Controlling your connected car
enforcing privacy on telematics data using cryptographic techniques

de Graaff, R.

Award date:
2015
Controlling your Connected Car

Enforcing Privacy on Telematics Data using Cryptographic Techniques

Ramon de Graaff
ramondegraaff@gmail.com

Supervisors:
- dr. Boris Škorić, Security Group, TU Eindhoven
- ir. Henri Hambartsumyan, Deloitte Risk Services B.V.

Eindhoven, June 2015
Assessment committee:
dr. Boris Škorić, Security Group, TU Eindhoven
ir. Henri Hambartsyunyan, Deloitte Risk Services B.V.
dr. ir. L.A.M. Schoenmakers, Coding and Crypto Group, TU Eindhoven
Abstract

The topic of this thesis is enforcing privacy on telematics data that is being transmitted from the cars to third parties. Telematics data is information about the state of the car which is being sent towards a third party. Privacy has become an issue for car drivers the last couple of years. Related research has only shown the issue, but never proposed a solution to the problem.

In this thesis, we have researched how a higher level of privacy (compared to the current level) can be enforced using cryptographic techniques. Typically, very little actions have been taken to protect the privacy of the user. Based on the security and privacy issues in the context of Connected Cars, we identified technical requirements from which we constructed a cryptographic solution.

We have given an overview of the possible architectures to communicate between the car and a third party which resulted in a list of advantages and disadvantages per type of communication. By using cryptographic techniques, such as homomorphic cryptography, we were able to eliminate a lot of disadvantages. The Semi-Trusted Third Party architecture is chosen as the best option in the context of our thesis.

Based on that architecture, we have provided sequence diagrams for the relevant scenarios, which should give the reader a clear understanding of the information flow. This includes the steps involved in storing the data, requesting delegated keys and aggregating data for a third party.

This all contributed to defining technical requirements for a cryptographic solution. We proposed a modification of two existing cryptographic systems that enables the aggregation of multiple individual messages. To accomplish this, these cryptographic systems use homomorphic cryptography without revealing individual messages. However, important is that the proposed solution should not limit the functionality. For instance, it should still be able for the car owner to view his individual messages. This would, for instance, allow the car owner to view the current location of the car in case it gets stolen. In this thesis, we have analyzed to what extent the security and privacy requirements are being satisfied by the proposed cryptographic solutions.

Although we have only tested our proposed solution on a modern CPU, we have compared the CPU with the microcontrollers in the car by using a general benchmark performance called Dhrystone. In order to do so, we have made a proof-of-concept implementation and researched the execution time for key generation, encryption and decryption. Furthermore, we have analyzed the theoretical complexity of both proposed cryptographic solutions and based on that made an estimation on how well the cryptographic solutions would perform on a car.

The outcome of this thesis is a cryptographic solution that increases the privacy of the user, while it does not limit the functionality required by the user or the involved third party. Moreover, we show that the proposed cryptographic system has an acceptable performance for an application in Connected Cars.
# Contents

Contents vii  
List of Figures ix  
List of Tables xi  

1 Introduction  
1.1 Problem description  
1.2 Related Work  
1.3 Research questions  
1.4 Contribution  
1.5 Outline  

2 Background  
2.1 Functionality  
2.2 Extended connectivity perimeter  
2.3 Attack surfaces  
2.4 Threat actors  
2.5 Vehicle internal network  
2.6 Simplified connectivity perimeter  
2.7 Summary  

3 Security  
3.1 Security threats  
3.2 Legal requirements  
3.3 Security requirements  
3.4 Prevention of attacks on the internal network  

4 Privacy  
4.1 The privacy concepts of Solove  
4.2 Privacy threats  
4.3 Legal privacy requirements  
4.4 Privacy requirements  

5 Overview of possible trust architectures  
5.1 Roles  
5.2 Possible architectures  

6 Proposed architecture  
6.1 Technical requirements  
6.2 Cryptographic techniques  
6.3 Choosing the architecture: sTTP  
6.4 Sequence diagram for storing data  

Controlling your Connected Car vii
## CONTENTS

6.5 Sequence diagram for requesting delegated key ........................................ 41
6.6 Sequence diagram for STTP-side aggregation ............................................ 42
6.7 Sequence diagram for car-side aggregation .............................................. 44

7 Proposed cryptographic systems ................................................................. 47
  7.1 Cryptographic requirements ..................................................................... 47
  7.2 Preliminaries ............................................................................................ 47
  7.3 Bresson et al.-based solution .................................................................. 48
  7.4 Improvement: Ateniese et al.-based solution ........................................... 52

8 Analysis of the proposed cryptosystems ......................................................... 57
  8.1 Key size ................................................................................................... 57
  8.2 Analysis of security requirements ............................................................ 58
  8.3 Analysis of privacy requirements .............................................................. 60
  8.4 Comparison Bresson vs. Ateniese ........................................................... 63
  8.5 Benchmark Bresson ................................................................................. 65
  8.6 Processors ............................................................................................... 69

9 Conclusions .................................................................................................... 71
  9.1 Future work ............................................................................................. 71

Bibliography ........................................................................................................ 73

Appendix ............................................................................................................... 77

A Proof of Concept implementation Bresson ..................................................... 77

B Pseudo code algorithms for the proposed cryptosystems ................................ 80
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Extended connectivity perimeter of the connected car</td>
<td>9</td>
</tr>
<tr>
<td>2.2</td>
<td>CAN bus data packet structure (base frame format) in bits [1]</td>
<td>12</td>
</tr>
<tr>
<td>2.3</td>
<td>Simplified connectivity perimeter</td>
<td>14</td>
</tr>
<tr>
<td>3.1</td>
<td>Restricting access to read-access</td>
<td>20</td>
</tr>
<tr>
<td>5.1</td>
<td>Direct communication architecture</td>
<td>33</td>
</tr>
<tr>
<td>5.2</td>
<td>Semi-Trusted Third Party Architecture</td>
<td>34</td>
</tr>
<tr>
<td>5.3</td>
<td>Trusted Third Party Architecture with TTP-side aggregation</td>
<td>36</td>
</tr>
<tr>
<td>6.1</td>
<td>Sequence diagram for storing ciphertexts</td>
<td>41</td>
</tr>
<tr>
<td>6.2</td>
<td>Sequence diagram for requesting delegated key</td>
<td>42</td>
</tr>
<tr>
<td>6.3</td>
<td>Sequence diagram for sTTP-side aggregation</td>
<td>43</td>
</tr>
<tr>
<td>6.4</td>
<td>Sequence diagram for car-side aggregation</td>
<td>45</td>
</tr>
<tr>
<td>8.1</td>
<td>Time consumption of the key generation per keylength</td>
<td>66</td>
</tr>
<tr>
<td>8.2</td>
<td>Performance of the encryption algorithm</td>
<td>67</td>
</tr>
<tr>
<td>8.3</td>
<td>Performance of the decryption algorithm using $x$</td>
<td>68</td>
</tr>
<tr>
<td>8.4</td>
<td>Performance of the decryption algorithm using $\lambda(n)$</td>
<td>68</td>
</tr>
<tr>
<td>8.5</td>
<td>Performance of the ciphertext multiplication</td>
<td>68</td>
</tr>
</tbody>
</table>
# List of Tables

2.1 Key Electronic Control Units (ECUs) within our cars, their roles, and which CAN buses they are on. [2] .......................... 13

8.1 Complexity comparison between Bresson et al.-based and Ateniese et al.-based solution ............................................. 63

8.2 Performance comparison between automotive microprocessors ............................................. 69
Chapter 1

Introduction

1.1 Problem description

Digital Security in vehicles is becoming more important which is the result of the fast development in potential cyber-attacks against vehicles. With the increase in “Internet of things” equipment, devices including vehicles are becoming connected. One of the market leaders is Tesla Motors with a lot of features accessible through an application over the Internet. Not solely the customer of the vehicle is becoming more connected through his vehicle, also the car manufacturers and other third parties are becoming more connected as they use wireless channels to get access to the data in the cars. For example, Tesla Motors’ Customer Privacy Policy states that they are capable of system monitoring and remote vehicle analysis. This is understandable for the improvement of services towards the customer. However it raises huge privacy issues as they are also capable of gathering information such as GPS location, fuel consumption, but also contacts, browsing history, navigation history and radio listening history. Note that Tesla is not the only car manufacturer that might collect this data. Several news articles and privacy statements indicate the collection of information by other car manufacturers as well.

Some people might argue that there is nothing to hide, but the point is not that there is data that is a priori “wrong” or illegal, but that “innocent” data can (later) be used against the sender. Moreover, it might be in violation with privacy regulations.

1.2 Related Work

The number of connected cars is rapidly increasing: 80% of the cars that will be sold in 2016 will be a connected car [3]. That means that security and privacy should be considered in the design of new cars. When security and privacy properties cannot be guaranteed, car drivers might face serious physical danger, because their car could potentially be taken over by an attacker. But they can also face privacy issues, such as personal data that is being stolen or sold to advertisement parties. A report by Capgemini Consulting [4] shows that 19% is not willing to share personal data and 34% is only willing to share personal data anonymously for research and statistics purposes. Within

---

1 “Telematics Log Data: To improve our vehicles and services for you, we collect certain telematics data regarding the performance, usage, operation, and condition of your Tesla vehicle, including the following: vehicle identification number, speed information, odometer readings, battery use management information, battery charging history, electrical system functions, software version information, infotainment system data, safety related data (including information regarding the vehicle’s SRS systems, brakes, security, e-brake), and other data to assist in identifying and analyzing the performance of the vehicle. We may collect such information either in person (e.g., during a service appointment) or via remote access.
Remote vehicle analysis: We may be able to dynamically connect to your Tesla vehicle to diagnose and resolve issues with it, and this process may result in access to personal settings in the vehicle (such as contacts, browsing history, navigation history, and radio listening history). This dynamic connection also enables us to view the current location of your vehicle, but such access is restricted to a limited number of personnel within Tesla.” [Accessed: 15-10-2015]
the set of questionnaire participants, 31% say that they are willing to share data in return for service or incentive and 16% of the participants are willing to share the data without restrictions. This shows that the people have privacy concerns about the data that is being shared with the car manufacturer. It shows that 53% of the participants does not or only limited want to share personal information with the car manufacturer. Moreover, 31% percent will share their data for something in return, while this is not always necessary to establish the particular service. The report “2014 Harris Poll AutoTECHCAST Study” describes that “65 percent said they feared that owning a connected car could compromise their privacy.” [5]. With almost 14,000 participants, this is a clear sign that car owners do care for a privacy friendly solution.

In November 2014, the Global Alliance of Automakers released seven Principles [6] which the participating Members should adhere to. A large part (19) of the car manufacturers from North America are affiliated with the Global Alliance of Automakers. The seven Principles are defined as follows:

1. **Transparency.** The data subjects, in this case the car drivers, should get notices about the collection, usage and sharing of the data that is generated by the car. This includes the type of information, the purpose of collection, the entities it is shared with, information about the deletion or de-identification, the choices and whether and how the information can be accessed.

2. **Choice.** The data subject should be given a choice with respect to the collection, usage and sharing of data. He should explicitly give consent in case the personal information is being used for marketing purposes or when the information is being shared with unaffiliated parties for their own purposes.

3. **Respect for context.** The participating Members commit to only use the collected information for a predefined purpose and they will not use the information for other purposes. The information should be used within the context for what it has been collected for.

4. **Data minimization, de-identification and retention.** The participating Members commit to not use and store the personal information for other purposes than communicated to the data subject. Moreover, the personal information should be de-identified whenever possible and it should be stored no longer than necessary as required by the business purposes.

5. **Data security.** The personal information which is being collected and stored by the car manufacturers should be protected against loss and unauthorized access or use. The participating Members commit to apply reasonable measures, which includes standard industry practices.

6. **Integrity & access.** Participating Members should protect the accuracy of personal information. Data subjects should be given means to review and correct the personal information which has been collected by the car manufacturer.

7. **Accountability.** The Participating Members commit to ensure they and other Members adhere to these principles.

After these privacy principles were released by the Global Alliance of Automakers, Senator Markey investigated whether the car manufacturers adhered to these principles. It resulted in a report [7] about the security and privacy practices of 16 major automobile manufacturers. The report released the follows key findings:

1. Almost 100% of today’s cars include wireless technology that allows devices (e.g., mobile phones) and services (e.g., collision notifications) to be connected to the car. The wireless signal could be subject to hacking or privacy intrusions, which might lead to access to the internal systems of the car.

2. Most of the car manufacturers are unable to detect or are unaware of past hacking incidents in their manufactured cars. Most of them cannot report on past hacking incidents.
3. The car manufacturers vary in the implemented security measures which should prevent remote access to vehicle electronics. Some of them do have some security mechanisms in place while others only have minor security measures in place. Many of the car manufacturers did not seem to understand the questions which are related to this topic.

4. Only two automobile manufacturers described capabilities to diagnose or respond to an infiltration in real-time. They claimed that they monitor the internal network of a car with proprietary software, anomaly detection, firewall and watchdog systems, message authentication, intrusion detection and seed-keying.

5. Car manufacturers collect large amounts of data from the car. This includes geographical locations (e.g., current and previous geographical location), system settings for Event Data Recorder devices (e.g., crash events, brake application and fault codes) and operational data (e.g., vehicle speed, direction and engine status).

6. Half of the questioned car manufacturers (50%) use technologies that collect and transmit data to off-board data centers. A majority respond that information about the driving history is being stored off-board and some of that data is being used by third parties who offer certain services which require that data.

7. Car manufacturers usually state to use personal vehicle data to “provide services” and “remote diagnostic features” or for research purposes. However, they all claim not to sell the collected information to third parties. Moreover, the data that is being stored is often retained longer than necessary.

8. Car manufacturers often make the customers aware of data collection by stating so in the user manual, privacy statements or the terms and conditions of the vehicle sale. However, the car owner is often not aware of the data collection and data processing. An opt-out is sometimes simply not possible or the customer might lose critical services such as navigation.

With the clear need on privacy friendly solutions, it is remarkable that this subject has only little been touched by researchers. Most of the published articles describe the security and privacy issues within the internal network of the car, security and privacy issues between cars in a Vehicle-to-X (V2X) setting [8], privacy issues with respect to the Tire Pressure Monitoring System or security issues with gaining access to a car via a key fob. The V2X setting is different from our work, because V2X mainly focuses on providing the car with information to avoid traffic jams and collisions, while our setting focuses on providing information about the car or car driver to third parties without leaking privacy sensitive data. The work most related to this thesis is the work of Borsch [9]. The authors investigate how to construct a privacy friendly way to perform driving pattern-based insurance plans. In order to do so, they use certain techniques to determine whether a particular value is bigger than another value. This information can be used to check whether a certain threshold has been reached and the driver can get a penalty for his misbehavior. They use Secure Multi-Party Computation and homomorphic cryptography to satisfy the properties they need. Secure Multi-Party Computation is a technique to securely and collaboratively evaluate a function without revealing the individual inputs to the other participating parties. Homomorphic cryptography is a technique to perform an operation on multiple ciphertexts without the actual knowledge of the underlying plaintexts. This can be done by performing an operation on multiple ciphertexts, for instance $C(m_i) \cdot C(m_j) = C(m_i + m_j)$. The authors of [3] state that in the coming years, the amount of data that is transmitted per vehicle per month will grow from 4 MB to 5 GB. Somewhat related, BMW released a press statement in January 2015, stating that technology companies and advertisers were putting pressure on carmakers to pass on data collected by connected cars [10]. In [11], the author describes privacy design strategies to support privacy by design. Especially design strategy #7: Enforce, is a strategy which we will focus on throughout this thesis.
CHAPTER 1. INTRODUCTION

1.3 Research questions

As we stated in our problem description, we want to find a solution that provides more privacy to the car owner/car driver, while maintaining the current functionality to the car owners and the third parties. The purpose of this research is to identify the security and privacy problems related to connected cars and find a technical solution to mitigate these problems. The outcome of this thesis should be the proposal of an architecture in combination with a cryptographic system. This cryptographic system should not limit the current possibilities, but enable the stakeholders to exchange information in a privacy friendly manner. We have defined the following main research question:

How can the privacy be technically enforced in the transfer of data towards car manufacturers or other third parties?

To be able to answer this main research question, we break this question down into five related subquestions. The sub research questions are defined as follows:

1. What are the security requirements for data which is transmitted to third parties? What are the legal requirements and what security attacks are possible?

2. What are the privacy requirements for data which is transmitted to third parties? What are the legal requirements according to the privacy regulations and what privacy attacks are possible?

3. What trust architecture should be implemented and how is the information flow between the parties?

4. What techniques are known which can help in enhancing the privacy of the user but can still provide information to the third parties?

5. How should these techniques be applied in the context of the Connected Car?

1.4 Contribution

We have contributed to the current state of mainly privacy issues which are related to connected cars. Summarizing our contribution, we have:

- Identified and described the threat actors and attack surfaces with respect to connected cars.
- Identified the threat with respect to security issues, described the legal requirements and described relevant security requirements.
- Identified the threats with respect to privacy issues, described the legal requirements and described relevant privacy requirements.
- Researched advantages and disadvantages of the possible trust architectures which enable the involved parties to communicate to each other.
- Identified the trust architectural requirements and motivated which trust architecture should be used.
- Provided a generally applicable information flow diagram (sequence diagram) for the functionality that is related to a privacy friendly implementation in the connected car setting.
- Described two cryptographic systems which meet the cryptographic requirements and provide privacy friendly features.
- Modified the existing cryptographic systems to meet a certain homomorphic property.
• Proposed a method to use the cryptographic system such that it is able to aggregate multiple messages.
• Analyzed the security and privacy requirements of the two described cryptographic systems.
• Provided a Proof-of-Concept implementation for one of the two described cryptographic systems and analyzed the performance of that implementation.
• Performed a comparison between the two cryptographic systems and related the performance to current automotive microcontrollers.

1.5 Outline

The research questions defined in Section 1.3 have been divided into chapters that correspond with the research questions. The thesis is structured as follows:

**Chapter 2** provides the necessary background information about the car and its functionality, attack surfaces and threat actors.

**Chapter 3** provides the reader with information about the relevant security threats, the legal requirements according to the Data Protection Directive with respect to security, the security requirements which are related to connected cars and the security of vehicle internal networks.

**Chapter 4** gives an introduction to the concept of privacy, identifies the privacy threats, provides the legal context based on the Data Protection Directive with respect to privacy and finishes with the definition of the relevant privacy requirements.

**Chapter 5** provides an overview of the roles involved in a system where information is being exchanged between the car and a third party. We provide an overview of the possible trust architectures.

**Chapter 6** will provide the reader with the relevant technical properties for a trust architecture and we describe some techniques which can be used to improve the privacy of the car owner. We will choose one architecture and describe how this architecture should be used in general terms to enhance the privacy of a car owner.

**Chapter 7** provides the cryptographic requirements and provides two cryptographic systems which meet these requirements.

**Chapter 8** provides an analysis of both cryptographic systems, a benchmark for one of the cryptographic systems and a comparison between commonly used processors in the automotive industry.

**Chapter 9** concludes the thesis and provides possible future work opportunities.
Chapter 2

Background

Modern cars are no longer solely mechanical, they are being controlled by multiple computers inside the car. These computers register a lot of information through their sensors and process the information to control and monitor the car. Initially these systems were placed to improve efficiency and safety. Currently, the world is moving towards connected cars, which means that the cars are being connected with other devices such as mobile phones and servers.

This chapter will give the reader background knowledge on how modern vehicles are connected to the parties that provide some service or functionality to the car owner. We will show what the relevant functionalities for the car owner and the third party are. We will define the scope of our research and we will show what roles are relevant our research. After reading this chapter the reader should have a clear understanding of what connected cars are, how they are connected and what possible attack surfaces are relevant.

2.1 Functionality

Connected cars obviously offer extra functionality to the car compared to traditional non-connected cars. Since this thesis focuses on connected cars, the reader should understand what extra functionality the connected cars offer and how they are used. Within connected cars, we can distinguish between two types of functionality, that is, functionality for the car owner and functionality for the third party.

We have selected a few examples of functionality for the car owner from [12], which are automatic collision notification, roadside assistance, destination guidance, remote functions, vehicle locater and POI Search.

- **Automatic Collision Notification.** Once a vehicle has been part of a collision, sensors can automatically detect and alert emergency services. These sensors can even detect the severity of the crash and the possible impact on the passengers. Either an employee of the car brand will be brought in contact with the passengers or the call is immediately forwarded to the public emergency services.

- **Roadside Assistance.** Roadside Assistance is a service which provides the driver of the car with services he needs in case his car breaks down. The driver can call the roadside assistance who will lookup nearby service providers. The assistance service of the car brand can direct the assistance the driver needs to his GPS location. Several services are provided, such as lock-out service (when the driver has been locked out of his car), tow truck service, flat tire change, battery jump start and emergency fuel delivery.

- **Destination Guidance.** We see “Destination Guidance” as the function that is essentially the same as GPS navigation. It is the function built within the car that gives the driver directions towards its destination. Sometimes the navigation automatically adapts the route depending on potentially upcoming traffic jams and such.
CHAPTER 2. BACKGROUND

• **Remote Functions.** Remote Functions function allow the car owner to remotely control certain functions of the vehicle. Examples of these functions are remote door lock/unlock, remote horn/lights, remote window control, remote temperature control and remote vehicle diagnostics information. This can either be done through a mobile application where the user logs into his account and can control the functions there or via a web application where the user authenticates himself and is able to control the door via the web application.

• **Vehicle Locater.** Vehicle Locater is the function that allows the owner of the vehicle to receive the GPS location of the vehicle in a mobile application or via a web portal of the car brand. For instance, when the driver does not remember the location where he parked his car, he will be able to locate the GPS location via the car brand. Another scenario where the vehicle locater might become handy is when the car owner lend his car to somebody else and he wants to check how the progress of the journey is going.

