University of West Bohemia
Faculty of Applied Sciences
Department of Computer Science and Engineering

Security Testing in Safety-Critical Networks

PhD Study Report

Nils Weiss

Pilsen, 2020

Contents

1 Introduction

The increasing connectivity of our modern world creates safety-critical networks in an exponential manner. Every connected computer or microcontroller participates in a communication network to deliver smart services. The security of a network is not inheritable. This leads to the fact that two individual secure networks won't form a secure network after their connection. Every new connection of a system can be used as an attack surface. If at least one of the participants in a network is a safety-critical component, the combined network forms a not secure safety-critical network. This process happens every day uncountable times in our life. A smartphone connects to a car, a car connects to a Wireless Local Area Network (WLAN)-network, a WLAN-network connects to a pacemaker and industrial control systems get remotely maintained over the Internet. Even if a connection is not persistent, it doesn't mean, that this connection can't be used as an attack surface to the safety-critical system.

Nowadays, a formal proof of the security of a system is only possible on very small and limited systems. If we just consider the complexity of the latest Internet of Things (IoT)-microcontrollers this already rules out any possibility for a formal security verification, which inherits this problem into any network such a microcontroller may be connected to.

To conquer this increasing problem of our modern world, automated security testing of safety-critical networks becomes necessary. Semi- and fully-automated security testing of safety-critical networks can identify known bugs, vulnerabilities and exploits of a system. This thesis focuses on automotive networks as safety-critical systems. A modern vehicle has various remote attack surfaces and safety-critical communication networks for its internal functions. Security vulnerabilities in this context can lead to devastating results. Even the next step in automotive technology, autonomous vehicles, is only achievable, if a certain level of safety and security can be guaranteed. Different approaches and methods for semi-automated and fully-automated security testing of safety-critical vehicular networks will be summarized in this thesis.

2 Vehicular Networks

Modern vehicles are more similar to a computer as to a mechanical device. Up to 100 different computers, called Electronic Control Units (ECUs), connected together by hundreds of cables (more than 2 km cable in total) form the inner network of a modern vehicle. Additionally to this wired connections, multiple ECUs support wireless connectivity, mainly for convenience but also for emergency features.

This Section introduces relevant protocols and existing network topologies of recent vehicles. The application of a protocol in a vehicles network is introduced and the relevance for security evaluations is discussed briefly.

2.1 Physical Protocols

More than 20 different communication protocols exist for the vehicle internal wired communication. Most vehicles make use of five to ten different protocols for their internal communication. The decision, which communication protocol is used from an Original Equipment Manufacturer (OEM), is usually made by the trade-off between the cost for a communication technology and the price of the final car. The four major communication technologies for inter-ECU communication are Controller Area Network (CAN), FlexRay, Local Interconnect Network (LIN) and Automotive Ethernet. For security considerations, these are the most relevant protocols for wired communication in vehicles.

2.1.1 LIN

LIN is a single wire communication protocol for low data rates. Actuators and sensors of a vehicle exchange information with an ECU, acting as LIN master. Software updates over LIN are possible, but the LIN slaves usually don't need software updates through their limited functionality. This thesis excludes LIN as protocol for security relevant considerations. Attacks on LIN are possible [1] but the attack range is very limited. Also an attack propagation through LIN into a topological higher network is unlikely and wasn't shown, yet.

2.1.2 CAN

CAN is by far the most used communication technology for inter-ECU communication in vehicles. In older or cheaper vehicles, CAN is still the main protocol for a vehicles backbone communication. Safety-critical communication during a vehicles operation, diagnostic information and software updates are transferred between ECUs over CAN. The lack of security features in the protocol itself, combined with the broad use, makes CAN to the major protocol for security investigations.

2.1.3 FlexRay

The FlexRay consortium designed FlexRay as successor of CAN. Modern vehicles have higher demands on communication bandwidth. By design, FlexRay is a fast and reliable communication protocol for inter-ECU communication. FlexRay components are more expensive than CAN components which leads to a more selective use by OEMs. The lack of open source hardware and software for FlexRay communication rules out the consideration of FlexRay in this thesis.

2.1.4 Automotive Ethernet

Recent upper class vehicles implement Automotive Ethernet the new backbone technology for vehicle internal communication. The rapidly grown bandwidth demands already replace FlexRay [2]. The major reasons for these demands are driver-assistant and autonomous-driving features. Only the physical layer (layer 1) of the Open Systems Interconnection (OSI) model distinguishes Ethernet (IEEE 802.3) from Automotive Ethernet (BroadR-Reach). This design decision leads to multiple advantages. For example, communication stacks of high level operating systems can be used without modification as well as routing, filtering and firewall systems. Automotive Ethernet components are already cheaper than FlexRay components. This will lead to vehicle topologies, where CAN and Automotive Ethernet are the major protocols for communication.

2.2 Topologies

The used topology of a vehicle network is mainly dependent from the OEM and the price category of the vehicle itself. OEMs implement vehicle topolo-

gies through construction kits for different vehicle categories. This means, vehicle models from the same OEM with similar selling prices have very similar, sometimes even identical, ECUs and network topologies.

Dependent on the vehicle topology, the attack-ability of the overall vehicle, can be completely different [3]. To distinguish, if a network or a sub-network is safety-critical, the participating ECUs have to be analyzed regarding their functionality. The attack-ability of a vehicle can be rated lower, if a network separation between safety-critical ECUs and ECUs with remote attack surfaces, exist [4]. The same applies vice versa.

2.2.1 Line-Bus

The first vehicles with CAN bus used a single network with a line-bus topology, as illustrated from Figure 1. Some lower priced vehicles still use one or two shared CAN bus networks for their internal communication, nowadays. The downside of this topology is its vulnerability and the lack of a network separation. All ECUs of a vehicle are connected together on a shared bus. Since CAN doesn't support security features from its protocol definition, any participant on this bus can communicate directly with all other participants. This allows an attacker to affect all ECUs, even safety-critical ones, by compromising one single ECU. The overall security level of this network is given from the security level of the weakest participant.

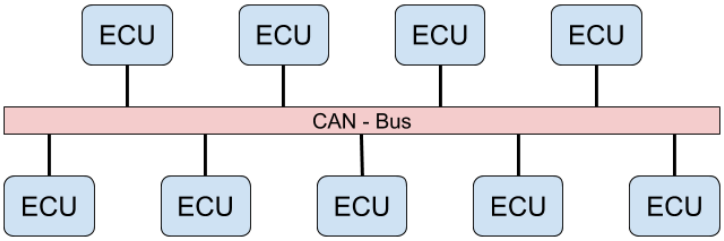

Figure 1: Line-Bus Network Topology

The famous attack from Miller and Valasek was possible because this topology was used in the car they attacked [5]. Attackers can escalate an

attack of a vulnerable ECU over the network to interfere directly with safety-critical ECUs.

