 Bits</th><th>4 Bits</th><th>4 Bits</th><th>4 Bits</th><th>4 Bits</th><th>4 Bits</th></tr>
<tr><td>Version</td><td>IHL</td><td colspan="2">Service Type</td><td colspan="4">Total Length</td></tr>
<tr><td colspan="4">Identification</td><td>Flag</td><td colspan="3">Fragmentation Offset</td></tr>
<tr><td colspan="2">Time To Live</td><td colspan="2">Protocol</td><td colspan="4">Header Checksum</td></tr>
<tr><td colspan="8">Source IP Address</td></tr>
<tr><td colspan="8">Destination IP Address</td></tr>
<tr><td colspan="8">Options (optional field)</td></tr>
</table>

Figure 2-5: IPv4 header

* supporting simultaneous bi-directional (two-way) communication

2.1.2.2.2.3 IPv6

In 1981, when IPv4 was introduced no one thought that more than 4 billion (2^{32}) addresses would be needed to address all of the devices connected to the Internet. However, this need for a very large number of routable addresses was anticipated in 1999 when IPv6 was created. The plan was to switch to IPv6 as soon as possible, but today, almost 20 years later - the number of users who access Google services over IPv6 is ~18% [12p. 6].

IPv6 will **not** be explained in detail; however, necessary details will be mentioned (as needed). It is important to know that IPv6 has a much bigger address space, namely 2^{128}, employs different protocols on some levels, and according to the number of reported common vulnerabilities and exposures (CVE) is more vulnerable to attacks than IPv4 [13]. It has to be pointed out that IPv6 is not by definition more vulnerable, on the contrary, it has more security mechanisms in place. Nevertheless, the problem is due to poor implementations, hence more reported vulnerabilities.

The IPv6 header is shown in Figure 2-6.

<table>
<tr><th>4 Bits</th><th>4 Bits</th><th>4 Bits</th><th>4 Bits</th><th>4 Bits</th><th>4 Bits</th><th>4 Bits</th><th>4 Bits</th></tr>
<tr><td colspan="2">Version</td><td colspan="2">Traffic Class</td><td colspan="4">Flow Label</td></tr>
<tr><td colspan="5">Payload Length</td><td colspan="2">Next Header</td><td>Hop Limit</td></tr>
<tr><td colspan="8">Source IP Address (128 Bit)</td></tr>
<tr><td colspan="8">Destination IP Address (128 Bit)</td></tr>
</table>

Figure 2-6: IPv6 header

2.1.2.2.2.4 Address Resolution Protocol (ARP)

In IPv4, ARP is used to map an IP address to a MAC address. If a node wants to communicate with another node over a link (such as Ethernet), it needs to know its own MAC address and the destination's MAC address in order to produce a frame that can be sent to the desired destination. For that purpose, each host keeps a list (called the ARP cache) of other hosts with their respective IP and MAC addresses. If a host wishes to communicate with another host that is not present in its ARP cache, then the host broadcasts an ARP request. This request asks the host with the target IP address to respond with its MAC address. The format of an ARP frame (for use with Ethernet) is shown in Figure 2-7.

<table>
<tr><th>8 Bits</th><th>8 Bits</th></tr>
<tr><td>Hardware Type</td><td>Protocol Type</td></tr>
<tr><td>Hardware Address Length</td><td>Protocol Address Length</td></tr>
<tr><td colspan="2">Sender Hardware Address (48 Bit)</td></tr>
<tr><td colspan="2">Sender Protocol Address (32 Bit)</td></tr>
<tr><td colspan="2">Target Hardware Address (48 Bit)</td></tr>
<tr><td colspan="2">Target Protocol Address (32Bit)</td></tr>
</table>

Figure 2-7: ARP frame

2.1.2.2.2.5 Neighbor Discovery Protocol (NDP)

NDP is the IPv6 equivalent of ARP. The NDP frame for neighbor advertisement and discovery is shown in Figure 2-8. A neighbor solicitation frame is similar but does **not** have the Router (R), Solicited (S), and Override (O) flags.

4 Bits	4 Bits	4 Bits	4 Bits	4 Bits	4 Bits	4 Bits	4 Bits
Type			Code		Checksum		
R	S	O	Reserved				
Time To Live			Protocol		Header Checksum		
Target IPv6 Address (128 Bit)							
Options							

Figure 2-8: NDP frame

2.1.2.2.2.6 Internet Control Message Protocol (ICMP)

ICMP is used to send control and error messages. The "*ping*" command used for testing the reachability of a host is based upon an ICMP Echo Request and Reply. ICMP protocol is used by both versions of IP, although there is a slight difference in the frames; therefore, ICMP in IPv6 is commonly called ICMPv6.

The ICMP frame is shown in Figure 2-9.

8 Bits	8 Bits	8 Bits	8 Bits
Type	Code	Checksum	
Content (optional)			

Figure 2-9: ICMP message

2.1.2.2.2.7 Transport Control Protocol (TCP)

TCP is a transport layer protocol and it adds session management to the connectionless internet protocol. A TCP session is established with a three-way handshake. TCP is a powerful and therefore more complicated protocol than many other IPv4 protocols.

Assume we have two hosts: "A" and "B". If "A" wants to communicate with "B" it first sends B a TCP segment with the SYN flag set and a sequence number say X (where X is a random number that will be increased by the number of bytes received whenever a new segment is sent). When "B" receives that segment it allocates the necessary resources and responds with a segment containing its own sequence number Y and both the SYN and ACK flags set. The Acknowledgment number with which "B" replies is X+1. Finally, after receiving this segment "A" replies with an ACK flagged packet containing the sequence number X+1 and the acknowledgment number Y+1. The TCP connection is now set up and subsequent user data transmission can begin.

A graphical representation of the handshake assuming X=250 and Y=700 is shown in Figure 2-10. A TCP connection is terminated by setting the FIN flag [14]. A TCP segment is shown in Figure 2-11.

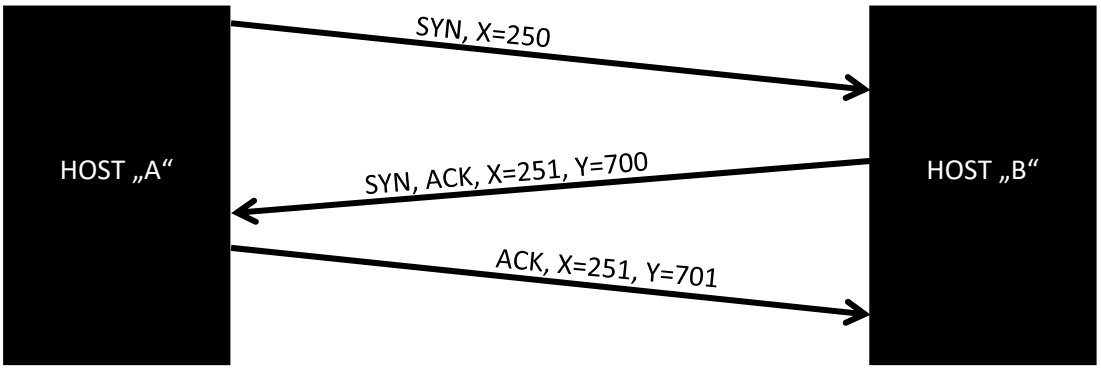

Figure 2-10: Three-Way TCP Handshake

16 Bits										16 Bits
Source Port										Destination Port
Sequence Number										
Acknowledgment Number										
Data Offset (4 Bit)	Reserved (3 Bit)	NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window
Checksum										Urgent Pointer
Options (0 or more 32 Bit Words)										
Data										

Figure 2-11: An example of a TCP segment

2.1.2.2.2.8 User Datagram Protocol (UDP)

UDP is a connectionless transport layer protocol, meaning there are no handshakes, no acknowledgments, no way to account for lost datagrams, and no flow control. A UDP datagram is shown in Figure 2-12. There are two benefits which UDP adds to the internet layer: (1) an optional checksum which can confirm the correctness of the datagrams and (2) UDP ports which can multiplex and demultiplex UDP data streams and hence can help manage multiple users and their requests. The overhead and latency of UDP is much lower than that of TCP. However, in the case of UDP, packet loss can occur and datagrams are not guaranteed to be received in the same order in which they were sent. In gaming or media streaming where low latency is much more important than a small fraction of lost datagrams, UDP is more suitable than TCP. UDP can also be used for lossless communication, but in this case, the application has to deal with reordering out of order datagrams as well as providing retransmission of lost datagrams [15].

16 Bits	16 Bits
Source Port	Destination Port
Length	Checksum
Data	

Figure 2-12: UDP datagram

2.1.2.2.2.9 Dynamic Host Configuration Protocol (DHCP)

DHCP is an application layer protocol. The DHCP message can be seen in Figure 2-13. A DHCP server automatically assigns IP addresses and provides other configuration information, thus avoiding the need for manual configuration of newly added nodes to a network. For IPv6 again there are slight differences in implementations and this protocol is called DHCPv6.

8 Bits	8 Bits	8 Bits	8 Bits
Operation Code	Hardware Type	Hardware Address Length	Hops
Transaction Identifier			
Seconds		Flags	
Client IP Address			
Your IP Address			
Server IP Address			
Gateway IP Address			
Client Hardware Address			
Server Name (optional)			
Boot Filename (optional)			
Options (optional)			

Figure 2-13: DHCP message format

2.2 Conventional Automotive Networking

This section introduces conventional automotive networking solutions and standards. Some of these will only be mentioned, while others (such as CAN bus and automotive Ethernet) are explained in detail. Ethernet is described in detail as its security is the main focus of this thesis, while CAN bus is included because there are certain attacks which rely on utilizing CAN bus. Moreover, CAN bus is important because of its control over essential peripherals, such as the brakes.

2.2.1 CAN bus

Today CAN bus is the most widely used automotive bus communication system. It will be described in more detail than other busses. However, details of its technical functioning will be left out since these details are out of the scope of this thesis. If interested the reader should refer to the cited work.

2.2.2 History and Standardization

CAN is the first widely accepted automotive bus protocol. It was developed by Robert Bosch GmbH in 1983, and published in 1986. Before that time cars were not as complex, thus manufacturers could afford to have dedicated wired connections between all the parts which needed to interact. To put this all into perspective it is worth remembering that the first motorcar was made in 1885 [16] and it did not have

any electronic parts*. The transistor was introduced in 1947 [17] and the first integrated circuits in 1958 [18]. The first airbags were deployed in cars in 1973 [19]. Subsequently, Automatic Braking System (ABS) and additional electronics were added to cars.

In 1991 a high-end car had 25 interconnected ECUs [20]. Interconnecting all of them pairwise was obviously not an option. Mercedes Benz was the first manufacturer to use CAN for connecting electronics in its S-class cars in the year 1992. From then on, CAN became so widely used that nowadays almost all semiconductor manufacturers offer CAN solutions. While the first suppliers were Intel and Motorola [1].

CAN is designed as a serial bus. It provides very reliable data transmission and satisfies real-time requirements. Today it is widely deployed inside cars.

Given the vast number of car manufacturers in the world, standardization is important in the industry. CAN was standardized in 1994. The standard consists of 4 ISO documents: the CAN protocol, the low speed (40-125 Kbit/s), the high-speed (up to 1Mbit/s) physical layers, and a time-triggered communication option so that deterministic communication can be assured (since the protocol defines its own event-driven communication) [21].

2.2.3 CAN in terms of the ISO Model

The CAN specification defines the data-link layer and part of the physical layer. The remaining higher layers and the rest of the physical layer are left to the system designer. If one prefers non-proprietary protocols for the implementation of the remaining layers, these protocols can be used together with the underlying CAN specification [22].

2.2.4 CAN Topologies

Within the automotive industry, a linear bus topology is the most widely used CAN topology. In this topology, the maximum wiring length should not exceed 40 m and the recommended stub length is a maximum of 30 cm. Other potential topologies include single star, twin star, and hybrid topologies.

2.2.5 Physical Restrictions

Table 2-2 summarizes some of the CAN's most important physical properties and restrictions. It is worth mentioning that the bus needs termination in order to prevent signal reflections. For that purpose 120 Ω resistors are used at both ends of the bus [23].

---

* However, a model electric car was first invented in 1828 by Ányos Jedlik. While in 1834, Thomas Davenport built an electric car that could run on a test track. Battery powered electric cars were introduced in 1837.

Table 2-2: CAN bus physical specifications

Physical medium	1 pair UTP copper cable
Maximum data rate	1 Mbit/s
Maximum network extension	40 meters
Maximum number of nodes defined by ISO 11898[24]	32

2.2.6 Local Interconnect Network (LIN)

LIN was developed to serve as a standard and low-cost alternative to CAN for less demanding applications. LIN is a serial bus which operates at a data rate of 20-25 Kbit/s. It uses a master/slave model and supports up to 16 slaves. LIN is usually used for **noncritical** subsystems which only require simple control, such as adjusting seats & mirrors or opening windows [25].

2.2.7 FlexRay

FlexRay evolved as an alternative for more demanding tasks which needed better performance than CAN could provide. It operates at a higher data rate of 10 Mbit/s and is used for applications such as Drive-by-wire, active suspension, and adaptive cruise control. FlexRay is an innovative communication protocol which combines event-driven and deterministic communication with static and dynamic segments. A static segment is reserved for deterministic data, while a dynamic segment is used for priority based event-driven communication [26].

2.2.8 Media Oriented System Transport (MOST)

Although the use of MOST is decreasing with the advent of Ethernet, for the sake of completeness it is mentioned here. MOST is a high-speed protocol developed for infotainment and media oriented communication. Its maximum data rate is 24.8 Mbit/s and it can support up to 64 nodes [27].

2.3 Automotive Ethernet and IP

IEEE 100Base-TX Ethernet is already used for diagnostics purposes inside cars. However, for a long time, it was thought that Ethernet would not be further adopted in the automotive sector. However, as the amount of data has increased specialized Ethernet protocols and services on top of it are being developed.

The four main areas for automotive Ethernet are:

- Advanced Driver Assistance Systems (ADAS) such as 360-degree camera systems
- Diagnostics over IP (DoIP)
- Infotainment
- Communication backbone

Figure 2-14 shows all the Ethernet and IP protocols used in the automotive domain organized according to the OSI model (with the new automotive protocols in red and the standard well-known protocols in gray). The left-hand column gives the names for the ISO/OSI layers, while the columns to the right are different fields of application: Diagnostic access, Audio/Video TimeSync, Smart Grid, Control Communications, Service Discovery, Address Configuration, Address Resolution, Signaling, etc.).

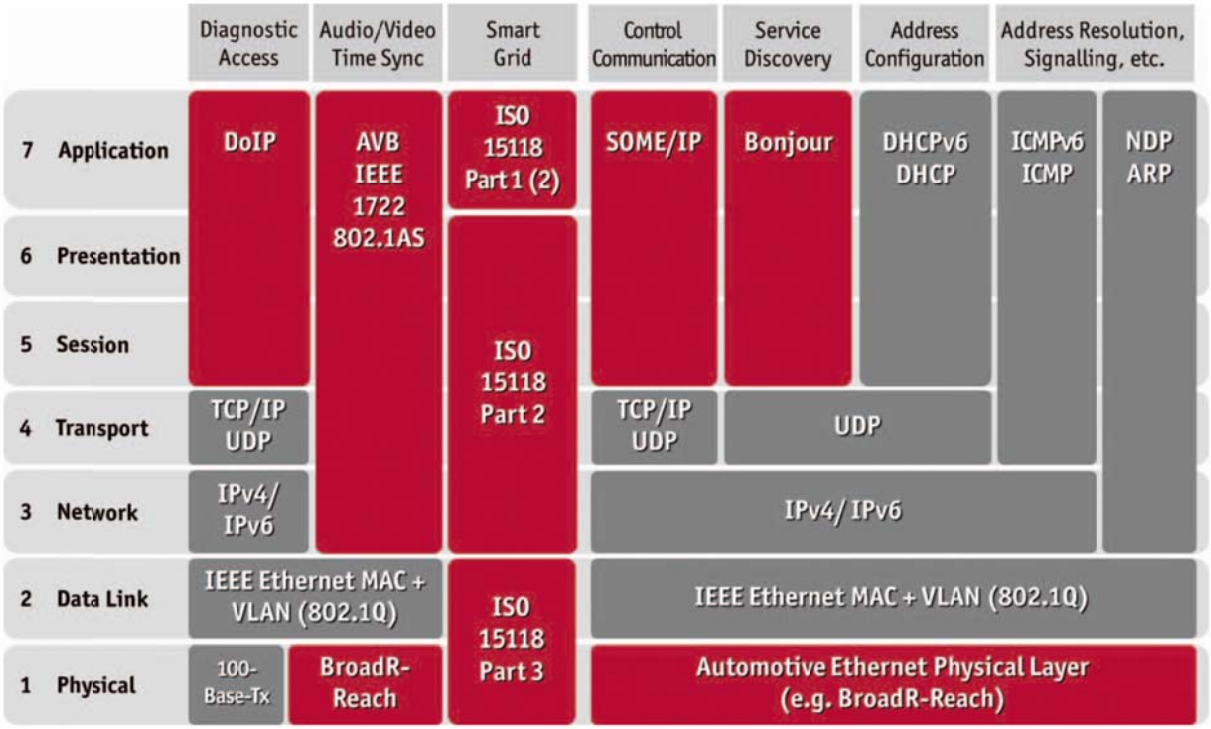

Figure 2-14: Automotive Ethernet [28]*

2.3.1 BroadR-Reach – 100Base-T1

BroadR-Reach is the most commonly used physical layer protocol in automotive Ethernet. The reason for this is that it was specifically developed for automotive use. It uses just one pair of UTP cables and supports the same speed as 100Base-TX; therefore, it is much faster than any other automotive communication protocols and it requires only two wires while also supporting power distribution over these wires.

BroadR-Reach was originally developed by Broadcom but is now developed by the consortium for automotive Ethernet OPEN (One-Pair Ether-Net) Alliance Special Interest Group (SIG) [29]. The consortium submitted BroadR-Reach to IEEE for standardization where it was standardized as the IEEE 100Base-T1 standard. This standard operates at a maximum speed of 100 Mbit/s and supports full duplex communication.

* Figure courtesy of Vector Informatik GmbH.

The data is encoded with 4 bits to 3 bits (4B3B) or 3 bits to 2 ternary states (3B2T) and afterward modulated with Pulse Amplitude Modulation with 3 states (PAM3). In full-duplex mode, both nodes add their differential voltages to the two wires. The receiver simply subtracts its own signal and ends up with the signal sent by the other node.

