DEGREE PROJECT IN ,
FIRST CYCLE, 15 CREDITS
STOCKHOLM, *SWEDEN 2019*

A Study of Vulnerabilities and Weaknesses in Connected Cars

KORAY MUSTAFA KAYA

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

A Study of Vulnerabilities and Weaknesses in Connected Cars

KORAY MUSTAFA KAYA

Bachelor in Computer Science
Date: June 30, 2019
Supervisor: Robert Lagerström
Examiner: Örjan Ekeberg
School of Electrical Engineering and Computer Science
Swedish title: En studie av sårbarheter och svagheter i uppkopplade bilar

Abstract

Security vulnerabilities in connected cars can have devastating consequences. For this reason we compiled and analyzed vulnerabilities in connected cars using empirical data to gain an understanding of the security issues in the automobile industry. The data is gathered from the U.S. National Vulnerability Database (NVD) and analyzed with the help of the CVSS system and the CVE and CWE databases. 183 reports were found from the company Qualcomm and 28 reports were found from the rest of the industry. Qualcomm was analyzed seperately to avoid skewed results. Exploitability and impact trends of the vulnerabilities were analyzed and we found that the vulnerabilities generally were highly exploitable and had an high impact according to CVSS standards. The CWE classifications of the vulnerabilities were also analyzed. We found that the most common weaknesses among the major car companies were Protection Mechanism Failure, Information Exposure, Improper Restriction of Operations within the Bounds of a Memory Buffer and Improper Input Validation. The most common weaknesses for Qualcomm components were Improper Restriction of Operations within the Bounds of a Memory Buffer, Improper Input Validation, Improper Access Control, NULL Pointer Dereference, Improper Validation of Array Index and Information Exposure. Looking deeper into the vulnerable components we found that 47% of the vulnerabilities were in the Infotainment system and 39% were in the Telematics Control Unit.

Sammanfattning

Sårbarheter i uppkopplade bilar kan få allvarliga konsekvenser. Syftet med denna rapport är att sammanställa och analysera sådana sårbarheter genom empirisk data för att få en ökad förståelse av säkerhetsproblemen inom bilindustrin. Data samlades in från NVD (the U.S. National Vulnerability Database) och analyserades med hjälp av CVSS (Common Vulnerability Scoring System) och databasen för CWE:er (Common Weakness Enumeration). Totalt analyserades 211 kända sårbarheter. Varav 183 stycken kommer från företaget Qualcomm och resterande 28 från andra delar av bilindustrin, därför analyserades Qualcomm separat för att undvika förvrängda resultat. Enligt CVSS-klassificeringen var både exploitability och impact högt för många av de sårbarheter som analyserades. För bilsårbarheter generellt (exkluderat Qualcomm) var de mest förekommande svagheterna Protection Mechanism Failure, Information Exposure, Improper Restriction of Operations within the Bounds of a Memory Buffer and Improper Input Validation. Medan de mest förekommande svagheterna i Qualcomm-produkter var Improper Restriction of Operations within the Bounds of a Memory Buffer, Improper Input Validation, Improper Access Control, NULL Pointer Dereference, Improper Validation of Array Index and Information Exposure. 47% av svagheterna relaterade till Infotainment-systemet och 39% till Telematic Control Unit (TCU).

Contents

Chapter 1

Introduction

Connected cars is the next big step in the evolution of automobiles. In the past decade most of our devices have been connected to the internet and this advancement has started to spread to cars as well. Smartphone app controlled cars and cars with internet connected infotainment systems are the new buzzwords in the industry. The connected car has humble beginnings, the first connected car was launched 1996 with an emergency call system and the industry has grown ever since. With the recent trend of highly technological cars such as Tesla's different models this growth has picked up in speed.

Modern technologies can come with many advantages in terms of safety, convenience and vehicle efficiency. The modern car has infotainment systems with bluetooth and/or cellular connection capabilities for entertainment. Beyond entertainment, with the help of sensors and electrical control units the car can not only help the driver but even prevent life-threatening accidents [1].

However, the new technology also comes with new types of threats and challenges that the auto industry has not dealt with before. Numerous automobile attacks have been demonstrated in previous research [2, 3, 4, 5] where these new connectivities have been proven to have a lot of weaknesses where even life threatening accidents can be caused remotely. There is also discussion surrounding how attacks can look in the future [6] concerning these connectivities.

Vulnerabilities have proven to put end users at risk and cause serious damage to the manufacturers. They naturally lead to a loss of revenue for the car manufacturers because of necessary safety recalls. E.g. Fiat Chrysler recalled 1.4 million cars[1] the year 2015 after a demonstration of a Jeep getting hacked.

[1]BBC News, "Fiat Chrysler recalls 1.4 million cars after Jeep hack.", Mar. 2019. www.bbc.com/news/technology-33650491.

Certain car companies like Tesla have employed a preventive strategy of remote updates[2] to eliminate the need to recall cars.

The industry has reacted to the demonstrated attacks by expanding the industry standards [7]. Based on these standards, new research [8, 9, 10] has emerged proposing authentication protocols for the CAN bus which is one of the core weaknesses in automobiles [11, 12].

Certain companies such as Tesla even work together with the public through public bug and vulnerability hunts. These bug hunts use money prizes as incentive, the average payout being $2,110[3] in the past three months with a total of 336 vulnerabilities been found.

For the proposals of a future vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) [13, 14, 15] to be possible a cyber safe foundation must first be established.

1.1 Problem Statement

To be able to make meaningful progress in the automobile security industry we must collect key data and gain an overview of what the problems are. With this study we want to examine the properties of known vulnerabilities such as exploitability and impact to measure the urgency of this issue. We also want to identify weaknesses to gain an understanding of what kind of security problems there are in connected cars that we have to deal with. Finally we want to attribute these weaknesses to certain components so we can pinpoint where the weaknesses are concretely so we know where automakers must focus their security efforts. The questions we want to answer are:

- What are the properties of known vulnerabilities in connected cars?
- What weaknesses are the vulnerabilities related to?
- Which components can we attribute these weaknesses to?

1.2 Scope

The primary focus of this thesis is to evaluate the most common vulnerabilities found in connected automobiles. The vulnerabilities being observed are

[2] Tesla, "Software Updates." Mar. 2019. www.tesla.com/support/software-updates
[3] https://bugcrowd.com/tesla (accessed Mar. 2019)

related to software, hardware and firmware. The vehicles that are being considered are exclusively commercial passenger cars The metrics used to describe the severity of vulnerabilities is the Common Vulnerability Scoring System version 3.0 (CVSS v3.0).

It is not in the scope of this thesis to perform attacks or simulations of attacks, nor is it in the thesis scope to model attacks. The research is conducted through publicly available reports of attacks and vulnerabilities.

Chapter 2

Background

2.1 Types of Connectivity

A connected car is a car which has internet access and also sometimes different kinds of connectivities. These connectivities are referred to with the coding V2X (Vehicle-to-X) and there are 5 types of connectivities in the context of cars[1]:

- V2I "Vehicle to Infrastructure": Is usually a bi-directional connection where the car communicates information about itself to devices in the environment such as traffic lights, lane markers and similar devices that support a country's highway-system.
- V2V "Vehicle to Vehicle": The vehicle communicates information about itself and it's surroundings to the nearby vehicles. This information could be it's speed, position or current traffic conditions.
- V2C "Vehicle to Cloud": This connection would be a broadband cellular connection such as 3G, 4G or recently 5G. Through this connection the car can access multiple applications such as connected navigation, social media or music and movie streaming. This connection would likely be implemented together with an infotainment system.
- V2P "Vehicle to Pedestrian": The technology communicates with nearby personal mobile devices and gains information about their position and speed.

[1] http://autocaat.org/Technologies/Automated_and_Connected_Vehicles/

- V2X "Vehicle to Everything": A concept of an interconnected infrastructure where cars, boats, airplanes and the rest of the transport system is connected.

The different connectivities communicate through different connections such as 4G or Dedicated Short-Range Communications (DSRC). Safety-critical connectivities such as V2I, V2V or V2P are anticipated to use DSRC which has lower latency than other connections. The DSRC connection would operate in a 5.9GHz band. The V2C connection uses cellular connections such as 3G today.