• **POI Search.** Point-of-Interest (POI) Search is a function within the car that allows the car owner to find a specific POI through the interface of the car. The driver will select a category and the search engine will provide a list of POI within the area.

The functional benefits for the third party are not explicitly stated somewhere, so we came up with a few examples of functionality where the third party might benefit from, i.e., behavioral information, personal interests, telematics information and location information.

• **Behavioral information.** Behavioral information is information about the behavior of the driver, such as what places he/she visits, where he/she does his grocery, what garage he/she goes to, which gas station he/she uses, what restaurant he/she likes to go to and so on.

• **Personal interests.** Slightly related to that, personal interest is information about what the driver likes to do in his free time, what the hobbies of the driver are and what the driver likes to spend his money on.

• **Telematics information.** Telematics data is information about the state of the car, which is for instance the status of the motor, the status of the brakes, whether any issues are known, information about the fuel spending, the mileage, speed and so on. This information can be useful for third parties, such as car manufacturers, who can use this information to improve their product and adjust the components of the car according to the liking of the drivers.

• **Location information.** Location information is information about the geographical location of the car. This might be handy for third parties who can use this information to construct a profile of the individual. The location is also useful for parties who want to prove that the individual was at a certain place at a certain time.
2.2 Extended connectivity perimeter

We can divide the connectivity to multiple devices, other vehicles or back-end services in different zones, which all have their own properties. In Figure 2.1 an illustration of the extended connectivity perimeter of the connected car is shown, which divides the services into five zones that are connected. From the first zone, which happens in the car or really close to the car, to the fifth zone which is data access by third parties via the service providers through data requests.

2.2.1 Zone 1

In the first zone, devices such as a media players, cell phones, CDs and USB-sticks are connected to the car. Directly connecting the devices to the car is not the only way of connecting to the vehicle. This is not limited to external devices that are connected to the network of the car, but does also include the internal network of the car (CAN bus) and the On-Board Diagnostics (OBD) port. Another possible way to connect to the vehicle is via wireless channels, i.e., cellular links, Bluetooth and Wi-Fi connections connect the vehicle wirelessly to devices such as cell phones, iPads, PDAs and other wireless devices.

2.2.2 Zone 2

The second zone is the connection between vehicles (V2V) and between Vehicles and the Infrastructure (V2I). This connection is relatively close to the vehicle and ranges from a few meters to a few hundred meters. Information such as the speed and direction of the car is being shared with the other vehicles and the infrastructure. This information is being sent over an unlicensed wireless channel (5.9 GHz), which is the standard for dedicated short-range communication (DSRC).

2.2.3 Zone 3

The third zone is the connection from the car to the OEM (car manufacturer) or supplier. This information includes information about the status of the vehicle, which might include information
that can be used for sales, servicing and marketing. For instance, when the car detects that the replacement of a vehicle part is necessary, the car can send a message to the OEM that a specific part is needed and they can already order it for the customer.

2.2.4 Zone 4

The fourth zone is the connection between the car and the service providers, which either get the needed information to provide the services through the third zone or directly from the first zone. These services can be provided by the OEMs, but are not limited to the OEMs and can also be provided by other parties. The car or the car owner can request a certain piece of information and the service provider provides the information that is based on the information in the request. This may concern parking or traffic information based on the current location, but it may also concern information which needs to be answered by the customer service.

2.2.5 Zone 5

The last zone is the connection from the car to the zone with third parties who have a particular interest in the information that is being collected in the other zones. Information about for instance the past locations of the vehicle can be requested by government agencies and speed or throttle level information can be requested by insurance companies. The information which is being collected in the third and fourth zone can also be shared with parties (customers) such as advertisement parties who might have an interest in the information that is being collected by the parties in the other zones.

2.3 Attack surfaces

The internal network of a car can be accessed via two different types of attacks. An attacker either needs physical access to the car or access through a wireless interface. Such as interface gives access to the car through a component which is connected to the internal network of the car. Such a component is called an Electronic Control Unit (ECU) and the internal network of the car is called the Controller Area Network (CAN). In Figure 2.1 we defined multiple zones for the multiple parties. In the first zone we have depicted several attack surfaces, wired and wireless, such as cellular, Wi-Fi, Bluetooth, cell phones, CDs and USBs.

- Cellular communication has known vulnerabilities [13], so both an inside- and outside attacker can use this to passively sniff the data which is being communicated over the network connection. An example of attacking the cellular network communication on cars has been shown in [14, 2, 15].

- Wi-Fi communication also has its limitations and is, depending on its configuration, vulnerable to known attacks [16]. For instance, dictionary or brute-force attacks might be possible. Once the key is known, the communication between the paired devices can be decrypted.

- Bluetooth also has known vulnerabilities [15]. When the car is paired with a device, the attacker only needs to brute force the PIN, which is being used to encrypt the traffic, to gain access to the Bluetooth module in the car.

- Keyless Entry Systems have also shown their weaknesses in the past, allowing the attacker to clone a key by exploiting weaknesses in the used cryptography [15].

- In [17], the authors show that it possible to manipulate the Tire Pressure Monitoring System (TPMS) by sending spoofed messages to the monitoring sensors. The receiving sensor operates on a frequency of 433 MHz. Due to a lack of security measures messages can be injected to the TPMS and will be interpreted by the ECU.
The authors of [2] show that it is possible to override firmware of the audio player in the car by inserting a CD which exploits a vulnerability in the software of the media player. The same holds for exploiting the firmware by playing an infected MP3 via a USB drive. As a result the ECU is compromised and the attacker is able to send packets onto the internal network of the car.

The cellphone is also an attack surface, as the cellphone of a driver often is connected to the vehicle, either directly via a cable or via a wireless channel. When the cellphone is infected with malicious software that exploits vulnerabilities of a reachable ECU, an attacker might be able to attack the car via that channel. Moreover, connected cars often offer remote functions to the driver through a mobile application. When the cellphone is infected, the attacker might be able to exploit these functions or exploit weaknesses in the corresponding ECU.

ECUs are not always produced by the car manufacturer himself. The software in these ECUs might have been infected by malicious software. As all the devices on the CAN-bus can broadcast messages, malicious ECUs can cause malfunctioning of the car.

2.4 Threat actors

In the context of connected cars, there are multiple parties involved. Parties involved are, as shown in Figure 2.1, cars, OEMs, suppliers, dealers, marketing and servicing parties, service provides and other third parties. To be able to identify threats, we need to identify the threat actors of the parties involved. These threat actors will help us to identify the relevant security and privacy threats. We divide the threat actors into two categories, the malicious insider and the malicious outsider.

2.4.1 Malicious insider

A malicious insider is a party who is part of the system, but does not act honestly. The malicious insider has access to particular information and resources where an outsider might not have access to.

Malicious car owner

A malicious car owner is a car owner who is part of the connected car network which is provided by the car manufacturer, but does not behave honestly. The car owner might be able to reach other cars, because the honest car owner and the malicious car owner belong to the same network. For instance, he can try to attack other vehicles, his own car (for insurance purposes) or the server of a third party.

Malicious person with physical access

A malicious person with physical access can be a mechanic who needs to perform maintenance on the vehicle, the dealer who originally sold the vehicle to the car owner, a friend who is allowed to drive in the car or a burglar who has physically access to the vehicle.

Malicious third party

A malicious third party is a third party that is not being honest and therefore behaves maliciously towards the owner of a car. If the car allows the sending packets onto the internal network of the car by a third party, this third party might be able to control certain components of the vehicle. An example of a malicious third party is a car manufacturer who delivers functionality such as opening windows and locking doors.
2.4.2 Malicious outsider

A malicious outsider is an attacker who is not part of the system and attacks components of the system, such as the involved parties or the communication between the parties. Outsiders might not be able to reach the connected cars within the same network, because they are not part of the network. For instance, malicious outsiders could try to attack the cellular network of the vehicles and try to manipulate the packets. As the malicious outsider is not part of the system of that connected car, they do not use a similar vehicle to attack and they are not part of any organization which gives them access to the system of the connected car of the victim.

2.5 Vehicle internal network

In the past the internal network of the vehicles was merely mechanical, but the increasing oil prices demanded a more efficient use of the gas. This resulted in embedding micro-controllers into the car which measures multiple values of the car to improve the efficiency of the fuel usage. These components are called Engine Control Units (ECUs). Modern vehicles have between 70 and 100 ECUs, which are able to monitor and control the car. In Table 2.1, we have depicted a table from [2]. This table gives an overview of the most critical ECUs of the car and the communication bus they are on. The ECU is either connected to the high speed CAN bus or to the low speed CAN bus, or both. The difference is that the high speed communication bus is used for critical ECUs that control the car. The low speed communication bus is used for the less critical systems. Please note that some of the ECUs are connected to both communication buses.

These ECUs communicate with each other over a communication bus, which is called the CAN bus. An ECU broadcasts packets to all the nodes, all the nodes receive all the packets and they determine whether the identifier matches the identifier of the ECU. In Figure 2.2, the CAN data frame packet specification is shown, which is specified in [1]. The ECUs decides, based on the identifier in the packet, whether the packet is destined for that ECU. Moreover, when the packet is destined for that specific ECU, the ECU does a Cyclic Redundancy Check (CRC) to detect accidental changes in the data. In Figure 2.2, we have used abbreviations for some of the field names which do not add value to our context.

![Figure 2.2: CAN bus data packet structure (base frame format) in bits][1]
<table>
<thead>
<tr>
<th>Component</th>
<th>Functionality</th>
<th>CAN Low Speed</th>
<th>CAN High Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECM</td>
<td><em>Engine Control Module</em></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Controls the engine using information from sensors to determine the amount of fuel, ignition timing, and other engine parameters.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EBCM</td>
<td><em>Electronic Brake Control Module</em></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Controls the Antilock Brake System (ABS) pump motor and valves, preventing brakes from locking up and skidding by regulating hydraulic pressure.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCM</td>
<td><em>Transmission Control Module</em></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Controls electronic transmission using data from sensors and from the ECM to determine when and how to change gears.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BCM</td>
<td><em>Body Control Module</em></td>
<td>✓ ✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Controls various vehicle functions, provides information to occupants, and acts as a firewall between the two subnets.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Telematics</td>
<td><em>Telematics Module</em></td>
<td>✓ ✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Enables remote data communication with the vehicle via cellular link.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RCDLR</td>
<td><em>Remote Control Door Lock Receiver</em></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Receives the signal from the car’s key fob to lock/unlock the doors and the trunk. It also receives data wirelessly from the Tire Pressure Monitoring System sensors.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HVAC</td>
<td><em>Heating, Ventilation, Air Conditioning</em></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Controls cabin environment.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SDM</td>
<td><em>Inflatable Restraint Sensing and Diagnostic Module</em></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Controls airbags and seat belt pretensioners.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IPC/DIC</td>
<td><em>Instrument Panel Cluster/Driver Information Center</em></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Displays information to the driver about speed, fuel level, and various alerts about the car’s status.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Radio</td>
<td><em>Radio</em></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>In addition to regular radio functions, funnels and generates most of the in-cabin sounds (beeps, buzzes, chimes).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TDM</td>
<td><em>Theft Deterrent Module</em></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Prevents vehicle from starting without a legitimate key.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2.1: Key Electronic Control Units (ECUs) within our cars, their roles, and which CAN buses they are on. [2]
2.6 Simplified connectivity perimeter

Figure 2.1 has shown the extended perimeter for the connected car environment with a distinction between multiple zones. The ECUs in a car are connected to each other via the CAN-bus, which is explained in Section 2.5. One of the ECUs, the telematics end-point, is the connection between the car and the external parties, such as the car manufacturer, repair shops, informational services and government agencies. These parties are examples of the last three zones, zone 3, 4 and 5 as we showed in Figure 2.1. Although the goals of the parties differ, all the parties might end up with information about the car owner. Figure 2.3 is an illustration of a simplified perimeter scheme of the connected car. TM is the Telematics Module which allows a network connection between the car and the server of the third party. This simplification is sufficient for the purposes of our research, because although the different parties might have different interests in different types of information of the car owner, we still need to get control over the connection between the car and an external party. So basically in our context it does not matter who the third party is, but it helps in determining the possible security and privacy issues.

2.7 Summary

In this chapter, we have given an overview of the functionality which is being used in connected cars. We have also extensively shown what parties are involved in the perimeter of the connected cars and what attack surfaces a connected car has to be able to pinpoint the security and privacy related threats. Related to that, we have identified the roles from a security and privacy perspective. Furthermore, we have shown what the structure of a CAN-packet is and we have given an overview of the most important ECUs in the car. This information will be very useful to identify the security and privacy related threats. It will help to understand what motives the involved parties might have and how they relate to security and privacy requirements. Lastly, we have shown how we can simplify the perimeter, which we will use for the more technical identification and implementation of risks.

Figure 2.3: Simplified connectivity perimeter
Chapter 3

Security

In this chapter we identify the security threats with respect to the connected car settings. We summarize the legal requirements which are obliged by the Data Protection Directive [18] and define general security requirements and explain them in the context of our research. After this chapter the reader should have a clear understanding of the possible security attacks on a connected car and how the security requirements should be explained in this context.

3.1 Security threats

In this section we describe the security attacks that are possible in the context of connected cars. These are either attacks on the communication between the car and a third party or it is an attack on one of the parties themselves. Within the security threats, we can distinguish between two types of attacks, i.e., passive and active attacks.

3.1.1 Passive attacks

A passive attack is an attack where the attacker can view the communication, however he does not tamper with the communication.

We give a few examples of passive attacks which can happen in a context of connected cars:

- In Section 2.3, we have shown the attack vectors for a connected car. A passive attack using these attack vectors can be the collection of information about the person who is driving the car. Very sensitive personal information such as GPS location and other information can become known to an attacker. For instance, information about the car and the behavior of the driver.

- The Tire Pressure Monitoring System (TPMS) is used to monitor the tire pressure. Every TPMS-sensor has a unique identification number, which passively can be monitored to track a vehicle. According to [15], these messages can be received up to 40 meters distance from a car.

3.1.2 Active attacks

An active attacker is an attacker who is able to listen, but also tampers with the communication. An attack either affects the car owner or the third party, because the attack is directed towards the server of a third party or towards the vehicle of a car owner.

- A possible security threat is a malicious car owner who can attack other cars through the same network as to what he is connected to with his own car.
Furthermore, as the car will be able to communicate with the server of the third parties, he will also be able to attack these systems from the inside as he is a legitimate user of the system.

The information on the server can also become corrupt when an attacker is able to tamper with the data which is transferred to the server.

Another active attack is when an attacker is able to inject packets onto the internal network (CAN-bus) of the car. The vehicle might damage as a result of these injected packets. A scenario is where a repairing shop might be able to gain customers by damaging nearby cars. When a particular brand is targeted, that could lead to serious reputation and financial damage.

Closely related to this, when packets are sent onto the CAN-bus, the attacker might be able to control the steering wheel, gears, brakes and other critical components at any given time [2, 19]. This can result in serious damage to the driver of that particular vehicle.

If data from the server of the third party to the car owner is being tampered, the functionality of the car might become incorrect, unavailable or replayed. A scenario where this could happen is when the customer service of a car brand helps the targeted car owner and digitally accesses the car, the attacker might be able to do the same.

### 3.2 Legal requirements

In this section we describe the legislative requirements from a security point of view on the basis of European Data Protection Directive 95/46/EC (DPD) [18].

The DPD has basically one article dedicated to security, i.e. Article 17 [18]. Article 17(1), states that “Member States shall provide that the controller must implement appropriate technical and organizational measures to protect personal data against accidental or unlawful destruction or accidental loss, alteration, unauthorized disclosure or access, in particular where the processing involves the transmission of data over a network, and against all other unlawful forms of processing.”. This sentence can be translated into security requirements in terms of the CIA-triad, i.e., confidentiality (unauthorized disclosure or access), integrity (alteration) and availability (accidental or unlawful blocking of access, destruction or loss).

Furthermore, Article 17(2) states that in case the controller is not processing the data by himself, he must “choose a processor providing sufficient guarantees in respect of the technical security measures and organizational measures governing the processing to be carried out, and must ensure compliance with those measures” [18] and this must be governed by a contract or legal act.

Article 12(b) of the DPD states that “Member States shall guarantee every data subject has the right to obtain from the controller (...) the rectification, erasure or blocking of data where the processing of does not comply with the provisions of this Directive, in particular because of the incomplete or inaccurate nature of the data”, which means that the data subject should be in control of his own data.

Article 6(d) of the DPD states that the processing of data must be “accurate and, where necessary, kept up to date”, which touches the authenticity and utility security requirements where accurate could be interpreted as authentic, but accurate can also be interpreted as being useful (the utility requirement).

The Data Protection Directive provides in basic terms how security should be handled with respect to the processing of personal data. Other than in general terms stating the security requirements there is no implementation guideline or code of conduct on this matter available.
CHAPTER 3. SECURITY

3.3 Security requirements

The CIA-triad [20] is a well known security model consisting of three classic security elements which are used to identify security problems. In this section we attempt to identify relevant security requirements with respect to the involved parties, i.e., the car owner/driver and the third party. Furthermore, we will generally describe the security requirements from [21]. We will first give a definition for the requirements in general. Furthermore, we will show why the requirements are relevant in the context of connected cars.

3.3.1 Confidentiality

Confidentiality is defined as “protecting sensitive information from unauthorized disclosure or intelligible interception” [20]. In ISO27001, confidentiality is defined as “the property that information is not made available or disclosed to unauthorized individuals, entities, or processes” [22]. In the context of this thesis, personal information should be kept confidential. A way to do this is to use cryptographic primitives to encrypt the personal information.

Below, we provide a list of elements for which the confidentiality should be preserved:

- The personal data of the driver of the car should be kept confidential. Only authorized entities should get knowledge about personal information about the car owner or car driver.
- Confidentiality should also be guaranteed for the private keys which are used in cars [23], because these values might be used to provide security guarantees for the data that is being transmitted.
- The communication between the car and the third party should be kept confidential as personal information, such as GPS-location and usage of the car might be transmitted over an untrusted communication line.
- On the server of the third party, the confidentiality of the information about the car owner or car driver should be kept confidential.

3.3.2 Integrity

Integrity is defined as “safeguarding the accuracy and completeness of information and computer software” [20]. In ISO27001, it is almost identical, and defined as “the property of safeguarding the accuracy and completeness of assets” [22]. In ISO7498, data integrity is defined as “ensuring that information has not been modified, added, or deleted during its storage or transit” [24]. One can argue that the last definition is more a definition of achieving integrity than a definition for integrity itself. One way to achieve this, is to use cryptographic tools such as signatures, MACs or encryption. The receiving party can verify whether the message has been adjusted, because the verification would fail.

Below, we provide a list of elements for which the integrity should be preserved:

- Integrity of data within the car should be guaranteed. Undetected modification of data can lead to misbehavior of the car, which could lead to damage to the driver and the reputation damage to the car brand.
- The integrity of the data in transmission between the car and a third party should also be guaranteed. Undetected tampered data is not desirable for both the third party and the car driver, as it might lead to incorrect data on the server and informational services to the car driver.

---

1ISO7498-2 provides a general description of security services and related mechanisms and defines the positions within the Reference Model where the services and mechanisms may be provided. Moreover, it extends ISO7498, to cover secure communications between open systems.
CHAPTER 3. SECURITY

- Integrity of data that control functions of the car should be preserved, as these components should only accept messages from trusted parties which are not modified.

3.3.3 Availability

Availability is defined as “ensuring that information and vital services are available to users when required” [20]. ISO27001 defines availability as “the property of being accessible and usable upon demand by an authorized entity” [22]. This basically means that any party who is authorized to access certain data should be able to retrieve that information at any given time. One way to achieve this is by having adequate recovery policies, even when a certain system fails. Adding redundancy can ensure that certain systems or services will still be accessible, even when some of the systems fail.

We can distinguish between vital and not vital services. An example of vital services are crash avoiding sensor information or emergency call information. Below, we provide a list of elements for which the availability should be preserved:

- A lack of availability of vital systems can cause physical damage to the driver and financial or reputation damage to the third party/OEM.

- The automatic emergency call (eCall) should be available at any time. Unavailability might lead to a delay in emergency services and could cause unnecessary damage to the passengers of the car.

- The availability of driving information of the driver is important to the OEM. They will partially base and modify their car models on the driving behavior of the driver.

3.3.4 Non-repudiation

Non-repudiation is defined as a property that is achieved through methods “which prevents an individual or entity from denying having performed a particular action related to data” [25]. Non-repudiation is closely related to the integrity property, however, integrity merely focuses on detecting changes in the message and non-repudiation merely focuses on being able to link a message to a person.

The non-repudiation requirement is important to both the car driver and the third party as an entity or component involved in malicious activity cannot deny his actions within that activity. With respect to the transmission of messages, the car cannot deny having sent certain information towards a third party and a third party cannot deny having asked for certain information.

3.3.5 Authentication

Authentication is defined as “the act of verifying the credentials of an entity”. A server can demand that the authenticating party proves the knowledge of certain credentials, which the server, in turn, can verify.

Authentication is very important to the car owner, as this can prevent an adversary from posing as a legitimate entity. When an adversary gets wrongly authenticated, he might be authenticated as a user with higher or other clearances. Similarly, cars/car owners should not be authenticated as different car owners, because this might lead to wrong authorization.

3.3.6 Authorization

Authorization is defined as “the act of determining whether an entity will be allowed access to a resource”. The authorizing party can determine whether access is provided to the requesting party.
Authorization should guarantee that access to components or services by unauthorized users is prevented. For instance, remote functionality which allows the car owner to control certain functions of his car should not be authorized to other parties. Wrong authorization can also result in access to (personal) data of other users.