With this standard only point-to-point communication is possible; therefore, switches are used to interconnect multiple devices.

2.3.2 Scalable service-Oriented MiddlewarE over Internet Protocol (SOME/IP)

SOME/IP is a middleware solution mainly used for control messages. It was very carefully designed so that it can fit on devices ranging from simple cameras up to full-fledged head units. Additionally, infotainment and traditional CAN scenarios have been kept in mind during its design. SOME/IP supports a wide range of features including [30]:

- Serialization,
- Remote Procedure Call (RPC),
- Service Discovery (SD),
- Publish/Subscribe (Pub/Sub), and
- Segmentation of UDP datagrams.

SOME/IP introduces a philosophical shift in automotive data transmission. All of the prior standards and protocols have been signal-oriented. In contrast, SOME/IP introduces *service-oriented* transmission of information.

In service-oriented communication data is transmitted only if it is needed by at least one receiver, while in signal-oriented communication data is sent whenever the sender finds it necessary, for example when a value is updated. The obvious benefit of service-oriented communication is that it avoids unnecessary data flooding of the channel. However, this assumes that a server (i.e., the provider of the data) is informed about whether any receiver(s) needs the data [31].

2.3.3 Audio Video Bridge / Time-Sensitive Network (AVB/TSN)

The AVB/TSN protocol was developed since most of the data exchange in an automotive environment happens *within* a Local Area Network (LAN) and because low latency is very important. AVB/TSN resides on levels 3 to 6 of the OSI model and works directly on top of Ethernet.

In 2012, IEEE decided to rename the Audio Video Bridging Working group to be the Time-Sensitive Networking Working Group. This group is continuing to work on the AVB/TSN standards. There is also an industry consortium called AVnu Alliance [32] that establishes and certifies interoperability of AVB/TSN standards. AVnu Alliance members are from various domains ranging from automotive and consumer electronics to industrial manufacturing.

As the name indicates AVB/TSN is mostly used for streaming data, such as camera video streams or multimedia. There are three types of nodes defined by the

protocol: the "*talker*" - the data source, the "*listener*"- a data sink, and the bridge/switch - the main coupling element. In the case of AVB/TSN, a standard Ethernet switch is insufficient as the switch **must** support AVB/TSN.

The Audio Video Transport Protocol (AVTP) is used for the actual transmission of data. This data is generally audio and video, but can be control data as well. The frames of a stream and control header are shown in Figure 2-15 and Figure 2-16 (respectively). The AVTP Timestamp field contains a time in the future at which the audio/video needs to be played. When that time comes, all output devices (e.g. speakers) will play this stream at the same time.

Since timing and low jitter are very important the Precision Time Protocol (PTP) is used to synchronize the device clocks. With the help of PTP and by prioritizing traffic it is possible for the latency between talker and listener to be less than 2 ms [33].

8 Bits		8 Bits				8 Bits	8 Bits	
Subtype	S V	Versio n	M R	FSD	T V	Sequence Number	Format Specific Data 1	T U
Stream ID								
AVTP Timestamp								
Format Specific Data 2								
Stream Data Length (Octets)							Format Specific Data 3	
Stream Data Payload								

Figure 2-15: AVTP frames of a stream

8 Bits		8 Bits	8 Bits	8 Bits
Subtype	S V	Version	Format Specific Data	Control Data Length (Octets)
Stream ID				
Control Data Payload				

Figure 2-16: AVTP control header

2.3.4 Diagnostics over Internet Protocol (DoIP)

DoIP has been used for quite some time. DoIP is defined in ISO 13400 [34] as a standard diagnostic transport protocol. DoIP is usually used in combination with diagnostic protocols, such as Key Word Protocol 2000 (KWP2000) Unified Diagnostic Services (UDS). High transmission rates are very attractive in diagnostics because this decreases the time needed to flash new software both during production and when the vehicle is being repaired/maintained. The physical medium does not have to be BroadR-Reach or Ethernet but can be any protocol supporting IP, such as Wireless Local Area Network (WLAN) or even Long Term Evolution (LTE).

In order to avoid the need to have a diagnostic interface on every single ECU, gateways make it possible to use DoIP over a standard bus, such as CAN and FlexRay. This gateway has to know the address of the desired ECU and the communication protocol to use to reach it, and then the gateway takes the received messages, packs them in such a way that the target will understand them and sends

them. When a reply arrives, the gateway repacks this reply in a format understandable to the diagnostics tool. By adding information about the source ECU and protocol the gateway ensures that multiple requests to multiple ECUs over multiple buses can be sent at the same time *without* the need for sequential answers [31]. The DoIP frame can be seen in Figure 2-17.

8 Bits	8 Bits	8 Bits	8 Bits
Protocol Version	Inverse Protocol Version	Payload Type	
Payload Length			
Payload			

Figure 2-17: DoIP frame

Although the flash bootloader relies on DoIP but is not directly part of DoIP, it has been integrated into this subsection as it is a potentially major attack vector.

The flash bootloader is a standalone software application which resides in the ECU along with the standard application software. It is designed to be triggered only on rare occasions when the application software is corrupted or the ECU's programming mode is entered. As its name indicates, this programming mode enables an external entity to program/flash new software onto the ECU. Due to hardware limitations of the flash memory, a bootloader is necessary. More specifically, the flash memory cannot be used **and** programmed at the same time because it is erased in blocks.

The flash bootloader may support different security levels. The most basic version uses seed key verification prior to erasing and reprogramming the memory. More sophisticated (expensive) versions use cryptographic signatures and strong encryption to prevent unauthorized flashing. Consequently, these more sophisticated methods are used only in those ECUs which should *never* be flashed by *third parties*.

2.3.5 eXtended Calibration Protocol (XCP)

XCP is a universal measurement and calibration protocol standard set by the Association for Standardization of Automation and Measuring Systems (ASAM) e.V. in 2003 [35]. It evolved from the CAN Calibration Protocol (CCP) when the automotive industry adopted protocols such as LIN, FlexRay, and MOST. Today XCP also supports Ethernet.

XCP makes it possible to do debugging, calibration, measurements, and even flash programming *without* having to connect a debugger to the ECU. XCP integrates itself in the existing communication channel without disrupting it and accesses the ECU's memory addresses of interest. XCP uses a master-slave setup where the master is the device requesting data (e.g. a personal computer with CANape installed), while the slave is an ECU with an XCP driver running on it. A webinar with more details on XCP can be found at [36].

2.4 Related Work

Car hacking has always been a hot topic, from the days when it was possible to hotwire a car (start the ignition without a key by shorting wires), until today when there is more and more technology inside the car. For attackers, the increasing number of ECUs and interfaces present additional potential means to get access to the vehicle and its data.

2.4.1 CAN bus based hacking

SANS Institute's "*Developments in car hacking*" [37] explains that (until recently) security has *not* been one of the top concerns of car manufacturers. The paper investigates the inherent insecurity of standard bus protocols, such as CAN, which also did not have security in mind when it was developed. The paper lists car hacking attacks during the period of 2010 to 2015 and gives suggestions about how to increase security by using encryption, authorization, and a few other methods.

Karl Koscher, et al. [38] showed in 2010 that it is possible to gain full control over a car through its OBD port. They showed the consequences of getting control over internal buses and anticipated the increased attack surface exploited in the following years due to advances in telematics [39] [40], Vehicle to Vehicle (V2V), and Vehicle to Infrastructure (V2X) communications. Although this proved that better security is necessary, their paper was initially met with skepticism because physical access to the car's internal OBD port was required.

However, even without getting into the car, Stephen Checkoway, et al. [41] showed that it is possible to do attacks *without* physical access. Building on the fact that it is possible to gain remote control, Miller and Valasek [42] reverse engineered the CAN buses of a Ford and a Toyota car, enabling them to control most parts of the car. The results can be seen in their video [43]. The tools they used are publicly available at [44]. Next, they published a white paper called "*A survey of Remote Automotive Attack Surfaces*" [45] that lists multiple car brands from 2006 to 2014 and investigates their internal architecture and peripherals. It rates each of these cars' "*hackability*" based on the size of their attack surface, architecture complexity, and physical features. In 2015, the same researchers using a 2014 Jeep Cherokee, which their research had shown to be most hackable, gained remote control over it [46]. A video of this is available at [47].

Craig Smith in his book "*The Car Hacker´s Handbook*" [48] gives a Metasploit payload for exploiting an infotainment system running on Linux OS that opens the doors of the car by sending CAN messages over UDP.

2.4.2 DoIP and related attacks

There are also studies on the security of DoIP [49], SOME/IP [50], and AVB [51]. All of these studies proposed security mechanisms which are being included in these standards, but in the end, it is the manufacturer's decision whether they are going to implement them or not.

Johan Lindberg's thesis [49] lists all the protocols DoIP depends upon and possible attacks for every one of them. He states that they can all "*be abused in one way or another*". The draft documents of the ISO 13400 protocol at the time of writing did *not* offer any secure alternatives. However, there are ways to protect against all of the mentioned attacks, but again doing so is up to the manufacturer.

2.4.3 Anomaly detection

Herold, et al. [50] propose an anomaly detection system implemented on an event processing engine. It detects protocol violations and possible attacks. They have made all the tools available on Github [52] [53] under a General Public License (GPL).

2.4.4 Security of Ethernet-based solutions

Robert Boatright and Joseph Tardo [51] pointed out the security benefits of Ethernet over conventional bus-based systems. They believe that Ethernet is inherently safer due to its switched network architecture. Moreover, Ethernet supports a vast number of security-oriented protocols, unlike busses which were not designed for security and mostly rely on "*security through obscurity*". Their paper also cites examples which prove that obscurity is no longer a security option.

2.4.5 Reaction of car manufacturers to security problems

A critical aspect which needs to be mentioned is how car security is handled by car manufacturers. Until recently, security has not been very high on the manufacturer's agendas. Recent research has shown that car manufacturers are still not sufficiently concerned with security. Reactions to the research results of earlier researchers ranged from suing the authors of [54] so they could not publish their work [55] to not fixing the vulnerability within 5 years, even while the researchers were kind enough not to publish which car's security flaws were exploited. When researchers did publish details, then the fix came a lot faster (assuming they were not sued by the manufacturer) [56]. The best possible example of how to handle security problems is that of Tesla. Tesla hires security specialists [57] and even has a bug bounty program in which it pays a sum of US$100-10 000 for every bug someone reports [58].

2.4.6 Further details of Hacking CAN

Even though CAN is a completely different communication system from automotive Ethernet, there are some things which can be learned from CAN hacking. First of all, CAN is a quite an old protocol. Throughout the years there have been many hacking attempts. Although CAN and automotive Ethernet standards may have nothing in common, the first hacking steps are similar; as both communication systems are within the car. The first thing that can be learned and applied to Ethernet is with regard to how to gain access to the bus.

Looking at a car from the outside someone who has no understanding of CAN might never know that it even exists, as the wires are not visible on the exterior of the car. However, for those who know that it does exist and know what CAN is used for we can assume that it would be possible for them to find the relevant wires. One way thieves gained access to the CAN bus in some cars was ripping off the side mirrors. With automotive Ethernet this is not as simple. Assuming there is a camera on the outside of the car, ripping it off would not give control over the whole network as it is the case with a bus system, but would only lead to the next switch.

Another common feature is that in the same way that we need an interface for CAN we also need an interface for Ethernet. Again due to the wide deployment of CAN, there are many CAN capable devices available at prices as low as 10€. For Ethernet the situation is a bit different, as the standards are new, hence the manufacturers have a limited number of products that supporting these standards. However, transceiver hardware such as the BCM89810 can be easily purchased (for ~8€), so a device that can interact with automotive Ethernet is relatively straightforward to build.

3 Methodology

This chapter introduces the research methodology followed throughout the remainder of the work. As pointed out earlier the Open Source Security Testing Methodology has been chosen as the methodology for this work. Since the methodology is a very extensive, and its manual alone is a 200-page book, there will not be enough space to go into detail about all of it. Hence parts which are irrelevant have been skipped.

OSSTMM [6] is a great resource with tangible examples of every defined term and process. Furthermore, the manual is a great read for gaining a wider perspective on security in general. The reader is advised to refer to this manual when the explanations provided in this thesis are insufficient.

Since security has become a broad field itself, there were multiple methodologies to choose from. However, OSSTMM (version 3.02) has been found to be most suitable due to its wide area of applicability, the freedom it gives for choosing tools, and its novel risk assessment approach. The following sections will introduce OSSTMM.

3.1 Research Process

Being a very broad methodology OSSTMM can be applied to a wide range of fields. These fields in which interaction can occur have been defined as channels. There are five channels defined by the methodology, specifically: human, physical, wireless, telecommunications, and the data networks. This research will focus on the last one because only Ethernet is of interest, and this technology falls into the domain of data networks.

For a thorough check of every channel, the methodology defines 17 test modules which all include multiple tests. Arranged in the "*one methodology*" workflow (as shown in Figure 3-1), they cover the whole system's security. They have been organized into four phases:

1. Induction (yellow)
2. Interaction (orange)
3. Inquest (blue)
4. Intervention (red)

Some modules focus more on users, the legal, or the business side; hence they are not relevant for this research. Details about every module and how they relate to different channels can be found in the OSSTMM manual.

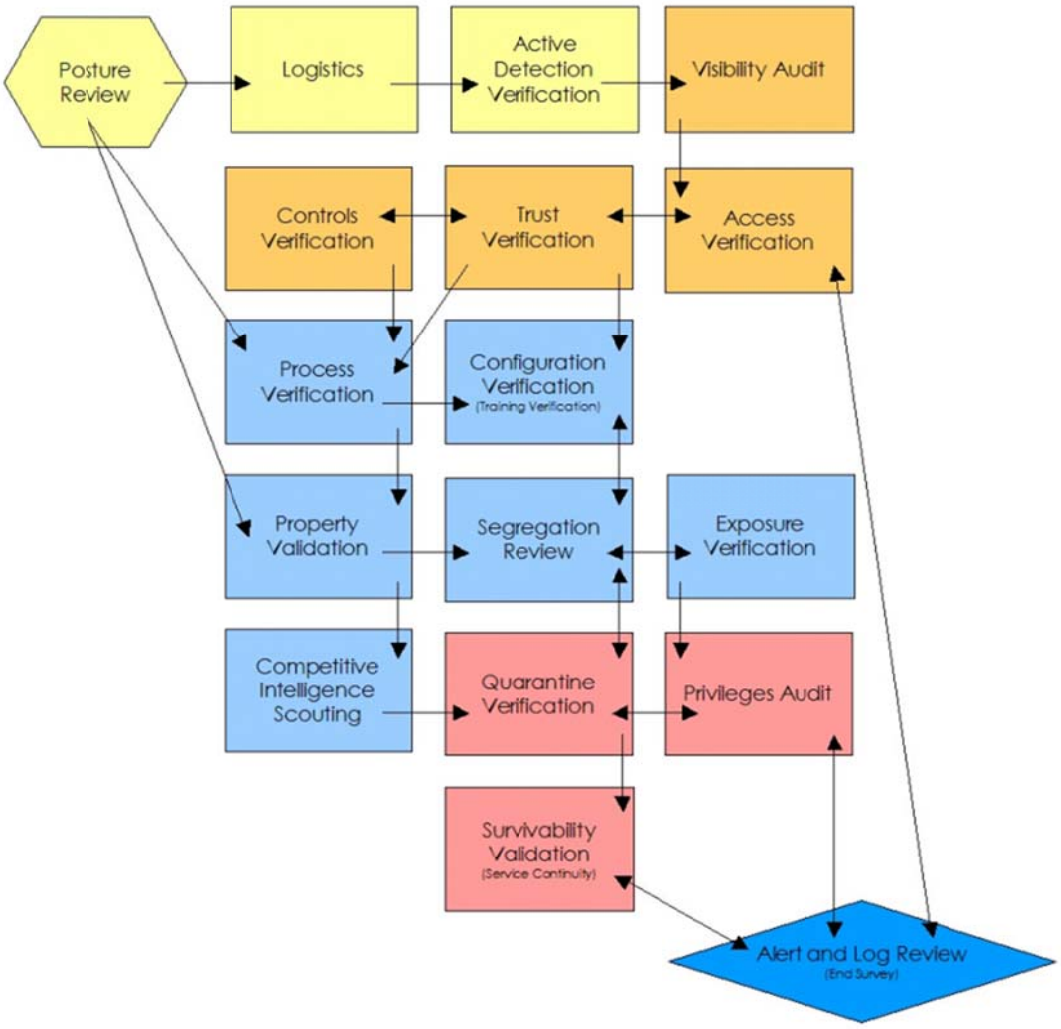

Figure 3-1: "*One Methodology*" Workflow [6]*

All of these modules consist of multiple tests. Every test has to be well defined. OSSTMM gives a 7 step process for defining a test:

First, the **assets** have to be defined, as these are what need to be protected. The asset in most tests will be the ECU itself including its firmware, the things it controls, and its normal operation. Next **control** mechanisms are put in place to ensure the safety of the asset. The aim of the test is to see if there are any weaknesses or vulnerabilities in these controls, leading to so-called **limitations**.

The engagement zone is represented by the asset, controls, and processes around it. This is where interaction with the asset happens.

The test **scope** includes the engagement zone and everything around the asset that is necessary for the asset to function. This includes the electricity, regulations, processes, protocols, and resources, even though some of these might be outside of the tester's direct control.

---

* Figure courtesy of OSSTMM's creative commons Creative Commons Attribution-Non-Commercial-No Derivative Works license.

Interactions within the scope and outside the scope are documented since they represent the (potential attack) **vectors**. These include interactions between the elements as well as their direction.

The equipment necessary for the execution of the test has to be defined. Since in some cases vectors may require interaction on multiple levels the details of the **channel** have to be specified.

The information expected from the test needs to be defined. Is the expected information just the response from the system or does it also include the effect of the interaction on the system? Depending on the answer to this question a specific **test type** is chosen. OSSTMM defines 6 types of tests ranging from blind tests in which the analyst has no prior knowledge of the target, but the target knows all about the audit, to the reverse, where the analyst knows everything about the target which in turn knows nothing about the audit. In order to speed things up, but still thoroughly test the system a combination of a double-blind (black box) and a double gray (white) box testing will be used for most tests. In this way, the system's response will be checked in all cases, but time will not be spent on known dead ends, hence more targeted attacks will be possible.

The last step is normally making sure that the **Rules of Engagement** are followed. They are guidelines concerning sales, contracts, negotiations, scope, planning, and the test process itself. Since this research is done solely *within* a test environment there is no need for rules of engagement.