2.2 Control Area Network (CAN)

To gain a comprehensive understanding of a path of attack in a connected vehicle a fundamental knowledge of the anatomy of such a vehicle is required. Thus, an overview of connected components and their functions in a car will be provided in this section.

The CAN bus is a standardized vehicle bus that lets sensors and microcontrollers communicate. The modern car has many sensors such as temperature sensors, tilt pressure sensors and many more. These sensors are used to control different functions in the car through up to 70 electronic control units (ECU) such as airbags or steering brakes[2]. Modern ECUs connected to drive-by-wire systems can have even more luxurious functions such as automatic parking or automatic braking if something is in front of the car in close proximity.

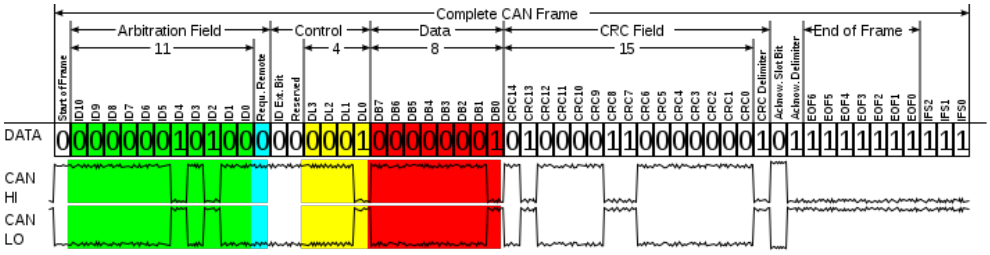

Figure 2.1: A simple CAN bus protocol

A single CAN bus like the one above has many security flaws. The ECUs listen to messages which have certain arbitration codes, so in theory any sensor can interact with any ECU. If we put into consideration a case where an

[2]CSS Electronics, "CAN Bus Explained - A Simple Intro", Mar. 2019 www.csselectronics.com/screen/page/simple-intro-to-can-bus

attacker can interact with the CAN bus we quickly realize that the attacker can control many components of the car by spoofing messages - even critical ones such as steering, braking or throttle. Modern cars may have multiple CAN buses for different services[3], seperating services. Communication between the CAN buses are made through a firewall.

2.2.1 Drive-by-Wire

Cars have traditionally carried out vehicular functions through mechanical linkages but this has changed in the latest years. Functions such as steering, throttle, shifting, braking and parking have been replaced by electronic systems. The drive-by-wire systems that are most widespread today are throttle, parking and shifting. The less common systems are braking and steering. Drive-by-wire braking and steering is however commonly implemented alongside traditional systems, this is referred to as an hybrid system.

Drive-by-wire systems bring many benefits to the driver but also security weaknesses as demonstrated by Miller et al. [16] who managed to control a car programatically in lower speeds.

2.3 Infotainment

Infotainment systems are the next evolution in both in-car information and entertainment. Infotainment systems in cars are becoming more common. They can fulfill their original purpose of playing the radio and giving information such as time, date and temperature, but also much more. A big advancement in new infotainment systems is the connectivity; while the old systems was connected to radio centrals, the new systems are utilizing V2C connections. The modern infotainment system is also connected to the CAN bus to carry out tasks within the car or gather information about the car[4].

[3] Ariel Nuñez, "An Introduction to the CAN Bus: How to Programmatically Control a Car.", June 2017. www.news.voyage.auto/an-introduction-to-the-can-bus-how-to-programmatically-control-a-car-f1b18be4f377.

[4] Anshul Saxena, "Everything You Need to Know About In-Vehicle Infotainment Systems", May 2019, www.einfochips.com/blog/everything-you-need-to-know-about-in-vehicle-infotainment-system/.

Figure 2.2: Tesla Models S's infotainment system
Source: https://www.flickr.com/photos/tecdencias/7212515632

Modern infotainment systems require complex hardware and software to run which of course bring vulnerabilities with them. Chips and operating systems are not perfect - they have bugs, errors and vulnerabilities. The infotainment systems vulnerabilities can be exploited to access the cars CAN buses which in turn means that an potential attacker can control parts of the car. This was demonstrated by Miller et al. [16]. The attack vector in this case was through the Uconnect infotainment system and affected the CAN bus.

Infotainment systems are connected to the internet, and naturally this can lead to malicious interactions. It is possible to download malicious files or executing malicious code thorugh the V2C connection. A remote attacker can through the cloud exploit vulnerabilities in the infotainment system and in theory remotely control the car. This gives attackers a brand new interface to work on. Such an attack was demonstrated on an Tesla model S by the Keen Security Lab of Tencent[5]. Attacks have previously been limited to local attack vectors, but it is now possible to pull of complex remote attacks.

[5]Keen Security Lab of Tencent, "Car Hacking Research: Remote Attack Tesla Motors", Sept. 2016, https://keenlab.tencent.com/en/2016/09/19/Keen-Security-Lab-of-Tencent-Car-Hacking-Research-Remote-Attack-to-Tesla-Cars/

2.3.1 Qualcomm Chips

Qualcomm is a system-on-chip manufacturer that has produced chips for smartphones in the past and has now expanded their business to automobile infotainment systems. Qualcomm's Snapdragon 620A and Snapdragon 820A[6] system-on-chips are specially developed to be used for infotainment systems in cars. The chipsets hold the operating system for each component and govern the interactions within. Among the features for the 820A are connectivities such as an LTE-A connection to the internet and a local device connection.

2.4 Telematics Control Unit (TCU)

While Infotainment systems are more focused on entertainment, the Telematics Control Unit (TCU) is more focused on lower level information. The TCU can be seen as a "black box", it tracks vehicle diagnostics, driving behaviour, location and other information about the car. It is not unusual for car manufacturers to also produce apps which the driver can use to get important information about their cars. With the help of information such as tire pressure and other diagnostic information the driver can easily see if it is time for an maintenance. The TCU is connected to the CAN bus which means that it can cause damage in a potential compromise[7].

2.5 Vulnerability Sources and Evaluation

The data in this study was gathered from NVD and the vulnerabilities were evaluated by their CVSS metrics and CWE classifications.

2.5.1 The National Vulnerability Database (NVD)

All the data in this study was gathered from the U.S. National Vulnerability Database (NVD). NVD is a U.S. government repository for vulnerabilities which uses the Security Content Automation Protocol (SCAP)[8]. In this thesis the CVSS, CWE and CVE aspects of this protocol will be relevant. NVD was

[6]https://www.qualcomm.com/products/snapdragon-820-automotive-platform

[7]Embitel, "Technology Behind Telematics Explained: How does a Vehicle Telematics Solution Work?", Dec. 2018, www.embitel.com/blog/embedded-blog/tech-behind-telematics-explained-how-does-a-vehicle-telematics-solution-work

[8]https://nvd.nist.gov/

proven to be the most consistent compared to 4 other vulnerability databases by Johnson et al. [17].

2.5.2 Common Vulnerabilities Scoring System (CVSS v3.0)

When communicating the severity of computer system vulnerabilities there is a need for standardized tools to represent security risks in a way that enables organizations to gain an overview and perform risk assessment after degree of severity, this is where the CVSS system comes into play. The Common Vulnerabilities Scoring System (CVSS) is an open standard for evaluating the severity and potential impact of vulnerabilities found in computer systems.

CVSS provides a range of metrics and a scoring system ranging from 0.0 to 10.0, the latter being more severe. The scoring system is referred to as the *base score*. The score is calculated from several metrics which are predefined in the CVSS specification. CVSS has multiple versions with different scoring standards. However, only CVSS[9] version 3.0 is considered in this thesis.

The metrics used when calculating the base score are divided into three groups. The first group is the base metric group, it is the most central group and also the only mandatory group. The two other groups are the temporal metric group and the environmental metric group. These two groups are not commonly used in the evaluation of vulnerabilites in connected cars so it will not be necessary to understand them any deeper.