### 3.3.7 Accounting

Accounting is defined as “the act of collecting information on resource usage for the purpose of capacity planning, auditing, billing, or cost allocation”. Accounting is very useful for determining who has accessed certain information and therefore that information is evidence of certain actions and events with respect to connected cars. This evidence can be used for a posteriori checks.

Accounting, with respect to connected cars, should guarantee that a posteriori checks are possible on certain events. This information can contain the logging of the transmission between the car and the third party. In case of malfunction of the car, these loggings can provide evidence of whether there was a mechanical error or a security breach. These audit loggings can also be useful for both the third party and the car owner to view what information has been accessed by whom.

### 3.4 Prevention of attacks on the internal network

In Section 2.3 and Section 3.1 we have shown the attack surfaces and the attack threats in the context of connected cars. One of the attack surfaces is the cellular network. In Table 2.1, we have shown that the Telematics Module enables the remote data communication via a cellular link. When information from within the car is solely used for statistics by a third party and no messages need to be sent onto the CAN-bus, access to the CAN-bus should only be read access. A way to do this is shown in Figure 3.1. TX stands for transmitter and RX stands for receiver. When the communication line from the microcontroller to the bus is physically cut (indicated by the red cross), it is not possible for the ECU to send messages onto the CAN-bus network. This way a lot of security threats that affect an attack on the internal network of the car can be mitigated. However, when remote functionality such as remotely unlocking the car is needed, this does not provide a solution. In this case, security mechanisms should be in place that ensure that abusing the ECU is not possible. A good security measure to apply in this situation is ensuring that only the specific messages related to the remote functionality are allowed over the CAN-bus network.
Figure 3.1: Restricting access to read-access
Chapter 4

Privacy

In this chapter we attempt to define what is privacy in the context of connected cars and what can be considered a privacy violation. After that we summarize the legal requirements which are stated in the Data Protection Directive [18]. After we have listed the legal requirements we will define the technical privacy requirements which are based on the privacy threats.

4.1 The privacy concepts of Solove

Before defining the technical privacy requirements, we must first understand the concept of privacy. In conceptualizing privacy, we attempt to distinguish privacy from other things [26]. In other words, we look for a common set of necessary and sufficient elements that single out privacy as unique from other concepts [26].

In this section we explore the different attempts made in the literature to isolate a common similarity of privacy. Although every concept has its limitation, if it is interpreted in a general notion, it gives an idea to the concept behind privacy. This list of privacy concepts was constructed by Solove [26].

1. The right to be let alone

The article “The Right to Privacy”, written by Warren and Brandeis resulted in a raise of interest and awareness in privacy. The article is seen as the foundation of the current privacy laws in the United States [27]. The authors define privacy as the “right to be left alone”, which refers to a sentence in a treatise written by Judge Thomas Cooley in 1888 [28]. The underlying principle was, as the authors describe, “that of inviolate personality”. Until then, tort law was mostly protecting the physical aspect of a human being and not the psychological aspect as can be seen in “Olmstead v. United States”, where the Supreme Court stated that “wiretapping was not a violation under the Fourth Amendment because it was not a physical trespass into the home” [29]. As a reaction to that, Brandeis wrote a dissent in which he stated that the Framers of the Constitution conferred to the violation of the right to be left alone [29]. In Katz v. United States [30], justice Fortas adapted Brandeis’ view and interpreted the right to be left alone as “the right to live one’s life as one chooses, free from assault, intrusion or invasion except as they can be justified by the clear needs of community living under a government of law” [31]. Clearly, one can see that the definition of the right to privacy as the right to be left alone has its limitations. The definition points into a direction, but fails to elaborate on how privacy should be valued against other interests, i.e. when we should be let alone.

2. Limited access to the self

Closely related to the right to be let alone is another concept of the right to privacy, namely the “limited access to the self”. In other words, limited access to the self can be explained as
“the desire for concealment and for being apart from others” [26]. However, it cannot be seen as equivalent to solitude, which is more reaching towards being alone. Sissela Bok defines privacy as “the condition of being protected from unwanted access by others – either physical access, personal information or attention.” [32]. David O’Brien’s view on the limited access concept as the right to privacy is that there is a distinction where some see limited access as a choice, while others see it as a state of existence. Arguing for the “state of existence”, O’Brien state that “Privacy is not identical with control over access to oneself, because not all privacy is chosen. Some privacy is accidental, compulsory, or even involuntary.” [33]. In order to select whether some matters should be considered private, it must be defined what matters constitute a privacy violation and what not. Another attempt to give a more specific definition for privacy in the light of limited access is made by Gavison, where he states that limited access consists of “three independent and irreducible elements: secrecy, anonymity and solitude” [34] and limited access should not be given to matters which involve solitude, secrecy and anonymity. While this gives a more specific interpretation for limited access, in some cases it is too strict as some matters will not fall under this definition where they actually should. One example of this is the collection, storage and computerization of information, which often do not reveal secrets, destroy anonymity or thwart solitude.

3. Secrecy
Privacy is commonly seen as an establishment on the secrecy of certain matters. Secrecy can be seen as a subset of limited access to the self, i.e., a way to limited access to the self. Information privacy is a constitutional right which has its foundation in this concealment of information. According to Judge Richard Posner, privacy can be divided into two interests, i.e., being left alone and the concealment of information [35]. He defines the concealment of information as the “right to conceal discreditable facts about himself” [36]. His opinion is that when people are concerned about their privacy, “they want more power to conceal information about themselves that others might use to their disadvantage” instead of the desire of seclusion [36]. In a legal contexts, once something is disclosed it can no longer be private with respect to the concept of privacy as secrecy. One can argue that the concept of privacy as secrecy is too strict, as there is no distinction between disclosing information to a specific group and disclosing information to the public, total secrecy vs. selective secrecy. Philosopher Julie Inness states that in privacy as secrecy there is no element of control, while the function of privacy might be to have control over certain aspects of one’s life [37].

4. Control over personal information
Another way to describe privacy is having the control over personal information. Alan Westin defines privacy as “the claim (...) to determine (...) when, how, and to what extent information (...) is communicated to others” [38]. Similarly Charles Fried defines privacy as “the control we have over information about ourselves” [39] and the Supreme Court stated that privacy is “control over information concerning his or her person.” [40]. The control over personal information notion of privacy can, similarly to secrecy, be seen as a subset of the limited access to the self concept. It excludes non-informational matters which can certainly be privacy violations, so the formulation might be too narrow. When information is included in the definition, it must be defined what types of information should be in control. The next challenge is to describe what control over information means. Control over information is often seen as a form of ownership in that particular information. However, information can easily be transmitted from one to another. Moreover, the revocation of information is hard as information can easily be remembered, hence a claim of privacy is not equal to the claim of ownership.

5. Personhood
Privacy can also be defined as a form of protecting personhood, which is quite different from previous ways of describing privacy as the definition of personhood basically protects the integrity
of the personality. One form of personhood is the protection of individuality, which Edward Bloustein defines as the protection of “demeaning to individuality, (...) an insult to personal dignity (...) or an assault on human personality.” [41]. Others define it as respect for individual’s capacity to choose. Another perception of privacy in the light of personhood is that this is not a privacy issue, but a liberty and autonomy issue. Personhood can also be described as antitotalitarianism where the state may not interfere with our identity. Often in privacy as the protection of personhood, the term personhood is not well defined. Another limitation is that personhood is mostly focused on state intervention and not on the private sector.

6. Intimacy

The last term often associated with privacy is privacy as a form of intimacy. Intimacy “expands moral personhood beyond simple rational autonomy” [42], which means that relations between human beings is also important to privacy. Intimacy can be defined as the motives of an individual that matter. All relationships between people are intimately different, some actions might be considered a privacy infringements in one relationship where this is not the case in another relationship. Another definition for privacy as intimacy is “the consciousness of the mind in its access to its own and other bodies and minds, insofar, at least, as these are generally or specifically secluded from the access of the uninvited.” [43]. This claim is quite similar to access to the self, but fails for the same reason in scope. Most of the definitions for privacy as intimacy have an inadequate scope, where the definition is too narrow or too broad.

All in all, the differences between the interpretations of privacy are very subtle. Most of definitions have a change in scope compared to the other definitions. The definitions end up being too narrow (excluding matters that should not be excluded) or too broad (including matters that should not be included).

4.2 Privacy threats

With the development of “Connected Cars” the market of current vehicles is moving towards vehicles with remotely controlled functions and access to data in the vehicles [4]. This also means that a malicious party might be able to remotely access or remotely gather information from the vehicle. Car manufacturers may be using the data available in cars to gather all kinds of information including personal identifiable information (PII). PII is defined as “any information about an individual maintained by an agency (...) such as name, Social Security number, date and place of birth, mother’s maiden name, or biometric records and any other information that is linked or linkable to an individual (...)” [44]. It has several implications to the data subject, listed below.

As a consumer, you might have a problem with the collection of your personal data by a car manufacturer or dealer [4]. Given the possible problem from a customer perspective, we have enumerated some privacy threats with their impact below.

- Government agencies such as tax and criminal investigation agencies might have a specific interest in the telematics data or personal data. As an example, a criminal investigation might benefit from the (GPS) data collected by the car manufacturer when an accident has happened. It is stated in Article 13(1d) of the Data Protection Directive (DPD) that some obligations and rights given in the Directive may be restricted if this is a necessary measure to safeguard “the prevention, investigation, detection and prosecution of criminal offenses, or of breaches of ethics for regulated professions”. Accordingly, the Member States may adopt legislation that enables the police to claim the information gathered by the car manufacturers. The impact in this scenario is quite high while the data was originally not collected for that purpose.

- Another possible attack is when tax agencies want to check whether certain tax statements are legitimate and a criminal investigator might be able to prove where a person has been at
a specific point in time. Again, the effect is that the data is used for another purpose than it was originally collected for.

- Insurance parties also have an interest in the telematics data. If they can prove that the actions of the claimant were in violation of the traffic rules, the claimant may have to bear his own losses or be held liable in person for the losses of a third party. This frees them from having to pay for high claims.

- Another situation is where the car manufacturer might have a particular interest with respect to warranty of the car. If they have monitored strange behavior with the car, such as excessive use of the motor, they might not be responsible for some defects on the car, which could save them a lot of money.

- The same car manufacturer can also use the collected personal information for marketing purposes, where they could use the information for targeted offers or sell the personal information in a contract with other companies who will use the information for targeted offers and services such as maintenance planning with selected repair shops when issues are found.

- The privacy of the user can also be violated if the driving behavior of the driver is being recorded and the information is used to offer the driver deals relating to his behavior. This can range from restaurants to offers related to the replacement of the tires of the car. The car owner might not like that the information from his car is being used for targeted advertisement.

4.3 Legal privacy requirements

This section describes the legal privacy requirements as laid down in the legal documents such as the Data Protection Directive [18] and the European Convention on Human Rights [45]. The implementation of the directive in the Netherlands, the “Wet bescherming persoonsgegevens” (WBP) is not being described as it is based on this directive and does only add more detail to the existing directive. Moreover, the WBP will be replaced by the “Data Protection Regulation”, this regulation is still pending. The reason to not describe this regulation yet is because it is not final yet and can still significantly change.

4.3.1 Data Protection Directive

The European Data Protection Directive (DPD) [18] protects individuals with respect to processing of their personal data and the free movement of such data. We will show how some of these articles are applicable to the privacy of data subjects with respect to cars and give an overview of the most relevant articles of the DPD.

Article 2 gives a definition for personal data which refers to it as “any information relating to an identified or identifiable natural person”. An identifiable person is defined as “one who can be identified, directly or indirectly, in particular by reference to an identification number or to one or more factors specific to his physical, physiological, mental, economic, cultural or social identity.” In the context of connected cars, personal data can range from the geographical location and vehicle identification number (VIN) to processing contact information and browsing data, as it (indirectly) identifies an individual person. Processing of personal data is defined as “any operation or set of operations which is performed upon personal data, whether or not by automatic means”. This means that once the car manufacturer does anything with this data (even collecting), it is considered the processing of personal data.

Article 3 states that this Directive does not apply “in any case to processing operations concerning public security, defence, State security (...) and the activities of the State in areas of criminal law”. As a consequence, the processing of personal information by the Member States...
under these conditions is only bound to their national law and European Treaties, such as the European Convention on Human Rights [45], of which Article 8 (Right to respect for private and family life) is in particular applicable.

**Article 6** lays down the principles relating data quality, which basically means that the quality of personal data should be sufficient. It says that the controller must ensure that the personal data is processed “fairly and lawfully” and should be “collected for specified, explicit and legitimate purposes”. That is, the personal data can only legitimately be processed when these conditions are met, which means that the data processor should explicitly specify for what legitimate purpose he is collecting the personal data. Furthermore, the processing must be “adequate, relevant and not excessive” and “accurate and kept up to date”. The importance of these principles can be illustrated in the connected car context, where for instance a geographical location should only be processed if it was collected for an explicit and legitimate purpose. Article 6 also states that personal data should be “kept in a form which permits the identification of data subjects for no longer than is necessary for the purposes for which the data were collected or for which they are further processed”. In the connected car context, when a car manufacturer or any other party is collecting personal data from a car driver, that party is considered a data processor and must follow these principles. This means that when that party is processing the geographical location explicitly and solely for a one-time purpose of providing the driver with the location of his car, the processor cannot keep the location in its database as that would be excessive and longer than necessary.

**Article 7** specifies the criteria for making data processing legitimate. The processing of personal data may only be processed if “the data subject has unambiguously given his consent” or processing is necessary for a contract with the data subject as a party, a legal obligation, the vital interests of the data subject, the public interest or the legitimate interests by the controller. Especially the last condition (legitimate interest) is important in the context of connected cars. This means that if the controller has a demonstrably legitimate interest in processing the personal data, he can process them without consent as long as it is not in violation with the fundamental rights and freedoms of the data subject, stated in Article 1(1) [18]. Whether an interest is legitimate and not in violation with the Charter of Fundamental Rights of the EU is often decided by the European Court of Justice in a legal case and might differ on the specific situation. As an example, the process of buying a new car possibly involves signing a contract or consent. When the car is transferred to a new owner, the consent the car manufacturer applies does not apply to the owner of the car anymore and the situation results in processing the personal data of the new owner while he has not given his consent.

**Article 10** basically states that the controller should inform the data subject about “the identity of the controller”, “the purpose of processing” and any other necessary information. Other necessary information can be information about the recipients of the personal data and the existence of the right of access and rectify. This Article is interesting in the context of the privacy related to connected cars, as the car manufacturer (controller) is obliged to inform the data subject regarding its processing activities. If you take the same example, the second-hand car, the controller has to inform the data subject (the new car owner) about his processing activities.

**Article 12** specifies the right of access to information about whether data relating to the data subject are being processed, the purpose of processing, the categories, the recipients, the logic involved in any automatic processing and the rectification, erasure or blocking of data. This means, the data subject must be able to get information regarding the processing activities of the controller (the car manufacturer).

**Article 17** states that the “controller must implement appropriate technical and organizational measures to protect personal data against (...) loss, alteration, unauthorized (...) access, in particular where the processing involves the transmission of data over a network”. For the automotive
industry, this means that in particular the transfer of data towards the car manufacturer must be appropriately protected, but also that the storage of this data must have appropriate security measures.

**Article 25** basically states that if the personal data is transferred to a country outside the EU, the country in question has to ensure an adequate level of protection. With respect to the controllers in car security, this means that the transference of personal data from within the EU to an inadequately protected controller who is not situated within the EU is in principle prohibited.

**Article 26** lays down the conditions for the transfer of personal data from Member States to inadequately protected countries outside the EU. These conditions are basically the same as stated in Article 7, with the exception of Article 7(f). The difference is that where car manufacturers could rely on a legitimate interest (Article 7(f)) within the Member State and within Europe, a legitimate interest is not sufficient for the transfer of personal data to non-EU countries. The condition consent might be needed to transfer the data to insufficiently protected third countries. However, other treaties such as the International Safe Harbour Principles could be more specific in what is possible and what is not.

An example where this transfer of data between countries could become a problem is when a car manufacturer is situated in a non-EU country with insufficient protection and the data subject drives within a Member State of the EU. The transfer of personal data to the non-EU country might be prohibited and the car manufacturer might be in violation with the DPD.

Given the above Articles, the European Data Protection Directive is applicable to the car manufacturer when the data which is being processed by the car manufacturer contains personal data. To legitimately process the data, the data subject should either have given his consent or the car manufacturer should have a legitimate aim. Moreover, the period identification of the personal data is possible should be kept to a minimum. The data controller is obliged to inform the data subject about his processing activities on the personal data of the data subject. Next to all these conditions, the data controller also needs to implement appropriate technical and organizational measures to protect the personal data from unauthorized access, alteration and loss, which refer to the classical CIA principles (see Chapter 3).

### 4.3.2 Convention for the Protection of Human Rights and Fundamental Freedoms

The Convention for the Protection of Human Rights and Fundamental Freedoms, also known as the European Convention on Human Rights (ECHR), is an international treaty which protects the human rights and fundamental freedoms in Europe. The treaty, which was signed in 1950, affects all the member states of the Council of Europe.

In Article 7 of the Data Protection Directive, it is stated that one of the conditions on which personal data may be processed is the condition that “processing is necessary for the purposes of the legitimate interests pursued by the controller (...) except where such interests are overridden by the interests for fundamental rights and freedoms of the data subject”. This is where Article 8, the Right to respect for private and family life, of the Convention for the Protection of Human Rights and Fundamental Freedoms might be relevant. Before an applicant can appeal to the European Court of Human Rights, he will first need to exhaust domestic remedies. The local courts have to follow the European treaties and regulation, of which the Charter of Fundamental Rights of the EU is relevant in this context. The national courts may ask for the opinion of the European Court of Justice during a case where it is not clear whether it is in violation with certain rights, which is especially in cases where no relevant case law is available. When the applicant still feels that his human rights were violated, there is still the possibility to appeal to the European Court of Human Rights, which will only happen in special cases. The European Court of Human Rights will test the judgment of the national courts against the European Convention of Human Rights.
The most relevant part of Article 8 of the ECHR in the light of the situation described above is
the first part, Article 8(1), which states that "everyone has the right to respect for his private and
family life, his home and his correspondence.". The second part, Article 8(2) states the exceptions
on which this right may be violated by a public authority, i.e., "There shall be no interference
by a public authority with the exercise of this right except such as is in accordance with the law
and is necessary in a democratic society in the interests of national security, public safety or the
economic wellbeing of the country, for the prevention of disorder or crime, for the protection of
health or morals, or for the protection of the rights and freedoms of others.".

In Europe, there has been a case (Germany v. Uzun) in which Mr. Bernhard Uzun appealed
to The European Court of Human Rights, where he claims that GPS tracking of his vehicle was in
violation with Article 8 of the ECHR, because it disproportionately interfered with his private life.
Mr. Uzun was suspected of participation in bomb attacks for which the Anti-Imperialist Cell had
claimed responsibility. Germany states the GPS tracking was necessary in a democratic society
which is in accordance with the Article 8(2) of the ECHR and therefore had a legal basis to do so.
The European Court of Human Rights stated that the use of GPS tracking was a proportionate
interference due to the gravity of the offenses he had been suspected for.

Although this case is not directly applicable to the case where car manufacturers can gather
your GPS location, the case gives some knowledge about how excessive GPS tracking is with
respect to the privacy of a human being. The European Court of Human Rights concludes that
the applicant’s observation via GPS amounted to an interference with the applicant’s private life
as protected by Article 8(1) of the ECHR [46]. The Federal Public Prosecutor General’s decision
to order the surveillance via GPS was in this case found legitimate on the grounds of necessity
in a democratic society. However, this fundamental right to privacy is also applicable between
a controller (car manufacturer) and a data subject, where the data controller probably cannot
satisfy the second part of Article 8 of ECHR.

4.4 Privacy requirements

Third parties with possession of personal data might construct a database containing personal
information about the geographical location of your car, the fuel consumption and your browsing
history.

In the next paragraphs we will outline, in our opinion, important privacy requirements which
are desirable for the data subject to protect his or her privacy. In the outline of these requirements,
we use three parties which are relevant with respect to telematics data, i.e., the sending party (the
car owner/driver), the communication network (the Internet) and the receiving party (the third
party).

4.4.1 Anonymity and pseudonymity

A requirement from a data subject could be that he does not want to be identifiable within the
given dataset. Identifiability is defined as “the state of being identifiable within a set of subjects,
the identifiability set” [47]. Anonymity and pseudonymity are sometimes confused because both
terms highly relate to de-identifying an identifiable object. Pseudonymization, according to [48],
is the replacement of true identities of entities in databases by pseudo-identities that cannot be
linked directly to their corresponding identities. Anonymity is the state of being not identifiable
within a set of subjects, the anonymity set [47].

Anonymity is often seen as a stronger notion than pseudonymity, because with anonymity a
subject cannot be identified within its set, which means that given multiple entries from the same
person in a dataset, one cannot identify the two entries as the same identity. With pseudonymity
however, one can link multiple entries but the original data subject cannot be identified. Another
type of anonymity is $k$-anonymity. A data subject is said to have $k$-anonymity when the infor-
mation released about that identity cannot be distinguished from at least $k - 1$ other identities
within the same dataset [49].
CHAPTER 4. PRIVACY

In the setting relevant to this research, the Connected Car setting, both the SIM and the server are owned by the car manufacturer. This means that anonymity is hard to obtain, because when there is data being sent to the server by the car, the car manufacturer might know (depending on the given environment by the ISP) to which SIM card and thus which car the data belongs, as the IP address is in the header of the IP packet on the communication layer. In the application layer, cryptographic signing might reveal to whom the data belongs, depending on how the public key infrastructure is used. Anonymity might be achieved by not directly talking to the server of the manufacturer.