After performing the test, the **Attack Surface** will be defined. This will show the vectors through which the scope is vulnerable.

3.2 Research Paradigm

OSSTMM backs away from the traditional security risk evaluation because risk is highly subjective and dynamic; whereas the attack surface is static and objectively measurable. It does this by defining the term *operational security*. In doing so it ignores the knowledge of the system's configuration in order to avoid being misled by assumptions about how the system should work, but rather enables us to find out how it, in fact, does work.

OSSTMM states that if a threat has no possible way of interacting with the system then the system is 100% secure. The problem is that in order for the system to operate there usually has to be some interaction. This is where the terms *control* and *safety* come into play. The methodology defines 10 controls that are there to ensure safety when interactions are possible. The methodology also defines *limitations*. These limitations represent exposures, vulnerabilities, weaknesses, and concerns of operations and controls, and anomalies which occur without clear reasons. Figure 3-2 shows the relation of all operations, controls and limitations, and their dependence on each other.

Category		OpSec	Limitations
Operations		Visibility	Exposure
		Access	Vulnerability
		Trust	
Controls	Class A - Interactive	Authentication	Weakness
		Indemnification	
		Resilience	
		Subjugation	
		Continuity	
	Class B - Process	Non-Repudiation	Concern
		Confidentiality	
		Privacy	
		Integrity	
		Alarm	
			Anomalies

Figure 3-2: Representation of Operations, Controls, and Limitations [6]*

According to OSSTMM perfect security is the correct balance between operations, controls, and limitations. As long as there is a control for every operation **and** this control has no limitations or the limitations are secured by other controls, the system is secure.

The author will avoid defining all these terms since it would take up an enormous part of this work. Moreover, OSSTMM is licensed under a creative commons license and hence is freely available (see [6]).

One important thing has to be defined here, that is *actual security*. This represents the attack surface of a system. Although the actual security is expressed in terms of percentages; the relation between operations, controls, and limitations are expressed on a **log**arithmic scale (see Section 3.6).

3.3 Data Collection

Data will be collected through a security audit of one channel. The whole process is visually represented in Figure 3-3.

* Figure courtesy of OSSTMM's creative commons Creative Commons Attribution-Non-Commercial-No Derivative Works license.

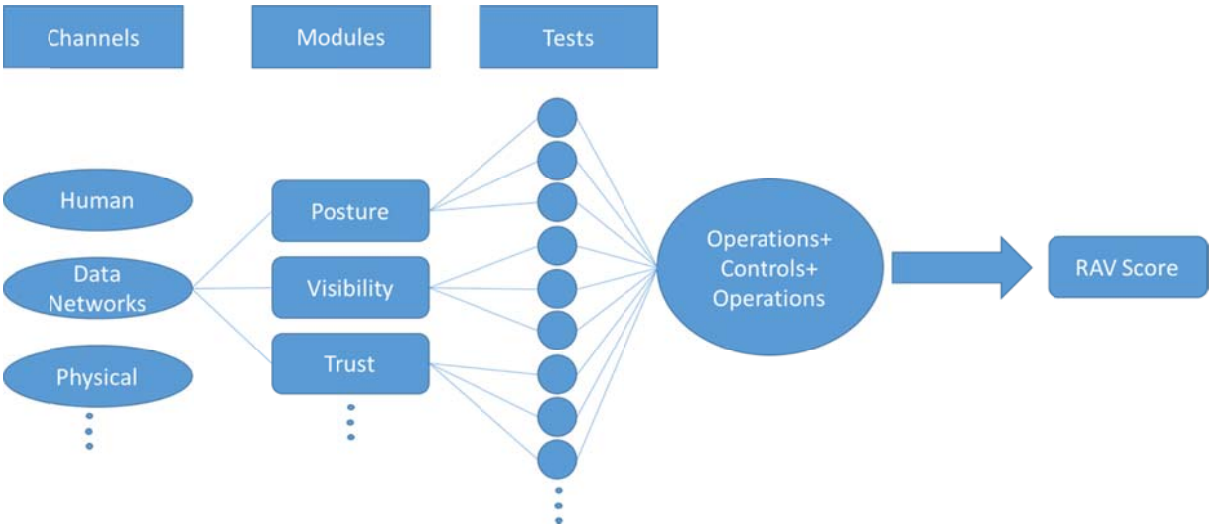

Figure 3-3: Workflow of getting a RAV score

Out of the 10 channels defined by OSSTMM the Data Networks channel is audited. It covers 17 different test modules consisting of multiple tests. After all the tests have been conducted, the results are categorized into operations, controls, and limitations which are used for calculating the RAV score.

The test results and details about more significant ones will be provided in Chapters 4 and 5 respectively.

3.4 Experimental Design/Planned Measurements

In this section, the testbed and the hardware/software which constitute the testbed will be described.

3.4.1 Testbed

Figure 3-4 shows the testbed. On the far right is the device under test, a Vector VC121-12 ECU (see Section 3.4.2.1). This ECU uses a 100Base-T1 physical layer (as noted earlier this is also known as BroadR-Reach). In order to communicate with the personal computer (PC), Vector´s network interface VN5610A (see Section 3.4.2.2) is necessary for interfacing between the two different physical layers 100Base-T1 of the ECU and 100Base-TX supported by the PC.

The green line in Figure 3-4 represents the automotive Ethernet connection from the ECU, while the blue line represents a standard 100Base-TX Ethernet going to an Ethernet-to-USB converter which is via USB connected to a PC running Windows 7. This PC has a virtual machine running Kali Linux [59]. Kali has been chosen for multiple reasons. First is the fact that it is an OS specially developed for penetration testing, hence a plethora of necessary tools comes preinstalled. Secondly due to the strict regulations of the company only programs approved by the data processing department may be installed on Windows. In a virtual machine, on the other hand, the only rule was to keep it contained and disable access to company files. Hence a Kali virtual machine was a perfect choice. The virtual machine´s virtual Ethernet

adapter is configured to be in bridged mode with the physical USB-to-Ethernet converter, effectively acting as a hub between Windows, Linux, and the ECU.

Since the Vector tools and other programming tools are developed for Windows, Configuration and setup (i.e. compiling, flashing, and debugging) are done directly from the PC. However, the network tools integrated into Kali exceed the testing capabilities of Windows. For that reason, testing is mostly done from the Kali virtual machine. The remaining element in the diagram, the iSYSTEM AG iC5000 On-chip Analyzer (see Section 3.4.2.3) is a debugger which has a direct JTAG connection to the ECU. This JTAG connection makes flashing and debugging of the ECU possible.

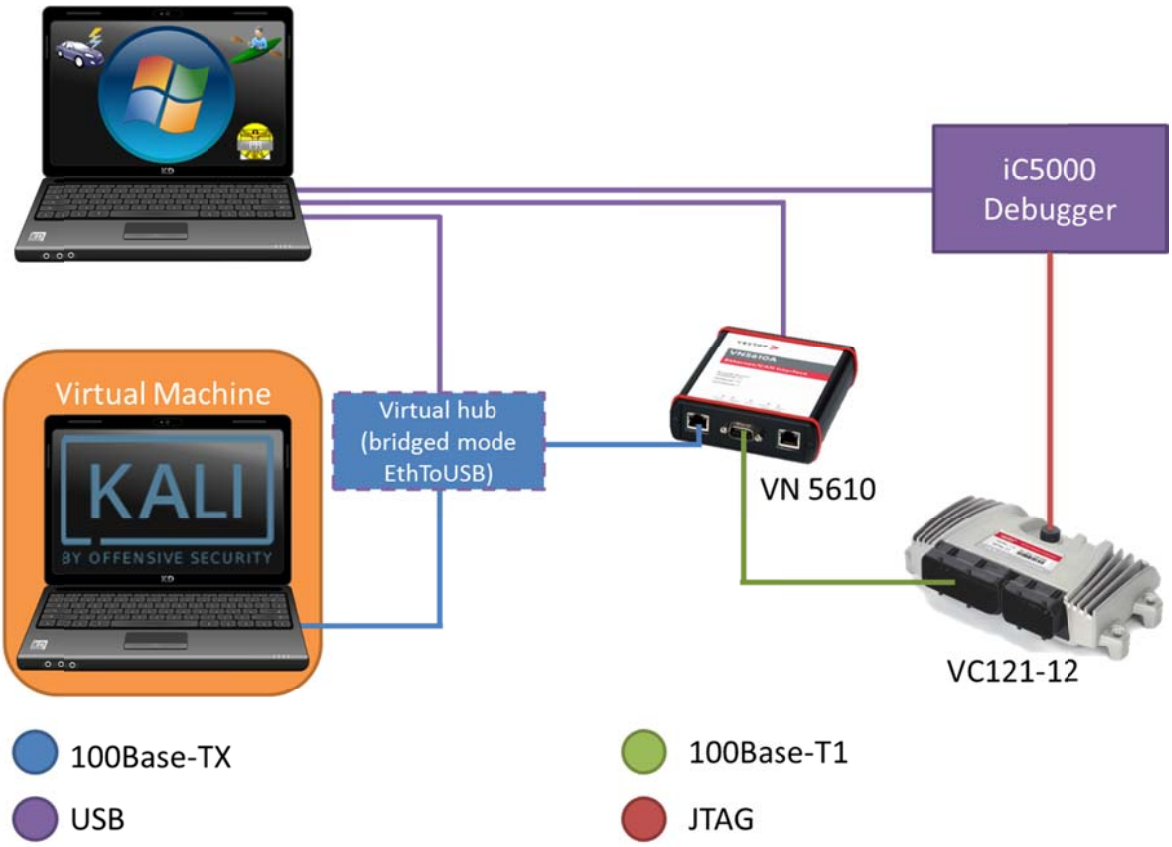

Figure 3-4: Testbed Diagram*

3.4.2 Hardware/Software to be used

This section lists all the hardware/software used in the testbed. The host machine (PC) on which all tests were conducted was a Dell Precision 7510 with an Intel i7-6920HQ processor running at 2.90 GHz, 64 GB of RAM, and Intel 1000Base-T Ethernet Connection I219-LM.

* vFlash, CANoe, DaVinci icons, VN5610, vc121-12 images courtesy of Vector Informatik GmbH
Kali illustration: https://commons.wikimedia.org/wiki/File:Kali_Linux_2.0_wordmark.svg
Laptop illustration: https://pixabay.com/de/laptop-notebook-computer-schwarz-158648/
Windows illustration: https://pixabay.com/de/fenster-start-button-kugel-47202/

3.4.2.1 VC121-12 Vector ECU

The VC121-12 Vector ECU was developed by Vector Informatik GmbH. It is a universal ECU. It is primarily meant to be used for prototyping and small-scale production of sports cars, agricultural vehicles, and such. Hardware wise this ECU is equipped with 1 FlexRay, 6 CAN, 2 LIN, and 1 Ethernet BroadR-Reach interface; three CPU cores, a Safety CPU, multiple analog and digital I/O pins, Pulse Width Modulation (PWM) outputs and much more. More details about it can be found in the datasheet [60].

3.4.2.2 VN5610 Ethernet/CAN Interface

The VN5610 is Vector's USB to CAN/Ethernet interface. It supports 100Base-T1, 10Base-TX, and 100Base-TX. Multiple usage modes are supported. It can be used as a standard network interface - which enables sniffing of CAN and/or automotive Ethernet communication through USB. In addition to sniffing, it can simulate absent network nodes in a restbus simulation (for details see Section 3.4.2.6). Another mode of operation is the standalone media converter mode. This mode was used for most experiments in this research as it creates a pass-through from 100Base-T1 to 100Base-TX and hence enables the PC to see all of the 100Base-T1 traffic. Additional information about these different configurations and the device itself can be found in the device's manual [61].

3.4.2.3 iSystems iC5000 On-chip Analyzer

The iC500 base platform is used in combination with the JTAG debug/trace module and the MPC5XXX cable adapter (used for debugging the ECU's microcontroller via USB). More information about all of these components can be found in [62]

3.4.2.4 Ethernet to USB adapter

Due to strict security measures in Vector Informatik GmbH in order to maintain Internet connectivity through the integrated network card of the PC, an additional Ethernet to USB adapter was used to create a sandbox environment for conducting the tests. This also makes it easier to track the packets since no filters for Internet traffic have to be set up. It should be noted that the experiments *could* have been conducted via the integrated network adapter as well, but the security policy of the company did not allow that. The adapter has to be used in bridged mode between the host and the virtual machine, so that it acts as a hub, hence all communication can be seen by both machines. The Startech USB13000S adapter was used for this research, but any 100Base-TX compatible adapter could have been used.

3.4.2.5 Wireshark

Wireshark is probably the most widely used network sniffer and protocol analyzer. It is a very successful open source community driven project started by Gerald Comby in 1998. There is even an annual SharkFest which brings together Wireshark users, developers, and enthusiasts [63]. This tool is particularly useful for capturing

packets, looking for patterns, and analyzing packets manually due to its highly customizable graphical user interface (GUI).

3.4.2.6 CANoe

CANoe is a program developed by Vector Informatik GmbH. It is designed for testing, diagnostics, and simulation. CANoe was specifically developed to sniff and analyze automotive networks and protocols through special interfaces such as the VN5610. It is, in essence, the equivalent of Wireshark for automotive networks. Another of its capabilities is so-called "*restbus simulation*". In a restbus simulation, it is possible to create a networked system where some of its elements are physical and others are simulated. This means that an ECU can be tested as if it were inside a real car, by simulating the other elements that would be connected to it. More information and videos of CANoe in action can be found in the "*ECU Testing*" section on Vector's website [64].

3.4.2.7 Scapy

One of the tools which does not come preinstalled with Kali, but definitely should is Scapy. Scapy is an extremely powerful packet manipulator written in Python. It can sniff, analyze, assemble, and send packets at different layers. It is possible to send raw Ethernet frames, IP packets, UDP datagrams, and higher layer protocol messages. These transmissions can be manipulated at the level of each single bit, permitting even malformed packets to be sent. This makes Scapy perfect for fuzz tests. By using Scapy's sniffing capability, it can analyze and react to any packet it receives. More information about Scapy as well as a download link can be found at [65].

3.4.2.8 DaVinci Configurator Pro

DaVinci Configurator Pro is a tool used to configure, validate, and finally generate the basic software (BSW) and the runtime environment (RTE) present in every AUTOSAR (AUTomotive Open System ARchitecture) ECU. AUTOSAR has not been mentioned until now due to the fact that even without knowing what it is, it is possible to understand this research fully. Suffice it to say that AUTOSAR is an approach of the automotive industry to standardize all software components so that the same software need not be written over and over by different companies. It does this by defining an open standard architecture for ECUs. For the curious reader who wants to learn more, an introduction to AUTOSAR can be found at Vector's e-learning portal [66].

All of the 60+ basic software modules with their thousands of parameters can be easily configured using DaVinci Configurator Pro. This includes configuring everything from communications to error logging and diagnostics.

3.4.2.9 Wind River Compiler tool suite

A C/C++ compiler, assembler, and linker based on the Diab Compiler comprise the Wind River Compiler tool suite. This tool suite supports multiple target architectures

including the PowerPC architecture of the microcontroller used in the VC121-12. More information about this tool suite can be found on Wind River's website [67].

3.4.2.10 winIDEA

iSystem's winIDEA is an integrated development environment developed by the same company that developed the iC5000. This integrated development environment is used when developing software for the iC5000. It runs on Microsoft's Windows operating systems. More information about it is available at iSystem's website [68].

3.4.2.11 vFlash

Vector's tool for programming ECUs is called vFlash. It supports flashing via multiple communication protocols. Of these, DoIP over Ethernet is the one of most interest for this research. This tool offers a simple GUI with status and progress messages. Flashing of compressed and encrypted data is also possible. More information, as well as setup and usage instructions, can be found in its manual [69].

3.5 Assessing Reliability and Validity of the Data Collected

This section presents how reliability and validity of the data collected via the testbed are assured.

3.5.1 Reliability

Since the OSSTMM defines a workflow and tests which need to be conducted the reliability is more or less guaranteed (if one follows this methodology). The test environment is well defined, making it possible for someone else to redo the experiments and test the reliability of the data.

3.5.2 Validity

Any test errors are documented so that their impact on the validity of the test results can be evaluated.

3.6 Planned Data Analysis

The final result of the data analysis is the definition of an attack surface. As already mentioned this attack surface represents the vectors from which our scope is vulnerable. As such it is a very objective measure of security. More precisely the attack surface identifies the relations between operations, controls, and limitations.

OSSTMM defines RAV as a *scalar* metric for measuring the attack surface. The benefit of having a metric for security is the fact that it is applicable to all systems. Another benefit of RAV scores is that they can be easily recalculated when the system is changed or the impact of an additional control is to be evaluated. Figure 3-5 shows

how RAV scores can be used to track the security of a system over time. RAVs are expressed through a base 10 logarithmic equation. A RAV score of 100 represents a perfect balance between operations, controls, and limitations, thereby representing perfect security. Although that is usually interpreted as a percentage it is actually a logarithmic scale whose values can go above 100. Values greater than 100 indicate *redundant* controls. This might sound desirable, but can be problematic since additional controls increase complexity and in some cases add interactions.