2.2.2 Central Gateway

The central Gateway (GW) topology can be found in higher priced older cars and in every medium to low priced recent car. A centralized GW ECU separates domain specific sub-networks as shown in Figure 2. This allows an OEM to encapsulate all ECUs with remote attack surfaces in one sub-network.

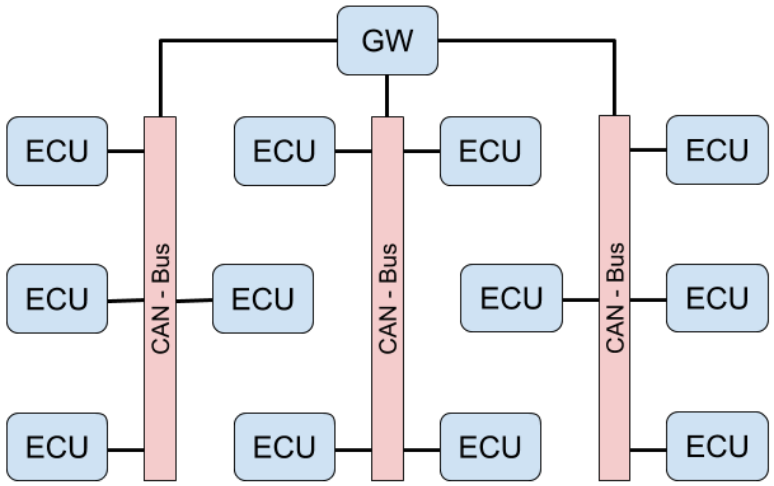

Figure 2: Network Topology with central GW ECU

ECUs with safety-critical functionality are located in an individual CAN network. Next to CAN, FlexRay might also be used as communication protocol inside a separated network domain. The security of a safety-critical network in this topology depends mainly on the security of the central GW ECU. This architecture increases the overall security level of a vehicle through separation. After an attacker exploited an ECU through an arbitrary attack surface a second exploitable vulnerability or a logical bug is necessary to compromise a safety-critical network inside a vehicle. This second exploit or logical bug is necessary to exploit the central GW ECU,

2.2.3 Central Gateway and Domain Controller

In latest higher priced vehicles, a new topology with central GW and Domain Controllers (DCs) can be found. The general structure of this topology is shown in Figure 3. The increasing demand on bandwidth in modern vehicles with autonomous driving and driver assistant features led to this topology. An Automotive Ethernet Network is used as communication backbone for the entire vehicle. Individual domains, connected through a DC with the central GW form the vehicles backbone. The individual DCs can control and regulate the data communication between a domain and the vehicles backbone.

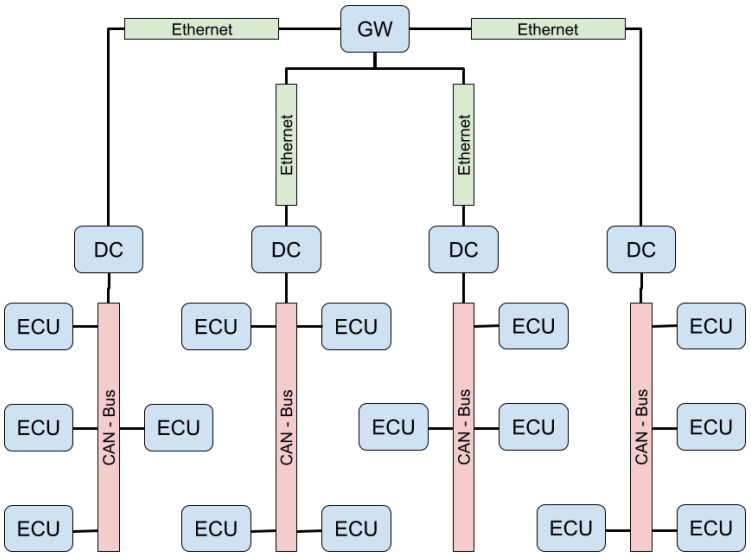

Figure 3: Network Topology with Automotive Ethernet Backbone and Domain Controller

This topology achieves a very high security level through a strong network separation with individual DCs, acting as gateway and firewall, to the vehicles backbone network. Next to this security improvement, OEMs have the advantage of dynamic information routing which is an enabler for Feature on Demand (FoD) services.

2.2.4 Summary

The topology of a vehicles' network has a very high impact on the vehicles security and safety properties. Figure 4 shows the dependencies between a vehicles price, a vehicles age and the used network topology.

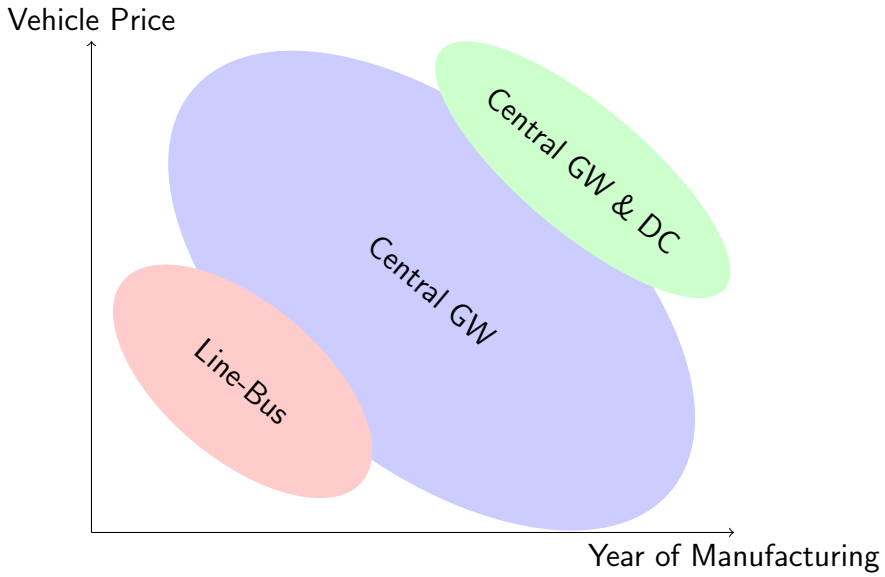

Figure 4: Overview of used topology, vehicle price and year of manufacturing

Only a few vehicles from the lower price segment or with a higher age are still using one or two CAN buses for their internal communication. Most vehicles are equipped with a central GW ECU. After the publication of Miller and Valasek, even the cheapest cars received an upgrade of their internal network topology, and OEMs made use of central GW ECUs as mitigation for cyber attacks [5]. Only a few vehicles from the higher price segment already implement a network architecture with Automotive Ethernet as communication backbone. Nevertheless, this group will rapidly grow in the near future since the advantages will compensate the higher prices for the communication equipment.

2.3 Automotive Communication Protocols

This Section provides an overview of relevant communication protocols for security evaluations in automotive networks. In contrast to Section 2.1 this

Section focuses on properties for data communication.