The metrics are expressed through values that makes sense in the context. E.g. the impact metric can acquire a value among the values of None, Low or High while the Attack Vector metric can have the values Network, Adjacent, Local or Physical. These values are later referred through their abbreviations.

Naturally we have to question the credibility of the assigned CVSS scores. Johnson et al. [17] studied the credibility of the CVSS V2 system and found that it can be trusted. Johnson et al. states that even though the study evaluated the V2 scores, it is possible to draw conclusions on the CVSS system as a whole - including version 3.0. The study concluded that the Confidentiality, Integrity and Availability (CIA) impact metrics were the most accurate. The least accurate metric was the Attack Access metric according to the study. Johnson et al. further concluded that the NVD database which will be used in this study was the most consistent among the 5 databases.

[9] https://www.first.org/cvss/specification-document

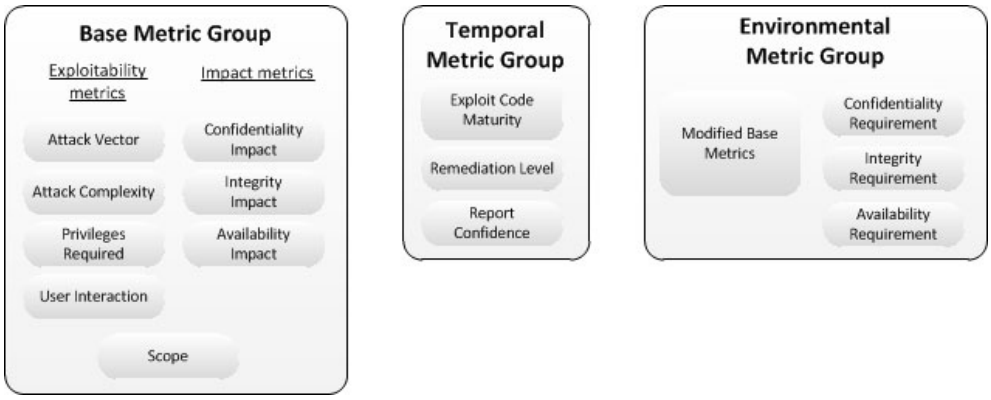

Figure 2.3: The CVSS metric groups (Taken from the CVSS website)

Exploitability Metrics

Among the base metrics, the metrics regarding the characteristics of the vulnerable component belong to the subcategory Exploitability Metrics. The exploitability metrics together measure the requirements for the vulnerability to be exploited.

Attack Vector (AV)
The Attack Vector describes the path from which an attacker can gain access to a vulnerable component. The full path is not necessarily specified, but the it's type is.

The possible values for the Attack Vector are **Network (N)**, **Adjacent (A)**, **Local (L)** or **Physical (P)**. If the exploitation was done remotely i.e. through the public Internet, the Attack Vector value would be Network. An adjacent vulnerability means that the device is in the same network stack, i.e. the exploit could be made through bluetooth or through a logical network. A local vulnerability means that the vulnerable component is not bound to a network stack and that the hacker has to perform the exploit locally. Local can mean that the attacker has to login to the system. A physical vulnerability means that the attacker needs physical access to the component. An example of a physical attack would be the Evil Maid Attack.

Attack Complexity (AC)
The attack complexity metric measures the amount of pre-conditions needed for the vulnerability to be exploitable. Only the conditions which are beyond the attackers control are put into account here.

There are only two values for the Attack Complexity metric, **Low (L)** and

High (H). A low attack complexity means that the exploit is fairly repeatable and in some cases that there does not even exist required pre-conditions. Respectively, a high attack complexity means that there are many required pre-conditions for the vulnerable component to be exploited and that the attack is not highly repeatable.

Privileges Required (PR)
Priviliges Required is the metric used to describe the level of privileges the attacker must have to exploit the vulnerability.
The different privileges the attacker can have are **None (N)**, **Low (L)** or **High (H)**. Some vulnerabilities can be exploited with no privileges at all, while others need some low level of privilege such as an average user privilege and other might even need a administrative level of privileges.

User Interaction (UI)
Some exploits require the user to either make a mistake or unknowingly assist the attacker in some way. A classic example of this is when a user unknowingly installs a virus.
The User Interaction attribute can either be **None (B)** or **Required (R)**. None means that the vulnerability can be exploited without any user interaction and Required means of course that user interaction is required.

Scope

Scope (S)
The scope metric of an vulnerability tells us how far potential exploits can affect the system. The scope metric has two possible values: **Unchanged (U)** or **Changed (C)**. The values are tell us if the exploit can affect systems beyond the scope or privileges of the vulnerability.

The CVSS 3.0 documentation gives us two great examples to demonstrate the values, the first one describes a situation where the scope is changed. Imagine a virtual machine running on windows, we have 2 authorities in this case: one is the software that enforces privileges for the virtual machine and it's user, the other is the OS which enforces the privileges for the host and the virtual machine. If an exploit within the virtual machine can corrupt or affect the host we would say that the scope of the vulnerability is changed. This is because the corruption was outside the initial authority - the virtual machine.

It is sometimes hard to see if the scope is changed or not; if a software such as Microsoft Word would corrupt other files within the OS the scope would remain unchanged since the authority in this case is the OS and the corruption

is still within that authority.

Impact Metrics

The Impact Metrics are used to measure the effects of an exploitation of the vulnerability.

Confidentiality Impact (C)
The Confidentiality Impact metric measures to which level the confidentiality of certain resources has been breached.

The Confidentiality Impact metric can have the values **None (N)**, **Low (L)** or **High (H)**. The None value means that there was no loss of confidentiality. The Low value means that there was some loss of confidentiality, i.e. some restricted information was obtained but the kind of information or the amount of information was constrained and the information did not cause a serious loss to the impacted component. If the confidentiality loss is High then it means that there was a serious loss of confidentiality. This could mean that the attacker has stolen a administrators password or private information that can lead to serious consequences.

Integrity Impact (I)
Integrity is the reliability of information and in a potential attack this reliability can be jeopardized.

Similar to the confidentiality metric, the integrity impact can be **None (N)**, **Low (L)** or **High (H)**. If no files have been lost of modified we can say that the Integrity Impact was none. If some data was modified but the attacker lacked control over the modifications or the amount of modifications were contrained we will say that the Integrity impact was low. Finally, if the attacker could successfully modify any/all restricted files and these files which were maliciously modified present a direct and serious consequence to the impacted component we say that the Integrity Impact was high.

Availability Impact (A)
The last of the impact metrics is the Availability Impact metric. While the previous Impact Metrics concerned the data and the damage to the data, the Availability Metric focuses on the vulnerable component itself. The Availability Metric tells us whether or not the component is still useable, this could for example be a service that is still online or offline due to the attack.

The values for the Availability Impact metric are **None (N)**, **Low (L)** or **High (H)**. A value of None means that there is no impact to availability. A low value means that the impacted component has reduced performance or interruptions in resource availability. This means that the attacker does not have the ability to fully deny the users from using the service or component. Lastly, if the value is high we can assume that the attack either can fully deny access to resources or that the little deniability the attacker has presents serious consequences.

Scoring

The metrics all have corresponding fixed numerical values which are used to calculate a score. The numerical values range from 0 to 1 in the base metrics. A value closer to 1 signifies a higher severity, for example a low attack complexity has a fixed value of 0.77 and a high attack complexity has a fixed value of 0.44.

The following algorithm is used to calculate the base score,

if $ISC <= 0$ **then**
| $Basescore = 0$
else if *Scope Unchanged* **then**
| $Basescore = Roundup(Minimum[(ISC + ESC), 10])$
else if *Scope Changed* **then**
| $Basescore = Roundup(Minimum[1.08 \times (ISC + ESC), 10])$
end

Where,

$$ISC = 1 - ((1 - Confidentiality) \times (1 - Integrity) \times (1 - Availability))$$

$$ESC = 8.22 \times AttackVector \times AttackComplexity \times PrivilegeRequired \times UserInteraction$$

2.5.3 Common Weakness Enumeration (CWE) and Common Vulnerabilities and Exposure (CVE)

The Common Weakness Enumeration[10] (CWE) is a system used for categorization of software vulnerabilities. Companies constantly try to identify and

[10] https://cwe.mitre.org/about/index.html

analyze software vulnerabilities but often come across the problematic lack of definitions and structure. CWE provides it's users with proper definitions for weaknesses and well documented information about them. This makes it possible for companies to publish and discuss vulnerabilities on an common platform where everyone has an understanding of what the definitions and terms represent. This also makes it easier to put a finger on what a vulnerability specifically is and which weakness it is due to.