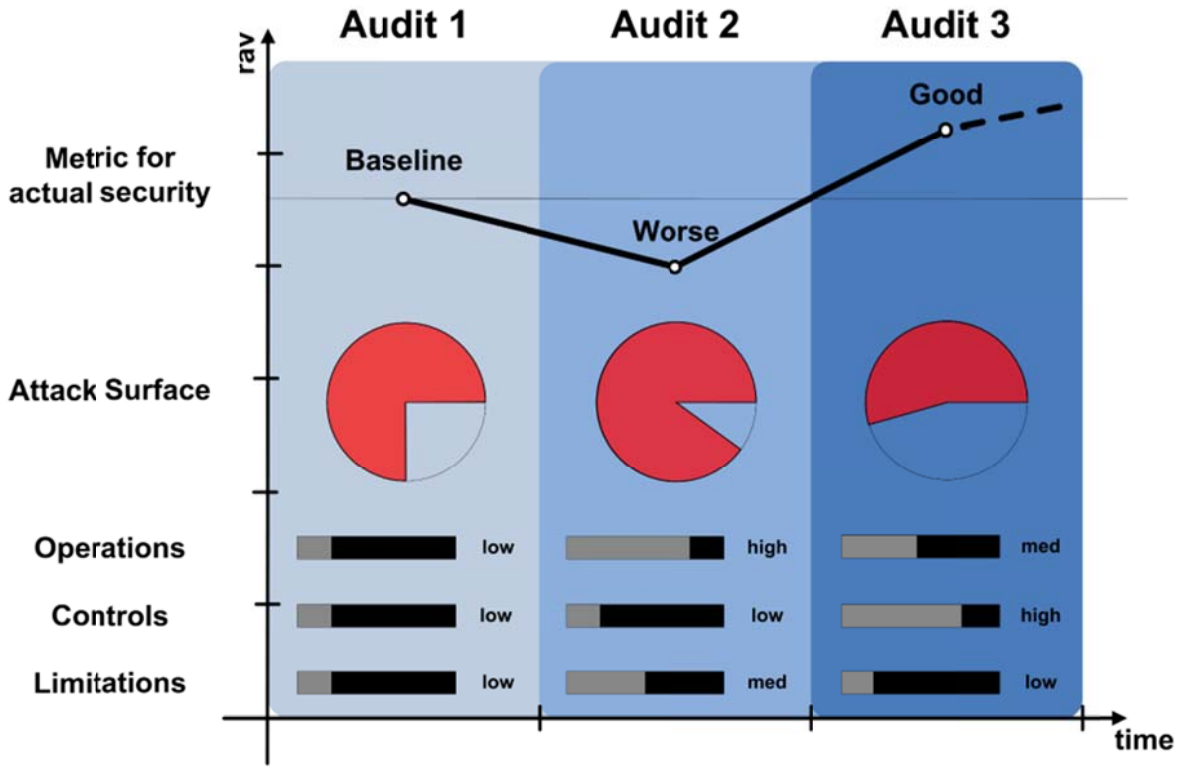

Figure 3-5: Security expressed in RAV scores over time [6]*

As the methodology manual says the fact that the scale is logarithmic helps "*understanding the big picture*" [6]. For this reason, the term actual security was introduced earlier. This term is applicable to all channels and conveys the same information. The manual gives the example of a 1000 computer system and a 10 computer system where the same RAV score is comparable. The manual goes further by comparing computer networks to buildings and shows that even in that case the RAV scores are comparable.

* Figure courtesy of OSSTMM's creative commons Creative Commons Attribution-Non-Commercial-No Derivative Works license.

3.6.1 Data Analysis Technique

Information collected through the data networks security tests will be used to calculate RAV scores, thereby describe the system in terms of operations, controls, and limitations.

3.6.2 Software Tools

The RAV calculator spreadsheet provided by ISECOM will be used for calculating the RAV score. This spreadsheet is available at [70].

3.7 Evaluation Framework

RAV scores are the basic means of evaluating the security of a system. They give an overall score for the security of the system, and it is easy to see the effects of any change to the system. For evaluation of this work, the author has decided to calculate multiple RAV scores for different configurations of the system so that they can be compared. The different systems have been defined as follows:

a) Base RAV score which represents the Vector TCP/IP stack implementation without any special security-related configurations.

b) RAV score for a standard TCP/IP stack implementation with all vulnerable protocols in place which have **not** been implemented in the Vector TCP/IP stack.

c) Multiple RAV scores for different security-related configurations which the Vector TCP/IP stack has implemented at the time of writing of this thesis (static ARP cache, disabled ICMP echo requests, and firmware signatures).

d) A RAV score in which all security-related configurations are combined in one system.

e) A RAV score with possible future implementations for increased security (e.g. encryption, signatures, etc.).

It is expected that the RAV score of the system (b) will have a lower RAV score than the base system (a) because it has more operations while maintaining the same number of controls. The remaining systems are expected to have higher RAV scores since they all implement controls which system (a) lacks, or get rid of redundant operations which are present in the system (a).

4 Security Audit

Multiple tests have been conducted to perform a security audit. This chapter describes the results of these tests. It starts by introducing the system and the aims of the tests. Please refer to the methodology and the seven steps outlined in Section 3.1. The details of carrying out these tests are given in the following chapter.

In the following sections, modules defined by OSSTMM will be tested and the results evaluated. Note that the name of the section corresponds to the OSSTMM module. All modules will be listed, but some will not be tested because they are irrelevant for this research. The sequence of the tests corresponds to the one specified in the methodology manual in "*Chapter 11 – Data Networks Security Testing*" [6]. This has been done so that it is easier for the reader to refer to them. Details about tests including the tools used can be found in the next chapter.

4.1 Posture Review

All experiments are run on a test environment consisting of hardware and software owned by Vector Informatik GmbH. Given the fact that the software running on the system is test software for internal purposes only, no personal data is present. Note that this test environment was isolated from any other network environments while carrying out the testing. As a result, there are no ethical concerns regarding the security testing.

Other aspects of the posture review such as policy, culture, and age (as specified in the OSSTMM tests) were irrelevant given the nature of this research.

4.2 Logistics

The tests from this section ensure that there are no false negatives or false positives which would lead to inaccurate results.

4.2.1 Framework

As the tests are run locally, there is no need for the steps outlined in the framework section of OSSTMM.

4.2.2 Network Quality

For assessing network quality multiple tests with sending packets of different protocols and measurements of parameters, such as speed and packet loss were conducted.

TCP, UDP, and ICMP requests have been sent out and the responses were recorded. Each request was sent 100 times, and the average speed and packet loss have been recorded. The results and details can be seen in Section 5.1 (starting on page 53).

a) Consistent data rates and 0% packet loss was observed for all the tested protocols.

b) This test looks at the network in segments since the test setup has just one direct connection between the ECU and the PC in this test will be omitted.

c) As expected, no packet loss was observed due to the local nature of the communication.

4.2.3 Time

Due to the nature of the experiments this type of testing was irrelevant since it involves actions such as synchronizing times, holidays, and customer interactions.

4.3 Active Detection Verification

Active Detection Verification tests are necessary to ensure there are no filters which would negatively affect communication with the target. These tests can be split into two subcategories: filter and active detection. Each will be described below.

4.3.1 Filtering

a) Incoming packets from the hardware were all visible via Wireshark; hence there were no problems with ingress filtering.

b) Outgoing communication from the PC to the hardware was initially completely blocked by the computer's firewall. After the firewall was disabled all of the communication with the testbed ran smoothly.

4.3.2 Active Detection

a) There were no active responses in form of silent alarms or the like. Malformed packets were dropped silently.

b) vFlash has optional reporting capabilities. However, it was not possible to know whether the ECU itself keeps any kind of log since such a log would be stored in its internal memory (and this memory was not examined).

4.4 Visibility Audit

These tests index the targets and systems via direct and indirect interaction between live systems

4.4.1 Network Surveying

These tests are performed with the aim of surveying the network. They identify all hosts and services running on them. Well-known ports and vulnerabilities are investigated.

a) The perimeter of the attack is local, with the system defined in Section 3.4.1.

b) The system has been run via the flash bootloader and in application mode. Several different interactions were initiated. In the two given applications*, the following protocols have been sniffed and identified: ARP, ICMP, TCP, DoIP, UDP, IGMP (Internet Group Management Protocol), Time-Triggered Protocol (TTP), SOME/IP, and AVB/TSN.

c) Due to the local nature of the testbed, there were no name servers to investigate.

d) ARP broadcast requests and unicast responses could be observed to and from the target. This is already an early indication that ARP cache poisoning could be possible since the cache is not static.

e) Traffic and routing protocols were not observed. It is known that those are not implemented (as of now).

f) ICMP responses to echo requests were observed from the target. Additional request types were not implemented and even echo requests can be disabled (in the IPv4 configuration through DaVinci Configurator Pro – see Section 3.4.2.8), such that even these packets are silently dropped.

g) Normally Simple Network Management Protocol (SNMP) is tested in this test, but since it is not being used nor implemented, testing of SNMP was completely skipped.

h) This test usually investigates the responses from targets on specific ports with a low time to live (TTL), but since it is known which ports are open, only the known open port 13400 and a random closed port 15463 were tested. In this way the system was still tested on open and closed ports, but time was not spend unnecessarily on multiple closed ports which behave in the same way. It was observed that:

   ICMP Echo requests are the only type of ICMP messages implemented as of now. Even they can be disabled in the configuration. The remaining ICMP message types were probably left out because of their accompanying security concerns.

   TCP On port 13400 an incoming SYN flagged packet is answered with a SYN and ACK flagged packet, while on port 15463 the answer is flagged with a RST and ACK packet. This proves the already known fact that port 13400 is open, while port 15463 is closed. Moreover, this behavior indicates that the ECU is acting in accordance with the RFC 793 [14]. The speculated downside of following the cited RFC 793 is that even though a closed port is probed an answer is received revealing there is a host at that IP address. This means that even though ICMP echo requests are disabled, probing a random TCP port would tell an attacker if there is an active host at that address or not.

* These applications were the flash bootloader and the sample application software.

UDP Since UDP is a connectionless protocol there is no handshake which can be tested out. Moreover, there is nothing comparable to an ICMP echo request. For this reason, UDP scans are hard to do. Sending a random packet to an open port will not yield any answer. On the other hand sending a packet to a closed port according to RFC 792 [71] may send an ICMP destination unreachable type message as a response. Since that type of ICMP messages was left out of the stack's implementation, open and closed ports behave in the exact same way, i.e., they do not respond at all. The only way to identify an open port is by sending an actual message for the service running on it. This has been done with DoIP diagnostic messages based on UDP.

i) This test requires tracing routes of ICMP packets to targets. Due to the fact that the target is on the same subnet as the testing equipment this test can be skipped.

j) Similar to the previous test, only for TCP packets, also skipped.

k) Similar to the previous test, only for UDP packets, also skipped.

l) The initial TCP sequence numbers seem to be completely random with no dependence on time. This makes it hard to guess the sequence numbers and perform attacks which rely on knowing the sequence number.

m) IP IDs increment by 1 with every new packet. This makes idle scanning [72], a stealthy scanning technique, possible. Even though the consequences of this are not extreme in the car environment, it is something that can easily be circumvented in the stack's implementation.

n) This test verifies the use of Loose Source Routing. As of now, loose source routing is not implemented, so this test can be skipped.

4.4.2 Enumeration

Most tests with regard to enumeration were inapplicable because they include tests such as searching newsgroups and investigating e-mail headers. The ones which could be conducted for enumerating services running on common TCP and UDP ports were skipped due to the fact that the only configured open ports are known; hence there is no need to check for all other ports as they are known to be closed. Packets with malformed checksums were sent to open ports, but all of them were silently dropped without causing any harm to the target.

4.4.3 Identification

Packets have been sniffed with Wireshark and CANoe. With the help of those two tools, it was possible to assign all observed packets to the corresponding services which are used in the application software and/or the bootloader. This knowledge has been used for constructing future attacks mentioned later and described in Chapter 0.

4.5 Access Verification

In this phase access points to the target are identified.

4.5.1 Network

With regard to network access verification tests d) through f) will be omitted as they are inapplicable due to relying on unused services. Instead, only the following points are considered:

a) UDP is used for DoIP and SOME/IP. UDP packets for DoIP are accepted from any IP address, while SOME/IP is configured for specific uses and has dedicated IP addresses and ports for those.

b) There are no VPN services at this point in time. There might be some in the future (for example, for over the air updates).

c) This test includes manipulating services and routing in order to get past access restrictions. Since VLANs are widely used, the first option was to try out a VLAN double tagging attack. However, it turns out that VLAN double tagging is **not** supported by the stack.

4.5.2 Services

The tests defined in the methodology for the services section are mostly related to protocols and services which are not used within a car. They range from HTTP to voice over IP. Additionally, these tests look at the uptime of the system and try to spot recent security patches which could not have been applied since the system was not restarted for long periods of times.

Regarding the automotive environment services for DoIP on known ports have been tested and responses have been attained. Additionally, SOME/IP has been observed to operate on UDP and different ports for different services. In the case of the application software which was running it - two different services of which one was a simple calculator and the other a continuous periodic chainsaw-like triangle signal.

4.5.3 Authentication

These tests investigate the use of authentication. They investigate where authentication is necessary and how it is done:

a) Operations requiring authentication are: deleting the memory and flashing new firmware through the flash bootloader.

b) Depending on the importance of the ECU, different authentication methods are possible. The most common one is seed key authentication in which both the communicating devices know an algorithm for calculating the key. The ECU provides a seed on the flasher's request. The flasher responds with a calculated key based upon the provided seed. The ECU checks whether the key

is correct and grants or withholds access. There are multiple security levels defined by the Unified Diagnostic Services (UDS) ISO14229 [73]. These different security levels provide different privileges. It is worth noting that this communication is done over DoIP which in turns uses TCP.

c) The identification which leads to authorization is the authentication process itself. Since the standard ISO13400 [74] defines the destination port to only be statically 13400, while the source port and IP address any random values; requests coming from any IP address and from any port will be responded to. It is sufficient to know the proper command codes (e.g., 0x0001 for a vehicle identification request message) in order to be able to communicate. Nevertheless, authentication through the seed key exchange is necessary for actions defined under a).

d) The authentication method used is a broadly used and verified seed-key verification method which is standard for the automotive industry.

e) The strength of authentication can vary because of the fact that every manufacturer defines their own seed-key algorithm. There are cases of very weak ones such as XORing middle two bytes of the seed [42]. Nevertheless, proper algorithms, long keys, and a limited number of retries make brute force attacks difficult and infeasible in a reasonable time.

f) The authentication process has been verified by observing the flashing session started by vFlash on the ECU. All communications including the seed-key exchange were observed with Wireshark. The fact that encryption is absent makes it possible to reverse engineer the algorithm if the key generation process is fairly simple and the key is short.

g) No logic errors in the application of the authentication have been identified during the test. It was observed that after two aborted connections in a row all future requests would be rejected and a reboot of the ECU was necessary to enable new connections.

4.6 Trust Verification

This section of tests concerns communication which does not need identification or authentication.

4.6.1 Spoofing

These tests are concerned with spoofing, which in the context of network security means successfully pretending to be a legitimate network host and being able to communicate with other hosts without them noticing and engaging in the data exchange. The following points are relevant:

a) Spoofing is generally possible due to the absence of authentication. When a packet is on the wire there is no way of knowing for sure where it came from since anyone can craft a packet with the source addresses of his/her choice.

No trusted hosts have been identified which have any special privileges related to communicating from a specific IP address.

b) The ARP cache can be poisoned if the ARP cache is not made static. Static cache is disabled by default, and needs additional configuration. It has to be kept in mind that AUTOSAR does not include static cache in the standard, but leaves it up to manufacturers to choose whether to implement it or not. Vector has added an optional static ARP cache in its stack implementation.

4.6.2 Phishing

Tests from this section are of no interest at this point because phishing in itself is based on the assumption that a person is the weakest link. Such interaction is outside the scope of this thesis.

4.6.3 Resource Abuse

These tests investigate to which degree if, at all, resource abuse is possible.

a) Usually, access to web servers without credentials are examined in this test, but since there are no web servers in the given test setup, this test will be omitted.

b) The test available at Ethernet Frame Padding has investigated whether any system specific data is being sent to the outside through padding as it was the case with the previously discovered vulnerability on some systems called Etherleak [75].

c) Here load balancing and other continuity measures are tested, but since they are not present in the system the test will be omitted.

4.7 Controls Verification

The following tests investigate controls in detail.

4.7.1 Non-repudiation

Non-repudiation basically means that based on the received data itself it is possible to know who the sender is. Non-repudiation is usually done with signatures which rely on asymmetric cryptography and signatures can only be issued by the owners of the private key, which by definition is only one host. They can be checked by the public key which is known by all the parties which need to communicate with the sending host. In the current implementation, other than firmware updates having signatures, there is no way of ensuring non-repudiation.

a) No logging mechanisms have been observed which would help support non-repudiation

b) None of the processes observed require any kind of identification

c) These tests assume that an identification is done by preceding interactions, hence these tests will be omitted.

d) Almost all communication defeats repudiation. There are no signatures to any messages except to the firmware in critical ECUs. All traffic can be forged, and there is no way of guaranteeing non-repudiation with the system as it is.

4.7.2 Confidentiality

This section involves investigating the use of encryption, but since encryption is not used at all at present these tests will be skipped.

4.7.3 Privacy

In these tests, the privacy of communications is examined. Two key points are:

a) Signatures are used in some cases for flashing when the ECU is responsible for handling safety-critical tasks.

b) There is one UDP port which by configuration should be open and used for something, but no outgoing traffic from that port has been identified, nor did it respond to DoIP requests. The type of request to which a response would come was not identified nor is it clear if the port is actually used for anything.

4.7.4 Integrity

There are a few ways to ensure the integrity of the sent data. There are checksums which guarantee that no data has been changed or malformed during transmission. Furthermore, the firmware of critical ECUs is signed so that the source can be verified.

4.8 Process Verification

These tests investigate the effectiveness of controls within the maintenance of the processes and their diligence.

4.8.1 Maintenance

No security notifications have been observed by the ECU itself. It has been observed that after two dropped TCP connections on the DoIP port all following connections are rejected until a restart of the ECU is performed. Other than that it has been observed that vFlash drops connections if something goes wrong in the flashing process.

4.8.2 Misinformation

While flashing the flash bootloader, vFlash will close the connection due to a broken connection. However, spoofed packets could be sent to it so that vFlash does **not** see the connection being dropped, hence does not close the connection. Alternatively, if

ARP poisoning has worked with a man in the middle attack, then the packets which should close the connection can be filtered out.

4.8.3 Due Diligence

No gaps between practice and requirements have been observed, hence there is nothing to document in this section.

4.8.4 Indemnification

Indemnification investigates how well the system is insured and secure from a legal point of view. The wording in the insurance contracts and security disclaimers are looked at in detail. Since that is not part of this research this section will be omitted.

4.9 Configuration Verification

This section investigates possible misconfigurations.

4.9.1 Configuration Controls

These tests check if the controls are configured properly.

a) Depending on the system the configuration changes, but in essence, there are very powerful configuration options which depending on the use case might or might not be practical. One example is static ARP cache which inside a normally static automotive environment can be used as an advantage.

b) This test investigates paths for bypassing access control lists. They do not exist in the setup as of now. What do come close to them are the different security levels which are available in the bootloader. Nevertheless, they are not linked to specific users or devices but require additional levels of authentication.

4.9.2 Common Configuration Errors

The tests in this section are there to investigate if any common configuration errors have been made.

a) All services have been found to be matched with the intended systems and not to be redundant.

b) Given the nature of the field, there are no standard settings like a standard administrator username and password which can lead to drastic consequences and have to be taken care of. All configurations are custom for the specific purpose.

c) No remote administration is possible. This complete separation makes the system secure from remote attacks. For local operations which are critical like for example erasing or rewriting program memory, proper security access is required. The security level is proportional to the severity of the action which it enables.