2.3.1 CAN

CAN was invented in 1983 as a message based robust vehicle bus communication system. The Robert Bosch GmbH designed multiple communication features into the CAN standard to achieve a robust and computation efficient protocol for controller area networks. Remarkable for the communication behavior of CAN is the internal state machine for transmission errors. This state machine implements a fail silent behavior in order to protect a safety critical network from babbling idiot nodes. If a certain limit of reception errors (REC) or transmission errors (TEC) occurred, the CAN driver changes its state, from active to passive and finally to off.

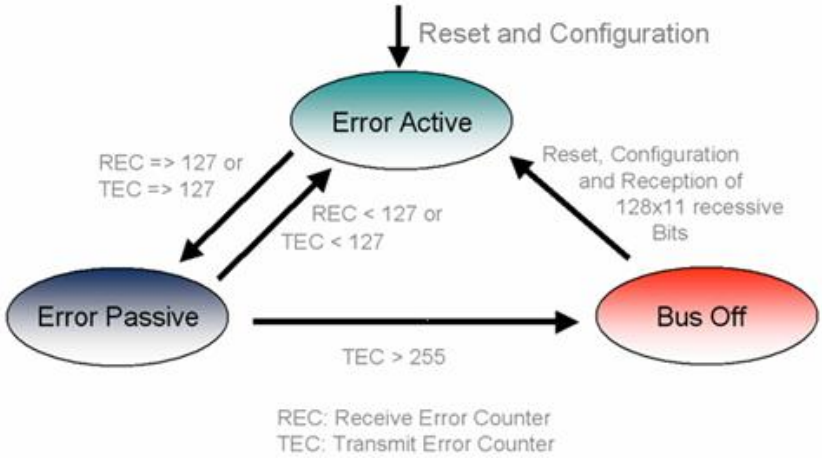

Figure 5: CAN bus states on transmission errors [6]

In recent years, this protocol specification was abused for Denial of Service (DoS) attacks and for information gathering attacks on the CAN network of a vehicle. Cho et. al. demonstrated a DoS attack against CAN networks by abusing the bus off state of ECUs [7]. Injections of communication errors in CAN frames of one specific node caused a high transmission error count in the node under attack. This forces the attacked node to enter the bus off state, as it can be seen in Figure 5. In 2019 Kulandaivel et. al. combined this attack with statistical analysis to achieve a fast and inexpensive way of network mapping in vehicular networks [8]. They combined statistics of the

CAN network traffic before and after the bus off attack was applied to a node. All missing CAN frames in the network traffic after an ECU was attacked, could now be mapped to the ECU under attack. This helps researchers to identify the origin ECU of a CAN frame.

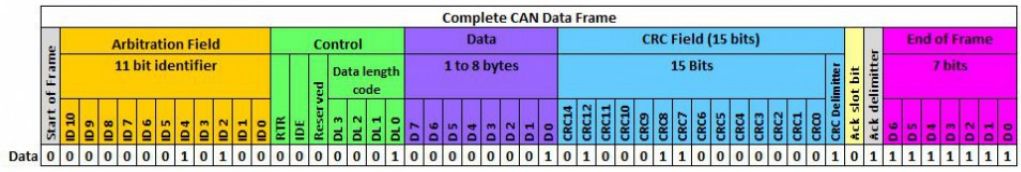

Figure 6: Complete CAN data frame structure [9]

Figure 6 shows a CAN frame and its fields as it is transferred on the network. For information exchange, only the fields arbitration, control and data are relevant. These are the only fields a usual application software has access to. All other fields are evaluated on a hardware-layer and in most cases are not forwarded to an application. The data field has a variable length and can hold up to 8 bytes. The length of the data field is specified by the data length code inside the control field. Important variations of this example are CAN frames with extended arbitration fields and the Controller Area Network Flexible Data-Rate (CAN FD) protocol. On Linux, every received CAN frame is passed to SocketCAN. SocketCAN allows the CAN handling via network sockets of the operating system. SocketCAN was created by Oliver Hartkopp and added to the Linux Kernel version 2.6.25 [10]. Figure 7 shows the frame structure, how CAN frames are encoded if a user-land application receives data from a CAN socket.

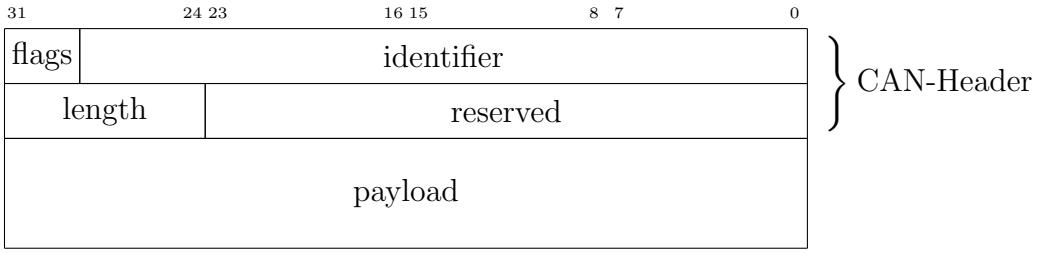

Figure 7: CAN frame as defined by SocketCAN [11]

The comparison of Figure 6 and Figure 7 clearly shows the loss of information during the CAN frame processing from a physical layer driver. Almost every CAN driver acts in the same way, no matter if an application code runs on a microcontroller or on top of Linux. This also means, that a standard application doesn't have access to the Cyclic Redundancy Check (CRC) field, the acknowledgment bit or the end-of-frame field.

Through the CAN communication in a vehicle or in a separated domain, ECUs exchange sensor data and control inputs. This data is mostly not secured and can be modified by attackers. Attackers can easily spoof sensor values on a CAN bus to trigger malicious reactions of other ECUs. This spoofing attack was described by Miller and Valasek during their studies on automotive networks [12]. To prevent attacks on safety-critical data transferred over CAN, Automotive Open System Architecture (AUTOSAR) released a specification for secure on-board communication [13].

2.3.2 ISOTP (ISO 15765-2)

The CAN protocol supports only eight byte of data. Applications like diagnostic services or ECU programming require much higher payloads than the CAN protocol supports. For this purposes, the automotive industry standardized the ISOTP (ISO 15765-2) protocol. ISOTP is a transportation layer protocol on top of CAN. Payloads with up to 4095 bytes can be transferred between ISOTP endpoints fragmented in CAN frames. Four special frame types are required by the ISOTP protocol handling.

The different types of ISOTP frames are shown in Figure 9. The payload of a CAN frame, shown in Figure 7 gets replaced by one of the four ISOTP frames from Figure 9. The individual ISOTP frames have different purposes. A single frame can transfer between 1 and 7 bytes of ISOTP message. The `len` field of a Single Frame or First Frame indicates the ISOTP message length. Every message with more than 7 bytes of payload has to be fragmented into a first frame, followed by multiple consecutive frames. This communication is illustrated by Figure 8. After the first frame is sent from a sender, the receiver has to communicate its capabilities for reception through a flow control frame to the sender. Only after this flow control is received, the sender is allowed to communicate the consecutive frames according to the receivers capabilities.