Common Vulnerabilities and Exposures (CVE) is a dictionary where all vulnerabilities are saved in unique entries and have unique CVE ids. Many different databases contribute to the dictionary, so does NVD. Each NVD entry is accompanied by a CWE classification and an CVE entry.

2.6 Related Work

Välja et al. [18] did a similar study on embedded systems in power networks using the NVD database. They used a similar method with data from NVD, CWE and ICS CERT to identify weaknesses in embedded systems in power networks which likely gave them a wider set of results. Välja et al. successfully collected 224 vulnerabilities in total. Similar to this study a big majority of their vulnerabilities were reported by a two companies. However, Välja et al. chose not to seperate their data, meaning that their results were skewed towards the two dominant companies who made up 205 of the 224 results. Välja et al. found that the most common weaknesses in embedded systems in power networks were Improper Input Validation, Cryptographic Issues, Improper Restriction of Operations within the Bounds of a Memory Buffer and Resource Management Errors, some of which we also found to be common in connected cars.

While our study relies on databases and reports, there is also a variety of studies [5, 4] that use a different method, they investigate a large amount of automobiles. Oka et al. [5] describes how an weakness current bluetooth pairing protocols can give the attacker access to the CAN system or other private information. They then survey how common the weakness is by fuzzing bluetooth passwords for over 100 vehicle IoT devices comprising of different infotainment systems and OBD-2 port dongles with bluetooth capabilities. They find that roughly 40% of external infotainment systems and 90% of OBD-2 port dongles allow for an attacker to pair with the device. By investigating the cars themselves they have the advantage of being independent of manufacturer reports. While the work is more cumbersome, the results are way less skewed towards any company which gives a better picture of the industry.

Currie [19] presented weaknesses in the CAN bus. In his paper Currie argues that this system which was developed more than 30 years ago is too vulnerable for todays standards and demonstrates the inherent flaws in the CAN bus by hacking a 2011 Honda Civic with the help of a laptop and a few inexpenssive wires. Koscher et al. [11] experimented with both lab tests and road tests. In the lab tests, hardware was extracted from certain vehicles and experimented with under lab conditions. Through simple Packet Sniffing, Probing and Fuzzing they were able to determine how the ECUs communicated and could easily reproduce data packets to control certain ECUs. Since the range of valid CAN packets is so small they could easily just iteratively test random data packets (fuzzing) to control or disrupt the vehicle. The data packets were then tested on a live car to see if the attacks could be reproduced in a real life situation, which they could. From these and other research [3, 12, 16] is easy to conclude that the CAN protocol is one of the main flaws in modern cars. Studies concerning the CAN bus repeatedly underline the fact that it was developed in a time where the automobile was assumed to be a closed system, which it is not today.

Finally, there is also a lot of studies researching how the security issues concerning the CAN in modern cars can be solved. Many of these studies focus on solving the lack of authentication in the CAN [9, 8, 10] and building new protocols that will authenticate messages to prevent spoofing. There is also an interesting study by Zweck et al. [20] that considers Automotive Ethernet as the future for in-vehicle networks. Zweck et al. proposes an embedded firewall for this new technology which will restrict non-authorized access to the network.

Chapter 3

Method

Before gathering the data for this study we researched articles and reports on security breaches related to connected cars to gain an overview of the situation today. The first one that stood out was the infamous hack featured on Wired[1] which was accompanied by an in-depth paper by the hackers Miller et al. [16]. By further researching articles such as this we were able to make out the underlying attack vectors and related components. The components we identified were CAN, TCU, Infotainment and Drive-By-Wire.

From reading previous studies on similar topics we were able to identify popular databases and the commonly used tools for vulnerability analysis. Välja et al. [18] informed us about the NVD and CWE databases which were used in this study. CVSS was also used by Välja et al. which together with paper *Can the Common Vulnerability Scoring System be Trusted? A Bayesian Analysis* by Johnson et al. [17] convinced us that CVSS can be trusted.

Additional vulnerability databases were considered such as X-Force, OSVDB, CERT-VN, and Cis. Having results from different databases could provide more information about the vulnerabilities or more vulnerabilities. Using multiple vulnerability databases would be preferable but due to time constraints we chose to use one. NVD was chosen due to the paper of Johnson et al. [17] where the database was proven to be the most consistent among the 5.

The first step to gather data in this thesis was to gather the names of the biggest car companies. We gathered the names of the 25 biggest car companies in Sweden by cars sold[2]. We also used keywords that were gathered when studying car components. A full list of the used search terms can be found

[1] Andy Greenberg, "Hackers Remotely Kill a Jeep on the Highway - With Me in It." Nov. 2018, www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/

[2] Mattias Rabe, "Sveriges mest sålda bilmärken – här är listan.", Apr. 2017, www.teknikensvarld.se/sveriges- mest-salda-bilmarken-har-ar-listan-316040/

in the appendix. We made sure to remove duplicates since different cars can contain the same component and therefore share a vulnerability, which still counts as one vulnerability. Only the results with a CVSS V3.0 scoring was used.

The next step in our research was to compile and analyze the data. The data was highly skewed due to the large amount of reports from Qualcomm so we chose to analyze this manufacturer seperately. We extracted the CVSS metrics and CWE classes from the data and compiled them into different forms of statistics which can be analyzed. The CVSS metric data was compiled into frequency diagrams which let us see how the values for the metrics were distributed among all the vulnerabilities.

The CWE statistics were used to identify the most common weaknesses among the vulnerabilities. The CWE classes were compiled into frequency charts and the top 25% of the vulnerabilities were picked and further analyzed to get more information of the main weaknesses.

We also extracted the sub-components that the vulnerabilities could be directly attributed to and the corresponding chips. The vulnerabilities were manually analyzed to identify the vulnerable component and categorize the vulnerabilities. 3 categories were formed: Infotainment, TCU and Other. With this data we created a pie chart to identify where the vulnerabilities were mainly located and the distribution among components. The vulnerable chips in each category were broke down and displayed in a flow chart.

Chapter 4

Results

We gathered totally 211 vulnerabilities from the NVD vulnerability database, 183 of which concern Qualcomm chipsets. Because of the overwhelming reports from Qualcomm we chose to analyze the results seperately by splitting the data into 2 sections - the results from companies (excluding Qualcomm) and the Qualcomm results.

To gain an useful overview of what kinds vulnerabilities and weaknesses existed in automobile components we summarized the data into diagrams.

4.1 Results ex. Qualcomm

4.1.1 Exploitability Metrics

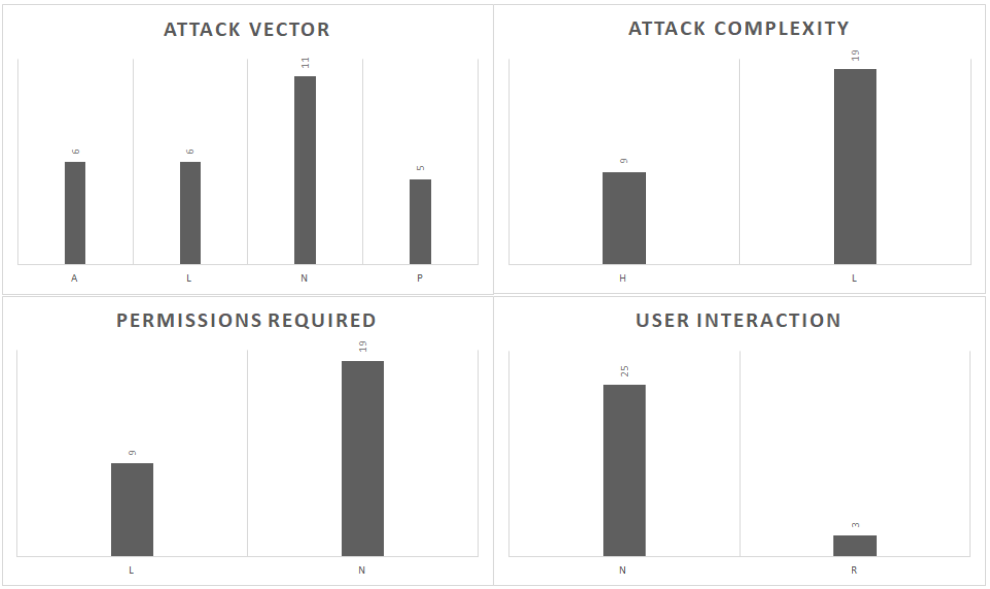

Figure 4.1: Frequenct chart of the Exploitability Metrics for the 28 vulnerabilities the different companies reported to NVD ex. Qualcomm.