Pseudonymization is easier to accomplish, as the real identifiable information of the sender only has to be pseudonymized. The relation between the characteristics is still intact, but the information behind is replaced by pseudonyms.

4.4.2 Untraceability

Normally, untraceability defines that for two records of an object or message, it cannot be determined whether they belong to the same object or not [50]. For instance, a MAC address that is unchanged can be traced within a shopping mall. It can be made untraceable when the MAC address is constantly changing. With respect to a connected car, specifically the GPS location can be used to trace a car. However, the traceability might be limited when only a limited amount of the geolocations is being collected, but the more locations the higher the chance that it can be linked to an identity. A data subject might not want his geolocation data to be recorded, but the telecom companies are obliged to record to which base stations the phone is connected to by the data retention directive [51]. Although this is less specific than geolocation, the data subject can still be traced. Moreover, metadata, source address, destination address and time of transmission, is also being collected by the telecom companies.

4.4.3 Unlinkability

The unlinkability requirement is satisfied when two or more items of interest occurring in the system cannot be related, because the probability that these items are related stays the same, before and after the attacker’s observation [47]. With respect to connected cars, items of interest could be the messages sent towards the car manufacturer and the messages sent towards the car owner. An attacker should not be able to relate multiple messages sent towards the car manufacturer. For the car manufacturer it should be impossible to link the message to the car owner.

4.4.4 Revocable privacy

Revocable privacy is the guarantee of an architecture that personal data is revealed only if a predefined rule has been violated. In the setting of this research, we add the notion that revocable privacy can also be revealed if the data subject wants to reveal his personal data. As an example of this nature, think of geolocation that is protected by the system architecture and can only be revealed by the data subject. This information could become handy in tracing the car when the car is stolen [52].

4.4.5 In control

Somewhat related to revocable privacy, we say the data subject is in control when he has the ability to modify or remove his personal information from a database where personal information is stored at any given moment in time. This is similar to privacy design strategy 6 which is being described in [11] as “agency over the processing of their personal information”. This is also related to the latest General Data Protection Regulation [53], where the Right to be Forgotten has been defined in Article 17. However, this does not mean the data subject has to be in control, but he has to have the ability to be forgotten. It would be desirable for the data subject to be able to
control which elements he wants to share with the car manufacturer, as shown in [4], where 54% of the respondents only want to share their data under certain conditions.

4.4.6 Data minimization

The data minimization principle derives from Article 6.1(b) and (c) of the DPD [18], which states that the personal data must be “collected for specified, explicit and legitimate purposes” and need to be “adequate, relevant and not excessive in relation to the purposes for which they are collected and/or further processed”. If the data subject consents to the collection of his personal data, he can still demand that the data that is being collected is only the strictly necessary information.

4.4.7 Non-profiling

The non-profiling requirement is that the data controller is not able to generate a profile of an identity based on the information it receives from the data subject. We see data processing as non-profiling when the goal of profiling is not targeted at profiling an identity. As an example, the car manufacturers want to be able to profile what conditions cause cars to have problems with their engines. Think of special offers to specific services which the car indicates it needs. A car manufacturer can make a deal with car repair shops. The car could indicate, in case of a low oil level or flat tires to direct to one of the nearest repair shops and indicate a special deal. Another example is of course fuel consumption and give a special deal when the gas tank is running out of gas.
Chapter 5

Overview of possible trust architectures

In this chapter we give an overview of the possible trust architectures, i.e., how the involved parties can communicate with each other and what the benefits and downsides of the different types of architectures are. After reading this chapter, the reader should have a clear understanding about the possible architectures, their benefits and downsides and the roles involved.

Given all the security and privacy requirements described in previous chapters, we need a system in which the car owner is as much as possible in control of the data his car produces and transfers it towards any other party. The system must satisfy the security and privacy requirements as much as possible. The processing party should only be able to perform the actions on the data which we define as aggregation. These operations should only be possible if they are allowed by the car owner. However, there are also some considerations about efficiency (communication load and computational load) and trust. Communication load is the amount of data that is being transmitted over the communication line and computational load is the amount of effort a computing unit (microprocessor) has to perform to achieve a certain goal.

5.1 Roles

The idea is that there is a data center for a set of cars, where these cars do not necessarily belong to the same brand. The car sends data to the data center for processing and aggregation. The data center can perform operations on the data, but has the least possible knowledge about the content of the data which is being stored in the data center. The data center is allowed to store the data, but should only send data to the third party service provider that is the least privacy infringing for the car owners, e.g. aggregate data with other data from other cars. The car owner must be able to explicitly give access to data which is stored in the data center. As indicated in Section 2.1, when data is needed for statistical analysis by the third party service provider (TSP), the data can also be aggregated in the data center before it will be sent to the third party service provider. Before we can give a description of the different architectures, we need to define the roles of the involved parties.

5.1.1 Car/Car owner

The car owner is the entity that stays the owner of the data his car produces by using a cryptographic system that allows this property. He must be the holder of a key, which allows the owner to decrypt any message which is being sent from the car. The car owner has the ability to generate and delegate a key which gives access to an aggregated set of the messages or an individual message. The car/car owner will generate a specific key and provide that key to the proxy/semi
trusted third party which will allow him to re-encrypt the aggregated message. The key will only work on a specific set of messages which the car/car owner has determined.

### 5.1.2 Data center/(Semi-)Trusted Third Party

The data center/(semi-)trusted third party is a facilitating party who is not allowed to decrypt the individual messages, but its role is to transfer individual messages or aggregate the data by performing mathematical operations on the (encrypted) data and transfer them after these operations. This party should aggregate the data in such a way that only the third party will be able to decrypt aggregated content. Ideally, it should be impossible for the data center/semi-trusted third party to collude with a third party service provider and be able to decrypt unaggregated messages which were not intended to be decrypted by that party or discover the secret key of the car/car owner. Also, it must not be possible for the data center/(semi-)trusted third party to decrypt or aggregate any future messages from the car to the third party service provider.

The data center must be (semi-)trusted, as it is the only party who has possession of the data. It has still the ability to perform a denial-of-service attack by not sending the data to the other parties. Furthermore, it has the ability to mess up the data which will result in corrupted data. The TSP also has the knowledge about what entries belong to what party, but we will elaborate more on the trust relations in the security section.

### 5.1.3 Third Party Service Provider

The third party service provider is the party who wants to know particular information from a car type or a specific car, for instance the average fuel consumption, average speed or the GPS location. The third party service provider will need to request the data center/(semi-)trusted third party for a specific type of information and a specific time range regarding one or multiple vehicles. Although this information can be collected by the data center for the third party service provider, in some situations the data center still has to request a key from the car/car owner to reveal the information.

### 5.2 Possible architectures

In this section we describe the setup of the possible trust architectures with the corresponding aggregation options. That is, aggregation on the car-side, TSP-side or (s)TTP-side. We will describe the advantages and disadvantages for the different aggregation options.

We distinguish between the following possible architectures:

- Direct communication architecture between car and TSP with TSP-side aggregation
- Direct communication architecture between car and TSP with car-side aggregation
- Semi-Trusted Third Party architecture with sTTP-side aggregation
- Semi-Trusted Third Party architecture with car-side aggregation
- Trusted Third Party architecture with TTP-side aggregation

### 5.2.1 Direct communication architecture with TSP-side aggregation

The first scheme is what we call the direct communication architecture, which is similar to a peer-to-peer network. In such a scheme, the cars are directly connected to the TSPs. As shown in Figure 5.1, another party such as a tracker might be needed to connect the parties. The Figure serves as an illustration of this scheme, where car, connects to a TSPx via a direct link. In this variant we will describe the advantages and the disadvantages from this architecture where the aggregation happens on the TSP-side.
Advantages
- Aggregation is more flexible on plaintext data compared to ciphertext data, as aggregation on ciphertext data is restricted to the capabilities of the cryptosystem and aggregation on plaintext data is not. This makes aggregation on plaintext data more attractive for the TSP compared to aggregation on ciphertext data.
- Changes in aggregation can be updated in a central place, that is on the server of the TSP, without involving the car. As the raw data is stored on the server, changing the aggregation rules does not affect the car.

Disadvantages
- Re-transfer of data is needed if different third party service providers need the same data. In that case, the data needs to be re-transferred from the car to a different TSP, while the data might be the same as an earlier transfer to a TSP.
- Anonymity is hard to obtain, as the TSP knows who is sending the data, because the TSP and the car are directly connected to each other. This allows the TSP to link an aggregated intermediate result to a specific car.
- In this case, the raw data is being aggregated on the TSP-side, so the unaggregated data which is being sent to the TSP could potentially contain sensitive data.
- The unaggregated data of the cars is stored on the server. This is a disadvantage as the car owner has no control over what aggregation will be performed on the data. Moreover, since the raw data is now in hands of the TSP, the car owner has to trust the security mechanisms of the TSP.

5.2.2 Direct communication architecture with car-side aggregation

With direct communication between the car and the TSP, there is also the possibility of aggregation in the car. Instead of sending the data in unaggregated form to the TSP, the data is already aggregated in the car. However, the aggregated result that will be shown to the TSP, will only reflect one car.

Advantages
- Raw (and potentially sensitive) data is not stored on the server of a TSP. In other words, the car owner has more control over his data. The data that is transferred to the TSP is already aggregated and does not contain raw (and potentially sensitive) data.
CHAPTER 5. OVERVIEW OF POSSIBLE TRUST ARCHITECTURES

Figure 5.2: Semi-Trusted Third Party Architecture

Disadvantages

- Re-transfer of data is needed if different third party service providers need the same data. In that case, the data needs to be re-transferred from the car to a different TSP, while the data might be the same as an earlier transfer to a TSP.

- When data is being aggregated in the car, this generates a high computational load for the cars, because the data every car generates has to be aggregated in every car separately.

- When aggregation changes, the software to aggregate has to be updated accordingly. That is, the software in the car needs an update which will contain the latest aggregating rules. Moreover, the firmware needs to be compatible for every individual car.

5.2.3 Semi-Trusted Third Party (sTTP) Architecture with sTTP-side aggregation

The “Semi-Trusted Third Party (sTTP) Architecture”, is shown in Figure 5.2. The idea behind this architecture is that there is one server which belongs to a semi-trusted third party. That party is not fully trusted, so the information that party receives must be kept to a minimum. Due to the nature of the architecture (not trusting the TSP on its aggregation) aggregation on the TSP server does not apply.

Advantages

- Aggregation is more flexible on plaintext data compared to ciphertext data, as aggregation on ciphertext data is restricted to the capabilities of the cryptosystem and aggregation on plaintext data is not. This makes aggregation on plaintext data more attractive for the TSP compared to aggregation on ciphertext data.

- Changes in aggregation can be updated in a central place, that is on the sTTP, without involving the car. Since the raw data is stored on the server of the sTTP, changing the aggregation rules does not require communicating with the car.

Disadvantages

- No separation between the sTTP of the cars and the TSPs. There is a trust relation between the car and the sTTP as well as between the sTTP and the TSP. The raw car data is trusted to the sTTP which has a trust relation with both parties, who might have a concern about the data ending up in the wrong hands (which are other parties than intended).
Preferences regarding the privacy of the car owner (which type of information can be used in what way) have to be stored at the sTTP as the aggregation is done at the sTTP. When a car owner wants to change these settings, these settings have to be transferred towards the sTTP, who is trusted to process the data accordingly.

The car owner cannot enforce what data is being aggregated and therefore has no absolute control. The car owner has to trust the sTTP on properly processing his data.

5.2.4 Semi-Trusted Third Party (sTTP) Architecture with car-side aggregation

There is also a system architecture possibility which we call “Semi-Trusted Third Party (sTTP) Architecture with car-side aggregation”, which is similar to the previous architecture with the exception of the place the aggregation happens. That is, instead of aggregation on the server of the sTTP, the aggregation will happen on the car-side, which has its own benefits and downsides.

Advantages

- Raw (and potentially sensitive) data is not stored on the server of the TSP, but on the car-side. This way, the car owner has more control over his data, because he can determine which information will be aggregated.

Disadvantages

- No separation between the sTTP of the cars and the TSPs. There is a trust relation between the car and the sTTP as well as between the sTTP and the TSP. The raw car data is trusted to the sTTP which has a trust relation with both parties, who might have a concern about the data ending up in the wrong hands (which are other parties than intended).
- When data is being aggregated in the car, this generates a high computational load for the cars, because the data every car generates has to be aggregated in every car separately.
- When aggregation changes, the software to aggregate has to be updated accordingly. That is, the software in the car needs an update which will contain the latest aggregating rules. Moreover, the firmware needs to be compatible for every individual car.

5.2.5 Trusted Third Party Architecture with TTP-side aggregation

The “Trusted Third Party Architecture”, is shown in Figure 5.3 where car\textsubscript{i} connects to TSP\textsubscript{x} through their respective TTP\textsubscript{ijk} and TTP\textsubscript{xyz}. The idea of this architecture is that the two groups (cars and TSPs) have strictly separated trusted third parties who they trust more than other parties, such as each other. The TTPs can be provided with keys or raw data from their respective group. When car\textsubscript{i} wants to send data to TSP\textsubscript{x}, the data will first be transferred through their respective TTPs, before it will reach TSP\textsubscript{x}. This results in a strictly separated environment. Due to the nature of the “TTP architecture”, there is no need for a distinction between car-side and TSP-side aggregation as this structure demands a trust on their respective TTPs. Due to this property, the aggregation will always happen at the corresponding TTP.

Advantages

- The unaggregated data is not stored on a central server which both the car owners and TSPs need to trust. That means that there is no overlap in trust relations, which is desirable for both parties. A result of this set up is that an incident at one party has limited effect on the other parties.
• Aggregation is more flexible on plaintext data compared to ciphertext data, as aggregation on ciphertext data is restricted to the capabilities of the cryptosystem and aggregation on plaintext data is not. This makes aggregation on plaintext data more attractive for the TSP compared to aggregation on ciphertext data.

• Changes in aggregation can be updated in a central place, that is on the server of the representative, without involving the car. As the raw data is stored on the server of the representative, changing the aggregation rules does not require communicating with the car.

Disadvantages

• The car owner cannot enforce what data is being aggregated and therefore has no absolute control. He has to rely on his representative.

• The representative has the possession over the raw data of its clients (i.e., cars or TSPs) which might form a security risk. The car owner needs to trust on the security of its representative.

• When the data of multiple car owners is stored in a central place (on the server of the representative) and there is a data breach, the data of multiple car owners will leak.

• Although it is the nature of this architecture, data accessibility of the server on the data of all its representative can still be seen as a disadvantage. It allows the representative to view all the data or even worse, the creation/alteration of data on behalf of the car owner.

• Preferences regarding the privacy of the car owner (which type of information can be used in what way) have to be stored at the representative. When a car owner wants to change these settings, these settings have to be transferred towards the representative, who is trusted to process the data accordingly.

• When a TSP requests data of cars that covers multiple representatives, the representative of the TSP needs to communicate with the different representatives of the multiple car owners. This results in a high communication load as the TSP needs to contact multiple representatives.
Chapter 6

Proposed architecture

In Chapter 5, we described possible architectural designs for the communication between the car and its TSP. This resulted in a list of designs of which we have shown the advantages and disadvantages. In this chapter we propose a design which in our opinion is the best option given the advantages and disadvantages. Moreover, we show techniques that can help in mitigating the disadvantages.

6.1 Technical requirements

The direct communication architecture either results in unnecessary retransmission of data or requires a high level of trust in the TSP. The TTP architecture requires an unrealistic revelation of data or secrets to the TTPs. The sTTP architecture either results in a single party who has the possession of all the data or impractical aggregation due to a high computational load and aggregation update procedures.

Given the advantages and disadvantages of the architectures, we can define specific technical requirements for the design of the architecture. The requirements for such an architecture are defined as follows:

- One of the most important requirements is achieving privacy for the car owners. This means that it is not self-evident that other parties beside the car owner himself have access to the raw data, unless explicit permission is given.

- Another important requirement is that the car owner is in full control of his own data. He decides whether certain parties get access to what type of information and in which form (raw or aggregated).

- It is also important to keep the bandwidth on the transmission of data as low as possible. This is valuable for the parties who need to pay the transmission of data.

- Moreover, the retransmission of data should be avoided as this is highly inefficient from a bandwidth perspective.

- Another requirement should be the ability of a car owner to change his privacy settings without the necessity to contact any other party. When the car owner needs to push his settings to a server, he has to trust that the server will process his request accordingly.

- No key escrow is also an important requirement, because without key escrow the car owner will have more control over his data. However, there should be a trapdoor which will always enable the car owner to view the data his car produced in the past.
6.2 Cryptographic techniques

In Section 6.1, we have defined the specific requirements that an architecture in our architecture would need to satisfy. The architectures in Chapter 5 have shown to be inadequate in its current form. However, the use of cryptographic techniques might be able to mitigate some of the disadvantages. Ordinary cryptography helps to Moreover, the techniques might enable some of the parties to perform a form of aggregation without the knowledge of the individual messages.

6.2.1 Homomorphic cryptography

Homomorphic cryptography is cryptography that allows arithmetic operations on the ciphertext domain. A cryptosystem that has the additive homomorphic property allows an operation on the ciphertext that will result in an addition within the plaintext domain. Mathematically we can describe this by \( f(\varepsilon(\alpha), \varepsilon(\beta)) = \varepsilon(\alpha + \beta) \), where \( f(x, y) \) is a function and \( \varepsilon(\alpha) \) is defined as the ciphertext of \( \alpha \). Similarly, a cryptosystem that has the multiplicative homomorphic property allows an operation on the ciphertext that will result in a multiplication within in the plaintext, which can mathematically be described as \( f(\varepsilon(\alpha), \varepsilon(\beta)) = \varepsilon(\alpha \cdot \beta) \). The given examples are known as partially homomorphic cryptosystems. There also exist fully homomorphic cryptosystems which allows both addition and multiplication. However, these fully homomorphic cryptosystems are still fairly slow and take a lot of computing power and disk space.

We will now show the advantages and disadvantages which specifically apply to homomorphic cryptography as opposed to ordinary cryptography:

**Advantages**

- The big advantage of applying homomorphic cryptography to the existing architectures is that it increases the privacy of the car owner, because the aggregating party does not have the decryption key.
- Due to the homomorphic properties of the cryptosystem, the encrypted data can directly be sent to the aggregating party, instead of the need to collaborate with other parties to achieve certain privacy goals. The only dependency is from an aggregating party (if the data needs to be aggregated) to a car, because that party needs the data to be able aggregate.

**Disadvantages**

- The computational load is higher for computing encrypted values compared to performing no operations for plaintext data. The car has to encrypt the values according to the cryptographic system. The execution time of the cryptographic system depends on the parameters of that system (e.g. key sizes).
- As a result, the bandwidth load is also larger compared to plaintext transmission.

6.2.2 Secure Multi-Party Computation (SMPC)

Secure Multi-Party Computation is a way of collaborative computations without the knowledge of the inputs of the other collaborating parties. That is, \( car_i, car_j \) and \( car_k \) compute together the function \( f(x_i, x_j, x_k) \) from their corresponding values \( x_i, x_j \) and \( x_k \). Let us assume the TSP wants to know a function over a certain set of values of a set of cars. The TSP would need to inform
the corresponding cars and ask them to compute a particular function together and release the output. There is no distinction between TSP-side and car-side aggregation as this architecture is designed to securely compute a function over certain values in a collaborative manner. We will now show the advantages and disadvantages which specifically apply to SMPC as opposed to ordinary cryptography:

Advantages

- The main advantage of this way of computing aggregated values is that it increases the privacy of the car owner, as it is impossible for the TSP to discover the individual values. The information about an individual is hidden within the aggregated information.

Disadvantages

- A large bandwidth load as the cars have to communicate multiple times with each other to be able to collaboratively compute the result of the function $f$. Every car needs to communicate with every other car which is connected to the same SMPC-group.

- A high computational load as multiple values have to be computed individually for every collaborative computation between two parties of the secure multi party computation.

- There is a high dependency on the other involved parties.

- Completely unrealistic when the values need to be computed through a Car-to-Car channel, because the vehicles would constantly be moving. This might not be the case when the data is transferred over the Internet.

6.3 Choosing the architecture: sTTP

In the past sections we have defined the technical requirements which need to be satisfied by the chosen architecture and we have shown cryptographic techniques which can help in improving the privacy of the car owner. In this section we will compare the possible architectures described in Chapter 5 and try to apply the techniques described in Section 6.2.

One of the biggest problems in the architectures described in Chapter 5 is that some other party besides the car/car owner is able to view the data which is being transferred or aggregated, without explicit permission from the car owner. A possible solution to this problem is the use of the cryptographic techniques we showed in Section 6.2, which might allow a set of messages to be aggregated without revealing the content of the messages.

Due to the high dependency on other parties to be able to compute a function over the data makes SMPIC not feasible. However, the use of homomorphic encryption seem to be useful to the architectures. Let us compare the architectures of Chapter 5 with the application of homomorphic encryption:

- **Direct communication architecture.** We have described two versions of the “Direct communication Architecture”, with aggregation on the car-side and aggregation on the TSP-side. Homomorphic encryption is not needed in the car-side aggregation as the data is generated there and therefore is also available in plaintext. In the TSP-side aggregation, homomorphic encryption can be quite useful, because it will ensure that the TSP will not have any insight in the data but still is able to aggregate according to the homomorphic properties of the cryptosystem. However, the storage and data transfer is not optimal as retransmission of data from the car to the TSP is still needed when multiple TSPs want the same data.
Semi-Trusted Third Party Architecture. In the “sTTP Architecture”, we can distinguish between two versions where aggregation happens either in the car or on the sTTP side. Again, the use of homomorphic encryption does not add anything to the car side aggregation. For the aggregation on the sTTP side, it can be very helpful. It allows the car owner to be in control of his data. The sTTP cannot read the raw data, but can perform operations supported by the homomorphic properties of the cryptographic system.