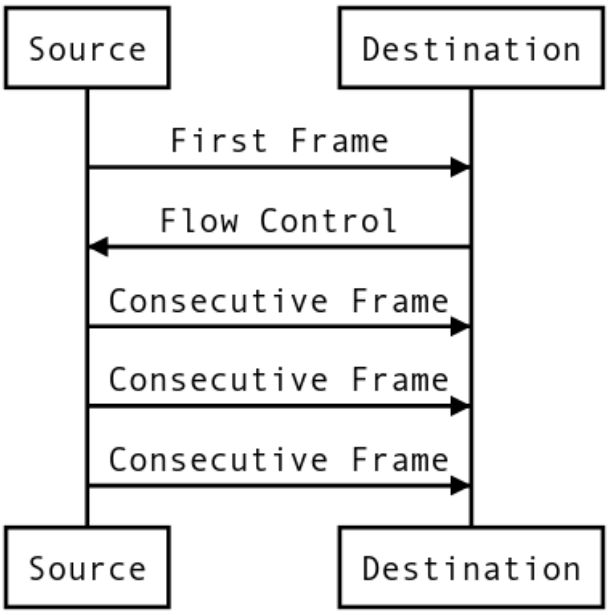

Figure 8: ISOTP fragmented communication

ISOTP acts as a plain transport protocol. In vehicles, ISOTP is mainly used as transport protocol for diagnostic communication. In rare cases, ISOTP is also used to exchange larger data between ECUs of a vehicle. Security measures have to be applied on the protocol transported through ISOTP, since ISOTP hasn't any capabilities to secure its transported data.

2.3.3 Diagnostic Protocols

Two examples of diagnostic protocols are General Motor Local Area Network (GMLAN) and Unified Diagnostic Service (UDS) (ISO 14229-2). GMLAN is used by the General Motors Cooperation. UDS is mainly used by German OEMs. Both protocols are very similar from a specification point of view and both protocols use either ISOTP or Diagnostic over IP (DoIP) messages for a directed communication with a target ECU. Since UDS is used by different OEMs, every manufacturer adds its custom additions to the standard. Also every manufacturer uses its individual ISOTP addressing

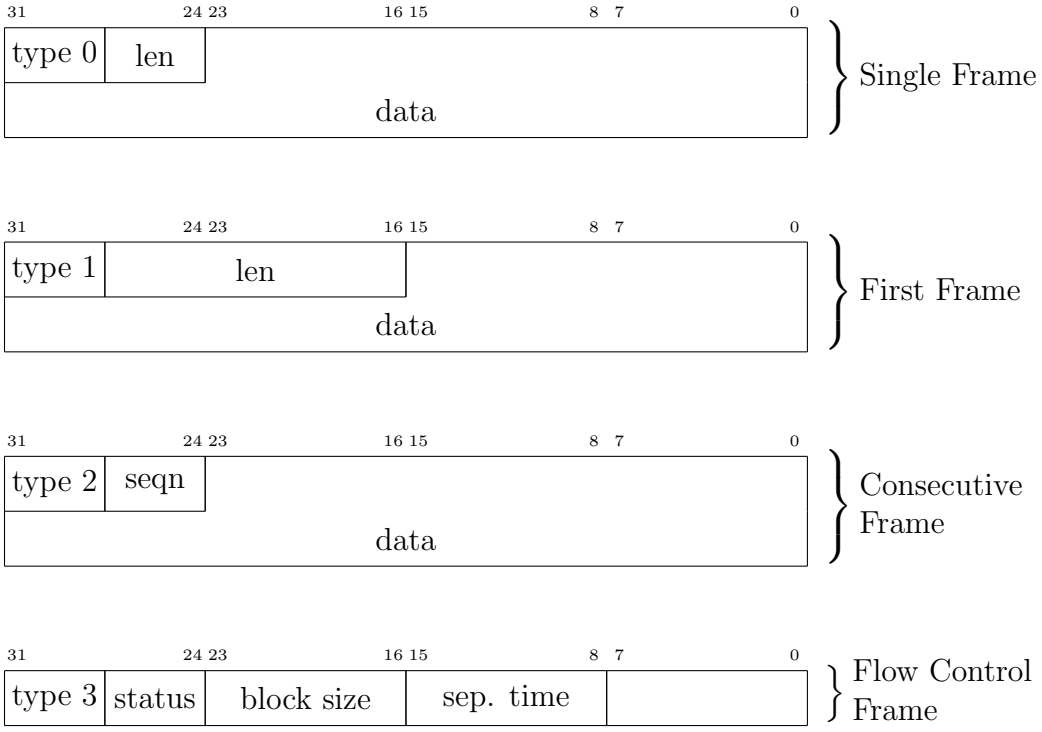

Figure 9: ISOTP frame types

for the directed communication with an ECU. GMLAN includes more precise definitions about ECU addressing and an ECUs internal behavior compared to UDS.

Both protocols, UDS and GMLAN, follow a tree-like message structure, where the first byte identifies the service. Every service is answered by a response. Two types of responses are defined in the standard. Negative responses are indicated through the service `0x7F`. Positive responses are identified by the request service identifier incremented with `0x40`.

Figure 10 and Figure 11 are given as an example how similar both diagnostic protocols are. The service identifier and the further specification of these request messages are identical.

Table 129 — Request message definition

A_Data byte	Parameter name	Cvt	Hex value	Mnemonic
#1	ReadDataByIdentifier Request Service Id	M	22	RDBI
#2 #3	dataIdentifier[] #1 = [byte#1 (MSB) byte#2]	M M	00-FF 00-FF	DID_ HB LB
:	:	:	:	:
#n-1 #n	dataIdentifier[] #m = [byte#1 (MSB) byte#2]	U U	00-FF 00-FF	DID_ HB LB

Figure 10: UDS ReadDataByIdentifier service definition [14]

Table 80: ReadDataByParameterIdentifier Request Message

Data Byte	Parameter Name [Note 1]	Cvt	Hex Value	Mnemonic
#1	ReadDataByParameterIdentifier Request Service Id	M	22	SIDRQ
#2 #3	parameterIdentifier #1 = [byte 1 (MSB) byte 2 (LSB)]	M	00 thru FF 00 thru FF	PID_ B1 B2
#4 #5	parameterIdentifier #2 = [byte 1 (MSB) byte 2 (LSB)]	U	00 thru FF 00 thru FF	PID_ B1 B2
:	:	:	:	:
#n-1 #n	parameterIdentifier #k = [byte 1 (MSB) byte 2 (LSB)]	U	00 thru FF 00 thru FF	PID_ B1 B2

Note 1: MSB = Most Significant Byte, LSB = Least Significant Byte.

Figure 11: GMLAN ReadDataByParameterIdentifier service definition [15]

2.3.4 DoIP

DoIP was first implemented on automotive networks with a centralized gateway topology. A centralized GW functions as DoIP endpoint which routes diagnostic messages to the desired network. This allows manufacturers to program or diagnose multiple ECUs in parallel. Since the Internet Protocol (IP) communication between a repair-shop tester and the GW is many times faster than the communication between the GW and a target ECU connected over CAN, the remaining bandwidth of the IP communication can be used to start further DoIP connections to other ECUs in different CAN domains. DoIP is specified as part of AUTOSAR and in ISO 13400-2.