The Exploitability Metrics give us information about how exploitable the vulnerabilities were. We can see that the Attack Vector was generally through a Network, the Attack Complexity was Low, there was no permission required and user interaction was not required either. This means that the attacks could be carried out remotely with easily repeateable methods without any previous permission or any user interaction. The scope value was Unchanged in all cases.

4.1.2 Impact Metrics

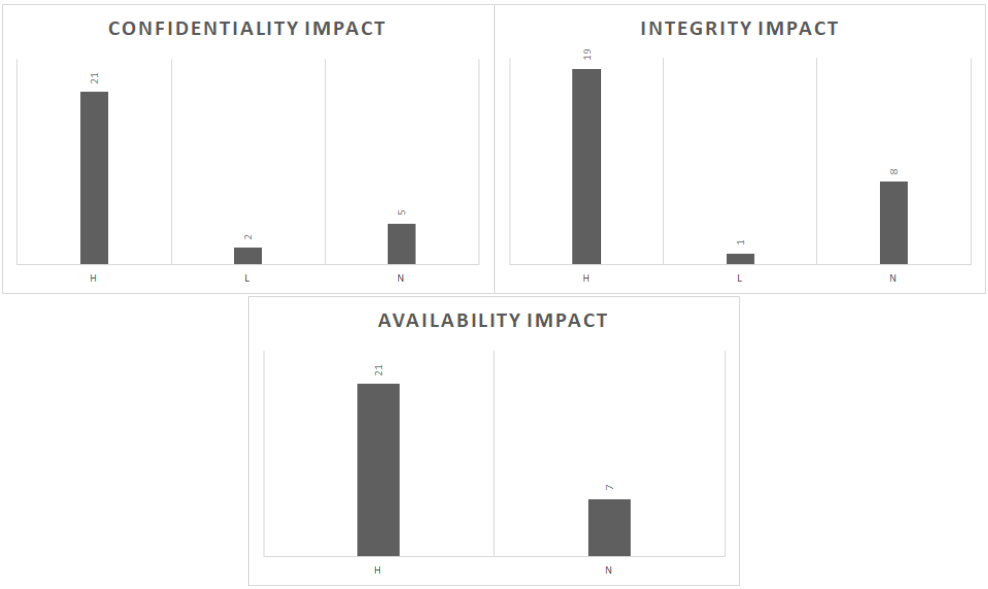

Figure 4.2: Frequency chart of the Impact Metrics for the vulnerabilities ex. Qualcomm.

The impact metrics tell us what kind of damage the vulnerabilities caused or could have potentially caused. We can see that generally the confidentiality, integrity and availability impact are all high (H). This means that generally confidential information was accessed, the hacker could successfully modify important information and control the availability of the component.

4.1.3 Weaknesses

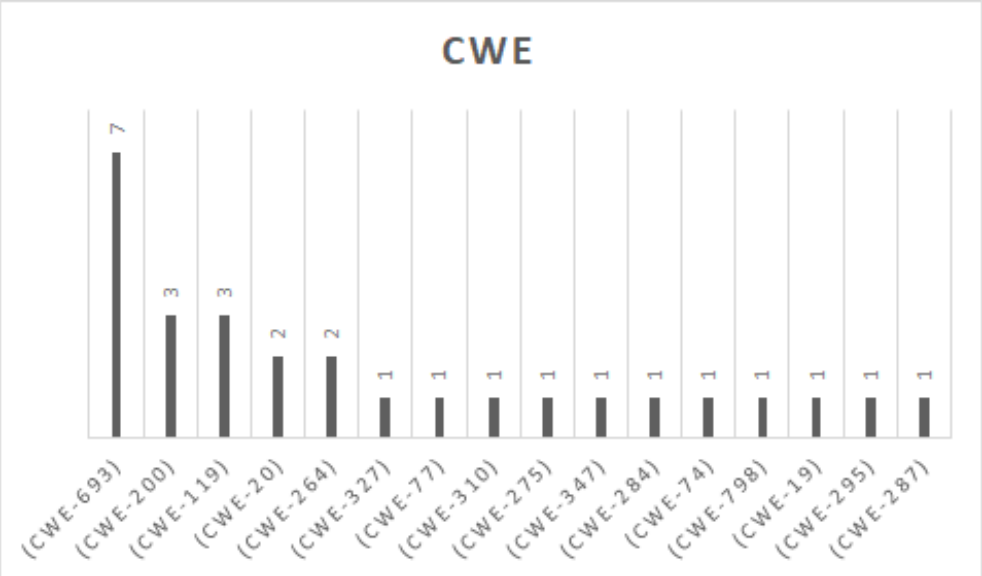

Figure 4.3: Frequency chart for the weaknesses associated with the vulnerabilities.

The top 25% of the CWE weaknesses in Figure 4.3 are described below,

CWE ID	Count	Description
CWE-693	7	Protection Mechanism Failure
CWE-200	3	Information Exposure
CWE-119	3	Improper Restriction of Operations within the Bounds of a Memory Buffer
CWE-20	2	Improper Input Validation

The CWE descriptions can be hard to understand but they are quite sensible when we break them down. A protection mechanism failure means that the protection mechanism which the developer installed is either missing, broken och bypassed by the hacker. An information exposure can mean that certain information was not properly stored e.g. not encrypted and was exposed to hackers. Improper Restriction of Operations within the Bounds of a Memory Buffer occurs with some low level languages such as C where you can directly access the memory space and unless restricted this can let the hacker access or change certain information. An Improper Input Validation means that the

software is incorrectly validating or invalidating input which can affect the data or control flow of a program.

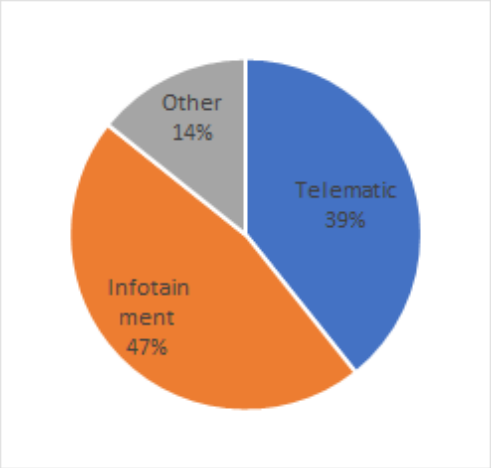

Figure 4.4: Pie chart for distributions of vulnerabilities by component.

Figure 4.4 gives us a deeper look into which component we can attrbute the vulnerabilities to. 47% of the vulnerabilities were in the infotainment system, 39% in the TCU and 14% were attributed to other components. The 4 vulnerabilities in the Other category was an airbag algorithm vulnerability in an undisclosed car model, 2 of the vulnerabilities was in a self-driving car solution and the last one was a cryptography issue in Tesla's Passive Keyless Entry and Start (PKES) system. It is worth mentioning that the Availability metric was None for both the airbag vulnerability and the PKES vulnerability.