4.9.3 Limitations Mapping

This section investigates whether there are any limitations like redundant services, default credentials, and known vulnerabilities.

a) No unnecessary services were discovered. The whole environment is extremely static and no unnecessary subsystems exist.

b) Due to the nature of the environment, there are no default credentials, or more precisely no place where to use them.

c) The system has been checked for various vulnerabilities. The fact that there is no encryption present enables spoofing on various levels. Implementation vulnerabilities have not been observed. The TCP/IP stack is still not fully implemented. It is kept to bare minimum removing all elements which pose security risks such as source routing, ICMP misuse, and alike.

4.10 Property Validation

This section goes into the legal and ethical aspects of the tests. As mentioned previously all the hardware and software tested is produced, developed and owned by Vector Informatik GmbH. Due to this reason, the tests within this section are omitted.

4.11 Segregation Review

The segregation review investigates whether private personal information is separated from business related information. Due to the fact that this is a test environment and there is no personal information involved, this section can be omitted.

4.12 Exposure Verification

In order to explain exposure verification let us assume there is an international company with offices in multiple locations. There is a known exposure in one location. The aim of these tests is to verify whether that same exposure or any other information from one location can lead to compromising the other locations. Now, this example does not exactly translate to a car because a car will not be differently configured as infrastructures in different locations might be. Instead, each and every car in a car series will have the same software version, the same hardware and so on; it is safe to assume that if one car has an exposure every other car from that series with the same equipment will have it as well. Furthermore, if it is something inside the firmware which is used in multiple series and has not been spotted for long it might even have consequences for multiple series. For this reason, it is even more important to make sure there are no leaks or exposures at all.

4.13 Competitive Intelligence Scouting

Since this section is mainly related to the business side of companies it will be omitted in this mainly technical research.

4.14 Quarantine Verification

No quarantine method has been observed so far other than the ECU rejecting all connection requests prior to a reset after two dropped connections.

4.15 Privileges Audit

In the privileges, audit operations which depend on authentication are tested with provided credentials or in this case key generators for the seed-key authentication.

4.15.1 Identification

No identification has been observed. The flashing process only requires authorization, which serves as identification at the same time. Requests from all IP source addresses and source ports are processed.

4.15.2 Authorization

This set of tests investigates whether it is possible to gain authorization and privileges through fraudulent ways.

a) The only means of authorization which has been found and investigated so far was the one prior to flashing new firmware. As already mentioned it is in most cases based on seed-key authorization. The absence of encryption makes TCP session hijacking possible for gaining authorization.

b) Default user accounts have not been found, nor is there any place where they could be used.

c) Session hijacking has been performed. Standard session hijacking tools could not be used due to the fact that they require known source and destination ports. As specified by ISO13400, vFlash uses a different port on every connection. This has been circumvented by sniffing all traffic, and developing a script which does not rely on the port, but rather on the contents of the packets, and adjusts the port accordingly. It should be noted that if the firmware is signed, hijacking the session and flashing custom code would have no effect since the ECU would reject any code without a correct signature. Details about the test can be found in TCP Hijacking.

4.15.3 Escalation

This set of tests investigates whether it is possible to escalate privileges by any means.

a) As of now no trusted persons/devices have been identified. This has still to be investigated when taking into consideration other nodes which communicate over Ethernet like cameras and the head unit. This does not fall under the thesis research since Vector Informatik GmbH is not working directly with those devices and none of them were available for testing.

b) Assuming there are trusted entities, this test would verify their privileges and investigate if there are means to escalate the privileges they have. As of now, this has not been done since no trusted devices have been identified, and privilege escalation as known in conventional operating systems is not possible since there are no privileges.

4.16 Survivability Validation

This section investigates how resistant the target system is to denial of service attacks and high load in general. These tests when performed on live systems have to be done with extreme caution since they may cause failures of the system. In the case of the test bed in this research, there are no restrictions which have to be considered.

4.16.1 Resilience

These tests check how resilient the system is, e.g. the tests investigate whether there are any choke points or how the system behaves when some parts of it are not functioning.

a) For Ethernet, generally routers and switches are single points of failure because all communication is done one to one, thus routers and switches must ensure that the data goes from point a to b. For the time being routing is still under development, and for switching, hardware switches are used.

b) This test investigates the impact of a system or service failure on accessing the ECU. It was observed that if two consecutive DoIP flashing sessions are dropped all following connection attempts are blocked until a complete ECU restart. Other attempts at bringing a service to failure by sending too long fragmented packets and fuzzed packets with corrupt data were not successful.

c) When it comes to gaining privileges from failure-induced access, it has not been observed that any privileges are gained by overloading the system with performing denial of service attacks, or by sending malformed, oversized, or elsehow corrupted packets. A timed TCP hijack attack on the other hand if counted did result in gaining privileges the actual programming device had. Those privileges included erasing the memory, and uploading new code onto the ECU.

d) A request for dropping the connection has been sent by the programmer (vFlash) immediately after it has noticed that the TCP connection got desynchronized. The ECU naturally ignores that request since it is in sync with the host that hijacked the connection. After some time the programmer sends an ARP request asking again for the ECU´s IP address. There the ECU

notices that something is happening and closes the connection. This safety measure can be taken out if forged packets are sent the programmer so it does not notice the connection being hijacked, or ARP cache poisoning has been performed and all packets are routed and can be filtered from the attacker.

4.16.2 Continuity

The continuity section investigates whether there are backup-systems in place and if they react as expected. Furthermore, they check if lock-out schemes affect only intruders or valid users as well.

a) Backup systems or alternate channels have not been identified.

b) Not many intruder lock-out schemes have been observed. It is quite hard to get to the right wires for performing an attack anyways since it is not a bus system in which anywhere on the bus would suffice. It has been observed though that the refusal of connections after two dropped connections to the DoIP 13400 port is blocking every possible connection. This is because the standard specifies a random source port, and there is no way of knowing which requests are legit and which are not. On the other hand, the fact that a simple ECU reset solves this problem leads to the conclusion that for a valid user it is easy to perform a reset of the ECU, while for an intruder it might not be that straightforward.

4.16.3 Safety

Dropped connections have been observed, but no complete shutdown of the Ethernet network or the ECU occurred. No gatekeepers in charge of disabling the whole network have been identified.

4.17 Alert and Log Review

This section analyses the alarms and logs which were triggered by attacks.

4.17.1 Alarm

The program vFlash issues warnings via its GUI. Optionally this program keeps logs in a user-defined location. These logs are not enabled by default. Their main purpose is for the user to be able to track where something went wrong in case it did.

4.17.2 Storage and Retrieval

This section is investigating the quality and robustness of the logs.

a) It is possible to circumvent the alarms with the help of man in the middle attacks.

b) The fact that the logs are optional degrades their value. It is unknown whether the ECU keeps internal logs.

5 Testing and attacks

This chapter describes the details of the testing and attacks that were performed in order to perform the security audit described in the previous chapter. Much of the information within the security audit did not require special testing. Some of the tests were skipped because the results were clear from the configuration of the system. Whatever was not clear from the configuration itself had to be tested. In many instances, multiple results were attained with one test. One such example is the encountered firewall filtering while performing the network quality tests.

All of the following tests have been documented in the same format. They provide an intro giving more details about the phenomenon under test. Then they continue with defining the aim of the test and pointing out the benefits of having the test done. The scope of the test is defined next followed by the process in which the steps which the test involves are listed. Finally, the results are presented and commented in the results section.

5.1 Network Quality Tests

Network quality tests are done to benchmark the network, i.e., the network behavior observed in network quality tests is taken as a reference point. Thanks to this reference point degraded network performance can be identified.

Aim of the test: Test the channel and have a reference point for future interactions with the system.

Scope: The test bed is as depicted in Figure 3-4. Only the Kali virtual machine is used for these tests. Packets are sent back and forth between the virtual machine and the ECU.

Process: Requests in different protocols are sent to the ECU. The response time and packet loss rate are measured. The tested protocols are ICMP (echo requests), TCP (initialization of a TCP connection with sending a packet with the SYN flag set), and UDP (used in DoIP vehicle identification request messages). All packets are sent one hundred times and the minimum, average, and maximum response times are documented.

Results:

The minimum, average, and maximum response times are shown in Table 5-1.

Table 5-1: Network Quality Test Results

	Response time (ms)			
Protocol	**Minimum**	**Average**	**Maximum**	**Packet loss**
ICMP	0.490	1.335	2.108	0%
TCP	0.898	3.363	6.374	0%
UDP	495.003	497.918	500.508	0%

For ICMP tests echo requests were sent, and the corresponding echo responses collected. The time between the moment a request is sent and the moment the corresponding response is received is measured as the response time.

In the case of the TCP tests, a TCP segment with a SYN flag was sent, and the corresponding TCP segment with the SYN and ACK flags set was captured. The time between the outgoing and incoming segment has been taken as the response time.

The measurement method has been slightly different for the case of UDP. Since UDP is a connectionless protocol, and there is no such thing as an UDP ping, alternative means had to be employed. When a datagram is sent to an open UDP port it simply consumes it and no response is sent out if none is requested. On the other hand, when a closed port receives a UDP datagram the RFC 792 states that: *"…the destination host may send a destination unreachable message to the source host."* [71]. However, since a destination unreachable is a type 3 ICMP message which is not implemented in the Vector stack, it will not be sent. This means that without sending a legitimate request to a known open port there is no way to do any measurements. Therefore, a UDP based DoIP packet was crafted and sent to the ECU. The request which was sent is a Vehicle Identification Request Message as specified by ISO13400 [74]. Since the request needs some processing time on the ECU the response time is much longer than in the case of the two other protocols.

5.2 Ethernet Frame Padding

Ethernet frames (see Figure 2-4) have to have a minimum length of 64 Bytes. Out of those 64 Bytes in an Ethernet II formatted frame, 22 Bytes belong to data with static lengths – such as the source and destination addresses, VLAN tag, and cyclic redundancy check. The preamble and start of frame delimiter are not counted in the 64 Bytes because they are handled on the physical layer and always same. The remaining 44 Bytes have to be either the payload or padding if the payload is too short.

RFC 894 [76] defines that the padding should consist of octets (bytes) of zeros. However, it has been observed that many manufacturers are not padding as the RFC requires, but rather use portions of memory, at times kernel memory for padding frames. In a faulty implementation an attacker can simply send multiple ICMP echo requests and in response will get potentially valuable memory information.

Aim of the test: Ensure that no sensitive data is shared through the padding of Ethernet frames.

Scope: The test bed was as depicted in Figure 3-4.

Process: ICMP echo requests are sent from the virtual Kali machine and the actual Windows 7 OS machine. Furthermore, padding of other packets is inspected.

Results: It has been observed that no sensitive data is leaked through the padding. The response packets to echo requests are implemented according to RFC 792 [71] meaning that the response will have the same payload as the request. Requests from Windows and Kali with their respective responses from the ECU can be seen in Figure 5-1. It was not possible to generate shorter ICMP messages from Windows or from Linux because the network interface card automatically adds padding. The author was unable to find a way to disable the automatic padding. Because of that, responses to TCP and UDP requests have been investigated as well. In those responses, the ECU had to add padding because the data was too short. That padding was found to be in accordance with RFC 894 [76], hence no vulnerability is present.

Kali Request			ECU Response		
[-] ICMPv4			[-] ICMPv4		
| Type:	Echo Request (PING)	08	| Type:	Echo Reply (PONG)	00
| Code:	0	00	| Code:	0	00
| Chcksum:	1908	0774	| Chcksum:	3956	0F74
| Id:	1814	0776	| Id:	1814	0716
| Seq Num:	2	0002	| Seq Num:	2	0002
[-] Payload			[-] Payload		
| Length	56 bytes		| Length	56 bytes	
DD 98 9A 59 00 00 00 00 A8 AE 0A 00 00 00 00 00 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37			DD 98 9A 59 00 00 00 00 A8 AE 0A 00 00 00 00 00 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37		
| ASCII			| ASCII		
Ý˜šY....¨®.....!"#$%&'()*+,- ./01234567			Ý˜šY....¨®.....!"#$%&'()*+,- ./01234567		

Windows Request			ECU Response		
[-] ICMPv4			[-] ICMPv4		
| Type:	Echo Request (PING)	08	| Type:	Echo Reply (PONG)	00
| Code:	0	00	| Code:	0	00
| Chcksum:	18747	493B	| Chcksum:	20795	543B
| Id:	1	0001	| Id:	1	0001
| Seq Num:	1056	0420	| Seq Num:	1056	0420
[-] Payload			[-] Payload		
| Length	32 bytes		| Length	56 bytes	
61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76 77 61 62 63 64 65 66 67 68 69			61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76 77 61 62 63 64 65 66 67 68 69		
| ASCII			| ASCII		
abcdefghijklmnopqrstuvwabcdefghi			abcdefghijklmnopqrstuvwabcdefghi		

Figure 5-1: ICMP Echo Requests and Responses

5.3 Denial of service (DoS) Attacks

The aim of DoS and distributed DoS attacks is to overload the communication channels with irrelevant data so that valid requests cannot be processed. There are multiple variants of such attacks and it is quite hard to protect against them. Normal DoS attacks can be prevented with filters and firewalls. The IP address from which the attack is coming can be blocked completely, or the communication coming from it can be filtered. When spoofing is done in combination with a DoS attack then there is the danger of blocking out a legitimate network host. A mechanism ensuring non-repudiation would help in filtering forged from legitimate communication.

Aim of the test: Hinder firmware flashing process with a TCP DoS attack.

Scope: The test bed was as depicted in Figure 3-4. A running firmware flashing session was taking place between the ECU and vFlash running on the PC. The virtual Kali machine was performing the attack.

Process: The flashing process is done via DoIP. Since large files are sent the DoIP communication runs over TCP. Tcpkill, part of dsniff, is a program built for network testing. It sniffs ongoing TCP connections on selected hosts and interrupts them by inserting RST flagged packets into the communication channel. This leads to a dropped connection.

Result: Figure 5-2 shows what a normal firmware flashing process with vFlash looks like. In the background, the Wireshark capture of TCP communication between vFlash and the ECU can be seen.

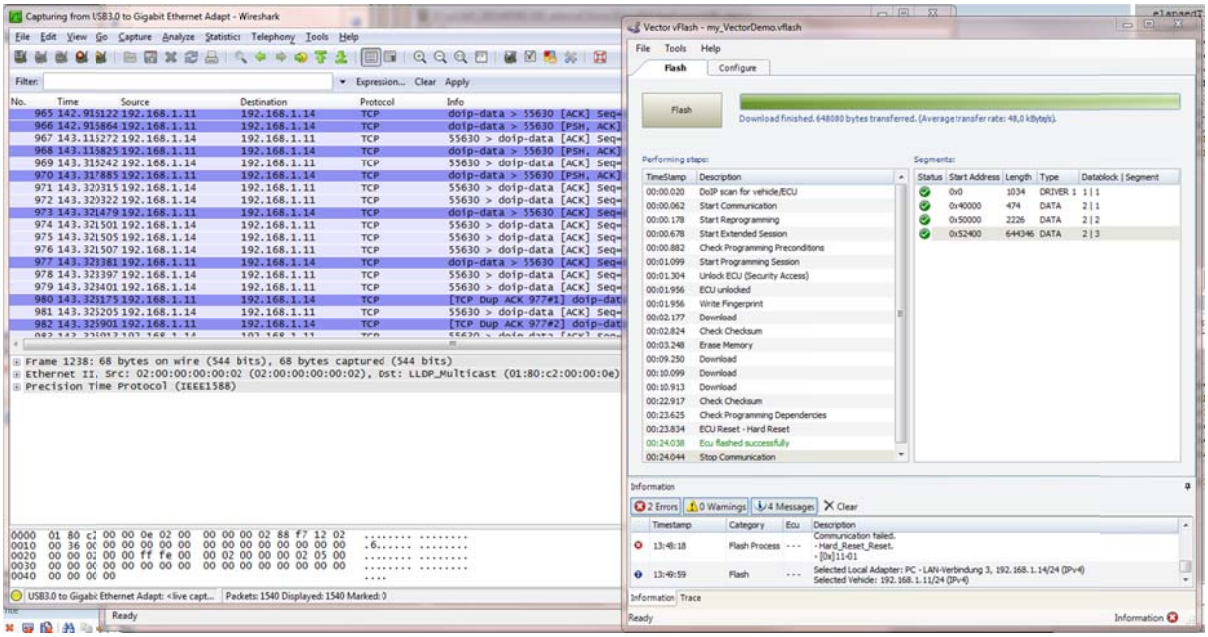

Figure 5-2: Normal vFlash Flashing Process

Figure 5-3 shows what happens if tcpkill is run before the flashing process is started. An error message saying "*Communication failed*" pops up and RST flagged TCP segments can be seen in the Wireshark window. This proves one of the vulnerabilities of the TCP protocol itself. A way to protect against this type of attack is to use encryption, hence the sequence numbers would not be in clear text and thereby harder to decipher. Another way would be cryptographic signatures which would ensure the non-repudiation of the source. Both of these could be realized by sending the TCP traffic through a virtual private network or by using IPsec.

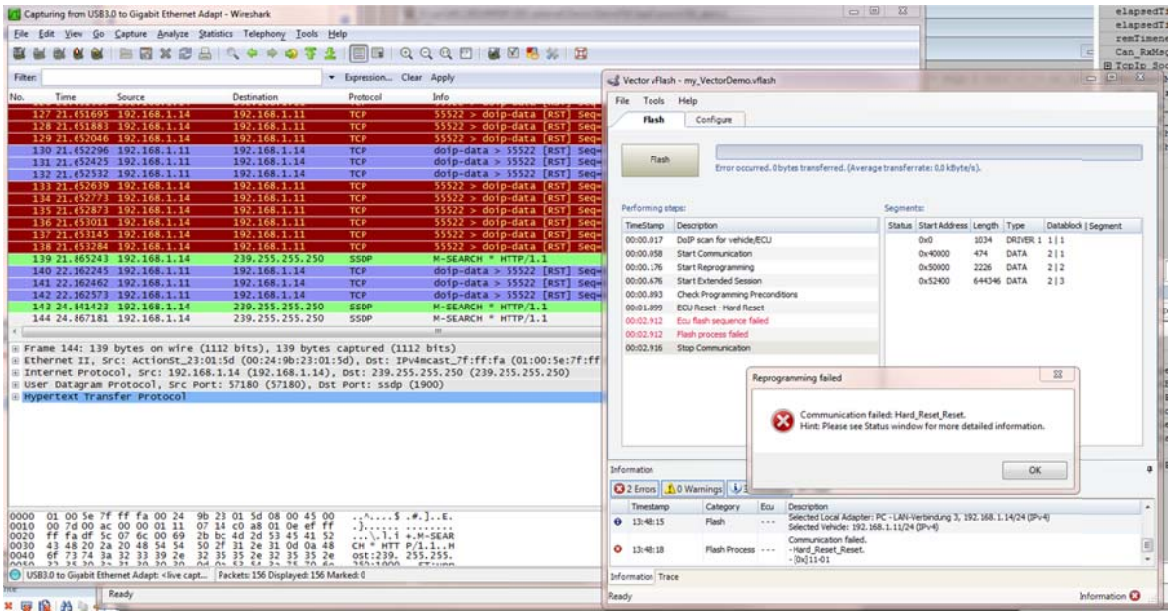

Figure 5-3: Flashing Process Interrupted by DoS Attack

5.4 SOME/IP Spoofing

In spoofing the attacker sends forged packets that the receiving host cannot differentiate from those which it actually expects.