Similar to ISOTP, DoIP doesn't specify special security measures. The responsibility regarding secured communication is delegated to the transported protocol.

2.3.5 SOME/IP

Scalable service-Oriented MiddlewarE over IP (SOME/IP) defines a new philosophy of data communication in automotive networks. In latest vehicular networks, as described in Section 2.2.3, SOME/IP is used to exchange data between network domain controllers. SOME/IP supports subscription and notification mechanisms. This allows domain controllers to dynamically subscribe to data provided by another domain controller dependent on the state of the vehicle. SOME/IP transports all the data, a vehicle needs during its normal operation, between domain controllers and the gateway.

The use-case of SOME/IP is similar to the use-case of CAN communication in a vehicle. The main purpose is the information exchange of sensor and actuator data between ECUs. This usage emphasizes SOME/IP communication as rewarding target for cyber attacks.

2.3.6 CCP/XCP

Universal Measurement and Calibration Protocol (XCP), the successor of the CAN Calibration Protocol (CCP), is a calibration protocol for automotive systems, standardized by ASAM e.V. in 2003. The main usage of XCP is during the testing and calibration phase of ECU or vehicle development. CCP is designed for the use on CAN. No message in CCP exceeds the 8-byte limitation of CAN. To overcome this restriction, XCP was designed with the aim of compatibility to a wide range of transport protocols. XCP can be used on top of CAN, CAN FD, Serial Peripherial Interface (SPI), Ethernet, Universal Serial Bus (USB) and FlexRay. The features of CCP and XCP are very similar, however XCP has a larger functional scope and optimizations for data efficiency. Both protocols have a session based communication procedure and support authentication through a seed and key mechanism between a master and multiple slave nodes. A master node is typically an engineering Personal Computer (PC). In vehicles, slave nodes are ECUs for configuration. XCP also supports simulation. An engineer has the possibility to debug a MATLAB Simulink model through XCP. In this case, the simulated model acts as XCP slave node. CCP and XCP have the ability to read and write to the memory of an ECU. Another main feature is data acquisition. Both protocols support a procedure that allows an engineer to configure a so called data acquisition list with memory addresses of inter-

est. All memory specified in such a list will be read periodically and be broadcast in a CCP or XCP Data Acquisition (DAQ) packet on the chosen communication channel.

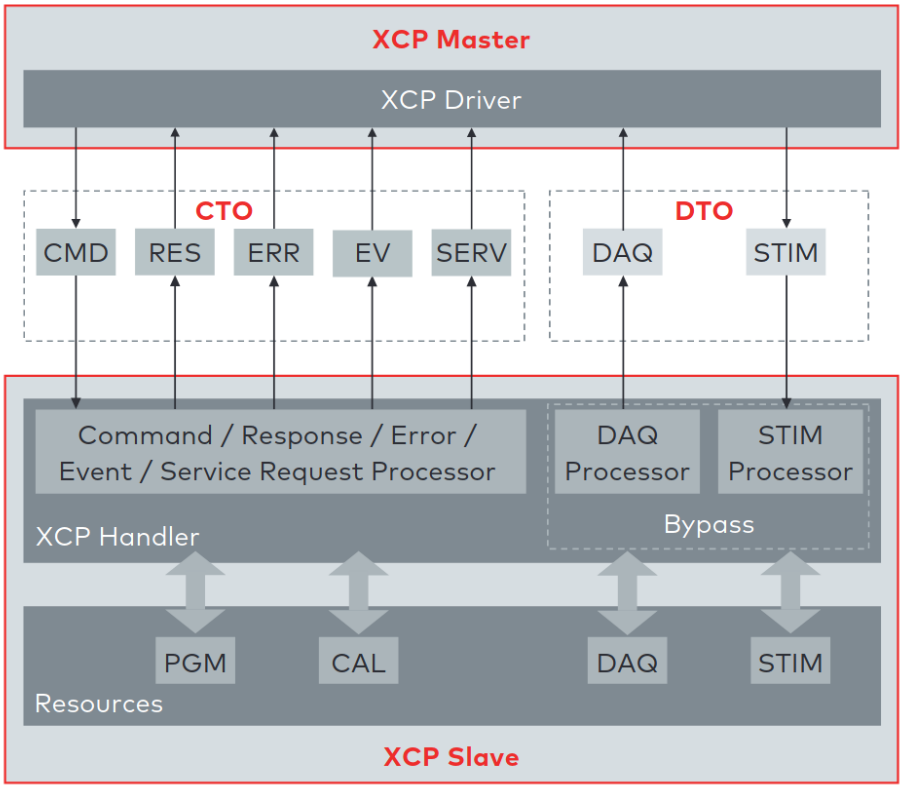

Figure 12: XCP communication model with CTO/DTO [16]

Figure 12 gives an overview about all supported communication and packet types in XCP. In the Command Transfer Object (CTO) area, all communication follows a request and response procedure always initiated by the XCP master. A Command Packet (CMD) can receive a Command Response Packet (RES), an Error (ERR) packet, an Event Packet (EV) or a Service Request Packet (SERV) as response. After the configuration of a slave through CTO CMDs, a slave can listen for Stimulation (STIM) packets and can periodically send configured DAQ packets. The resources Section of Figure 12 clearly indicates the possible attack surfaces of this protocol (Programming (PGM), Calibration (CAL), DAQ, STIM) which could be used by

an attacker. It's crucial for a vehicles security and safety that such protocols, which have their use only during calibration and development of a vehicle, are disabled or removed before a vehicle is shipped to a customer.

3 Electronic Control Units

Today vehicle architectures consist of a dense network of ECUs. An ECU is specialized on a small set of features and functions. The inner architecture of an ECU highly depends on its purpose. Every ECU is customized and cost optimized to serve its functions. OEMs design ECUs in a generic way to be able to reuse one ECU with different software and hardware configurations in different car models from the same vehicle construction kit. This development is caused by the high price pressure of automotive components. Further more, the extensive reuse of hardware and software components from previous car construction kits leads to an extremely heterogeneous mix of ECUs in a vehicle. The paper *On Threat Analysis and Risk Estimation of Automotive Ransomware* summarizes important vehicle properties for risk estimations of cyber-attacks [3]. Some of these properties directly depend on the individual implementation of an ECU. Two ECUs from the same vehicle can be completely different in terms of computation power, used operating system, supported network interfaces, supported update mechanisms and many more. Common between all ECUs is the circumstance that they have to communicate with other ECUs to serve their functions. This is another reason, why vehicular networks are an important attack surface for security investigations.