4.1.4 Reporting

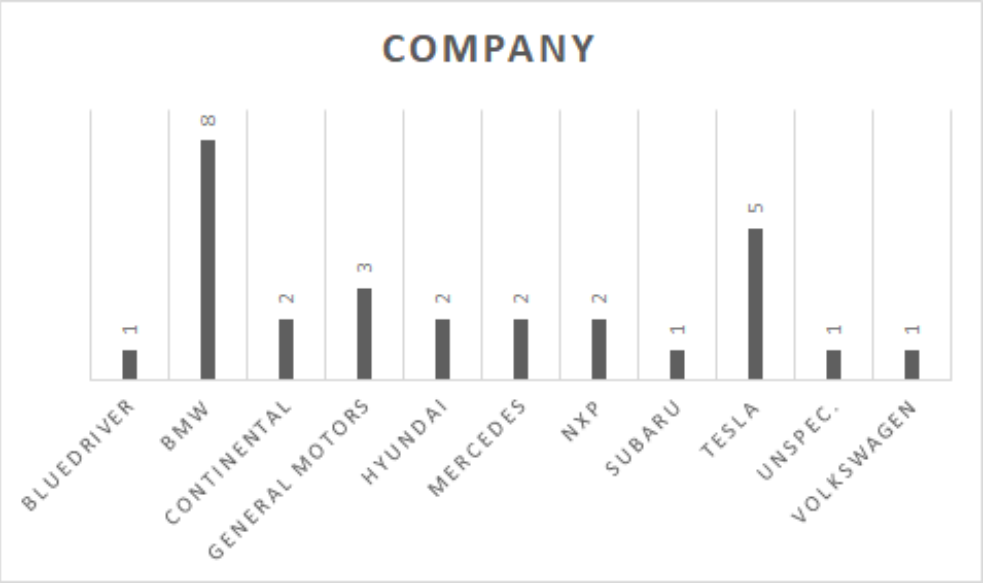

Figure 4.5: Frequency chart for reports by company.

The data is almost evenly distributed among the car companies BlueDriver, BMW, Ford, General Motors, Hyundai, Mercedes, Nissan, NXP, Subaru, Tesla and Volkswagen. BMW stands out with 8 entries compared to the average of 2.7 entries for the rest of the car manufacturers. There was also 1 vulnerability attributed to an undisclosed vehicle. The rest of the results are reported by Qualcomm and are analyzed in the next section.

4.2 Qualcomm

4.2.1 Exploitability Metrics

Figure 4.6: Frequency chart of the Exploitability Metrics for the 183 vulnerabilities for Qualcomm's Snapdragon Chipset from NVD.

The exploitability metrics results give us a good overview of what kind of vulnerabilities are present in the Qualcomm Snapdragon 802A chipset and how complex they are. We can see that there is a high amount of network vulnerabilities, the attack complexity is generally low, permission is usually not required and user interaction is very rarely required. The Scope metric is not displayed since the values were homogenous - the scope was Unchanged (U) in all cases. You might also notice that there are no physical attack occurences, this makes sense since the Snapdragon chipsets are exclusively used for "next-gen" infotainment; the system has strong network capabilities.

4.2.2 Impact Metrics

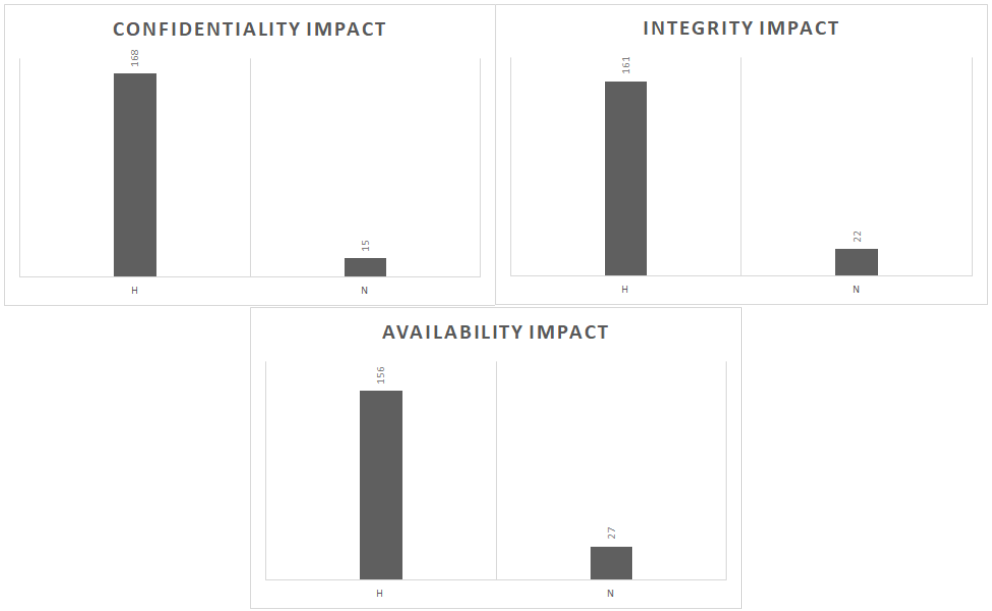

Figure 4.7: Frequency charts of the Impact Metrics for the Qualcomm Snapdragon 802A chipset.

The impact metrics tell us what kind of damage the vulnerabilities caused or could have potentially caused. We can see that generally the impact was high in all cases. The results are similar to those in the previous section; the attacker could generally read and change important data and control the components availability.

4.2.3 Weaknesses

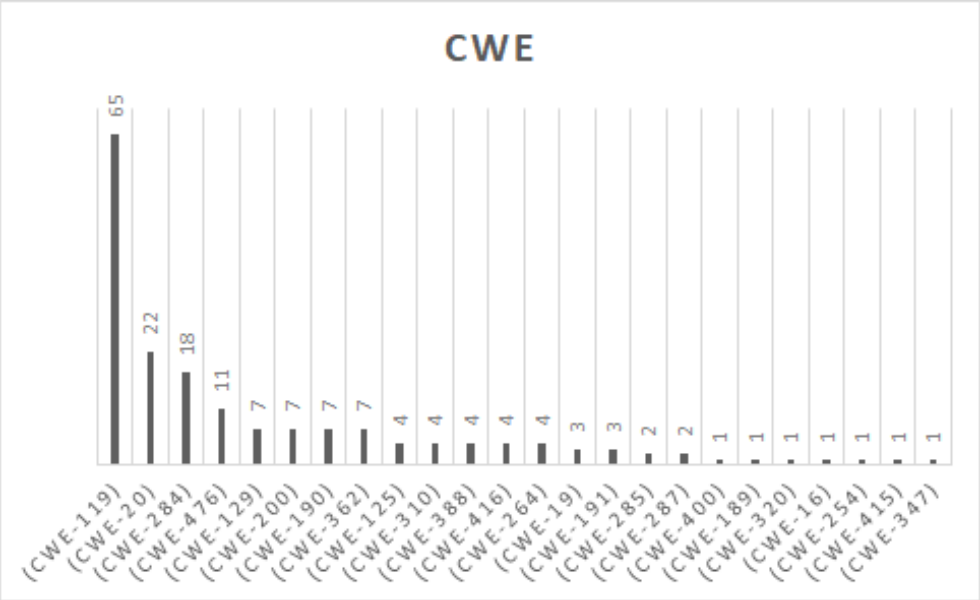

Figure 4.8: Frequency chart for the CWEs corresponding to the vulnerabilities.

The top 25% most frequent CWE weaknesses are explained below,

CWE ID	Count	Description
CWE-119	65	Improper Restriction of Operations within the Bounds of a Memory Buffer
CWE-20	22	Improper Input Validation
CWE-284	18	Improper Access Control
CWE-476	11	NULL Pointer Dereference
CWE-129	7	Improper Validation of Array Index
CWE-200	7	Information Exposure

The weaknesses in this section are similar to the **previous results** but we can notice that there are three new types of weaknesses here. Improper Access Validation means unauthorized actors can access the program. A NULL Pointer Dereference means that a pointer is unexpectedly null, leading to a crash. A Improper Validation of Array Index means that the software is trying to access an array index which does not exist.