Aim of the test: Prove that SOME/IP spoofing and spoofing, in general, is possible.

Scope: The test bed was as depicted in Figure 3-4. Application software with ongoing SOME/IP communication is running on the ECU.

Process: The ECU is exchanging SOME/IP data with the PC. The data is graphically represented as a continuous chainsaw-like function of multiple triangles (as depicted in Figure 5-4). A script which sends specially crafted packets with different data is run on Kali.

Result: As soon as the script is run the lines in the graph become malformed as shown in Figure 5-5. That proves that spoofing is possible. In the given example the consequences are simply a malformed graph. However, the same thing could be done with a camera live stream which is fed to the adaptive cruise control system which is in charge of maintaining a given the distance from the car in front. The consequences of such an attack are frightening. SOME/IP is still a relatively new protocol which is being worked on. There are some security options under considerations like Secure Onboard Communication (SecOP) and (TLS) [77]. Another solution would be the use of the Secure Real-Time Transport Protocol (SRTP) [78] which provides encryption, integrity and message authentication in combination with Multimedia Internet KEYing (MIKEY) [79] which is a key management solution that provides session keys for SRTP.

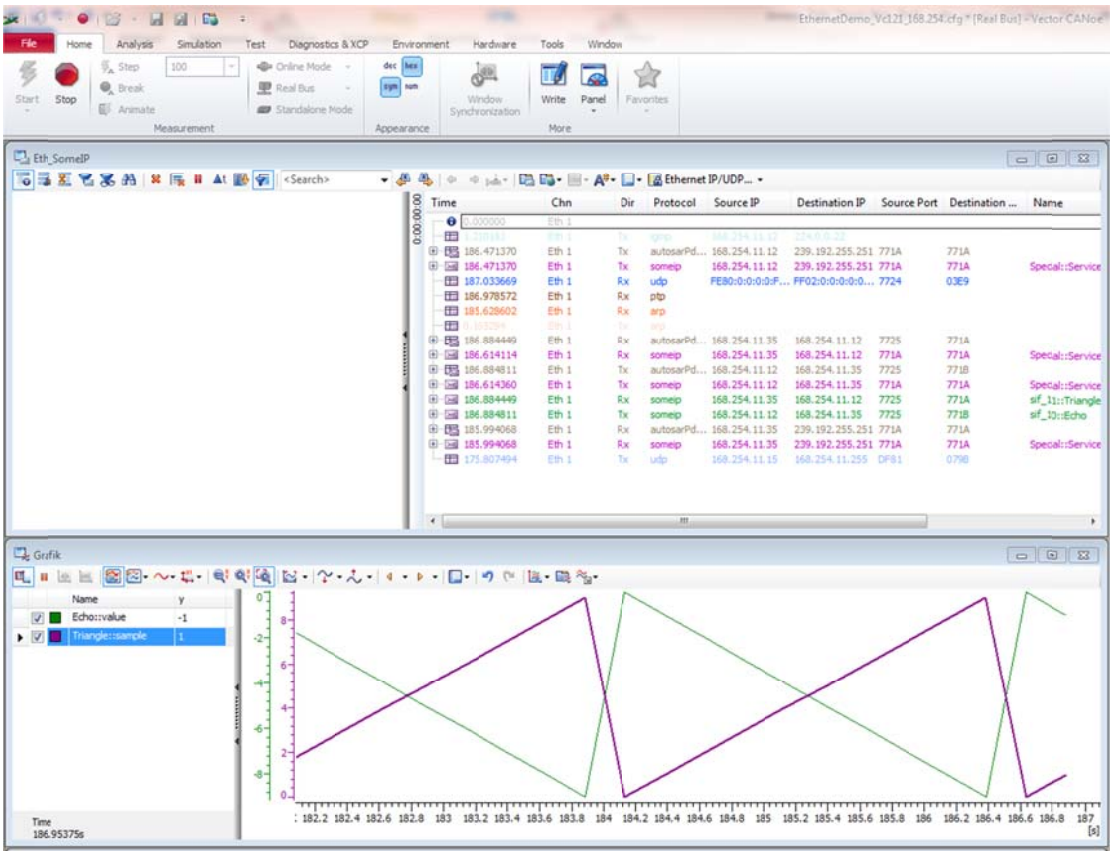

Figure 5-4: SOME/IP data presented graphically

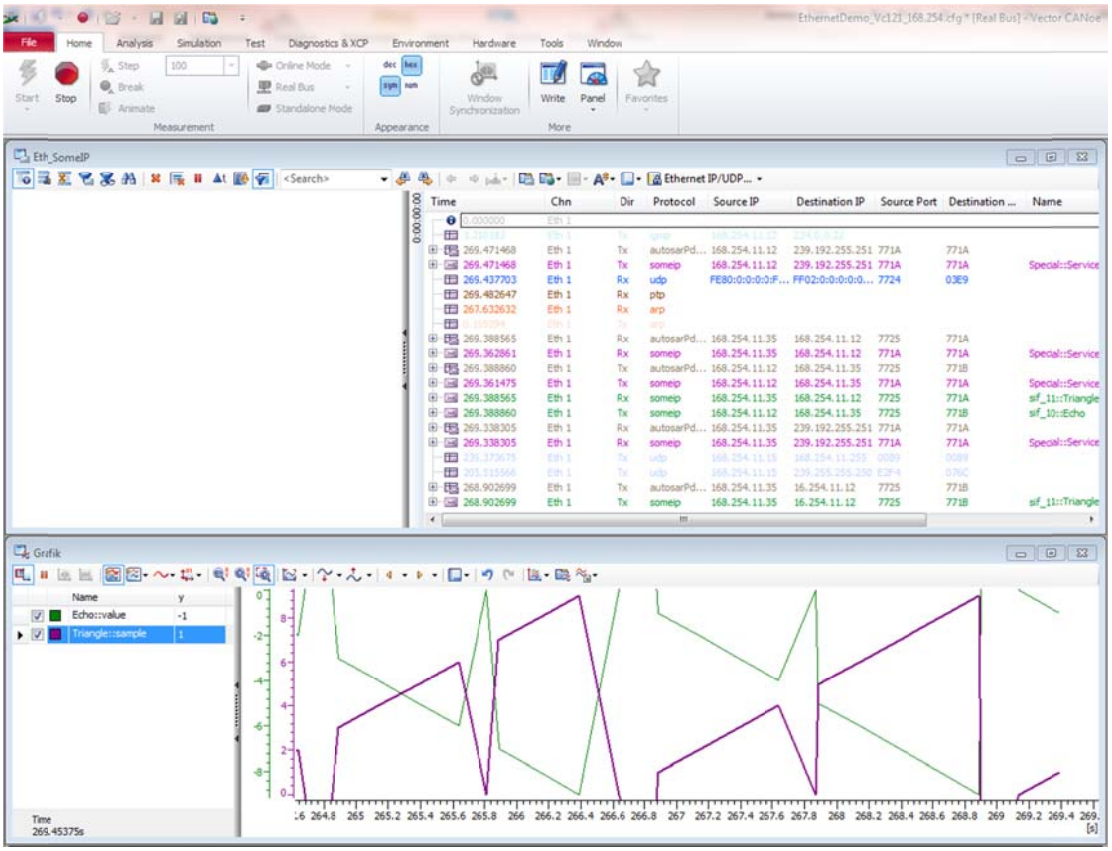

Figure 5-5: Spoofed SOME/IP packets presented graphically

5.5 ARP Cache Poisoning

The concept of the ARP protocol was briefly explained in Section 2.1.2.2.2.4. In ARP cache poisoning forged ARP reply or requests are sent out by the attacker in order to compromise (poison) the ARP cache of the host under attack. Thereby a legit IP address is used so that the host under attack associates the attackers MAC address to the legit IP address. After this has happened the attacker is getting all the data which was meant for the legit host. It can filter the data and forward some to the legit host, or take control over the whole communication without forwarding anything to the legit host. ARP cache poisoning is usually done for most man in the middle attacks. It can even be useful for the TCP Hijacking attack which is demonstrated in 5.6

Aim of the test: Demonstrate ARP cache poisoning and prove that it is possible to do with non-static ARP tables.

Scope: The test bed as depicted in Figure 3-4. The sample application software is running on the ECU. An Ethernet packet builder has been added to the setup in order to generate and send the forged packets.

Process: An ECHO Request packet and two ARP request/reply packets with the corresponding MAC addresses AA:AA:AA:AA:AA:AA and CC:CC:CC:CC:CC:CC have been crafted. The packet exchange happened as follows:

1. The ECHO request packet from 168.254.11.17 (random IP address) was sent to 168.254.11.35 (ECU)
2. The ECHO request packet is directly followed by an ARP reply with a crafted AA:AA:AA:AA:AA:AA MAC address (Shown in Figure 5-6)
3. After the ARP reply is received by the ECU and its cache is updated, an ECHO response comes out as expected to AA:AA:AA:AA:AA:AA (Shown in Figure 5-7)
4. Another ARP reply is sent, only this time with CC:CC:CC:CC:CC:CC being the MAC address (Shown in Figure 5-8)
5. Again an ECHO request follows
6. This time the ECU sends the ECHO response to CC:CC:CC:CC:CC:CC (Shown in Figure 5-9)

Result: The fact that the ECU replied to the same ECHO request to two different MAC addresses after receiving the corresponding ARP replies shows that ARP cache poisoning is possible. This can be easily prevented by enabling the “Static ARP tables” option within DaVinci Configurator, but then all devices in the network need to be entered into the ARP table, and it will not be possible to add devices on the fly while the system is operating. Every device addition would require flashing the ECU firmware. Nevertheless, in a static environment as the car is this is not such a bad option given all the benefits it carries with it. Many man in the middle attacks would be harder or even impossible to do if ARP cache cannot be poisoned.

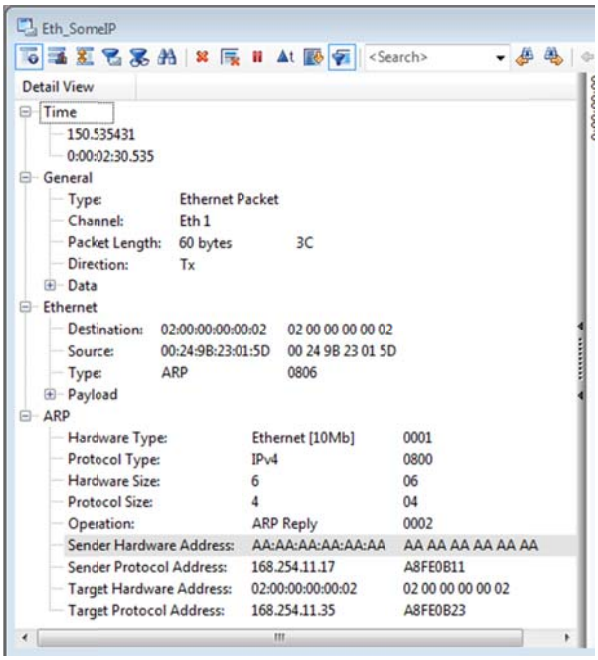

Figure 5-6: ARP Reply from Step 2.

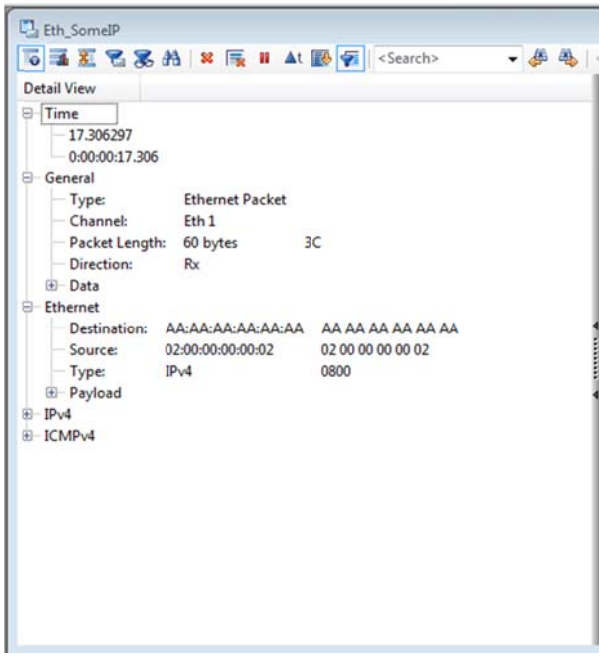

Figure 5-7: ECHO Response from Step 3.

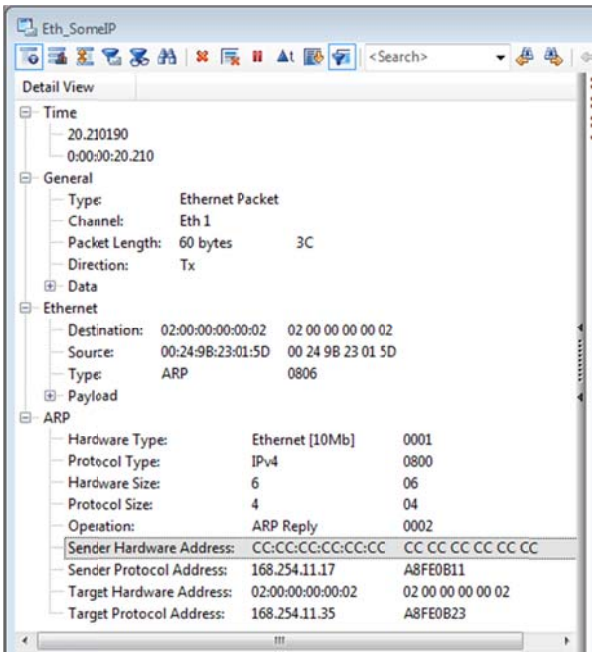

Figure 5-8: ARP Reply from Step 4.

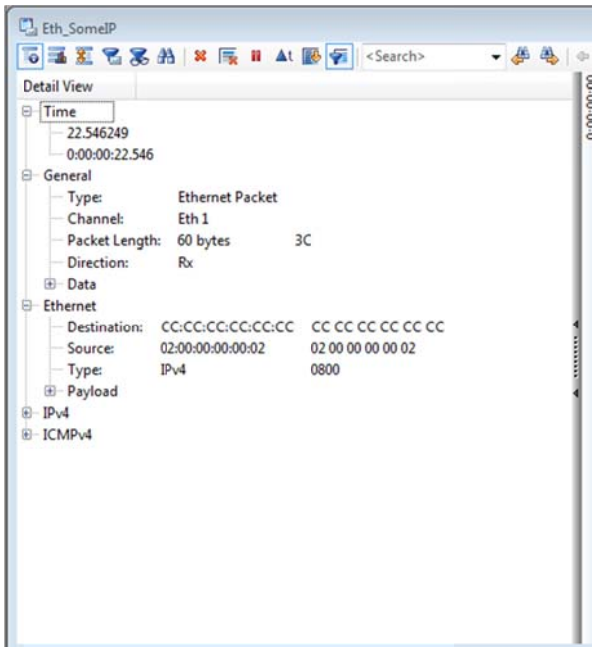

Figure 5-9: ECHO Response from Step 6.

5.6 TCP Hijacking

TCP hijacking is based on the fact that TCP is a connection-oriented protocol. The connection is based on the sequence numbers which are randomly determined during the handshake (see Section 2.1.2.2.2.7). They get increased with by the length of every new packet which is sent. Packets with the wrong sequence numbers are dropped.

During the hijacking process, the attacker sniffs the ongoing communication session and finds out the sequence numbers. Then he inserts a carefully crafted packet (with spoofed information) into the communication session and thereby desynchronizes the ongoing communication, kicks out one legitimate host and continues communicating in its place. From this point on the packets coming from the legitimate host will be dropped as TCP retransmissions because of the outdated sequence numbers.

Aim of the test: Hijack a DoIP firmware flashing connection

Scope: The test bed as depicted in Figure 3-4. The PC is running vFlash for the flashing process, and the Kali virtual machine runs a Python script which sniffs packets and desynchronizes the connection when needed.

Process: The ECU is connected to the PC. The hijacking script is run on the Kali virtual machine. vFlash is started and a flashing session is begun. The script is sniffing the communication between vFlash and the ECU. Once the seed key exchange has happened the script jumps in and sends its own packet to desynchronize the ongoing connection from vFlash and is able to send its own payload instead.

Figure 5-10 shows the error message "*Security access failed…*" which vFlash generates after the session has been desynchronized.

Figure 5-11 is a screenshot of the communication captured by Wireshark. Important packets are marked as follows:

1. The injected TCP packet with spoofed sequence number for hijacking
2. vFlash´s attempt to send a legit packet which is recognized as TCP Retransmission because of the repeated sequence number
3. Contents of the crafted packet with payload which says "Hijacked"

Result: The hijacking process was successful. Conventional hijacking tools could not be used due to the fact that they require both, source and destination ports to be defined. vFlash, in accordance with ISO13400, does not start every session on the same source port. Due to that, a python script was written using scapy to sniff all packets and identify the packet which successful authentication after the seed key exchange. The session was desynchronized. vFlash immediately reports an error and sends a packet for closing the connection. This can be prevented in multiple ways. ARP cache can be poisoned and with a man in the middle attack, those packets can be filtered. Another option would be to forge packets to vFlash so it does not notice

that the connection has been desynchronized. Alternatively, it can be just turned off, but this would require precise timing.