3.1 Software Architecture

Between 2004 and 2006, the AUTOSAR standard for automotive software was released. OEMs and suppliers for automotive hard- and software defined AUTOSAR as open system architecture for automotive systems. AUTOSAR caused major changes in automotive software development. OEMs want to achieve ex-changeability between suppliers through a common software architecture of all electronic components. Unique functions of a vehicle are implemented on top of an AUTOSAR Operating System (OS) to abstract the specific hardware completely. This should theoretically allow the migration of functions from one ECU to another ECU, even if the physical hardware

of this ECUs is completely different.

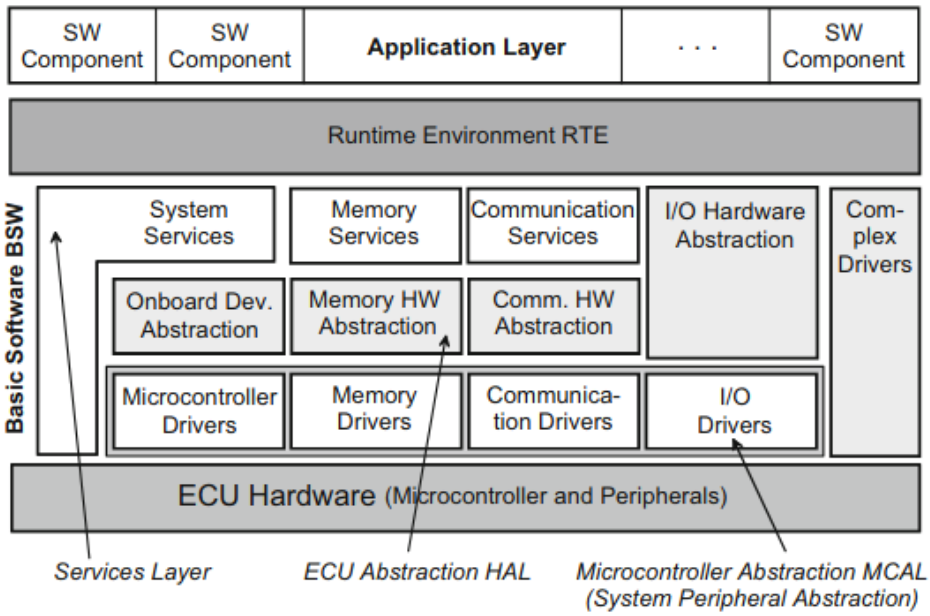

Figure 13: Layer model of the AUTOSAR software [17, p. 368]

Figure 13 shows different software components on top of a Runtime Environment (RTE) layer. This demonstrates the complete abstraction of application layer software components from the physical implementation of an ECU hardware.

3.2 ECU Programming

In general, an ECU needs at least three different types of firmware. The initial software component which is executed after reset is called Boot-Manager [17, p. 441]. This Boot-Manager runs internal checks and decides which firmware is executed next. On normal conditions, the ECU application is started and the ECU starts to serve its functions to the vehicle. In case of an update a special application, called Flash-Loader, is entered. This Flash-Loader has the ability to read and write to internal memories. For safety reasons, the software libraries and drivers for persistent write operations on internal memories are shipped on demand over the network to the Random Access

Memory (RAM) of the ECU processor. In general, update mechanisms are a main attack surface of ECUs. The design choice of loading drivers for memory operation on demand can be an extremely critical security vulnerability, if this mechanism is implemented in an insecure way.

3.3 Functionalities

The functionalities of a ECU can be split in two different categories.

1. Functions of an ECU during the desired use of a vehicle that are real-time functions for input to output data processing. These functions show a deterministic run-time behavior and process physical inputs or input data from a communication channel to generate the desired outputs. The network traffic during these operations caused by this ECU is deterministic and precisely timed. ECUs broadcast their computation results on a shared network to offer these results as input to other ECUs.

2. Next to functions that process input data, an ECU serves multiple diagnostic and configuration services. The execution of such services is triggered upon request. Over the network, a service request is sent to an ECU which causes the execution of this service. The result is delivered to the requester over the chosen communication channel. This operations are performed through a higher level communication protocol since most messages usually exceed the message size of 8 byte, the maximum size of a CAN frame specified in ISO 11898-3.

3.4 Security Measures

The publication *Cybersecurity Evaluation of Automotive E/E Architectures* from Ring et. al. gives an overview about security mechanisms in automotive architectures and compares the state of implementations in 2018 with an estimation of their evolvement in 2023 [18]. Vehicle manufacturers and suppliers work on a security in depth strategy to overcome the modern challenges in terms of security and safety. This security in depth strategy is a layered approach which separates the individual security measures for ECUs in levels according to their effective range.

- Chip-Level
  - Hardware Security Module (HSM)
  - Hardware Trust Anchors
- ECU-Level
  - Secure Boot / Authenticated Boot
  - AUTOSAR
  - Signed software updates
- Network-Level
  - Domain separation
  - Secure on-board communication
- Vehicle-Level
  - Firewall
  - Intrusion Detection System (IDS)
  - Intrusion Prevention System (IPS)
  - Secure diagnostic services
- Fleet-Level
  - Secure back-end communication
  - Security defense center
  - Over-the-Air software updates

The previous list of security measures and levels is not complete but should provide a general overview. A lot of research is ongoing in every level.

4 Automotive Penetration Testing

4.1 Automotive Security Research

The first notable academic publications related to security analyses of modern vehicles were published from Koscher et al. in 2010 [19], followed by

another publication from Checkoway et al. in 2011 [20]. The whole research field received a broad audience after the publications of Miller and Valasek in 2013, 2014, and 2015 were they demonstrated a remote exploitation of an unaltered passenger vehicle [12][4][5]. The German car manufacturer BMW was targeted from Spaar who demonstrated a remote attack on the locking mechanism in 2015 [21]. The story continues with publications from the Tencent Keen Security Lab about a remote exploitation of a Tesla Model S in 2016 and remote exploitation of multiple BMW models in 2018 [22][23]. A smaller publication targeted the car brands Volkswagen and Audi in 2018 [24].

In 2016, Craig Smith published the book *The Car Hacker's Handbook* [25]. This book documents techniques and tools used by the Open Garages community which started a movement around all kind of software modifications of vehicles in 2014.

Since 2019 the Pwn2Own competition, hosted from the organization Zero Day Initiative, added a Tesla car to their targets for hacking [26]. Every year from that onward, attendees showed their abilities and successfully broke into the cars web browser, which rewards them with the ownership of the hacked car next to a cash prize.

These important publications raised public awareness for security in the safety-critical system, passenger vehicle. OEMs couldn't ignore the necessity of security engineering and security testing for their vehicles, anymore. The effects of these publications are already visible in the electrical design of modern vehicles. Insecure network topologies were abolished and security measures for ECU software were taken. Nonetheless, the challenge to provide secure vehicles is immense. Over the years security wasn't a major part of vehicle engineering and the ecosystem that OEMs built around connected cars is huge and complex. The software and firmware management in a modern car, even in a single ECU is already a challenging task.