4.3 Component Breakdown

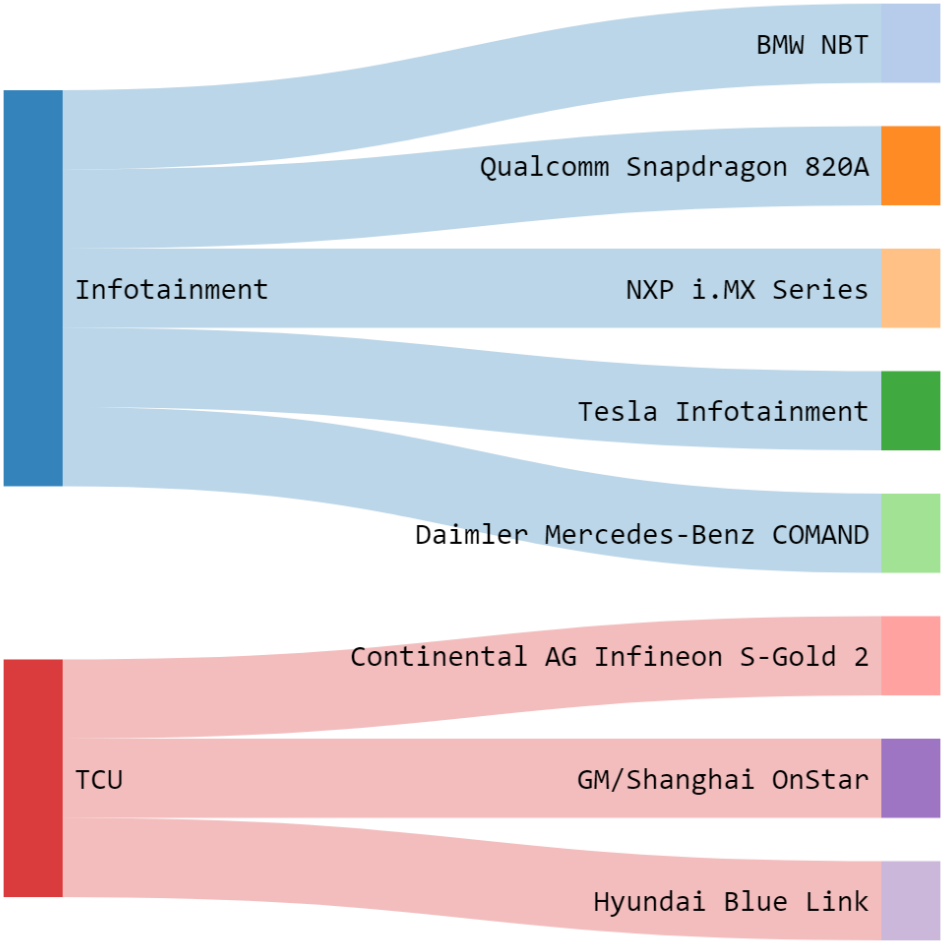

Figure 4.9: Flow chart breakdown of the components associated with vulnerabilities.

Figure 4.9 gives us an even deeper look and lets us further investigate the components which were vulnerable. The chipsets flowing from the Infotainment node are different chipsets that each hold their own operating system, used in different cars. The chipsets flowing from the TCU node are, naturally, different Telematic Control Units. These chips together accounted for all the vulnerabilities discovered in this study.

Only one of the vulnerabilties among the 189 Qualcomm vulnerabilties concerned the Snapdragon 602A (and also concerned the Snapdragon 820A), the rest concerned only the 820A.

Chapter 5

Discussion

5.1 The Vulnerabilties

The vulnerabilities show an alarming trend of high exploitability. Looking at the exploitability metrics for Results ex. Qualcomm we see that a majority of the attacks are through Network (N). 11 of the vulnerabilities have the Attack Vector value Network, 6 have Adjacent, another 6 have Local and 5 are Physical attacks. Connected cars seem to have a weakness against Network attacks compared to any other single attack vector.

Network attacks might be an issue due to its novelty in the automobile industry. The car industry has not dealt with cyber security in the past and one might even say that they are each others opposites. While the car industry has traditionally been unhackably mechanical, it is today implementing hackable technology. Being remotely hacked might have been unheard of in the past yet the industrys realization of this new threat will be quintessential for the future of the automobiles.

The automobile industrys ignorance to security is further reflected in the other exploitability metrics. 19 of 28 cases had low attack complexity, we could see this in Curries [19] paper where he hacked the car with relatively low attack complexity. Curries hack required no pre-conditions and could be executed with an average laptop and some inexpensive wires. We see the same numbers again in Permission Required where 19 of the cases did not require any permissions at all. 21 of the 28 cases required no user interaction either. The reason for the exploitability to be so high might be because of the car industry not expecting such an situation. If the exploitability was low or medium we could have argued that there was some effort put into cyber security by the car industry, but there seems to be no major resistance towards the hackers.

Similar to the exploitability metrics the impact metrics are also generally high. The confidentiality impact, integrity impact and availability impact were in most cases High (H). Not only are the vulnerabilities easy to exploit but also have serious consequences. The exact consequences of the reports are not known but we know that 86% of the attacks were in the Infotainment system and TCU, both of which are very likely to be connected to the CAN bus. With this knowledge we can assume that the attacker could cause an accident as demonstrated before. The automobile industry must realize the security flaws, especially in critical components such as the CAN bus.

5.2 The Weaknesses

The most common weakness for connected cars (ex. Qualcomm) was Protection Mechanism Failure (CWE-693). This means that the protection mechanism was bypassed, missing or malfunctioned. This weakness is especially hard to solve due to the life cycle of automobile software. Hacks and patches are many times a back and forth between hackers and developers. In traditional software we can expect that the vulnerability would be patched only for another vulnerability to be found and so on. We can not expect the same back and forth in connected cars since patching software is a major inconvinience as displayed in the case of Fiat Chrysler. Although, we might expect this in the future since it is probable for more car manufacturers to employ remote software updates since software is starting to play a bigger and bigger part in automobiles thus this software will undeniably need to be maintained with frequent updates. This might be something to look forward to if more smartphone component manufacturers such as Qualcomm turn to the automobile industry. With the similar in functions of smartphones and infotainment- and TCU systems we could expect to see some security measures in smartphones to be applied to infotainment- and TCU systems such as highly maintained and up-to-date universal operative systems. Android is an great example of such an OS.

The rest of the weaknesses are not weaknesses unique to the automobile industry. The Information Exposure weakness might be due to bad practices in the business or developers not following the companys guidelines. Encryption of information should be taken seriously. Weaknesses such as Improper Input Validation can be caused by human error or by ignorance. Softwares used in cars are complex which means that certain amounts of bugs should be expected due to the sheer amount of interdepent code.

5.3 The Components

The results reveal that there are known vulnerabilities attributed to some specific components in connected cars. We can see that a majority of the vulnerable components in our study are related to either the Infotainment system or the TCU, these systems are in turn connected to the CAN bus which explains why these vulnerabilities usually have an high impact on cars. Furthemore, only 4 (12%) of the vulnerabilities in Results ex. Qualcomm section are attributed to "Other" and 2 of these were not vulnerabilities which gave access to the CAN bus. Since the Availability impact of these 2 vulnerabilities were None (N) we can only assume that they did not have access to critical control units.

Qualcomm single handedly accounted for 87% of the results. The reason for this could simply be that they reported more of their vulnerabilities compared to other companies, but there might be another reason. When looking through Qualcomms NVD entries we see that the reported vulnerabilities applied for multiple chips, not only the Snapdragon 820A. The other chips in the reports were for other devices such as smartphones. My hypothesis is that the 820A chip shares software with other Snapdragon chips for other devices where vulnerabilities are more closely followed. This means that a vulnerability found in a smartphone can also be reported for the 820A. It makes sense that vulnerabilities are more likely to be identified in smartphones since they are more widely used and therefore gather more user feedback.

5.4 Lack of Reporting

If we compare the results to the used search terms we can notice that most of the researched car manufacturers are not among the results. This is most likely due to a lack of reporting, or due to that the vulnerabilities are being reported to different databases. The former reason can further be reflected upon, this lack of reporting can have many reasons.