This attack is possible if the attacker is able to sniff the whole unencrypted communication. In the case of this testbed, a great factor was the fact that practically all communication is visible to anyone. If that were not the case it would have been a lot harder to guess the sequence numbers. First implementations of the TCP protocol used to increase the initial sequence numbers after a time period. This of course was very weak since one could easily make a connection to the host, see which sequence number he would get, and by trying out numbers around that one he would have a high probability of hijacking another ongoing session. This is not the case nowadays and in this implementation where the numbers seem very random.

If the communication were encrypted it would have been a lot harder, if not impossible, to get the sequence numbers and perform hijacking.

It has to be noted that in case the firmware has a cryptographic signature it would still not be possible to install a custom firmware. With the signature and a public key the ECU is able to check the authenticity of the data, and whether it came from the right source.

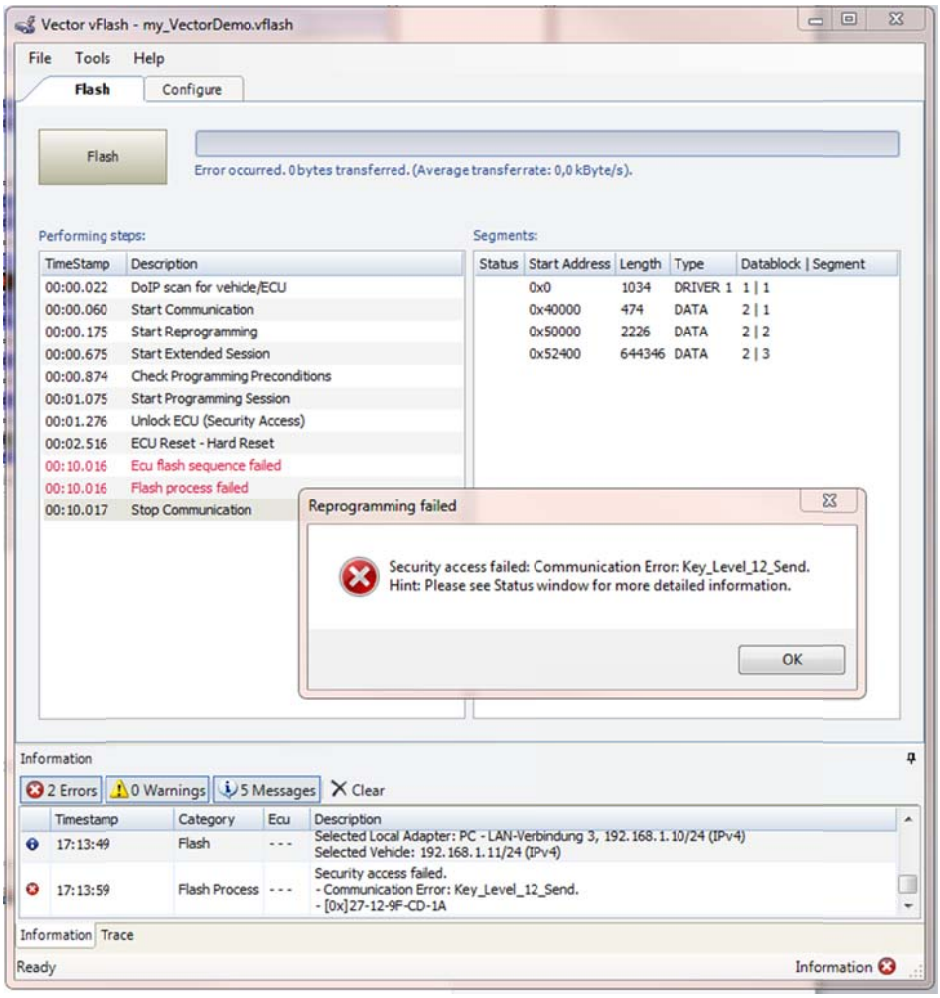

Figure 5-10: vFlash Security Access Failed

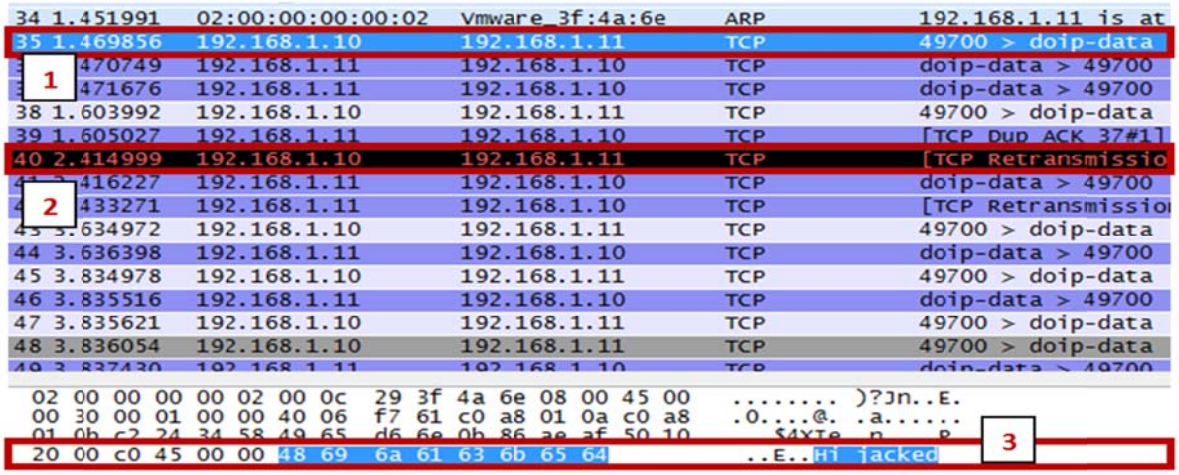

Figure 5-11: TCP Hijacking Wireshark

6 Analysis

The analysis consists of the Major Results (in Section 6.1) and the Discussion (in Section 6.2). It is possible to assess how secure the system is by calculating RAV scores for every system described in Section 3.7. Additionally, one can calculate how different variations and additions will affect this RAV score.

After the RAV scores are calculated and countermeasures for limitations and porosities are proposed these results will be discussed.

6.1 Major Results

Thanks to the fact that RAV is a scalar metric the different systems and configurations can be compared quite easily. In Table 6-1, the different systems are described. Table 6-2, shows impacts of single configuration changes of the existing Vector stack. It is kept separately because of ease of view and comparison of the configuration effects. The codes used for representing operations, controls, and limitations within those tables can be found in Table 6-3, Table 6-4, and Table 6-5 respectively.

Table 6-1: Actual Security Comparison of Different Stacks and Configurations

	Protocols by Definition	Vector Stack Base Configuration	Vector Stack Combined Improvements	Proposed Additions to Vector Stack
Visibility	V	V	V	0
Access	AC1,AC2,AC3,AC4	AC1,AC3,AC4	AC3,AC4	AC4
Trust	T1,T2	T1,T2	T1	0
Authentication	A1	A1	A1	A1
Indemnification	0	0	0	0
Resilience	R1	R1	R1	R1
Subjugation	0	0	0	0
Continuity	0	0	0	0
Non-Repudiation	0	0	NR1	NR1
Confidentiality	0	0	0	CF1
Privacy	0	0	0	0
Integrity	0	0	I1	I1
Alarm	A1,A2	A1,A2	A1,A2	A1,A2
Vulnerabilities	VW1	VW1	VW1	0
Weaknesses	W1, VW1	W1, VW1	W1, VW1	W1
Concerns	0	0	0	0
Exposures	E1;E2;E3;E4;E5	E1;E2;E3	E2;E3	E3
Anomalies	0	0	0	0
Actual Security	82,4038 RAVs	83,7185 RAVs	85,2647 RAVs	91,8459 RAVs

Table 6-2: Actual Security Comparison of Different Vector Stack Configurations

	Vector Stack Base Configuration	ICMP Disabled in Vector Stack	Static ARP Cache Enabled in Vector Stack	Firmware Signatures & Hashes
Visibility	V	V	V	V
Access	AC1,AC3,AC4	AC3,AC4	AC1,AC3,AC4	AC1,AC3,AC4
Trust	T1,T2	T1,T2	T1	T1,T2
Authentication	A1	A1	A1	A1
Indemnification	0	0	0	0
Resilience	R1	R1	R1	R1
Subjugation	0	0	0	0
Continuity	0	0	0	0
Non-Repudiation	0	0	0	NR1
Confidentiality	0	0	0	0
Privacy	0	0	0	0
Integrity	0	0	0	I1
Alarm	A1,A2	A1,A2	A1,A2	A1,A2
Vulnerabilities	VW1	VW1	VW1	VW1
Weaknesses	W1, VW1	W1, VW1	W1, VW1	W1, VW1
Concerns	0	0	0	0
Exposures	E1;E2;E3	E2;E3	E1;E2;E3	E1;E2;E3
Anomalies	0	0	0	0
Actual Security	83,7185 RAVs	84,2657 RAVs	84,0709 RAVs	84,2372 RAVs

Table 6-3: Mapping of Operations to Codes

Type of Operation	Code	Explanation
Visibility	V	There is only one host so it is either visible or not
Access	AC1	ICMP probe response
Access	AC2	UDP probe response
Access	AC3	TCP probe response
Access	AC4	DoIP probe response
Trust	T1	Spoofing possible
Trust	T2	ARP cache poisoning possible

Table 6-4: Mapping of Controls to Codes

Type of Control	Code	Explanation
Authentication	A1	Seed-Key Authentication
Resilience	R1	Rejecting connections after two dropped ones
Non-Repudiation	NR1	Signatures
Confidentiality, Integrity	CF1	Encryption
Integrity	I1	Hashes
Alarm	A!1	Notifications from vFlash
Alarm	A!2	Logs in vFlash

Table 6-5: Mapping of Limitations to Codes

Type of Liability	Code	Explanation
Exposure	E1	Responses to ICMP Echo requests
	E2	Responses to TCP SYN probes
	E3	Responses to DoIP requests from any port and IP
	E4	Responses to probes on closed UDP ports
	E5	VLAN hopping with double tagging
Vulnerability, Weakness	VW1	TCP hijacking
Weakness	W1	DoS attacks

Countermeasures to porosities and limitations are listed in Table 6-6. Some are discussed in more detail the tests in Chapter 5. All the countermeasures can be grouped into the following three categories:

1. Making use of existing configuration settings (e.g. Static ARP tables, and disabled ICMP echo responses)
2. Doing further development of the stack implementation and adding features like encryption and signatures with hashes to all communication
3. Changing the existing stack implementation so that it does not conform to protocol specifications for the greater good. An example here is sending RST flagged TCP packets when SYN flagged packets are received on closed TCP ports. This would not have great consequences on the functionality of the system but would remove one exposure.

Table 6-6: Countermeasures

Code	Countermeasure
E1	Turn off ICMP responses in the configuration
E2	Change implementation so that no RST flagged packets are sent when probing closed TCP ports
E3	Change implementation so that it does not respond to any IP address which sends DoIP commands
VW1	Implement encryption so that sequence numbers cannot be sniffed
W1	Add filters and firewalls to secure against DoS attacks
T1	MIKEY and SRTP
T2	Turn on static ARP cache

6.2 Discussion

As can be seen from the previous section, the actual security can be increased by 1.5 RAVs by just changing configurations of the existing stack implementation *without* doing any further development. With further development, an increase of 8 RAV would be possible. There are two things which need to be kept in mind:

> Firstly, the RAV scores are seemingly low because of the fact that certain controls have not been investigated at all as they are not relevant for this research.
>
> Secondly, some security measures such as encryption will need additional resources and might degrade system performance. Therefore, there is a tradeoff between security and performance which has to be evaluated on a case by case basis in a given scenario.

In conclusion, it can be said that initial goals of assessing the security and investigating the impact of proposed controls to encountered limitations (as defined by goals 1 through 5 in Section 1.4 on page 3.) have been met.

7 Conclusions and Future Work

In this chapter, the conclusions and suggestions for future work will be presented. The chapter also discusses some of the limitations of this work and offers some reflection concerning ethical issues and sustainability.

7.1 Conclusions

Ethernet-based communication has come a long way from when it was first developed to today. In its early days security was not one of its highest priorities, but over time many security measures have been added, secure protocols developed, and inherent vulnerabilities fixed.

While in conventional computer networks the main sources of insecurity are insecure services on open ports, a car will not have a single more open port than is needed, and even then these ports will be secured. In most cases all network devices are known to the endpoints, hence they can be statically defined. Another feature that increases the security of the system thereby raises a hurdle for an attacker, is fact that there is no conventional OS running inside a car. This means that all existing exploits which attack known vulnerable services and escalate privileges are inapplicable. The network stack implementation in the tested device is very solid and no implementation faults which could be exploited were identified.

The TCP/IP stack implementation has been carefully implemented. For example, protocols (such as ICMP) which have many inherent vulnerabilities have not been fully implemented. This is in part because they are unnecessary in the automotive domain. Additionally, not implementing these functionalities has enabled a higher level of security. Out of the whole ICMP protocol, only ECHO requests and responses have been implemented, and even those can be disabled in the configuration.

The automotive environment is completely static. This fact can be used as an advantage, hence this has been kept in mind during the development of the network stack. The ARP cache can, for example, be configured to be static. In this case, ARP cache poisoning and many man in the middle attacks are impossible.

The cumulative effects of all these omitted protocols, static configurations, and even proposals for additions to the stack have been shown through the calculated RAV scores (for various different systems and configurations). The current stack in its basic setup proved to be more secure than a system with all of the protocols implemented according to their specifications. The use of static configurations means that the implementations deviate slightly from protocol specifications. A few additions to the stack such as encryption and signatures with hashes proved to increase security by roughly 10%. However, since some of these additions require tradeoffs between security and performance, their adoption has to be decided on a case by case basis.

All of the goals defined in Section 1.4 have been met. When it comes to the question of what the author would have done differently if he would do the same

research again, the answer would be to have made use of a specific methodology from an earlier point in time. OSSTMM proved to be great resource.

Unfortunately, no yes/no answer was found to the question: "*Is automotive Ethernet together with its accompanying protocols secure and safe enough for the automotive industry?*" It can be said that automotive Ethernet and its accompanying protocols still have room for improvement. Most protocols are still in the standardization and development process. Security is worked on and kept in mind. Recent studies have made manufacturers aware of the close link between security and safety. The industry is moving in the right direction. There is one important thing to keep in mind with security though, and that is the fact that a chain is only as strong as its weakest link.

7.2 Limitations

All of the research was conducted using hardware and software produced and owned by Vector Informatik GmbH. Therefore, many of the most vulnerable devices were not included in this research. As related work in Section 2.4 shows the most vulnerable entry points are the telematics system, and third party (non-automotive) specific hardware and accompanying software, such as the Bluetooth and Wi-Fi modules.

The initial setup of the testbed took considerably longer than expected, but eventually, a running system with both the flash bootloader and the sample application software was set up and ready for testing.

IPv6 was left out from most of the research since it was not used by the services which were investigated. The flash bootloader had an older version of the network stack, so most tests were done on the application software which had a more recent version of the network stack. For instance, the older version of the network stack did not support packet fragmentation.

Since the field of security is concerned with vulnerabilities and dangers, quite frequently all results are kept secret (for the obvious reasons). As a result, the author was unable to find examples of security audits performed according to OSSTMM which would aid in performing the audit and categorizing the data from it into operations controls and limitations more accurately. Luckily the methodology manual is so detailed that a deeper investigation enabled the author to perform an audit and calculate the RAV scores even without prior examples. Unfortunately, this process required more time than was initially allocated.

7.3 Future work

As mentioned above IPv6 was not tested thoroughly throughout this work because the services implemented in the application did not rely on it. They should be investigated further in future research.

AVB/TSN is another world on top of Ethernet which is new and has to be researched. Thus far only a limited amount of research has been reported in this area concerning security.

As many automotive TCP/IP stacks and other protocols are still under development, testing needs to be done as soon as any feature is added. The first new feature to be examined would be routing – once it is implemented to a greater extent.

Depending on the car manufacturer, there may be infotainment units running Linux systems. This opens these systems to the potential use of known exploits. Furthermore, if the car's network topology is organized in such a way that safety-critical systems can be controlled from the infotainment units, then there it is definitely a need for future security studies.

In the case of electric and hybrid cars, the charging stations are another area for research, as the communication between the charging stations and the car is mostly done over Ethernet.

7.4 Reflections

Cars are part of our daily life. Unfortunately, accidents also happen. According to the Association for Safe International Travel, 3287 people die daily in car crashes. According to the same source, an additional 20-50 million people are injured or become disabled. Globally this accounts for about US$518 billion of economic loss. Recent advances in ADAS are able to lower the accident rates by 28% [80]. That corresponds to an immense number of lives that can be saved or improved. Automotive Ethernet is the base for almost all of these advances. For that reason, it is necessary to ensure that systems that utilize automotive Ethernet are reliable and secure. The objective of this thesis was to do exactly that.

References

[1] Wolfhard Lawrenz, Ed., *CAN System Engineering*. London: Springer London, 2013, ISBN: 978-1-4471-5612-3 [Online]. DOI: 10.1007/978-1-4471-5613-0

[2] U. Abelein, H. Lochner, D. Hahn, and S. Straube, 'Complexity, quality and robustness - the challenges of tomorrow's automotive electronics', in *2012 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2012, pp. 870–871. DOI: 10.1109/DATE.2012.6176573

[3] S. M. Bellovin, 'Security Problems in the TCP/IP Protocol Suite', *SIGCOMM Comput Commun Rev*, vol. 19, no. 2, pp. 32–48, Apr. 1989. DOI: 10.1145/378444.378449

[4] M. W. Eichin and J. A. Rochlis, 'With microscope and tweezers: an analysis of the Internet virus of November 1988', in *Proceedings. 1989 IEEE Symposium on Security and Privacy*, 1989, pp. 326–343. DOI: 10.1109/SECPRI.1989.36307

[5] Free and Open Software Security Community, 'OWASP Website'. [Online]. Available: https://www.owasp.org/index.php/Main_Page. [Accessed: 12-Sep-2017]

[6] Pete Herzog, 'OSSTMM 3 - The Open Source Security Testing Methodology Manual: Contemporary Security Testing and Analysis'. ISECOM - Institute for Security and Open Methodologies, 2010 [Online]. Available: http://www.isecom.org/mirror/OSSTMM.3.pdf

[7] J. Postel, 'Internet Protocol', Internet Request for Comments, vol. RFC 791 (INTERNET STANDARD), Sep. 1981 [Online]. Available: https://www.rfc-editor.org/rfc/rfc791.txt

[8] S. Deering and R. Hinden, 'Internet Protocol, Version 6 (IPv6) Specification', Internet Request for Comments, vol. RFC 2460 (INTERNET STANDARD), Dec. 1998 [Online]. Available: https://www.rfc-editor.org/rfc/rfc2460.txt

[9] William Stallings, 'The Origins of OSI', 1998. [Online]. Available: http://williamstallings.com/Extras/OSI.html. [Accessed: 30-Apr-2017]

[10] Deon Reynders and Edwin Wright, *Practical TCP/IP and ethernet networking for industry*. Amsterdam; Heidelberg [u.a.: Newnes, 2003, ISBN: 978-0-7506-5806-5.