A new ISO/SAE standard, ISO/SAE 21434 *Road vehicles – Cybersecurity engineering*, which enforces security engineering and penetration testing during the development process of any vehicle for the European market will be valid in November 2020. This can also be interpreted as direct aftermath from the previous publications.

4.2 Tools for Penetration Testing

Next to publications mentioned in Section 4.1, which mainly focused on research regarding vulnerabilities and exploitation of automotive systems, many researchers published tools and frameworks for automotive penetration testing and security analysis. Multiple tools for penetration tests and CAN media access were analyzed from Pozzobon et al. in 2018 [27]. This research showed, that most tools and frameworks were not comprehensive and reliable enough to cover main aspects of automotive penetration testing tasks.

4.3 Penetration Testing Process

The general penetration testing process for IT systems is described in a publication from the German Federal Cyber Security Authority (BSI) [28]. In 2014 Sahri et. al. published an article about the penetration testing process in higher education institutes with focus on education of security experts [29]. This two publications give a detailed overview about the necessary steps during a penetration test of IT systems. Since automotive systems can be considered as IT systems most of this process can be directly applied to an automotive penetration testing process.

Figure 14: Penetration testing process by Wai, Chan [29] [30]

Main differences of a penetration testing process for IT systems, as shown in Figure 14, to a penetration testing process for a safety-critical automotive system is the threat modeling and the goal of the penetration test. Section 4.4 introduces some background information regarding threats of automotive systems. These threats have to be considered as soon as the goals of a penetration test are defined.

For the accomplishment of an automotive penetration test, several specialized skills might be required. Automotive systems and also automotive software engineering differs a lot from classical IT systems and their software. Dependent on the scope of a penetration test, a penetration tester needs a strong understanding of electrotechnical systems and automotive communication systems, as described in Section 2.1.

So far, automotive penetration testing is a fairly new field of work. In the recent years, all car manufacturers designed internal security processes which also involve penetration testing. For individual systems, the penetration process might still be very close to security research, since only very little information regarding security issues is publicly available.

4.4 Threats for Automotive Systems

Nowadays, the main threats of automotive systems are modifications of a system (tuning) and thievery including preparations to be able to resell the entire vehicle or components of it. In the near future new threats which involve the remote communication systems of a vehicle will happen in the public. One example could be automotive malware or ransomware [3]. Overall, threats for vehicular systems are highly profit oriented since attackers accept a high risk during an attack. This is caused by the physical presence of a car theft to its target. A game changer will be an automotive ransomware, which can infect new vehicles over remote interfaces. This wouldn't require the physical presence anymore. Backers of ransomware attacks can operate from any country in the world without risking legal issues in the country where their attack takes place. Also the hiding of their traces is much easier on a remote attack over the Internet. Another upcoming threat might be the blackmailing of car manufacturers. Similar to crimes in the past, where an offender blackmailed producers by poisoning their food products in supermarkets, cyber criminals could blackmail car manufacturers with harmful

modifications to their fleets.

All malicious cyber attacks against vehicular systems which are publicly disclosed are targeting immobilizer or entry systems of cars. Chip tuning isn't considered a malicious cyber attack in this case. Only attacks which lead to the loss of money or cause harm to people where considered in this statement.

4.5 ISO 21434 & UNECE WP.29

Upcoming regulations ISO/SAE 21434 and the UNECE WP.29 aim to standardize cyber security engineering for road vehicles. These regulations will force vehicle manufacturers to secure the entire development process and also maintain the security of sold vehicles. Security testing during and after the development process is a main goal of these regulations. This is an additional motivation for OEMs and suppliers to implement automated security tests in their safety-critical systems.

5 Conclusion

The motivations for additional security testing in safety-critical networks are obvious. Customers constantly ask for more connected and smart features on the one hand, authorities get new regulations regarding security of the ground on the other hand. A brief introduction into the automotive security research history showed that security tests are absolutely necessary to secure tomorrows vehicles. The modern network topologies all with a single point of failure and the complex software design in ECUs require further security research. Security flaws in these networks or their components can lead to devastating outcomes. The complexity of modern cars electrical architecture can only be handled with automated testing approaches. Functional and safety-related testing is well understood in that field. Security testing, the new challenge, still requires research in semi-automated and fully-automated security testing approaches.

A study from the consulting company Deloitte revealed that most customers have strong concerns regarding their safety and security in autonomous vehicles [31]. The whole new trend and technique of autonomous vehicles will

only be accepted, if customers can trust their autonomous vehicle. One single accident, caused by a cyber-attack, can irreversible break that trust and will delay the break through of a whole technology.

Vehicle manufactures, suppliers and insurance companies are facing completely new threats through the attack surfaces of connected vehicles. Automated security testing will be one important part, next to multiple other actions, to estimate and lower the overall risk of cyber-attacks.

5.1 Aims of Doctoral Thesis

- Evaluations of Open Source Software Projects suitable for security tests in safety-critical networks
- Implementation of software tools to support the penetration testing process in safety-critical networks
- Research on current security flaws of automotive systems
- Research on automation capabilities of security tests in automotive systems
- General consideration what kind of security flaws can be detected through automated testing

5.2 Title of Doctoral Thesis

Security Testing in Safety-Critical Networks

Acronyms

AUTOSAR Automotive Open System Architecture. 10, 13, 16, 19

CAL Calibration. 15

CAN Controller Area Network. 2–5, 7–10, 13, 14, 18, 21

CAN FD Controller Area Network Flexible Data-Rate. 9, 14

CCP CAN Calibration Protocol. 14, 15

CMD Command Packet. 15

CRC Cyclic Redundancy Check. 10

CTO Command Transfer Object. 15

DAQ Data Acquisition. 15

DC Domain Controller. 6

DoIP Diagnostic over IP. 11, 13

DoS Denial of Service. 8

ECU Electronic Control Unit. 2–5, 7–14, 16–20, 23

ERR Error. 15

EV Event Packet. 15

FoD Feature on Demand. 6

GMLAN General Motor Local Area Network. 11, 12

GW Gateway. 5–7, 13

HSM Hardware Security Module. 19

IDS Intrusion Detection System. 19

IoT Internet of Things. 1

IP Internet Protocol. 13

IPS Intrusion Prevention System. 19

LIN Local Interconnect Network. 2

OEM Original Equipment Manufacturer. 2–7, 11, 16, 20, 23

OS Operating System. 16

OSI Open Systems Interconnection. 3

PC Personal Computer. 14

PGM Programming. 15

RAM Random Access Memory. 17

RES Command Response Packet. 15

RTE Runtime Environment. 17

SERV Service Request Packet. 15

SOME/IP Scalable service-Oriented MiddlewarE over IP. 14

SPI Serial Peripherial Interface. 14

STIM Stimulation. 15

UDS Unified Diagnostic Service. 11, 12

USB Universal Serial Bus. 14

WLAN Wireless Local Area Network. 1

XCP Universal Measurement and Calibration Protocol. 14, 15

References

[1] Junko Takahashi, Yosuke Aragane, Toshiyuki Miyazawa, Hitoshi Fuji, Hirofumi Yamashita, Keita Hayakawa, Shintarou Ukai, and Hiroshi Hayakawa. Automotive attacks and countermeasures on lin-bus. *Journal of Information Processing*, 25:220–228, 02 2017.