Due to the rigid nature of software in cars it is understandable for manufacturers to keep some vulnerabilities unreported to protect their customers, but there could be other reasons as well. Reporting a vulnerability could mean that millions of customers are exposed to hackers. It is generally very hard to update software in cars. As with Fiat, in some critical cases car manufacturers can be forced to recall millions of cars with an economic impact to the company. However, we must also consider the possibility that some compa-

nies chose to not report their vulnerabilities with the purpose of maintaining an false image of security to their customers. Another reason for companies to not report vulnerabilities is simply ignorance. There might be an poor connection between the customer and the company causing a lack of feedback and reports. This is detrimental to the company and the car industry since the customers will suffer and the researchers will be unaware of the vulnerability.

When looking at the breakdown of the sub-components we can see that many car manufacturers use custom solutions. Tesla, GM, Hyundai, BMW and Mercedes-Benz all have their own software. The software might even been developed uniquely for a single model or series. A high amount of different softwares can be out in the market for a single manufacturer and this makes maintanenance a major inconvinience since they have to keep track of vulnerabilities in all the different softwares. Furthemore, the fact that different companies use different softwares for their cars eliminate the possibility of collaboration between manufacturers. Looking at smartphones we can find an operating system such as Android running on different hardware. Android is famously an open-source project which invites everyone to contribute. Vulnerabilities in such an software will naturally be easier to find due to the collective feedback and collaborative development. Employing an collaborative strategy could highly support the connected car industry in the future.

Chapter 6

Conclusions

The properties of the vulnerabilities were high exploitability and high impact. The most likely Attack Vector was proven to be the Network (N), but the other Attack Vectors were likely as well. The vulnerabilities have generally low attack complexity, no permission required and do not require user interaction. In a large majority of cases the confidentiality, integrity and availability impact of the attacks were high.

We also found that these vulnerabilities could be attributed to common weaknesses with the help of the CWE database. When we looked at the results (without the overwhelming Qualcomm reports) the main weaknesses we found by frequency was Protection Mechanism Failure. The second place was tied by Information Exposure and Improper Restriction of Operations within the Bounds of a Memory Buffer. The fourth most common weakness was Improper Input Validation.

The most common weaknesses we found in the Qualcomm vulnerabilities were Improper Restriction of Operations within the Bounds of a Memory Buffer, Improper Input Validation, Improper Access Control, NULL Pointer Dereference, Improper Validation of Array Index and Information Exposure.

A majority of the vulnerabilities were in the Infotainment system and the rest of the vulnerabilities were almost all in the Telematic Control Unit (TCU). Both the Infotainment system and the TCU is connected to the cars CAN bus which might have explained the overall high impact value of the vulnerabilities.

The vulnerabilities were found in the following chipsets for infotainment systems: BMW NBT, Qualcomm Snapdragon 820A, NXP i.MX Series, Tesla Infotainment and Daimler Mercedes-Benz COMAND. All except 4 of the rest of the vulnerabilities were in the following TCU systems: Continental AG

Infineon S-Gold 2, GM/Shanghai OnStar and Hyundai Blue Link.

Another conclusion we can draw from this study is the lack of reporting vulnerabilities in the automobile industry. Reporting of vulnerabilities is important for studies such as this one to be possible. Whether this lack of reporting is reasonable or not is up for debate and need its own research.

We must take into account that cyber security is something the automobile industry is new to, but we must also not forget that this is a very serious matter.

Bibliography

[1] Donna Glassbrenner. *An analysis of recent improvements to vehicle safety*. Tech. rep. 2012.

[2] Madeline Cheah et al. "Towards a systematic security evaluation of the automotive Bluetooth interface". eng. In: *Vehicular Communications* 9 (2017), pp. 8–18. ISSN: 2214-2096.

[3] Stephen Checkoway et al. "Comprehensive experimental analyses of automotive attack surfaces." In: *USENIX Security Symposium*. Vol. 4. San Francisco. 2011, pp. 447–462.

[4] Charlie Miller and Chris Valasek. "A survey of remote automotive attack surfaces". In: *black hat USA* 2014 (2014), p. 94.

[5] D.K. Oka et al. "Survey of vehicle IoT bluetooth devices". In: *Proceedings - IEEE 7th International Conference on Service-Oriented Computing and Applications, SOCA 2014*. Institute of Electrical and Electronics Engineers Inc., 2014, pp. 260–264. ISBN: 9781479968336.

[6] Yeongkwun Kim and Injoo Kim. "Security issues in vehicular networks". eng. In: *International Conference on Information Networking 2013 (ICOIN)*. 2013, pp. 468–472. ISBN: 9781467357401.

[7] J SAE. "3061: Cybersecurity guidebook for cyber-physical vehicle systems. 2016". In: *Society for automotive engineers* (2016).

[8] S. Nürnberger and C. Rossow. "vatiCAN: Vetted, authenticated CAN bus". In: vol. 9813. Springer Verlag, 2016, pp. 106–124. ISBN: 9783662531396.

[9] Jo Van Bulck, Jan Mühlberg, and Frank Piessens. "VulCAN: Efficient Component Authentication and Software Isolation for Automotive Control Networks". eng. In: *Proceedings of the 33rd Annual Computer Security Applications Conference*. Vol. 132521. ACSAC 2017. ACM, 2017, pp. 225–237. ISBN: 9781450353458.

[10] A.-I. Radu and F.D. Garcia. "LeiA: A lightweight authentication protocol for CAN". In: vol. 9879. Springer Verlag, 2016, pp. 283–300. ISBN: 9783319457406.

[11] K. Koscher et al. "Experimental Security Analysis of a Modern Automobile". In: *2010 IEEE Symposium on Security and Privacy*. May 2010, pp. 447–462. DOI: 10.1109/SP.2010.34.

[12] Marko Wolf, André Weimerskirch, and Christof Paar. "Security in automotive bus systems". In: *Workshop on Embedded Security in Cars*. 2004.

[13] M Maile et al. "Objective testing of a cooperative intersection collision avoidance system for traffic signal and stop sign violation". eng. In: *2011 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2011, pp. 130–137. ISBN: 9781457708909.

[14] F Ahmed-Zaid et al. *Vehicle safety communications–applications (vsc-a) final report*. Tech. rep. 2011.

[15] EL Suzanne et al. "Intersection collision avoidance-violation project: Final project report". In: *DOT HS* 810 (2007), p. 749.

[16] Charlie Miller and Chris Valasek. "Remote exploitation of an unaltered passenger vehicle". In: *Black Hat USA* 2015 (2015), p. 91.

[17] P. Johnson et al. "Can the Common Vulnerability Scoring System be Trusted? A Bayesian Analysis". In: *IEEE Transactions on Dependable and Secure Computing* 15.6 (Nov. 2018), pp. 1002–1015. ISSN: 1545-5971. DOI: 10.1109/TDSC.2016.2644614.

[18] Margus Välja, Matus Korman, and Robert Lagerström. "A Study on Software Vulnerabilities and Weaknesses of Embedded Systems in Power Networks". eng. In: *Proceedings of the 2nd Workshop on cyber-physical security and resilience in smart grids*. CPSR-SG'17. ACM, 2017, pp. 47–52. ISBN: 9781450349789.

[19] Roderick Currie. "Hacking the CAN bus: Basic manipulation of a modern automobile through CAN bus reverse engineering". In: *SANS Institute* (2017).

[20] Mert D Pesé, Karsten Schmidt, and Harald Zweck. *Hardware/Software Co-Design of an Automotive Embedded Firewall*. eng. 2017. URL: https://saemobilus.sae.org/content/2017-01-1659.

Appendix A

Search Terms

Volvo
Volkswagen
BMW
Audi
Toyota
Kia
Mercedes
Skoda
Renault
Ford
Peugeot
Nissan
Hyundai Fiat
Opel
Mazda
Seat
Subaru
Dacia
Citroën
Honda
Mitsubishi
Mini
Suzuki
Lexus
Porsche
Jeep
Land Rover
Jaguar
DS
Chevrolet
Alfa Romeo
Cadillac
Lancia
Smart
Saab
Rolls-Royce
Snapdragon Automobile

TRITA-EECS-EX-2019:488