Trusted Third Party Architecture. The nature of the “Trusted Third Party Architecture” is to trust a representative with the raw data. When homomorphic encryption is being used, the representative has no insight anymore in the plaintext data. This can be a good thing, but there is no need for separated representatives anymore, because the representative has no knowledge about the information that is being processed anyway. Moreover, due to the nature of trusting the representatives with your keys or raw data, the use of homomorphic encryption does not make any sense.

In our opinion the best solution is the Semi-Trusted Third Party Architecture. This is due to the nature of the TTP Architecture, where homomorphic encryption does not make any sense. Furthermore, direct communication has an overhead over the sTTP Architecture due to retransmission. Hence, the sTTP seems the best option.

In the Semi-Trusted Third Party Architecture the aggregation either happens in the car or in the sTTP. Some disadvantages can be mitigated by using cryptosystems with homomorphic properties. Using this technique, the preferences regarding the privacy can be stored on the carside and the car owner can be in full control of his data by only releasing certain keys that will only decrypt an aggregated end-result. We will explain the system in more detail in the next subsections.

6.4 Sequence diagram for storing data

In this and the next sections, we will describe explicitly how the flow of information is. These sequence diagrams should help the reader to understand how the parties operate with each other. Please note that this is a conceptual information flow to show the interaction between the parties and not a technical message flow which would represent a protocol. This means that if it is interpreted as a cryptographic protocol, it will be subject to numerous attacks. The communication between parties should happen over a secure channel. In Figure 6.1, we show step-by-step how the sTTP and the car should communicate to store the information on the sTTP.

Step 1: TransmitData

In this step, data is continuously sent from the car to the data center (sTTP). This will happen at predefined intervals, which can be determined by the car owner in his privacy settings. These intervals can, for instance, be based on time or distance. The data is encrypted with a key which only the car/car owner has. We denote TransmitData(carID, i, type, Ci) as the function that transmits the ciphertext from carID to the sTTP, i is the identifier that uniquely identifies the ciphertext and Ci is the ciphertext. Additionally, the type of the message is transmitted.

Step 2: StoreData

In the second step, the data will just be stored in the data center in its encrypted form. No aggregation of the data will take place yet, only the time of reception, the type of car and the type of message will additionally be registered. The type of message will contain whether the ciphertext for example contains GPS-data, error messages or telematics data. We denote StoreData(carID, i, type, Ci) as the function that stores the data Ci together with identifier i and carID and other additional properties such as the type of the message.

In this section, we will describe explicitly how the flow of information is. These sequence diagrams should help the reader to understand how the parties operate with each other. Please note that this is a conceptual information flow to show the interaction between the parties and not a technical message flow which would represent a protocol. This means that if it is interpreted as a cryptographic protocol, it will be subject to numerous attacks. The communication between parties should happen over a secure channel. In Figure 6.1, we show step-by-step how the sTTP and the car should communicate to store the information on the sTTP.
CHAPTER 6. PROPOSED ARCHITECTURE

6.5 Sequence diagram for requesting delegated key

One possible access scenario is when a third party wants to have full access to a certain type of information from the car. For instance, the car owner wants to have live updates on the current location of the car or the battery status of the car on his phone. In such a case, the TSP is the car owner. The TSP will request a delegated key of a specific type, with which the TSP can decrypt every message of that specific type. In the following subsections we show the steps involved to request such access. The sequence diagram to request such a delegated key is shown in Figure 6.2. Please note that this is a conceptual information flow and not a technical message flow.

Step 1: RequestDelegatedKey

In the first step, the party who wants to have access (TSP) requests the car for a delegated key. We denote RequestDelegatedKey(TSP, type), as the functions that takes as parameters the ID of the requesting party and the type of the messages the TSP wants to have access to.

Step 2: CheckPermission

In the second step, the car checks according to its privacy settings whether the requesting TSP is allowed to access specific messages of the requested type. We denote CheckPermission(TSP, type) as the function that takes as parameters the ID of the requesting party and the type of the messages (type).

Step 3: TransmitKey

In the third step, if access has granted by the car, the car (cary) transmits a key with which the TSP can decrypt the individual messages. We denote TransmitKey(k, type) as the function that takes as parameters the key k and the type of the messages (type) to which the key k allows access.

Step 4: Notify

In the last step, the car (cary) has to notify the sTTP about the delegated access to the specific type of information. We denote Notify(sTTP, TSP, cary, type) as the function that takes the IDs of the involved parties and the type of message the access has been granted for. The sTTP can now, without requesting permission from the car, give the ciphertexts of type type to the TSP when he requests it.

Controlling your Connected Car 41
CHAPTER 6. PROPOSED ARCHITECTURE

6.6 Sequence diagram for sTTP-side aggregation

Information about a car is requested by a third party. When this data request allows the result to be of an aggregated form, it can be an aggregation which covers multiple individual messages per car (e.g., message $m_i, m_{i+1}, m_{i+2}$ for car $x$) as well as multiple messages that cover multiple cars (e.g., aggregating data type $i$ over car $x, y$ and $z$). The goal is to perform privacy friendly aggregation (aggregation where the involved third party does not know the individual values) when aggregation is possible.

In this section, we show what steps are involved when the third party requests data that is being aggregated in the sTTP only. The aggregation type also applies to single messages, where no aggregation has to be performed. Please note that this is a conceptual information flow and not a technical message flow.

Step 1: RequestCarData

In the first step, a third party service provider (e.g., car manufacturer) will request information from the data center, which can either be of a general form or a specific form. An example of such a request could be the average fuel consumption of all the connected BMW X5 models for a specific timespan. It is also possible for the TSP to request information of specific form, such as the most recent location of a specific BMW model. We denote RequestCarData($car_0: n, type$) as the function that requests data of type $type$ from the sTTP.

Step 2: RequestPermission

In the second step, the sTTP determines to what ciphertexts the request applies and will forward a permission request for those ciphertexts to all the corresponding vehicles. We denote RequestPermission($TSP_x, ID_{C_0...j}$) as the function that requests permission for the aggregation of a specific set of ciphertexts. Additionally, the ID of the TSP will be provided, which ensures that the car can determine whether the requested data is allowed for the specific $TSP_x$.

Step 3: CheckPrivacy

In step 3, the $car_y$ checks, according to the privacy settings in the car, whether the $TSP_x$ is allowed to receive the aggregated data, which is $f(C_0,...,C_j)$. We denote CheckPrivacy($TSP_x, ID_{C_0...j}$) as the function that takes as input the ID of the TSP and the IDs of the messages that need to be aggregated. The car will check whether the aggregation of this information is allowed, by checking...
### Step 4: TransmitKey

In step 4, two things can happen, the car either grants the information access or it rejects information access. The car has checked in step 3 whether the information request is conform the privacy settings the user has given. When a privacy setting by the user allows the TSP to request information of general form (average fuel consumption), but the request only applies to one specific entry, there might be an issue. When a request gets rejected, this response will be communicated to the data center. When a request for information is granted, the car will transfer a value \( k \) which will allow the specific messages to be decrypted, that is, it will provide a value which will only be able to decrypt the aggregated ciphertext of a specific set of messages. We denote TransmitKey(\( car_y, f(m_0,\ldots,m_j) \)) as the function that provides the value to decrypt the aggregated ciphertext for car \( car_y \). \( f(m_0,\ldots,m_j) \) is a mathematical function over \( m_0 \) to \( m_j \), for instance addition, multiplication and others.

### Step 5: Decrypt

In the fifth step, mathematical operations will be performed on the ciphertexts that will enable the sTTP to decrypt the aggregated result \( g(C_0,\ldots,C_j) \) which results in the aggregated result \( f(m_0,\ldots,m_j) \). We denote \( g(x_1,\ldots,x_j) \) as a function that performs mathematical operations to the ciphertexts according to the cryptosystem specifications. Decrypt(\( k, g(C_0,\ldots,C_j) \)) is defined as the function that takes as an input the received value for \( k \) which the sTTP received from the car.
CHAPTER 6. PROPOSED ARCHITECTURE

Step 6: Aggregate

In step 6, if applicable, the data will be combined with aggregated data of the other cars. After this combination is done, the aggregated data will be transferred to the information requester who is now able view the aggregated result. We denote \( \text{Aggregate}(S_0, \ldots, S_n) \) as the function that takes as input the computed result of the selected cars and aggregate them. It returns the value \( S = h(S_0, \ldots, S_n) \), where \( h \) is a mathematical function such as addition.

Step 7: SendCarData

In the last step, the end result, which is computed in the 6th step, is transferred towards the TSP who requested the data. We denote \( \text{SendCarData}(S) \) as the function which takes the computed end result as an argument which will be transferred towards the TSP. It is possible that some cars did not allow their data to be decrypted by the TSP, which can result in an incomplete end-result. The TSP should decide whether this is acceptable or not. When the request has a general form, e.g., the average speed of all the BMW X5 models, this might not be an issue. When the request has a more specific form, such as the average speed of a specific set of cars and the majority is rejected, the data might not be useful for the TSP.

6.7 Sequence diagram for car-side aggregation

Another way of doing aggregation is doing aggregation on the car-side instead of on the sTTP-side. The sequence diagram is shown in Figure 6.4, which displays the information flow between the data provider (car), data requester (TSP) and a data center (sTTP). Please note that this is a conceptual information flow and not a technical message flow. We show step-by-step what actions need to be performed.

Step 1: RequestCarData

In the first step, a third party will request information from the data center. An example of such a request could be the average fuel consumption of all the connected BMW X5 models for a specific timespan. It is also possible for the TSP to request information of specific form, such as the most recent location of a specific BMW model. We denote \( \text{RequestCarData}(car_{0:n}, \text{type}) \) as the function that requests data of type \( \text{type} \) from the sTTP.

Step 2: RequestPermission

In the second step, in contrary to sTTP side aggregation, the sTTP does not determine to what ciphertexts the request applies to, but will forward the permission request to all the corresponding vehicles. This is because the cars will determine for themselves to what messages the request applies to. We denote \( \text{RequestPermission}(TSP_x, \text{type}) \) as the function that requests permission for the aggregation of a specific \( \text{type} \) of information. Additionally, the \( ID \) of the TSP will be provided, which ensures that the car can determine whether the requested data is allowed for the specific \( TSP_x \).

Step 3: CheckPrivacy

In step 3, the \( \text{car}_{ID} \) checks, according to the privacy settings in the car, whether the \( TSP_x \) is allowed to receive the aggregated data of the specified \( \text{type} \). We denote \( \text{CheckPrivacy}(TSP_x, \text{type}) \) as the function that takes as input the \( ID \) of the TSP and the \( \text{type} \) of messages that need to be aggregated. The car will check whether the aggregation of this information is allowed. The return value will be the \( IDs \) of the messages that are allowed to be aggregated, where \( m_{0:j} \) denotes messages 0 to \( j \).
**Step 4: Aggregate:**

In step 4, two things can happen, the car either grants the access or it rejects access. When a request for information is granted, in contrary to the sTTP aggregation, the car will aggregate the messages instead of transferring a value to decrypt the ciphertexts. We denote \( \text{Aggregate}(m_{0:j}) \) as the function that takes as an input the messages 0 to \( j \) which he has granted aggregation to in the previous step and the return value will be the aggregated value \( M = f(m_0, \ldots, m_j) \). The function \( f \) is a function that performs mathematical operations on the messages.

**Step 5: Encrypt**

In the fifth step, the aggregated messages \( (M) \) will be encrypted with a key \( (k) \) and sent to the sTTP, as denoted by the function \( \text{Encrypt}(k, M) \). Key \( k \) can for instance be the public key of \( TSP_x \). When the cryptosystem has homomorphic properties, the sTTP can aggregate the messages without the knowledge of \( k \). Key \( k \) can, depending on the used cryptosystem, be a symmetric key or an asymmetric key.

**Step 6: Aggregate**

In step 6, the data will be combined with the aggregated data of the other cars. After this combination is done, the aggregated data will be transferred to the information requester who is now able view the aggregated result. We denote \( \text{Aggregate}(S_0, \ldots, S_n) \) as the function that takes as input the encrypted result of the selected cars and aggregate them. The function returns the value \( S = g(S_0, \ldots, S_n) \), which is the aggregated and encrypted value for the message-type \( type \) over all the cars.
CHAPTER 6. PROPOSED ARCHITECTURE

Step 7: SendCarData

In the seventh step, the encrypted end result, which is computed in the 6th step, is transferred to the TSP who requested the data. We denote SendCarData(S) as the function which takes the computed encrypted end result as an argument which will be transferred to the TSP.

Step 8: Decrypt

In the last step, the encrypted result will be decrypted by the TSP, which allows the TSP to view the decrypted aggregated end result. We denote Decrypt(k,S) as the function that takes as an argument the key k and the encrypted data S and returns the plaintext of the ciphertext, denoted by value. Key k can, depending on the used cryptosystem, be a symmetric key or an asymmetric key.
Chapter 7

Proposed cryptographic systems

In the previous chapters, we have described how we can achieve aggregation without revealing the content of the data in the context of connected cars. We have described the trust architectures and we have motivated why a particular trust architecture should be chosen. Furthermore, we have shown how such an architecture can be used to perform the required operations on the data by providing information flow sequence diagrams for the relevant scenarios. In the current chapter, we describe the requirements the cryptographic system must satisfy and we describe two cryptosystems which have these capabilities. Furthermore, we show how these cryptosystems apply to the context of communication with connected cars.

7.1 Cryptographic requirements

Before we can select an appropriate cryptosystem, we need to know which specific cryptographic conditions it has to satisfy. We have three important properties that the cryptosystem must satisfy:

- The cryptosystem must have the ability to construct ciphertexts that can be revealed through one or more trapdoors.
- The cryptosystem must have the ability to aggregate the data by performing mathematical operations on the ciphertexts without the need to decrypt the ciphertext first. That is, it must have homomorphic properties, at least the additive homomorphic property.
- The cryptosystem must have a trapdoor to reveal the result of the additive operation on a specific set of messages. This trapdoor must not reveal the intermediate messages within the aggregated subset or any other messages.
- The decryption of individual messages in case of car theft or other functions that requires individual messages to be decrypted.

7.2 Preliminaries

In the description of the cryptosystems, we use several cryptographic concepts and terms. In this section, we give a short explanation of these concepts and terms, which we are going to use in the next sections.

7.2.1 Component-wise multiplication

In some of the proofs of correctness in Section 7.3 and Section 7.4 we will use component-wise multiplication. For any \( a, b, c, d \in \mathbb{N} \), component-wise multiplication is defined as:

\[
(a, b) \cdot (c, d) = (ac, bd)
\]  

(7.1)
Furthermore, we will use component-wise exponentiation throughout the proofs. For any \(a, b, c \in \mathbb{N}\), where \(c\) is the exponent, component-wise exponentiation is defined as:

\[
(a, b)^c = (a^c, b^c)
\]  

(7.2)

### 7.2.2 Carmichael’s theorem

In Bresson’s cryptosystem, the authors describe a trapdoor which is based on the factorization of \(n = pq\). To be able to understand why this trapdoor works and understand the proof we give in Section 7.3.3, we need to define a formula that uses the factorization of \(n\) to solve a generally hard problem. By Carmichael’s theorem \([54]\), for any \(\omega \in \mathbb{Z}_{n^2}^*\), where \(n = pq\) and \(\lambda(n) = \text{lcm}(p-1, q-1)\), where \(\text{lcm}\) stands for least common multiple, we have:

\[
\omega^{n\lambda(n)} = 1 \mod n^2
\]  

(7.3)

\[
\omega^{\lambda(n)} = 1 \mod n
\]  

(7.4)

### 7.2.3 Bilinear mapping

In Ateniese’s cryptosystem, the authors use a bilinear mapping to map an element in a group to an element in a second group. Let \(G\), \(\hat{G}\) and \(G_\tau\) be cyclic groups of prime order \(p\). A bilinear map from \(G \times \hat{G}\) to \(G_\tau\) is a function \(e : G \times \hat{G} \rightarrow G_\tau\) such that for all \(u \in G\), \(v \in \hat{G}\) and \(a, b \in \mathbb{Z}\):

\[
e(u^a, v^b) = e(u, v)^{ab}
\]  

(7.5)

### 7.3 Bresson et al.-based solution

The cryptosystem we propose to use is Bresson et al.’s cryptosystem \([55]\), which is based on Paillier’s cryptosystem \([56]\) and is a variant from Cramer and Shoup’s cryptosystem \([57]\). The Paillier cryptosystem is a public key cryptographic system and has the property of additive homomorphism, which means additive operations can be performed in the encrypted domain.

#### 7.3.1 Paillier’s cryptosystem

Since Bresson’s cryptosystem is based on the Paillier cryptosystem, we will first describe the general Paillier cryptosystem, to give a better understanding of Bresson’s cryptosystem. Paillier’s cryptosystem is defined as follows:

**Key generation:**

- Pick two safe random primes \(p = 2p' + 1\) and \(q = 2q' + 1\), where \(p'\) and \(q'\) are also prime \([58]\) and let \(n = pq\).
- Compute \(\lambda(n) = \text{lcm}(p-1, q-1)\), where \(\text{lcm}\) stands for least common multiple. The function \(\lambda(n)\) is called the Carmichael function.
- Select a random integer \(g \in \mathbb{Z}_{n^2}^*\) such that there exists the modular multiplicative inverse \(\mu = (L(g^{\lambda(n)} \mod n^2)^{-1} \mod n\) where \(L(u) = \frac{n-1}{u}, \text{ for all } u \in \{ u < n^2 \mid u = 1 \mod n \}\). 

Public parameters: \((n, g)\)

Private key: \(\lambda(n)\)

**Encryption:**

Let \(m\) denote a message where \(m \in \mathbb{Z}_n\).

- Select a random \(r \in \mathbb{Z}_n^*\).
Construct the ciphertext: \( C = g^m \cdot r^n \mod n^2 \).

**Decryption:**

Let \( C \) be the encrypted message, where \( C \in \mathbb{Z}_{n^2}^* \).

- Recover the message \( m = L(C^{\lambda(n)} \mod n^2) \cdot \mu \mod n \).

### 7.3.2 Bresson’s cryptosystem

Next, we describe the variant of Paillier which has been proposed by Bresson et al. [55]. We propose the use of this specific cryptographic system because of its special property of having two trapdoors, which is beneficial in the application we need and it has the property of being additively homomorphic. That is, it allows the decryption of individual messages without the need of the master private key. Moreover, this double trapdoor property can help in generating keys for an aggregated set of messages, without revealing the master private key.

The variant of Bresson et al. is defined as follows:

**Key generation:**

- Pick two safe random primes \( p = 2p' + 1 \) and \( q = 2q' + 1 \), where \( p' \) and \( q' \) are also prime [58] and let \( n = pq \).
- Compute \( \lambda(n) = \text{lcm}(p-1, q-1) \).
- Select a random integer \( g \in \mathbb{Z}_{n^2}^* \) of order \( \lambda(n) \) such that there exists the modular multiplicative inverse \( \mu = L(g^{\lambda(n)} \mod n^2)^{-1} \mod n \) where \( L(u) = \frac{u-1}{n} \), for all \( u \in \{ u < n^2 \mid u = 1 \mod n \} \). As remarked by Cramer and Shoup [57], such a \( g \) can be easily found by selecting a random \( a \in \mathbb{Z}_{n^2}^* \) and compute \( g = -a^{2n} \mod n^2 \).
- Select a random \( x \) in range \( \{ x \in \mathbb{N} \mid 1 \leq x \leq n^2/2 \} \).

Public parameters: \( (n, g) \)

Public key: \( h = g^x \mod n^2 \)

Private key: \( \lambda(n) \) or \( x \)

**Encryption:**

Let \( m \) denote a message where \( m \in \mathbb{Z}_n \).

- Select a random \( r \) in range \( \{ r \in \mathbb{N} \mid 1 \leq r \leq n/4 \} \).
- Construct the ciphertext: \( (A, B) = (g^r, h^r \cdot [1 + m \cdot n]) \mod n^2 \).

**Decryption:**

Let \( (A, B) \) be the encrypted message.

- If \( x \) is known, message \( m \) can be recovered as \( m = L(B/A^x \mod n^2) \mod n \). Please note that this operation is similar to the ElGamal cryptosystem [50].
- If \( \lambda(n) \) is known, message \( m \) can also be recovered from \( B \) as \( m = L(B^{\lambda(n)} \mod n^2) \cdot \lambda(n)^{-1} \mod n \).

**Commitment scheme:**

Let \( m \) denote a message where \( m \in \mathbb{Z}_n \).

- Select a random \( r \) in range \( \{ r \in \mathbb{N} \mid 1 \leq r \leq n/4 \} \).
- Construct the commitment: \( C(m, r) = (h^r \cdot [1 + m \cdot n]) \mod n^2 \).

The commitment can be verified by revealing \( r \). The message \( m \) can be recovered by using \( r \) as \( m = L(C(m, r)/h^r \mod n^2) \mod n \).
7.3.3 Proof of correctness

To give an idea about the cryptographic system proposed by Bresson et al. and its capabilities, we will show a proof of correctness. We will show that the cryptographic system has two trapdoors and that the ciphertext will be able to be decrypted either by $x$ or $\lambda(n)$.

**Lemma 7.3.1 (From [55]).** Given the ciphertext tuple $(A, B)$, message $m$ can be reconstructed by computing $m = L(B/A^x \pmod{n^2}) \mod{n}$.