[2] Ankita Sawanta and Lenina Svb. Can, flexray, most versus ethernet for vehicular networks. *International Journal of Innovations & Advancement in Computer Science*, 04 2018.

[3] Nils Weiss, Markus Schrötter, and Rudolf Hackenberg. On threat analysis and risk estimation of automotive ransomware. In *ACM Computer Science in Cars Symposium*, CSCS '19, New York, NY, USA, 2019. Association for Computing Machinery.

[4] Dr. Charlie Miller and Chris Valasek. A survey of remote automotive attack surfaces. DEF CON 22 Hacking Conference. Las Vegas, NV: DEF CON, August 2014.

[5] Dr. Charlie Miller and Chris Valasek. Remote exploitation of an unaltered passenger vehicle. DEF CON 23 Hacking Conference. Las Vegas, NV: DEF CON, August 2015.

[6] Rotti, Prasad (IE10). *CAN bus state*, 2007 (accessed February 14, 2020). http://can-bus.996267.n3.nabble.com/attachment/1784/0/image001.jpg.

[7] Kyong-Tak Cho and Kang G. Shin. Error handling of in-vehicle networks makes them vulnerable. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, page 1044–1055, New York, NY, USA, 2016. Association for Computing Machinery.

[8] Sekar Kulandaivel, Tushar Goyal, Arnav Kumar Agrawal, and Vyas Sekar. Canvas: Fast and inexpensive automotive network mapping. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 389–405, Santa Clara, CA, August 2019. USENIX Association.

[9] Pico Technology Ltd. *Complete CAN data frame structure*, 2020 (accessed February 14, 2020).

https://www.picotech.com/images/uploads/library/topics/_med/CAN-full-frame.jpg.

[10] *Readme file for the Controller Area Network Protocol Family (aka SocketCAN)*, 2020 (accessed January 29, 2020). https://www.kernel.org/doc/Documentation/networking/can.txt.

[11] *LINKTYPE_CAN_SOCKETCAN - Packet structure*, 2020 (accessed January 30, 2020). https://www.tcpdump.org/linktypes/LINKTYPE_CAN_SOCKETCAN.html.

[12] Dr. Charlie Miller and Chris Valasek. Adventures in automotive networks and control units. DEF CON 21 Hacking Conference. Las Vegas, NV: DEF CON, August 2013.

[13] *Specification of Secure Onboard Communication*, 2020 (accessed January 31, 2020). https://www.autosar.org/fileadmin/user_upload/standards/classic/4-3/AUTOSAR_SWS_SecureOnboardCommunication.pdf.

[14] *Road vehicles — Unified diagnostic services (UDS) — Specification and requirements*, 2020 (accessed February 27, 2020). http://read.pudn.com/downloads191/doc/899044/ISO+14229+(2006).pdf.

[15] *General Motors Local Area Network Enhanced Diagnostic Test Mode Specification*, 2020 (accessed February 27, 2020). https://www.docdroid.net/zTNov8C/gmw3110-2010.pdf.

[16] *XCP – The Standard Protocol for ECU Development*, 2020 (accessed January 30, 2020). https://assets.vector.com/cms/content/application-areas/ecu-calibration/xcp/XCP_ReferenceBook_V3.0_EN.pdf.

[17] Werner Zimmermann and Ralf Schmidgall. *Bussysteme in der Fahrzeugtechnik*. ATZ/MTZ-Fachbuch. Vieweg, Wiesbaden, 5 edition, 2014.

[18] Davor Frkat Martin Ring and Martin Schmiedecker. Cybersecurity evaluation of automotive e/e architectures. In *ACM Computer Science in*

Cars Symposium, CSCS '18, New York, NY, USA, 2018. Association for Computing Machinery.

[19] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage. Experimental security analysis of a modern automobile. In *2010 IEEE Symposium on Security and Privacy*, pages 447–462, May 2010.

[20] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, and Tadayoshi Kohno. Comprehensive experimental analyses of automotive attack surfaces. In *Proceedings of the 20th USENIX Conference on Security*, SEC'11, page 6, USA, 2011. USENIX Association.

[21] Dieter Spaar. *Auto, öffne dich!* c't 2015, Issue 5.

[22] Tencent Keen Security Lab. *Car Hacking Research: Remote Attack Tesla Motors*, 2020 (accessed February 14, 2020). https://keenlab.tencent.com/en/2016/09/19/Keen-Security-Lab-of-Tencent-Car-Hacking-Research-Remote-Attack-to-Tesla-Cars/.

[23] Tencent Keen Security Lab. *New Vehicle Security Research by KeenLab: Experimental Security Assessment of BMW Cars*, 2020 (accessed February 14, 2020). https://keenlab.tencent.com/en/2018/05/22/New-CarHacking-Research-by-KeenLab-Experimental-Security-Assessment-of-BMW-Cars/.

[24] Computest Group BV. The connected car - ways to get unauthorized access and potential implications. 2018. https://www.computest.nl/wp-content/uploads/2018/04/connected-car-rapport.pdf.

[25] Craig Smith. *The Car Hacker's Handbook: A Guide for the Penetration Tester*. No Starch Press, USA, 1st edition, 2016.

[26] Zero Day Initiative. *Pwn2Own Vancouver 2019: Wrapping Up and Rolling Out*, 2020 (accessed February 29, 2020). https://www.zerodayinitiative.com/blog/2019/3/22/pwn2own-vancouver-2019-wrapping-up-and-rolling-out.

[27] Enrico Pozzobon, Nils Weiss, Sebastian Renner, and Rudolf Hackenberg. A survey on media access solutions for can penetration testing. In *ACM*

Computer Science in Cars Symposium, CSCS '18, New York, NY, USA, 2018. Association for Computing Machinery.

[28] Bundesamt für Sicherheit in der Informationstechnik. *Studie Durchführungskonzept für Penetrationstests*, 2020 (accessed February 29, 2020). https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Studien/Penetrationstest/penetrationstest.pdf.

[29] Zulazeze Sahri, Eizan Aziz, Khairul Zolkefley, Roslan Sadjirin, and Mohd. Ikhsan Md. Raus. Implementing it security penetration testing in higher education institute. *Australian Journal of Basic and Applied Sciences*, 8:67–72, 06 2014.

[30] Chan Wai. *Conducting a Penetration Teston an Organization*, 2020 (accessed February 29, 2020). https://www.sans.org/reading-room/whitepapers/auditing/conducting-penetration-test-organization-67.

[31] Deloitte GmbH Wirtschaftsprüfungsgesellschaft. *Autonomes Fahren in Deutschland - wie Kunden überzeugt werden*, sep 2016. https://www2.deloitte.com/content/dam/Deloitte/de/Documents/consumer-industrial-products/Autonomes-Fahren-komplett-safe-Sep2016.pdf.