[11] Andrew L. Russell, 'OSI: The Internet That Wasn't - IEEE Spectrum'. IEEE Spectrum, 30-Jul-2013 [Online]. Available: http://spectrum.ieee.org/computing/networks/osi-the-internet-that-wasnt

[12] Google, 'IPv6 – Google', 2009-2017. [Online]. Available: https://www.google.com/intl/en/ipv6/statistics.html. [Accessed: 10-Sep-2017]

[13] Marc Heuse, 'IPv6 Insecurity Revolutions', Kuala Lumpur, 2012 [Online]. Available: https://www.yumpu.com/en/document/view/1924942/d1t2-marc-heuse-ipv6-insecurity-revolutions/1. [Accessed: 06-May-2017]

[14] J. Postel, 'Transmission Control Protocol', Internet Request for Comments, vol. RFC 793 (INTERNET STANDARD), Sep. 1981 [Online]. Available: https://www.rfc-editor.org/rfc/rfc793.txt

[15] Margaret Rouse, 'UDP (User Datagram Protocol)', *TechTarget*, 2014. [Online]. Available:

http://searchmicroservices.techtarget.com/definition/UDP-User-Datagram-Protocol. [Accessed: 06-May-2017]

[16] 'Karl Benz', *Biography.com*, Apr-2014. [Online]. Available: https://www.biography.com/people/karl-benz-9208256. [Accessed: 10-Sep-2017]

[17] ScienCentral Inc., 'An Outline of the History of the Transistor', *PBS: Public Broadcast Service*, 1999. [Online]. Available: http://www.pbs.org/transistor/album1/. [Accessed: 30-Apr-2017]

[18] ScienCentral Inc., 'Integrated Circuit', *PBS: Public Broadcast Service*, 1999. [Online]. Available: http://www.pbs.org/transistor/background1/events/icinv.html. [Accessed: 30-Apr-2017]

[19] Lisa Wade, 'A Short History of the Airbag', *Consumer Affairs*, 25-Sep-2006. [Online]. Available: https://www.consumeraffairs.com/news04/2006/airbags/airbags_invented.html. [Accessed: 10-Sep-2017]

[20] W. Ziebart, 'Car electronics-key factors of success for the '90s', in *1991 Eighth International Conference on Automotive Electronics*, London, 1991, pp. 1–6 [Online]. Available: http://ieeexplore.ieee.org/document/152000/?arnumber=152000

[21] Vector Informatik GmbH, 'Introduction to CAN', *Vector E-Learning*, 19-Oct-2016. [Online]. Available: https://elearning.vector.com/vl_can_introduction_en.html. [Accessed: 15-Apr-2017]

[22] Pat Richards, 'AN228 - A CAN Physical Layer Discussion'. Microchip Technologies Inc., 2002 [Online]. Available: http://ww1.microchip.com/downloads/en/AppNotes/00228a.pdf. [Accessed: 06-May-2017]

[23] Steve Corrigan, 'Controller Area Network Physical Layer Requirements'. Texas Instruments, 2008 [Online]. Available: http://www.ti.com/lit/an/slla270/slla270.pdf. [Accessed: 06-May-2017]

[24] International Organization for Standardization, *ISO 11898-1:2015 - Road vehicles -- Controller area network (CAN) -- Part 1: Data link layer and physical signalling*. 2015, p. 65 [Online]. Available: https://www.iso.org/standard/63648.html. [Accessed: 02-Jul-2017]

[25] Microcontroller Division Applications, 'AN1278 - LIN (LOCAL INTERCONNECT NETWORK) SOLUTIONS'. STMicroelectronics, 2002 [Online]. Available: http://www.st.com/content/ccc/resource/technical/document/application_note/10/30/18/b5/90/bc/4c/73/CD00004273.pdf/files/CD00004273.pdf/jcr:content/translations/en.CD00004273.pdf. [Accessed: 06-May-2017]

[26] National Instruments, 'Überblick über den Kommunikationsbus FlexRay für den Automobilbereich', *National Instruments*, 05-Oct-2009. [Online]. Available: http://www.ni.com/white-paper/3352/de/. [Accessed: 06-May-2017]

[27] Markus Schmidt, 'Automotive Bus Systems'. Atmel [Online]. Available: http://eng.umd.edu/~austin/enes489p/project-resources/SchmidAutoBusSystems.pdf. [Accessed: 06-May-2017]

[28] Hans-Werner Schaal, 'IP and Ethernet in Motor Vehicles: Challenges for the development tool, illustrated by today's applications'. Vector Informatik GmbH, Apr-2012 [Online]. Available: http://cn.vector.com/portal/medien/cmc/press/PON/Ethernet_IP_ElektronikAutomotive_201204_PressArticle_EN.pdf

[29] Open Alliance, 'Open Alliance Website'. [Online]. Available: http://www.opensig.org/. [Accessed: 06-May-2017]

[30] Lars Völker, 'Scalable service-Oriented MiddlewarE over IP (SOME/IP)'. [Online]. Available: http://some-ip.com/. [Accessed: 30-Apr-2017]

[31] 'Einführung in Automotive Ethernet'. [Online]. Available: https://elearning.vector.com/index.php?&wbt_ls_seite_id=1536276&root=376493&seite=vl_automotive_ethernet_introduction_de. [Accessed: 30-Apr-2017]

[32] Avnu Alliance, 'Avnu Alliance Website'. [Online]. Available: http://avnu.org/. [Accessed: 06-May-2017]

[33] Vector Informatik GmbH, 'Introduction to Automotive Ethernet', *Vector E-Learning*, 19-Oct-2016. [Online]. Available: https://elearning.vector.com/vl_automotive_ethernet_introduction_en.html. [Accessed: 06-May-2017]

[34] International Organization for Standardization, *ISO 13400-1:2011 - Road vehicles -- Diagnostic communication over Internet Protocol (DoIP) -- Part 1: General information and use case definition*. 2011, p. 15 [Online]. Available: https://www.iso.org/standard/53765.html. [Accessed: 12-Sep-2017]

[35] Association for Standardization of Automation and Measuring Systems, 'ASAM Website'. [Online]. Available: https://www.asam.net/. [Accessed: 06-May-2017]

[36] Vector Group, *XCP fundamentals: measuring, calibrating and bypassing based on the ASAM standard*. [Online]. Available: https://www.youtube.com/watch?v=uEmgbCXij-w. [Accessed: 06-May-2017]

[37] Roderick Currie, 'Developments in Car Hacking'. SANS Institute, 05-Dec-2015 [Online]. Available: https://www.sans.org/reading-room/whitepapers/ICS/developments-car-hacking-36607. [Accessed: 06-May-2017]

[38] Karl Koscher, Alexei Czeskis, Franziska Roesner, Shwetak Patel, Tadayoshi Kohno, Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, and others, 'Experimental security analysis of a modern automobile', in *Security and Privacy (SP), 2010 IEEE Symposium on*, 2010, pp. 447–462 [Online]. Available: http://ieeexplore.ieee.org/abstract/document/5504804/. [Accessed: 06-May-2017]

[39] Antuan Goodwin, 'Ford unveils open-source Sync developer platform', *Roadshow*, 29-Oct-2009. [Online]. Available: https://www.cnet.com/roadshow/news/ford-unveils-open-source-sync-developer-platform/. [Accessed: 06-May-2017]

[40] Steve Mollman, 'From cars to TVs, apps are spreading to the real world', *Cable News Network*, 08-Oct-2009. [Online]. Available: http://edition.cnn.com/2009/TECH/10/08/apps.realworld/. [Accessed: 06-May-2017]

[41] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, Tadayoshi Kohno, and others, 'Comprehensive Experimental Analyses of Automotive Attack Surfaces.', in *USENIX Security Symposium*, San Francisco, 2011 [Online]. Available: http://static.usenix.org/events/sec11/tech/full_papers/Checkoway.pdf. [Accessed: 06-May-2017]

[42] Charlie Miller and Chris Valasek, 'Adventures in Automotive Networks and Control Units', presented at the DefCon 21, Las Vegas, NV, USA, 2013 [Online]. Available: http://illmatics.com/car_hacking.pdf. [Accessed: 06-May-2017]

[43] Andy Greenberg, 'Hackers Reveal Nasty New Car Attacks--With Me Behind The Wheel', *Forbes*, 24-Jul-2013. [Online]. Available: http://www.forbes.com/sites/andygreenberg/2013/07/24/hackers-reveal-nasty-new-car-attacks-with-me-behind-the-wheel-video/. [Accessed: 06-May-2017]

[44] Chris Valasek and Charlie Miller, 'Car Hacking Tools'. [Online]. Available: http://illmatics.com/content.zip. [Accessed: 06-May-2017]

[45] Charlie Miller and Chris Valasek, 'A Survey of Remote Automotive Attack Surfaces', in *Black Hat USA*, 2014 [Online]. Available: https://sm.asisonline.org/ASIS%20SM%20Documents/remote%20attack%20surfaces.pdf. [Accessed: 06-May-2017]

[46] Charlie Miller and Chris Valasek, 'Remote Exploitation of an Unaltered Passenger Vehicle', in *Black Hat USA*, 2015 [Online]. Available: https://securityzap.com/files/Remote%20Car%20Hacking.pdf. [Accessed: 06-May-2017]

[47] Andy Greenberg, 'Hackers Remotely Kill a Jeep on the Highway—With Me in It', *Wired*, 21-Jul-2015. [Online]. Available: https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/. [Accessed: 06-May-2017]

[48] Craig Smith, *The car hacker's handbook: a guide for the penetration tester*. San Francisco: No Starch Press, 2016, ISBN: 978-1-59327-703-1.

[49] Johan Lindberg, 'Security Analysis of Vehicle Diagnostics using DoIP', Master of Science Thesis, Chalmers University of Technology, Göteborg, Sweden, 2011 [Online]. Available: http://publications.lib.chalmers.se/records/fulltext/143639.pdf. [Accessed: 06-May-2017]

[50] N. Herold, S. A. Posselt, O. Hanka, and G. Carle, 'Anomaly detection for SOME/IP using complex event processing', in *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, 2016, pp. 1221–1226. DOI: 10.1109/NOMS.2016.7502991

[51] Robert Boatright and Joseph Tardo, 'Security Aspects of Utilizing Ethernet AVB as the Converged Vehicle Backbone', *SAE Int. J. Passeng. Cars - Electron. Electr. Syst.*, vol. 5, no. 2, pp. 470–478, Sep. 2012. DOI: 10.4271/2012-01-0735

[52] Egomania, 'SOME-IP_Analyzer', *GitHub*. [Online]. Available: https://github.com/Egomania/SOME-IP_Analyzer. [Accessed: 06-May-2017]

[53] Egomania, 'SOME-IP_Generator', *GitHub*. [Online]. Available: https://github.com/Egomania/SOME-IP_Generator. [Accessed: 06-May-2017]
[54] Roel Verdult, Flavio D. Garcia, and Barıs Ege, 'Dismantling Megamos Crypto: Wirelessly Lockpicking a Vehicle Immobilizer', in *Supplement to the Proceedings of the 22nd USENIX Security Symposium*, Washington D.C., 2013 [Online]. Available: https://www.usenix.org/sites/default/files/sec15_supplement.pdf. [Accessed: 06-May-2017]
[55] Catalin Cimpanu, 'Volkswagen Sued Researchers for 2 Years to Prevent Them from Publishing a Security Flaw', *Softpedia News*, 16-Aug-2015. [Online]. Available: http://news.softpedia.com/news/volkswagen-sued-researchers-for-2-years-to-prevent-them-from-publishing-a-security-flaw-489347.shtml. [Accessed: 06-May-2017]
[56] Andy Greenberg, 'GM Took 5 Years to Fix a Full-Takeover Hack in Millions of OnStar Cars', *Wired*, 09-Oct-2015. [Online]. Available: https://www.wired.com/2015/09/gm-took-5-years-fix-full-takeover-hack-millions-onstar-cars/. [Accessed: 06-May-2017]
[57] Fred Lambert, 'Tesla hired Chris Evans from Google's Project Zero to lead the company's security team', *Electrek*. 07-Aug-2015 [Online]. Available: https://electrek.co/2015/08/06/tesla-hired-chris-evans-from-googles-project-zero-to-lead-the-companys-security-team/. [Accessed: 06-May-2017]
[58] 'Tesla's bug bounty program', *Bugcrowd Inc.* [Online]. Available: https://bugcrowd.com/tesla. [Accessed: 06-May-2017]
[59] Offensive Security, 'Kali Linux Website', *Kali*. [Online]. Available: https://www.kali.org/. [Accessed: 12-Sep-2017]
[60] Vector Informatik GmbH, 'VC121-12 Fact Sheet'. Vector Informatik GmbH, Oct-2014 [Online]. Available: https://vector.com/portal/medien/cmc/factsheets/VC121-12_FactSheet_EN.pdf
[61] Vector Informatik GmbH, 'VN5610/VN5610A Ethernet/CAN Interface Manual'. Vector Informatik GmbH, 2016 [Online]. Available: https://vector.com/portal/medien/cmc/manuals/VN5600_Manual_EN.pdf
[62] iSYSTEM, 'iC5000 On-Chip Analyzer Hardware Reference'. iSYSTEM, Apr-2017 [Online]. Available: http://www.isystem.com/files/products/OnChip/iC5000/IC50000.pdf
[63] Wireshark, 'Wireshark Website'. [Online]. Available: https://www.wireshark.org/. [Accessed: 02-Jul-2017]
[64] Vector Informatik GmbH, 'CANoe - ECU Development & Test', *Vector*. [Online]. Available: https://vector.com/vi_canoe_en.html. [Accessed: 02-Jul-2017]
[65] Philippe Biondi, 'Scapy 2.3.3-dev documentation', *Read the Docs*, Sep-2017. [Online]. Available: https://scapy.readthedocs.io/en/latest/. [Accessed: 10-Sep-2017]
[66] Vector Informatik GmbH, 'Introduction to AUTOSAR', *Vector E-Learning*, 19-Oct-2016. [Online]. Available: https://elearning.vector.com/vl_autosar_introduction_en.html. [Accessed: 02-Jul-2017]

[67] Wind, 'Wind River Website'. [Online]. Available: https://www.windriver.com/products/development-tools/. [Accessed: 02-Jul-2017]
[68] iSYSTEM, 'iSystem Website', *iSYSTEM*. [Online]. Available: http://www.isystem.com/products/software/winidea. [Accessed: 02-Jul-2017]
[69] Vector Informatik GmbH, 'vFlash Manual'. Vector Informatik GmbH, 2015 [Online]. Available: https://vector.com/portal/medien/cmc/manuals/vFlash_Manual_EN.pdf
[70] ISECOM, 'OSSTMM V3 RAV Calculator'. ISECOM [Online]. Available: http://www.isecom.org/mirror/rav_calc_OSSTMM3.xls
[71] J. Postel, 'Internet Control Message Protocol', Internet Request for Comments, vol. RFC 792 (INTERNET STANDARD), Sep. 1981 [Online]. Available: https://www.rfc-editor.org/rfc/rfc792.txt
[72] Gordon 'Fyodor' Lyon, *Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning*. Nmap Project, 2009, ISBN: 978-0-9799587-1-7 [Online]. Available: https://nmap.org/book/. [Accessed: 12-Sep-2017]
[73] International Organization for Standardization, *ISO 14229-1:2013 - Road vehicles -- Unified diagnostic services (UDS) -- Part 1: Specification and requirements*. 2013, p. 392 [Online]. Available: https://www.iso.org/standard/55283.html. [Accessed: 12-Sep-2017]
[74] International Organization for Standardization, *ISO 13400-2:2012: Road vehicles - Diagnostic communication over Internet Protocol (DoIP) -- Part 2: Transport protocol and network layer services*. 2012, p. 68 [Online]. Available: https://www.iso.org/standard/53766.html. [Accessed: 12-Sep-2017]
[75] Ofir Arkin and Josh Anderson, 'Etherleak: Ethernet Frame Padding Information Leakage', *Securiteam*, 07-Jan-2003. [Online]. Available: http://www.securiteam.com/securitynews/5BP01208UO.html. [Accessed: 12-Sep-2017]
[76] Charles Horning, 'A Standard for the Transmission of IP Datagrams over Ethernet Networks', Internet Request for Comments, vol. RFC 894 (INTERNET STANDARD), Apr. 1984 [Online]. Available: https://www.rfc-editor.org/rfc/rfc894.txt. [Accessed: 12-Sep-2017]
[77] Jonas Wolf, Eduard Metzker, and Armin Happel, 'Ethernet-Security am Beispiel SOME/IP', Wolfsburg, 21-Oct-2015 [Online]. Available: https://vector.com/portal/medien/solutions_for/Security/Ethernet-Security_SOMEIP_Lecture_VDI_2015.pdf. [Accessed: 06-May-2017]
[77] M. Baugher, D. McGrew, E. Carrara, M. Naslund, and K. Norrman, 'The Secure Real-time Transport Protocol (SRTP)', Internet Request for Comments, vol. RFC 3711 (INTERNET STANDARD), Mar. 2004 [Online]. Available: https://www.rfc-editor.org/rfc/rfc3711.txt
[78] J. Arkko, E. Carrara, F. Lindholm, M. Naslund, and K. Norrman, 'MIKEY: Multimedia Internet KEYing', Internet Request for Comments, vol. RFC 3830 (INTERNET STANDARD), Aug. 2004 [Online]. Available: https://www.rfc-editor.org/rfc/rfc3830.txt
[80] Sean Kilcarr, 'Study: Driver assistance technology could prevent 28% of crashes', *Fleet Owner*, 30-Sep-2015. [Online]. Available:

http://fleetowner.com/technology/study-driver-assistance-technology-could-prevent-28-crashes. [Accessed: 23-Sep-2017]

TRITA-ICT-EX-2017:180