**Proof.** Given is the ciphertext $(A, B) = (g^r, h^r\cdot(1+m\cdot n)) \mod{n^2}$. We can decrypt the ciphertext $(A, B)$ by using $x$ as follows: $BA^{-x} \mod{n^2} = h^r(1+mn)g^{-rx} \mod{n^2} = 1+mn \mod{n^2}$. Applying $L$ yields $m$. □

The other method can decrypt a ciphertext $(A, B)$ by using a second trapdoor. This trapdoor uses $\lambda(n)$, which is based on the factorization of $n$. We show the correctness for the second trapdoor as well.

**Lemma 7.3.2 (From [55]).** Given the ciphertext tuple $(A, B)$, message $m$ can be reconstructed by computing $m = L(B^{\lambda(n)} \mod{n^2}) \cdot \lambda(n)^{-1} \mod{n}$.

**Proof.** Given the constructed ciphertext $(A, B) = (g^r, h^r\cdot[1+m\cdot n]) \mod{n^2}$. We can decrypt the ciphertext $(A, B)$ by using $\lambda(n)$ as follows: $B^{\lambda(n)} \mod{n^2} = h^r\lambda(n)(1+mn)\lambda(n) \mod{n^2}$. Applying Equation 7.4 results in $(1+mn)\lambda(n)$. This can be reduced to $(1+mn\lambda(n))$ by applying binomial expansion, which is possible due to the order of $B$, which is $|\lambda(n)|$. Furthermore, applying $L$ yields $m\lambda(n) \mod{n}$, which reveals $m \mod{n}$ after the multiplication with $\lambda(n)^{-1}$. □

7.3.4 Homomorphic properties

An application the cryptosystem is used for uses the homomorphic property which has been mentioned by Bresson et al.. We will show the correctness of the additive homomorphic property.

**Lemma 7.3.3 (From [55]).** Given the ciphertext tuple $C(m, r) = (g^r, h^r[1+mn]) \mod{n^2}$, the additive homomorphic property $(m_i + m_j)$ is satisfied by $C(m_i, r_i) \cdot C(m_j, r_j)$.

**Proof.** Let $C(m_i, r_i) = (A_i, B_i)$ be the ciphertext of message $m_i \in \mathbb{Z}_n$. By applying component multiplication, according to Equation 7.1, we get $C(m_i, r_i) \cdot C(m_j, r_j) \equiv (A_i A_j, B_i B_j) \mod{n^2}$. $(A_i A_j, B_i B_j) \mod{n^2} = (g^{r_i+r_j}, h^{r_i+r_j}[1+mn+m_i n]) \mod{n^2}$, because we are computing modulo $n^2$. The resulting formula can be rewritten to $(g^{r_i+r_j}, h^{r_i+r_j}[1+(m_i + m_j)n]) \mod{n^2}$, which can be seen as $C(m_i + m_j, r_i + r_j)$. □

7.3.5 Cryptosystem implementation

In this subsection, we describe how the cryptosystem should be used in the context of communication between cars and third parties. As we showed in Chapter 6, a party can only decrypt a ciphertext by using a trapdoor. In the system of Bresson et al., this is either possible by a delegated key ($x$), the masterkey $\lambda(n)$ or he has to receive a value from the encrypting party which enables him to reveal the message. We can distinguish between two types of messages to be decrypted:

- **Individual messages.** Individual messages are messages that are meant to stay in the same form throughout the whole communication process. All the messages are stored in this form on the sTTP, as shown in Figure 6.1. In Figure 6.2, we showed how a delegated key should be requested to the car owner. In the cryptosystem of Bresson et al., the key $k$ from Figure 6.2 should be the value $x$, which is solely used for one specific type of information. This key allows the receiver of that key $x$ to decrypt past and future messages used with that key. Another way of revealing individual messages to a party is by using the commitment scheme for this cryptosystem, which is defined by $C(m, r) = h^r \cdot (1+m\cdot n) \mod{n^2}$, which is
actually equal to the $B$ part of the previously described ciphertext tuple $(A,B)$. Whenever the party who encrypted the message $m$ with key $x$ and used randomness $r$ wants to reveal one specific value, he can send the specific value of $r$ instead of $x$. One might think why the car does not send the message $m$ itself, i.e., he will not always be able to do so because this value has been removed from the car. However, the pseudo-random function for computing $r$ is still available.

- **Aggregated messages.** Aggregated messages are messages on which certain operations have been performed. We have described how we can achieve aggregated messages without revealing the content of those messages, i.e., by using the homomorphic properties of the cryptosystem. This means that messages can be added while only using the ciphertexts of those messages. As described in Section 7.3.4, Bresson et al.’s cryptosystem benefits from the additive homomorphic property.

  - Given is a set of ciphertexts encrypted for party $\alpha$: $C_i = (g^{x_i} \cdot h^{r_i} \cdot [1 + m_i n]) \mod n^2$, where $i \in I$ and $I$ is the set of identifiers. Please note that $g$, $h$ and $n$ are unique per car but publicly available.
  
  - Party $\alpha$ can release $R = \sum_{i \in I} r_i$.
  
  - The sTTP calculates values $A = \prod_{i \in I} A_i = g^R$ and $B = \prod_{i \in I} B_i = h^R \cdot [1 + (\sum_{i \in I} m_i n)] \mod n^2$.

As shown above, the ciphertext will have the form of a commitment ciphertext. Releasing $R$ will reveal the $\sum_{i \in I} m_i$. Any party who only wants to reveal a subset of the messages can add the specific values for $r_i$ and reveal that value to the party who wants to receive the aggregated (added) message.

### 7.3.6 Security of Bresson et al.’s cryptosystem

Bresson et al.’s cryptosystem is a variant of the Paillier cryptosystem, but uses the properties described in [57]. The Bresson et al. cryptosystem has only been proved semantically secure in the standard model, based on the Decisional Diffie-Hellman assumption modulo a square composite number. That is, they proved that the scheme is semantically secure, if the Decisional Diffie-Hellman assumption holds. The simplified scheme that we described has only been proven to be semantically secure and we refer to the original work [57] for the full proof against adaptive chosen-ciphertext attacks in the standard model.

Also, they prove the one-wayness of the scheme provided that the Lift Diffie-Hellman problem is hard. The Lift Diffie-Hellman assumption states that it is hard to compute $Z^t = g^{x_y} \mod n^2$ from $X,Y$ and $Z$, where $X = g^x \mod n^2$, $Y = g^y \mod n^2$ and $Z = g^{x_y} \mod n$. They also prove that the Lift Diffie-Hellman problem is hard if the Partial Discrete Logarithm is hard. The Partial Discrete Logarithm assumption states that it is hard to compute $a$ (without the factorization of $n$) from $g,h$ where $g$ is of maximal order in $G$, $h = g^a \mod n^2$, where $a$ in range $\{a \in \mathbb{N} \mid 1 \leq a \leq \text{ord}(G)\}$ and $n$ is the product of two safe primes. Given these conditions, the partial discrete logarithm problem is equivalent to the composite residuosity class problem [56].

Furthermore, an insider attacker (TSP) could identify intermediate values by receiving the aggregated $m_i$ and $m_i + m_j$ and $i,j$ are identifiers. This would enable the TSP to discover the value $m_j$ by computing $(m_i + m_j) - m_i$, while this might be disallowed. The car should keep track of what messages are being requested by whom, to overcome this problem.
7.4 Improvement: Ateniese et al.-based solution

The cryptosystem from Bresson et al. has a few disadvantages compared to the cryptosystem of Ateniese et al.. In the Bresson et al.-based solution, one can only aggregate over multiple cars in the ciphertext domain when the car aggregates its data in the car. In the Ateniese et al.-based solution, the data can be aggregated over multiple cars in the ciphertext domain without the requirement to aggregate the data in the car first. Another disadvantage of the Bresson et al.-based solution is the size of the ciphertexts. The Ateniese et al.’s cryptosystem [60] is based on BBS [61] and ElGamal public key cryptosystem [59]. We will first describe BBS, then the proxy re-encryption scheme proposed in [60] and after that we will show how we can use this to generate a variant of this to satisfy the properties we want.

7.4.1 Blaze, Bleumer and Strauss (BBS)

BBS [61] proposed by Blaze et al. is very similar to the public key cryptosystem proposed by ElGamal [59]. The proposed cryptosystem operates over a group \( \mathbb{Z}_p \) as we show in the description of the cryptosystem, which works as follows:

**Key generation:**

- Pick a prime \( p \) of form \( p = 2q + 1 \) where \( q \) is also a prime [58].
- Pick a generator \( g \in \mathbb{Z}_p^* \).
- Pick a private key \( x \in \mathbb{Z}_2^* \) (i.e., relatively prime to \( p \)).

**Public parameters:** \((p, g)\)

**Public key:** \( h = g^x \)

**Private key:** \( x \)

**Encryption:**

Let \( m \) denote a message where \( m \in QR_p \). \( QR_p \) is defined as \( \{ y \in \mathbb{Z}_p^* : \exists x \in \mathbb{Z}_p \ y = x^2 \mod p \} \).

- Select a random \( r \) where \( r \in \mathbb{Z}_{2q}^* \).

Construct the ciphertext: \((A, B) = (h^r, m \cdot g^r) \mod p\).

**Decryption:**

Let \((A, B)\) be the encrypted message.

- Message \( m \) can be recovered as \( m = B/A^{\frac{1}{x}} \mod p \).

This scheme is based on ElGamal and uses a tuple as ciphertext to represent a message \( m \). Although it has beneficial properties, it does not yet allow a ciphertext to be transformed into another ciphertext without decrypting it first and it only satisfies multiplicative homomorphism.

7.4.2 Ateniese et al.

To satisfy the property of re-encrypting ciphertexts, we propose to use the cryptosystem which has been proposed by Ateniese et al. [60]. It is a proxy re-encryption scheme which is based on BBS and ElGamal. It allows for the use of a proxy or a third party to re-encrypt data which was originally intended for party \( \alpha \) to re-encrypt it into a ciphertext for \( \beta \) without the revelation of the message.

The scheme is operating over two groups \( G, \hat{G} \) of prime order \( p \) with a bilinear map \( e : G \times \hat{G} \to G_{\tau} \). The system parameters are random generator \( g \in G \) and \( Z = e(g, g) \in G_{\tau} \). The system is defined as follows:
CHAPTER 7. PROPOSED CRYPTOGRAPHIC SYSTEMS

Key generation:
- Pick a prime $p$ of form $p = 2q + 1$ where $q$ is also a prime.
- Pick a generator $g \in G$.
- Pick a private key $x \in \mathbb{Z}_p$.

Public parameters: $(p, g)$
Public key: $h = g^x$
Private key: $x$

Re-Encryption key generation:
Let $k_{\alpha \rightarrow \beta} = (g^\beta)^\frac{1}{x} \in G$ be the re-encryption key to transform a second-level ciphertext into a first-level ciphertext. Let $\alpha, \beta$ be the private keys of the involved parties.

Encryption:
Let $m$ denote a message where $m \in \hat{G}$.
- Select a random $r$ where $r \in \mathbb{Z}_p$.
- First level encryption: Construct the ciphertext $(A_1, B_1) = (e(g, h)^r, m \cdot Z^r)$ mod $p$.
- Second level encryption: Construct the ciphertext $(A_2, B_2) = (h^r, m \cdot Z^r)$ mod $p$.

Re-Encryption:
Anyone with possession of the re-encryption key between two parties $k_{\alpha \rightarrow \beta} = h^\frac{1}{x}$ with $h = g^\beta$ can re-encrypt a second-level ciphertext for $\alpha$ using $k_{\alpha \rightarrow \beta}$ into a first-level ciphertext for $\beta$ by applying $A_1^\prime = e(A_2, k)$.
This way the ciphertext will transform from $(g^{\alpha r}, m \cdot Z^r)$ mod $p$ to $(Z^{\beta r}, m \cdot Z^r)$ mod $p$.

Decryption:
Let $(A_i, B_i)$ be the encrypted message with the public key $h$, where $i$ denotes the level of the ciphertext.

- First level decryption: Message $m$ can be recovered as $m = B_1 / A_1^\frac{1}{x}$ (mod $p$).
- Second level decryption: Message $m$ can be recovered as $m = B_2 / e(A_2, g)^\frac{1}{x}$ (mod $p$).

Given this cryptosystem, we satisfy the property of re-encrypting the ciphertext from $\alpha$ to $\beta$ without revealing the content to the proxy and without leaking $\alpha$ and $\beta$. Although not mentioned in their paper, their cryptographic system is homomorphic multiplicative capable, which means that the system can multiply two ciphertexts which results in the multiplication of the plaintext messages as well. Although this is a nice feature, in our application we rather need addition instead of multiplication in the encrypted domain.

Similarly to what is shown in [62], we can use $Z^m$ instead of $m$ for a small message space, to satisfy the additive homomorphic property. Although this gives a nice feature, decryption involves the computation of a discrete logarithm, which is generally hard. When this feature is used on a fairly small space it can be done efficiently. When we take the “Pollard $\lambda$-algorithm” to compute the discrete logarithm, the running time complexity to compute $Z^m$ of this algorithm is $O(\sqrt{b-a})$, where $a < x < b$. We can also build a hash table where a lookup has $O(1)$ running time complexity, but $O(n)$ storage.
CHAPTER 7. PROPOSED CRYPTOGRAPHIC SYSTEMS

7.4.3 Proof of correctness

To give an idea about the cryptographic system proposed by Ateniese et al. and its capabilities, we will show a proof of correctness. We will show that the holder of the secret key in the cryptographic system can decrypt both levels of ciphertext.

Lemma 7.4.1 (Partially from [60]). Given the constructed first-level ciphertext tuple \((A_1, B_1)\), message \(m\) can be reconstructed by computing \(m = \log_Z \left( \frac{B_1}{A_1} \right) \mod p\).

Proof. Given the constructed first-level ciphertext \((A_1, B_1) = (e(g, h)^r, Z^m \cdot Z^r) \mod p\) where \(Z^m \in \mathbb{G}_r\). Message \(m\) can be recovered by: \(B_1A_1^{-1/2} \mod p = Z^mZ^r \cdot (e(g, h)^{-r/x}) \mod p = Z^mZ^r \cdot Z^{-xr/x} = Z^m \mod p\). Applying \(\log_Z(Z^m \mod p)\) yields \(m\), which can be feasible in a small message space or immediately with a lookup table.

Lemma 7.4.2 (Partially from [60]). Given the constructed second-level ciphertext tuple \((A_2, B_2)\), message \(m\) can be reconstructed by computing \(m = \log_Z \left( \frac{B_2}{e(A_2, g)^{r/v}} \right) \mod p\).

Proof. Given the constructed second-level ciphertext \((A_2, B_2) = (h^r, Z^m \cdot Z^r) \mod p\) where \(Z^m \in \mathbb{G}_r\). Message \(m\) can be recovered by: \(B_2A_2^{-1/2} \mod p = Z^mZ^r \cdot (e(h^r, g)^{-1/x}) \mod p = Z^mZ^r \cdot Z^{-xr/x} = Z^m \mod p\). Applying \(\log_Z(Z^m \mod p)\) yields \(m\), which can be feasible in a small message space or immediately with a lookup table.

7.4.4 Homomorphic properties

We have shown that the cryptographic system is correct and the decryption works for both the first-level and second-level ciphertexts. Next, we show the homomorphic capabilities of the system, which are similar to the homomorphic properties of exponential ElGamal. The usual homorphic tricks, addition and multiplication in the plaintext domain by performing operations on the ciphertexts, work exactly as expected. We first show that when a ciphertext is raised to the power of \(v\), where \(v\) is a numeric value, the messages is multiplied by \(v\).

Lemma 7.4.3. Given the first-level ciphertext tuple \(C_1(m, r) = (e(g, h)^r, Z^m \cdot Z^r) \mod p\), the scalar homomorphic property \((m \cdot v)\) is satisfied by \(C_1(m, r)^v\).

Proof. Let \(C_1(m, r) = (e(g, h)^r, Z^m \cdot Z^r) \mod p\) be the constructed first-level ciphertext. Relatively easy can be shown, by applying component exponentiation from Equation 7.2, that the scalar homomorphic property is satisfied as \(C_1(m, r)^v \mod p = (e(g, h)^{rv}, Z^mvZ^rv) \mod p = C_1(mv, rv) \mod p\).

Lemma 7.4.4. Given the second-level ciphertext tuple \(C_2(m, r) = (h^r, Z^m \cdot Z^r) \mod p\), the multiplicative homomorphic property \((m \cdot v)\) is satisfied by \(C_2(m, r)^v\).

Proof. Let \(C_2(m, r) = (h^r, Z^m \cdot Z^r) \mod p\) be the constructed second-level ciphertext. Relatively easy can be shown, by applying component exponentiation from Equation 7.2, that the scalar homomorphic property is satisfied as \(C_2(m, r)^v \mod p = (h^{rv}, Z^mvZ^rv) \mod p = C_2(mv, rv) \mod p\).

Furthermore, we can show that two ciphertexts can be multiplied which results in an addition in the plaintext domain. We show this for first-level encryption as well as for second-level encryption.

Lemma 7.4.5. Given the first-level ciphertext tuple \(C_1(m, r) = (e(g, h)^r, Z^mZ^r) \mod p\), the additive homomorphic property \((m_1 + m_2)\) is satisfied by \(C(m_1, r_1) \cdot C(m_2, r_2) \mod p\).

Proof. Let \(C_1(m, r) = (e(g, h)^r, Z^mZ^r) \mod p\) be the constructed first-level ciphertext. Relatively easy can be shown that, by applying component exponentiation from Equation 7.1, the additive homomorphic property is satisfied for the first-level ciphertext as \(C_1(m_1, r_1) \cdot C_1(m_2, r_2) \mod p = C_1(e(g, h)^{r_1+r_2}, Z^{m_1+m_2}Z^{r_1+r_2}) \mod p = C_1(m_1 + m_2, r_1 + r_2) \mod p\).
Lemma 7.4.6. Given the second-level ciphertext tuple \( C_2(m, r) = (h^r, Z^m Z^r) \mod p \), the additive homomorphic property \((m_1 + m_2)\) is satisfied by \( C(m_1) \cdot C(m_2) \mod p \).

Proof. Let \( C_2(m, r) = (h^r, Z^m Z^r) \mod p \) be the constructed second-level ciphertext. Relatively easy can be shown, by applying component exponentiation from Equation 7.1, that the additive homomorphic property is satisfied for the second-level ciphertext as \( C_2(m_1, r_1) \cdot C_2(m_2, r_2) \mod p = C_2(h^{r_1 + r_2}, Z^{m_1 + m_2} Z^{r_1 + r_2}) \mod p = C_2(m_1 + m_2, r_1 + r_2) \mod p \).

7.4.5 Cryptosystem implementation

In this subsection, we describe how the cryptosystem should be used in the context of communication between cars and third parties. As we showed in Chapter 6, a party can only decrypt a ciphertext by using a trapdoor. In Ateniese et al.’s cryptosystem, a re-encryption key will be provided by \( A \) to the sTTP when re-encryption is allowed for party \( B \). We described in Chapter 6 how the parties should communicate to be able to aggregate data in a privacy-friendly manner. When a TSP wants information from a single car or multiple cars, the TSP should contact the corresponding sTTP to ask for the information. The sTTP, in turn, contacts the cars and asks them to provide a re-encryption key for the requested information. Details about such a process has been described in Chapter 6.

The steps required to acquire the correct information also applies to the Ateniese et al.-based cryptosystem. The benefit of this cryptosystem is that there is no need to use car-side aggregation to be able to aggregate over multiple cars. This is only possible when the parameters of the cryptosystem are globally agreed. The public parameters generator \( g \) and prime \( p \) must be used by all the cars to be able to aggregate over multiple cars.

We can distinguish between two types of messages to be decrypted:

- **Individual messages.** Individual messages are messages that are meant to stay in the same form throughout the whole communication process. All the messages are stored in this form on the sTTP, as shown in Figure 6.1. In Figure 6.2, we showed how a delegated key should be requested to the car owner. In the cryptosystem of Ateniese et al., the key \( k \) from Figure 6.2 should be the re-encryption key \( k_{\alpha \to \beta} \). This key allows the receiver of that key \( k_{\alpha \to \beta} \) to decrypt past and future messages which have been encrypted with the public key of \( \alpha \) by using \( k_{\alpha \to \beta} \).

- **Aggregated messages.** Aggregated messages are messages on which certain operations have been performed. We have described how we can achieve aggregated messages without revealing the content of those messages, namely by using homomorphic properties of the cryptosystem. This means that messages can be added while only using the ciphertexts of those messages. As described in Section 7.4.4, our modified version of the Ateniese et al.’s cryptosystem benefits from the additive and multiplicative homomorphic property. Using this property, party \( \alpha \) can release a re-encryption key which only applies to a subset of messages. This can be done by performing the following steps:
  - Given is a set of ciphertexts encrypted for party \( \alpha \): \( C_i^{(\alpha)} = (h_i^{r_i}, Z^{m_i} Z^{r_i}) \mod p \), where \( i \in I \) and \( I \) is the set of identifiers.
  - Party \( \alpha \) can release \( \rho_{\alpha \to \beta} = h_{\beta}^{R} \) where \( R = \sum_{i \in I} r_i \) and \( h_{\beta} = g^\beta \).
  - The sTTP calculates value \( B = \prod_{i \in I} B_i = Z^{R \sum_{i \in I} m_i} \mod p \).
  - The holder of \( \rho_{\alpha \to \beta} \) calculates value \( A = e(\rho_{\alpha \to \beta}, h_{\alpha}) \mod p = e(g^{R\beta/\alpha}, g^\alpha) \mod p = e(g, h_{\beta})^R \mod p \).
  - The ciphertext will have form: \( (e(g, h_{\beta})^R, Z^{R \sum_{i \in I} m_i}) = C^{(\beta)}(\sum_{i \in I} m_i, R) \mod p \).

As shown above, the ciphertext will have the form of a first-level ciphertext for party \( \beta \).
CHAPTER 7. PROPOSED CRYPTOGRAPHIC SYSTEMS

7.4.6 Security of Ateniese et al.’s cryptosystem

For simplicity reasons, we have described the simplest variant of Ateniese et al.’s cryptosystem, which the authors refer to as the “second attempt”. One might think that if the car (A) releases a re-encryption key for a TSP (B), if the sTTP (proxy) and TSP collude, they can also transform a first-level encryption to A into a first-level encryption of B. However, collaborating sTTP and TSP can extract \( g^a \) by \( (g^b)^a = g^{ab} \), which allows them to decrypt second-level encrypted messages, which were actually supposed to be decrypted by B using the re-encryption key \( k_{A\rightarrow B} = g^{\frac{a}{b}} \).

Using \( g^{\frac{a}{b}} \), the best the collaborating sTTP and TSP can do is \( e(g, g^a) = e(g, g)^{\frac{a}{b}} = Z^{\frac{a}{b}} \), however they cannot retrieve \( Z^{ar} \) as \( Z^{ar} \cdot Z^{\frac{a}{b}} = Z^{ar + \frac{a}{b}} \).

Exactly due to this property, the security of the cryptosystem is determined by how hard it is to find \( g^a \) given \( g, g^a \in G \). This is related to the \( q \)-Decisional Bilinear Diffie-Hellman Inversion (\( q \)-DBDHI) assumption [63], which states that it is hard to decide if \( Q = e(g, g)^{\frac{a}{b}} \) or not, where a random \( g \in G, a \in Z_p, Q \in \hat{G}_{\tau} \) and given \( (g, g^a, g^{a^2}, \ldots, g^{a^q}, Q) \). Furthermore, the security of the scheme is also determined by the assumption that given \( (g, g^a) \), the value \( a \) cannot be derived from seeing the \( a \)-th root of a polynomial set of random values. Although this seems plausible in a prime order group, they have adjusted their “second attempt” into a “third attempt” [60], which uses two secret values instead of one and is based on fewer and more standard assumptions, such as the extended Decisional Bilinear Diffie Hellman (eDBDH) assumption. As stated before, for simplicity, we have described the “second attempt” variant, but please note that the “second attempt” can be transformed into the “third attempt” without losing any functionality. In the “third attempt”, they use \( (g^r, mZ^{ar}) \) instead of \( (g^{ar}, mZ^r) \) for a second-level encryption and use a re-encryption key \( k_{A\rightarrow B} = g^{ab} \) instead of \( k_{A\rightarrow B} = g^{\frac{a}{b}} \). After re-encryption, this results in \( (Z^{abr}, mZ^{ar}) \) instead of \( (Z^r, mZ^r) \), which has the same properties as the “second attempt”, but has an extra \( a \) in both parts of the ciphertext. They prove in the “third attempt” that the security of first-level ciphertexts only depends on Decisional Diffie Hellman in \( \hat{G} \). Moreover, they prove that the scheme is correct and secure assuming the extended Decisional Bilinear Diffie Hellman (standard security) and discrete logarithm (master secret security).

Furthermore, our modification of the scheme leads to an attack that also applies to the previous cryptosystem. That is, an inside attacker (TSP) could identify intermediate values by receiving the aggregated \( m_i \) and \( m_i + m_j \) where \( i, j \) are identifiers. This would enable the TSP to discover the value \( m_j \) by computing \( (m_i + m_j) - m_i \), while this might be disallowed. The car should log the requests that are being made to overcome this problem.
Chapter 8

Analysis of the proposed cryptosystems

In this section, we will determine the complexity of the proposed cryptosystems, we will measure the running time complexity by providing a proof-of-concept implementation and we will analyze the current cryptosystems with respect to the security and privacy requirements defined in Chapter 3 and Chapter 4.

However, what should be considered before applying any analysis is that we did not provide a technical message scheme, but simply provided an information flow diagram. That means, in order to be secure against some attacks, we need to have a secure channel between the sTTPs and the cars and between the TSPs and the sTTPs. The ciphertext is only meant to hide information from the sTTP and the TSP as long as necessary, to avoid an excessive amount of information disclosure.

8.1 Key size

In order to determine an acceptable keysize for the cryptosystems, we need to identify which underlying problem defines the security for that particular cryptosystem. We have described two cryptosystems, i.e., Bresson et al.-based solution and Ateniese et al.-based solution.

8.1.1 Bresson et al.-based solution

One of the trapdoors in the Bresson et al.-based cryptographic system is based on factoring the modulus. The factorization of \( n \) enables anyone with the possession of these factors to decrypt any message that is encrypted using those factors. According to Lenstra [64], cryptographic systems which are based on factoring the modulus provide security which corresponds to the keylength of \( n \). In the context of connected cars, if we assume that a car will switch to new owners within 5 years we would need a keylength that provides security until 2020. The keys in the car will switch to a new key, because keeping the key will allow the new owner to view the data of the previous owner. Moreover, the previous owner will be able to track his car if he keeps the master key. The calculation in [64] on page 24 in Table 3 shows that with optimistic estimations we need a keylength of 1387 bits to reach the required security level. Conservative calculations show that \( n \) should be at least 1569 bits to achieve the security goal of 5 years. However, note that these estimations are based on what should be theoretically possible by researchers with a large budget.

The second trapdoor is based on the discrete logarithm problem. The groupsize of \( g \) is \( n^2 \), which has a keylength between \( n \) and \( 2n \). According to Lenstra [64], the requirements for the groupsize of a discrete logarithm cryptosystem are equal to the requirements of a cryptosystem that is based on factoring the modulus. In this case, that is between 1387 and 1569 bits. Given the private key \{ \( x \in \mathbb{N} \mid 1 \leq x \leq n^2/2 \} \), the keylength of \( x \) is approximately between \( n \) and \( 2n \).
CHAPTER 8. ANALYSIS OF THE PROPOSED CRYPTOSYSTEMS

According to Lenstra [64], the keysize of a private key in a discrete logarithm problem should be approximately 163 bits, which is much lower than the defined $n$ bits for $x$. So the second trapdoor is at least as hard as the first trapdoor.

8.1.2 Ateniese et al.-based solution

The trapdoors in the Ateniese et al.-based cryptographic system are based on the discrete logarithm problem. Similar to the Bresson et al.-based solution, the groupsize of $G$ and $G_t$ should be at least between 1387 and 1569 bits for security until 2020. That means that $p$ should be chosen according to those sizes. The secret key and the random value $r$ are chosen in the group $\mathbb{Z}_p$ where $p$ only needs to be 163 bits.

8.2 Analysis of security requirements

In Chapter 3, we have defined the general security requirements which apply to the context of connected cars. In this section, we will interpret these security requirements with respect to the proposed cryptographic systems, i.e., the Bresson et al.-based solution and the Ateniese et al.-based solution. We did not include the security requirements authentication, authorization and accounting, because they are not dependent on the cryptographic system, but require cryptographic protocols and mechanisms to ensure these conditions rather than security requirements from cryptographic systems.

8.2.1 Security requirements Bresson et al.-based solution

Confidentiality

This cryptosystem achieves the confidentiality requirement to a certain extent. That is, outsiders should not be able to gather knowledge about the sensitive information which is encrypted with this cryptosystem. However, the way the information is disclosed in this application of the cryptosystem might be too much. Information about the aggregated value per car is disclosed to the sTTP in the sTTP-side aggregation. In the car-side aggregation, this is not revealed. However, there is still an attack possible which could reveal individual messages. This has already been described in the security section of the cryptographic system.

Furthermore, when the private key $x$ is released to any party, it will enable that party to perform cryptographic operations on behalf of original key owner. We proposed to use a different key $x$ for every type of message. This way, the party with the possession of $x$ will be restricted to a specific type of messages. The requirement to reveal $x$ to a party to be able to give knowledge about messages for more than once without the need for re-communication with the car is not desirable for the confidentiality.

The third trapdoor, based on the knowledge of $r$, as described in Chapter 7, allows specifically the decryption of a single message or an aggregation of single messages. This preserves the confidentiality for the rest of the messages (the messages that cannot be decrypted through that trapdoor).

The trapdoor that uses $\lambda(n)$, which is based on the factorization of $n$, is only used as a master trapdoor which can decrypt any message encrypted with the corresponding $g$ and $n$. The value $\lambda(n)$ should not be revealed to anyone other than the car owner.

Integrity

This cryptosystem does not provide integrity by itself. The way we propose to use this cryptosystem, i.e., for homomorphic operations, does not allow a signature in the ciphertext, because the intermediate party (sTTP) will generally not be able to view the content of the ciphertext. In this case, the ciphertext can be treated as the message and the car can append a signature for that
ciphertext. However, the involved parties should use a secure channel to transfer that data and the sTTP can verify that the ciphertext was generated by the car.

Using sTTP-side aggregation, the sTTP will see the aggregated value per car. Hence, the car could also use a homomorphic signature scheme which enables the sTTP to verify the signature on the aggregated data. Furthermore, aggregated ciphertexts can be signed by the sTTP, which allows the TSP to verify the aggregated result generated by the sTTP.

**Availability**

The availability requirement is satisfied to a certain extent, it differs per situation. When the computation of a value for a trapdoor is needed, the availability of the data depends on the availability of the car. This is because the car needs to compute either the aggregated value or release a value to open the aggregated value.

When a trapdoor is used that does not require the communication with the car, because the value to use the trapdoor is already available, there is no need to contact the car. The availability of the data only depends on the sTTP which has the possession of the data.

**Non-repudiation**

Another security requirement which we defined is non-repudiation, in which a party cannot deny having taken a certain action. In the setting of connected cars where cars connect to TSPs in a privacy friendly manner, a TSP should not be able to deny having requested certain values from a car. The non-repudiation is closely related to the integrity requirement, because this problem can be resolved by the same measure, applying signatures. A way to do this is by signing the request with the private key. However, when the private key $x$ is shared with other parties for longer term purposes, one cannot distinguish between the original owner and the “delegated” party anymore.

A better solution would be when the data is being signed with the private key $\lambda(n)$ and verified with the corresponding public parameters. This way of signing is better than using $x$ because $\lambda(n)$ should never be shared with anyone but the car owner.

### 8.2.2 Security requirements Ateniese et al.-based solution

**Confidentiality**

This cryptosystem achieves the confidentiality requirement to a certain extent. Eavesdroppers should not be able to gather knowledge about the content of the ciphertext. For an outsider, this is not possible, but the system allows the transformation of a ciphertext for $A$ into a ciphertext for $B$ with re-encryption key $k_{A\rightarrow B}$. This allows party $B$ to transform any ciphertext from $A$ into a ciphertext destined to him, which might violate the confidentiality in some situations.

However, aggregation can be done in the sTTP per car as well as over multiple cars without revealing any plaintext content to the sTTP. In contrary to the cryptosystem of Bresson, the sTTP will not need to decrypt aggregated values to be able to aggregate over multiple cars.

Another trapdoor can be created by combining the re-encryption key with a set of known values for $r$. This will allow the owner of the data to specifically control for which information the confidentiality is preserved.

In this cryptosystem, subtracting aggregated messages can reveal individual messages. This is an attack on the confidentiality which is similar to the Bresson et al.-based cryptosystem and has also been described in the security section of the definition for the cryptosystem.

**Integrity**

This cryptosystem does not provide integrity by itself. Similar to the previous cryptosystem, the system generally does not allow a signature in the ciphertext, because the intermediate party (sTTP) will not be able to view the content of the ciphertext. In this case, the ciphertext can be treated as the message and the car can append a signature for that ciphertext. However, this
CHAPTER 8. ANALYSIS OF THE PROPOSED CRYPTOSYSTEMS

should be sent over a secure channel between parties. Another way to transform signatures is
to use proxy re-signatures, which allow the transformation of signatures from $A$ into signatures
from $B$ with the use of a key $k_{A\rightarrow B}$ is described by Ateniese and Hohenberger in [65].

Another possibility is the use of a homomorphic signature scheme which might enable the TSP
to verify the signature on the aggregated data for a single car. In the worst case, the sTTP can
still verify the origin of the ciphertext and can sign that he has performed aggregation on the
ciphertexts and send the result to the TSP through a secure channel.

**Availability**

The availability requirement is satisfied to a certain extent, it differs per situation. This is similar
to the previous cryptosystem. When the re-encryption key is released to the sTTP, there is no
dependency on the car anymore. This is different from the previous cryptosystem as releasing a
re-encryption key does not allow the sTTP to encrypt messages on the other party his behalf, but
he will only be able to transform the ciphertext into another ciphertext. However, when the car
owner only wants to release aggregated data, a special trapdoor that only reveals the aggregated
content has to be requested every time the aggregated data is needed. This obviously affects the
availability of the data, as it is dependent on the availability of the car.

**Non-repudiation**

The non-repudiation requirement is similar to the previous cryptosystem. A TSP should not be
able to deny having sent a certain information request, because that could lead to the attack where
individual messages are being revealed. When signatures are being used, a party cannot deny that
it had sent a request or message provided that they did not disclose their private keys. This can
be done by using a secure channel which provides non-repudiation protection or by applying the
 technique that is presented in [65].

8.3 Analysis of privacy requirements

In Chapter 4, we have defined the seven privacy requirements to determine to what extent a certain
application respects the privacy of the subject of that application. In this case we will interpret
the privacy requirements laid down in Chapter 4, from the point of view of the car owner/car
driver.

8.3.1 Privacy requirements Bresson et al.-based solution

**Anonymity and pseudonymity**

As described in Chapter 4, we have stated that achieving anonymity requires the subject (car
owner) to not be identifiable within a set of subjects. When we use the Bresson et al.-based
cryptosystem, we can obtain this requirement between the sTTP and the TSP. However, depending
on the way of aggregation, the sTTP might still know the intermediate values per car. The car
might need to reveal certain information to the sTTP which will be hidden within a set of cars.
In this case, the information has become known to the sTTP and the anonymity requirement is
not fully satisfied. Only when car-side aggregation is used, the anonymity requirement is fully
satisfied.

**Untraceability**

Untraceability has been defined as the requirement of the inability of determination that two
records relate to the same object. In this context, that means that it cannot be determined by an
attacker that two ciphertexts relate to the same car. As already mentioned in Chapter 4, this is
impossible, because the car is equipped with a SIM-card which is owned by the car manufacturer.
Hence, this party is always able to trace the car. Whenever the SIM-card is not owned by the
Unlinkability

With respect to unlinkability, it should be impossible to relate multiple records in the dataset to the same object. That is, it should not be possible for an attacker to observe whether two messages belong to the same car. Because the content is encrypted with a secure cryptosystem, this is in general not possible for the plaintext. However, it is possible for ciphertexts, because when ciphertexts are unlinkable to cars they cannot be used to aggregate for specific cars. For an sTTP it is possible to determine what message belongs to what car, however, in car-side aggregation this information is encrypted and in sTTP-side aggregation this information is only revealed after it has been aggregated.

Revocable privacy

Given the definition in Section 4, “revocable privacy is the guarantee of an architecture that personal data is revealed only if a predefined rule has been violated”, this requirement is met to a certain extent in this cryptosystem. This can been seen in the application where the car is being stolen and a trapdoor can be used by the car owner to revoke the privacy of the thief. In this case a predefined rule (stealing the car) has been violated.

In control

We defined the requirement of being in control as the ability to control what information is being shared to whom and in what form. In the application we defined here, we try to satisfy this requirement as much as possible. The car owner should, via his privacy preferences in the car, be in control of his data. The car will not send information generated within the car to any party that is not in accordance with his privacy preferences. With the master key ($\lambda(n)$), the car owner is in full control of his data.

Data minimization

Somewhat related to being in control, the data minimization requirement says that the information stored about a person should not be excessive. In our application, the user is in control and determines the level of information disclosure. The TSP has no or only limited information about the car owner, due to the aggregation with other cars. The cryptographic system only allows the additive homomorphic on the encrypted domain, which results in a rather minimum amount of disclosure of raw personal information. However, the system still allows the raw information to be disclosed through a trapdoor, which could result in an excessive amount of information disclosure. Moreover, the amount of information between the sTTP and the car can be excessive, as with sTTP side aggregation, the amount of information that is being shared with the sTTP is in some cases excessive. The sender has a choice in the amount of information that is being disclosed, so we can safely say that this requirement is satisfied.

Non-profiling

In Section 4, we have defined the requirement of non-profiling. Although one of the goals of data gathering is generating a profile of the performance of certain cars, it should be as hard as possible to generate a profile on the individual behavior. Profiling car drivers using aggregated data is much harder than profiling car drivers using raw messages, but profiles might still be present in the dataset. This leads to the conclusion that the non-profiling requirement is satisfied to a certain extent.
8.3.2 Privacy requirements Ateniese et al.-based solution

**Anonymity and pseudonymity**
In the newly proposed cryptosystem, the anonymity of the car owner has been improved as the intermediate values are not revealed when doing sTTP-side aggregation. The ciphertext can be transformed and aggregated by the sTTP without the knowledge of the plaintext. The data of the car owner is now hidden within \( n \) other car owners. However, the ability to decrypt individual ciphertexts would reveal the identity, but is not an issue since we introduced it as a necessary feature.

**Untraceability**
The untraceability requirement remains unchanged for this newly proposed cryptosystem. The car might still be geographically traceable by monitoring the information within the GSM packets.

**Unlinkability**
The unlinkability requirement is improved compared to the Bresson et al.-based solution, as it is not possible for the sTTP to link plaintexts to cars in the general case. However, it is possible for ciphertexts, because when ciphertexts are unlinkable to cars they cannot be used to aggregate for specific cars.

**Revocable privacy**
Although in the current cryptosystem, the availability of two trapdoors has been disappeared, a new feature has become available which is ability to transform a ciphertext without the knowledge of the encryption key. When a new re-encryption key is created the privacy of the current car driver can be revoked. With this key or a correct private key, the privacy of the car driver can be revoked. Say that the private key is loaded in the car, but also has a backup on a chip. This will allow the car owner to revoke the privacy of the car user by releasing a re-encryption key for a specific party.

**In control**
The ability to be in control has not changed in the current cryptosystem, as the car will still not send information that is not in accordance with his privacy preferences to any other party. This will still allow the car owner to be in control of his data. However, when a re-encryption key of the form \( k_{\alpha \rightarrow \beta} = g^{\beta} \) has been released, all second level ciphertexts can be re-encrypted for party \( \beta \), that means individual messages as well. When this is not desirable anymore, the car owner can create a new key. Moreover, as explained, a re-encryption key can also be created for a specific set of messages.

**Data minimization**
The data minimization requirement has also not changed for this new cryptosystem. The cryptographic system only allows a few operations in the encrypted domain, so the amount of disclosure of raw personal information is still rather limited. By sending individual messages, it is possible in cryptosystem though.

**Non-profiling**
We stated in the previous system that it should be as hard as possible for the TSP to generate a profile on the individual behavior. This requirement is still satisfied as the information that is disclosed to the TSP is already processed and the individual elements can be removed in the aggregation as much as possible.
CHAPTER 8. ANALYSIS OF THE PROPOSED CRYPTOSYSTEMS

<table>
<thead>
<tr>
<th>Bresson et al.-based solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>NVM</td>
</tr>
<tr>
<td>Key generation $\mathcal{O}(n^2)$</td>
</tr>
<tr>
<td>Encryption $\mathcal{O}(n^2)$ + 2 · PNG + ME + RNG</td>
</tr>
<tr>
<td>Decryption with $x$ $\mathcal{O}(n)$</td>
</tr>
<tr>
<td>Decryption with $\lambda(n)$ $\mathcal{O}(n)$</td>
</tr>
<tr>
<td>CPU</td>
</tr>
<tr>
<td>Communication $\mathcal{O}(n^2)$</td>
</tr>
<tr>
<td>$2 \cdot$ PNG + ME + RNG</td>
</tr>
<tr>
<td>$\mathcal{O}(n^2)$</td>
</tr>
<tr>
<td>$\mathcal{O}(n)$</td>
</tr>
<tr>
<td>$\mathcal{O}(n)$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ateniese et al.-based solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>NVM</td>
</tr>
<tr>
<td>Key generation $\mathcal{O}(p)$</td>
</tr>
<tr>
<td>1st level encryption $\mathcal{O}(p)$</td>
</tr>
<tr>
<td>2nd level encryption $\mathcal{O}(p)$</td>
</tr>
<tr>
<td>Re-encryption key gen. $\mathcal{O}(p)$</td>
</tr>
<tr>
<td>Re-encryption $\mathcal{O}(p)$</td>
</tr>
<tr>
<td>1st level decryption $\mathcal{O}(p)$</td>
</tr>
<tr>
<td>2nd level decryption $\mathcal{O}(p)$</td>
</tr>
<tr>
<td>CPU</td>
</tr>
<tr>
<td>PNG + 2 · RNG $\mathcal{O}(p)$</td>
</tr>
<tr>
<td>BM + 2 · ME + RNG $\mathcal{O}(p)$</td>
</tr>
<tr>
<td>2 · ME + RNG $\mathcal{O}(p)$</td>
</tr>
<tr>
<td>ME $\mathcal{O}(p)$</td>
</tr>
<tr>
<td>MM $\mathcal{O}(p)$</td>
</tr>
<tr>
<td>DLP + ME + MM $\mathcal{O}(p)$</td>
</tr>
<tr>
<td>$\mathcal{O}(p)$</td>
</tr>
<tr>
<td>$\mathcal{O}(p)$</td>
</tr>
<tr>
<td>$\mathcal{O}(p)$</td>
</tr>
<tr>
<td>$\mathcal{O}(p)$</td>
</tr>
<tr>
<td>$\mathcal{O}(p)$</td>
</tr>
<tr>
<td>$\mathcal{O}(p)$</td>
</tr>
<tr>
<td>$\mathcal{O}(p)$</td>
</tr>
<tr>
<td>Communication $\mathcal{O}(p)$</td>
</tr>
<tr>
<td>$\mathcal{O}(p)$</td>
</tr>
<tr>
<td>$\mathcal{O}(p)$</td>
</tr>
<tr>
<td>$\mathcal{O}(p)$</td>
</tr>
<tr>
<td>$\mathcal{O}(p)$</td>
</tr>
<tr>
<td>$\mathcal{O}(p)$</td>
</tr>
<tr>
<td>$\mathcal{O}(p)$</td>
</tr>
<tr>
<td>$\mathcal{O}(p)$</td>
</tr>
</tbody>
</table>

Table 8.1: Complexity comparison between Bresson et al.-based and Ateniese et al.-based solution

8.4 Comparison Bresson vs. Ateniese

The Ateniese et al.-based cryptosystem has a few advantages over the other cryptosystem, that is, messages can be re-encrypted without the need to reveal the content to the sTTP. This was not possible in the Bresson et al.-based system. We can use two benefits of the two types of aggregation in one system, which means that we can aggregate in the sTTP without revealing the intermediate result to the sTTP and we can combine the results of multiple cars together.

Moreover, using car-side aggregation in the Bresson et al.-based cryptosystem for two different TSPs who want the same aggregated value would require a re-aggregation in the car or at least re-encryption of the same value under a different public key by the car. In the Ateniese et al.-based cryptosystem, these values can be computed in the sTTP and the car only has to release a new re-encryption key.

The cryptosystem that is using the Ateniese et al.-based cryptosystem for proxy re-encryption can be used in the same manner as we described for the Bresson et al.-based cryptosystem. The only difference is that the Ateniese et al.-cryptosystem is more efficient and hiding more information from the sTTP. The sequence diagram is also similar to the one described before, with the only difference that the ciphertext will not be decrypted in the sTTP, but re-encrypted for the corresponding TSP.

Again, we can distinguish between messages which are meant as individual messages, in which we just can use $B_i = m \cdot Z^r$ instead of $B_i = Z^m \cdot Z^r$ as they do not need to satisfy homomorphic properties. As an example a car owner can provide a TSP with raw information in case this is needed for repair service. Another application is when the car owner wants the information of the car to be available on other devices, such as his cellphone.

To ensure that messages in the past are not possible to re-encrypt for the newly added messages, the car has to change the key at every newly added delegate. Hence, the re-encryption keys have to be updated for every delegate, which only involves the proxy/sTTP.
8.4.1 Complexity analysis

In the previous section, we have compared the two cryptographic systems in terms of usability and with respect to security and privacy requirements. In this section, we will compare the two cryptographic systems in terms of complexity.

In Appendix B, we have given the pseudo-code for all the algorithms which are stated in Table 8.1. In the columns we show the non-volatile memory space complexity (NVM), the running time complexity (CPU) and the communication space complexity (communication). We did not include the amount of random access memory, because it depends on the algorithms used to solve for instance the modular exponentiation. The column for CPU is filled with abbreviations for the underlying mathematical running time complexity of the problems. The algorithms used to solve these problems will determine the complexity of the algorithms. The amount of optimization of these algorithms determine the performance. We assume that simple operations, such as addition and taking the modulus is trivial.

Modular Exponentiation (ME)

More complex is the algorithm to perform a modular exponentiation, which is denoted by ME. The complexity of an efficient algorithm has been described in [66] and is called the “repeated square-and-multiply”, also known as “binary exponentiation”. This algorithm has the complexity $O(\log_2 x)$ multiplications, for the computation of $g^x \mod n$ for any $g, n \in \mathbb{N}$.

Discrete Logarithm Problem (DLP)

Another important computation to measure is the computation of the a discrete logarithm, denoted by DLP. An algorithm to solve this problem is called the “baby-step giant-step”-algorithm, which has complexity $O(\sqrt{n})$ for $h = g^x \mod p$, where $\{ x \in \mathbb{N} \mid 1 \leq x \leq n \}$. Please note, that if we would use a lookup table that $O(\sqrt{n})$ can be replaced by $O(1)$.

Bilinear Mapping (BM)

In our Ateniese et al.-based solution, the complexity of the bilinear mapping, which is denoted by BM, is of great importance to measure the overall complexity. The authors in [60], describe that they used fast Tate-pairings described in [67] to compute the bilinear mapping. In [67], the authors show an efficient way to compute Tate pairings.

Modular Multiplicative Inverse (MI)

The modular multiplicative inverse of a value $a$ modulo $n$ is an integer $b$ which satisfies the following property: $ab \equiv 1 \pmod{n}$. This is used by Bresson et al.’s cryptosystem to compute the plaintext message through trapdoor $\lambda(n)$. A way of finding the multiplicative modular inverse of a value is by applying the Extended Euclidean Algorithm and has $O(\log(n)^2)$ running time complexity.

Prime Number Generator (PNG)

Also important to measure is the algorithm to generate random prime numbers of a certain bitlength. There are multiple ways of generating primes, such as generating probable primes and generating provable primes. For the method of Bresson et al. the primes have to be generated and should be kept secret because the factorization of the product of these primes can reveal the messages. However, with Ateniese et al. the prime does not need to be secret and is shared as a public parameter. This means that this prime can be a known and verified safe prime number.
CHAPTER 8. ANALYSIS OF THE PROPOSED CRYPTOSYSTEMS

Modular Multiplication (MM)

Modular multiplication is also regularly used by both the cryptosystems. The function will calculate the result of $c = ab \mod n$ where $a, b$ are the natural numbers to be multiplied. Normally the long multiplication has a running time complexity of $O(\log^2 n)$. However, complexity improvements can be made by applying methods such as Karatsuba’s method and Fast Fourier Transformation (FFT).

Also important to mention is that we assume the usage of a pseudo-random generator for the random value $r$, which will result in no disk space usage for several algorithms. Furthermore, notice that we used $n$ in the Bresson et al.-based solution and $p$ in the Ateniese et al.-based solution. This is due to the way it has been defined in the algorithms, but $p$ and $n$ both denote the size of the ciphertext and are equal in number of bits.

Both the solutions showed in Table 8.1 are similar with respect to the running time complexity, because computing a bilinear mapping is done efficiently with fast Tate implementation [67]. Moreover, solving the Discrete Logarithm Problem can be avoided by computing a large lookup table, which improves the running time complexity from $O(\sqrt{p})$ to $O(1)$. Given these reasons, the running time complexity of both solutions are similar, while the complexity for NVM and communication is improved for the Ateniese et al.-based solution.

8.5 Benchmark Bresson

In the previous sections, we have given a comparison of the two cryptosystems with respect to security, privacy and complexity. In this section, we will provide an implementation of the cryptographic system of Bresson and we will use that implementation to measure the performance of Bresson’s cryptographic system.

8.5.1 Proof-of-concept

In Appendix A, we provide a proof-of-concept implementation of the Bresson’s cryptosystem [55]. Please note that this implementation is a proof of concept and the primes are not generated in a cryptographically safe way.

We chose Java as the programming language as it has a lot of mathematical libraries and it would be relatively easy to generate and run the proof-of-concept code on (for instance) an processor that is controlled by Android as Android is also written in Java.

In Appendix A, we have defined the class Bresson, with a constructor which performs the key generation and two methods, namely the encryption and the decryption method. In the constructor, the primes $p$ and $q$ are being generated, a generator $g$ is computed, secret value $x$ is randomly selected, and $h = g^x$ is computed. Please note that the value $x$ can be different for different type of messages. However, in this example we keep the $x$ constant.

We created two encryption methods and two decryption methods. The first encryption method only takes the message $m$ as an argument, it will select a random $r$ and will return the ciphertext tuple $(A, B)$. The variables $g, h$ and $x$ are declared in the same class. The other encryption method takes $m, g, h$ and $n$ as arguments and will encrypt the message $m$ with the variables given as arguments, which are the public parameters from any other party.

For the decryption method, we have two variants. One which uses both variables of the tuple $(A, B)$ and one which only uses $B$. The first one uses the class value $x$ to decrypt the ciphertext $B$, by computing $A^{-x}$ and then multiply it with $B$. The second method uses the secret value $\lambda(n)$, which enables the holder of the factorization (and thus $\lambda(n)$) to decrypt the ciphertext $B$.
In the main function, we instantiate the class and the two messages $m_1$ and $m_2$. Then we encrypt the messages by calling the encryption method in the object. Next, we show that both decryption methods work and result in the plaintext messages we defined before. After that, we show that the homomorphic properties hold, by computing the product of the $A$ from all the tuples and the product of the $B$ of all the tuples. Then, we decrypt these multiplication of ciphertexts and compare the result to the original sum of the plaintext messages.

### 8.5.2 Performance of Bresson’s key generation algorithm

In order to test the performance of the key generation, we set up a test environment in which we tested the performance of the key generation of Bresson for the following keylength sizes: 256 bit, 512 bit, 1024 bit, 2048 bit and 4096 bit. Furthermore, we did the key generation 100 times per keylength, which results in Figure 8.1. We selected two processors, the Intel Core i5-4300U and the Qualcomm Krait. These processors are chosen because the Intel Core i5-4300U is a relatively new processor and thus a good representation of the current state of the art processor. The Qualcomm Krait is chosen because that specific central processing unit is also used by the Qualcomm Snapdragon 602A in modern automotives [68]. Furthermore, the Figure shows the time consumption in ms on the $y$-axis and the keylength in bits on the $x$-axis. It also shows the standard deviation for the 100 iterations we did the simulation for. The generation of cryptographic keys range from a few milliseconds for small keys (256 bit) to about 20 seconds for larger keys (4096 bit). This range is perfectly acceptable as this key generation phase is only rarely used: in the setup phase, when the masterkey is compromised or when the car owner changes.
8.5.3 Performance of Bresson’s encryption algorithm

In the previous subsection we have shown that generating the public and private keys is efficient. We are interested in how well ciphertexts can be generated. We only use the Intel Core i5-4300U, because we have already shown in Section 8.6 and Figure 8.1 how the Qualcomm Krait CPU performs compared to the Intel Core i5-4300U. We have implemented the cryptosystem according to Algorithm 2 in Appendix B. The result is shown in Figure 8.2, which is average time it takes to construct a ciphertext tuple \((A, B)\) per keylength. This was done by generating 1000 ciphertexts by 1000 different keys. On average, the construction of a single ciphertext will take at most a few hundred milliseconds.

8.5.4 Performance of Bresson’s decryption algorithms

In the previous subsection we have tested the performance of the encryption algorithm. In this subsection we will test the performance of the decryption algorithm. We have taken the generated ciphertexts from the previous subsection and we have used these 1000 random ciphertexts to measure the time consumption of the decryption algorithms. The implementation is according to Algorithm 3 and Algorithm 4 in Appendix B. In Figure 8.3, the average amount of computation time is shown for the decryption of one ciphertext by using trapdoor \(x\). The result is quite similar to the average encryption time of one ciphertext. In Figure 8.4, the average amount of computation time is shown for the decryption of one ciphertext by using trapdoor \(\lambda(n)\). Algorithm 4 that uses \(\lambda(n)\) is slightly faster than the decryption algorithm that uses \(x\).

8.5.5 Performance of constructing aggregated ciphertexts

Although the decryption process is similar to the decryption process of Algorithm 3, constructing an aggregated ciphertext that can be decrypted has not yet been measured. Constructing such an aggregated ciphertext is done by multiplying the components of the ciphertext tuple \((A, B)\). We determined the slope of the time consumption as a function of the keylength by generating 100 ciphertexts per key pair for 100 unique key pairs. We determined the slope per keylength by measuring the time consumption of multiplying 100 ciphertexts for 100 key pairs and we have measured the average over these 100 values. In Figure 8.5, the average amount of time consumption in milliseconds per multiplication is displayed per keylength. Based on these values, we can define the slope for a formula that would predict the amount of time consumption for the multiplication of ciphertexts. The multiplication of, for instance, 1 million ciphertexts of 2048 bits would take about 90 seconds, based on a slope of 0.09 \(ms\) per multiplication.
CHAPTER 8. ANALYSIS OF THE PROPOSED CRYPTOSYSTEMS

Figure 8.3: Performance of the decryption algorithm using $x$

Figure 8.4: Performance of the decryption algorithm using $\lambda(n)$

Figure 8.5: Performance of the ciphertext multiplication
8.6 Processors

In previous sections, we have analyzed the performance for Bresson’s cryptosystem and we related that to Ateniese’s cryptosystem. This analyses showed that the running time complexity of both cryptosystems is similar, but that Ateniese should perform better in terms of NVM, RAM and communication. We tested a proof-of-concept implementation on an Intel Core i5-4300U and Qualcomm Krait. The selected processors, especially the Qualcomm Krait processor, give a good indication for the performance, but relating these processors to current automotive processors would be very useful to estimate the real performance in current cars.

We have selected a set of processors, according to the availability of DMIPS/MHz values, from a non-profit organization which provides the Industry-Standard Benchmarks for Embedded Systems [69], but we also selected a few other examples which are known in the industry to provide processors for embedded systems, such as [70] and [68].

In Table 8.2, we have given the specifications of the selected processors from [69, 70, 68]. The DMIPS/MHz column is the amount of Dhrystone million instructions per second per megahertz, which is well-known unit for benchmarking microprocessors and has been developed by Weicker [71]. The last column, “Factor”, is the DMIPS/MHz as a factor of the DMIPS/MHz of the Intel Core i5-4300U, which we used to measure the performance of the Bresson cryptosystem in Section 8.5.

<table>
<thead>
<tr>
<th>Processor type</th>
<th>Clock Speed (MHz)</th>
<th>DMIPS/MHz</th>
<th>Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARM Cortex-M3</td>
<td>180</td>
<td>1.25</td>
<td>0.20</td>
</tr>
<tr>
<td>AMCC 405GP</td>
<td>400</td>
<td>1.52</td>
<td>0.24</td>
</tr>
<tr>
<td>AMCC 440GP</td>
<td>500</td>
<td>2.00</td>
<td>0.31</td>
</tr>
<tr>
<td>AMD K6-2E+ 500/ACR</td>
<td>500</td>
<td>1.40</td>
<td>0.22</td>
</tr>
<tr>
<td>ARM Cortex-A15</td>
<td>1700</td>
<td>3.50</td>
<td>0.55</td>
</tr>
<tr>
<td>ARM Cortex-A9</td>
<td>1400</td>
<td>2.50</td>
<td>0.39</td>
</tr>
<tr>
<td>ARM Cortex-M4</td>
<td>150</td>
<td>1.27</td>
<td>0.20</td>
</tr>
<tr>
<td>ARM Cortex-M7</td>
<td>300</td>
<td>2.14</td>
<td>0.33</td>
</tr>
<tr>
<td>Freescale MC7448</td>
<td>1700</td>
<td>2.30</td>
<td>0.36</td>
</tr>
<tr>
<td>IBM 970FX</td>
<td>2000</td>
<td>2.20</td>
<td>0.34</td>
</tr>
<tr>
<td>Infineon TC1796</td>
<td>150</td>
<td>1.70</td>
<td>0.27</td>
</tr>
<tr>
<td>Intel Atom E3800</td>
<td>1900</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>Intel Atom N450</td>
<td>1660</td>
<td>2.50</td>
<td>0.39</td>
</tr>
<tr>
<td>Intel Atom Z5xx</td>
<td>1000</td>
<td>2.50</td>
<td>0.39</td>
</tr>
<tr>
<td>Intel Celeron E3300</td>
<td>2500</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>Intel Celeron N2800/N2900</td>
<td>2000</td>
<td>2.86</td>
<td>0.45</td>
</tr>
<tr>
<td>Intel Core i5 4300U</td>
<td>1900</td>
<td>6.40</td>
<td>1.00</td>
</tr>
<tr>
<td>Intel Quark</td>
<td>400</td>
<td>1.20</td>
<td>0.19</td>
</tr>
<tr>
<td>MIPS 20Kc</td>
<td>600</td>
<td>2.40</td>
<td>0.39</td>
</tr>
<tr>
<td>NEC VR5500</td>
<td>1000</td>
<td>2.00</td>
<td>0.31</td>
</tr>
<tr>
<td>NXP LPC3180/Phytec board</td>
<td>208</td>
<td>1.10</td>
<td>0.17</td>
</tr>
<tr>
<td>PMC Sierra RM7000C</td>
<td>625</td>
<td>2.30</td>
<td>0.36</td>
</tr>
<tr>
<td>Qualcomm Krait</td>
<td>1900</td>
<td>3.40</td>
<td>0.53</td>
</tr>
<tr>
<td>Qualcomm Scorpion</td>
<td>1000</td>
<td>2.10</td>
<td>0.33</td>
</tr>
<tr>
<td>Renessas RX610</td>
<td>100</td>
<td>1.65</td>
<td>0.26</td>
</tr>
<tr>
<td>ST231</td>
<td>500</td>
<td>1.80</td>
<td>0.28</td>
</tr>
<tr>
<td>Transmeta Crusoe</td>
<td>1000</td>
<td>2.30</td>
<td>0.36</td>
</tr>
</tbody>
</table>

Table 8.2: Performance comparison between automotive microprocessors
8.6.1 Comparison on the performance of Bresson et al’s and Ateniese et al’s cryptosystem

Table 8.2 gives an indication on how the current microprocessors perform compared to the used Intel Core i5-4300U. Looking at the last column, the “Factor” column, one can see that the current automotive microprocessors have a performance capacity of between 20% and 50% per MHz compared to the used Intel Core i5. Some of the microprocessors have more or less the same amount of clock speed as the processor we used in the benchmark. For instance, the ARM Cortex-A15, Intel Celeron N2800/N2900 and Qualcomm Krait have around the same amount of megahertz and have more or less 50% of the capacity of the Intel Core i5. This shows that the microprocessor (Qualcomm Krait) we used was a representative choice to measure the performance on.

In Section 8.4.1, we have shown the complexity of the proposed cryptosystems. We can assume that $n$ and $p$ are of equal bitsize, because the group size of a factoring based problem should be more or less equal to the group size of a discrete logarithm problem, according to [64]. As we showed in Section 8.4.1, the algorithms are comparable in terms of running time complexity (CPU). This means that a benchmark for the Ateniese et al.-based solution will be similar to the performed benchmark of the Bresson et al.-based solution.
Chapter 9

Conclusions

In this thesis, we have researched how privacy can be enforced in the transfer of data towards car manufacturers or other third parties without restricting the functionalities. Reports show that many car drivers have a problem with sharing information with the car manufacturers.

In our research we have identified the security and privacy threats and the legal requirements from the Data Protection Directive in the context of Connected Cars. The security threats include attacks on the internal network of the car, which might enable the attacker to control certain functionalities of the car.

Privacy requirements such as anonymity/pseudonymity, untraceability, unlinkability, revocable privacy, in control, data minimization and non-profiling can be achieved, for instance, by aggregating messages that do not necessarily need to be revealed individually to the car manufacturer. The best way to do this is by using a semi-Trusted Third Party (sTTP) in combination with homomorphic cryptography. Achieving this requires multiple steps, such as storing the data in an encrypted form on the sTTP and asking the car for access, which will check that the information request is in accordance with the privacy policy of the car user. When the information request is approved, the sTTP should be provided with some kind of key to aggregate the data and send the result to the TSP.

In our thesis, we investigated two cryptographic systems, the Bresson et al.-cryptosystem and the Ateniese et al.-cryptosystem. Both cryptosystems are capable of performing additive homomorphic aggregation, which allows the addition of individual messages by performing an operation in the encrypted domain. We have shown how both cryptographic systems can be used in constructing aggregated messages. We had to adjust the Ateniese et al.-cryptosystem to make it additively homomorphic. It is based on bilinear pairings and has a benefit over the Bresson et al.-cryptosystem by being able to aggregate over multiple cars in the TSP without revealing information about the messages to the TSP. The ciphertext for a specific party can be transformed, by using a special key, into another ciphertext for a different party, without leaking individual private keys.

Additionally, we show in this thesis how the Bresson et al.-cryptosystem performs on a modern computer and a known modern Android phone. We have tested the performance of the key generation algorithm, the encryption algorithm and the two decryption algorithms. The result shows that the performance of the algorithm in perfectly acceptable, with an execution time of a few hundred milliseconds for a keylength of 2048 bits.

9.1 Future work

Our work proposes the use of a specific modified version of a cryptographic solution that allows car owners to be in control of their personal data. The work we did can be extended by investigating different cryptographic systems that satisfy the security and privacy requirements to a greater extent. Since we focused on the cryptographic system and cryptographic properties, an interesting
additional research topic is how a secure protocol should be constructed that is compatible with the properties stated in this thesis, but does not need secure channels to transfer the ciphertext.

Furthermore, it would be good to make an actual implementation of the applied cryptosystem on a real microcontroller of an actual car to benchmark the performance. When the implementations are performing poorly, it would be interesting to investigate an optimization of the code to make it less time consuming.

Sharing cars is an interesting concept with respect to the proposed cryptographic solution, because it introduces new security challenges. It should be investigated how and where the cryptographic keys should be stored, to be able to distinguish between the car owner and the borrowers. Our initial thought would be to store these keys on a chip inside the physical car key, but when a car is borrowed by someone who is not the owner of the car, it is hard to distinguish between the owner and the borrower. Maybe a possible solution is to create or use an existing chip/card to store or use the cryptographic keys, e.g. the identity card. Also related to that, the master key of the car owner is currently solely generated in the car. Currently, when a car is stolen, the master key can be re-generated and overwritten within the car, which disallows the car owner to view the raw data when his car is stolen and the key is overwritten. A possible solution to this might be to involve the car manufacturer in the re-generation of the master key.

Additionally, the construction of an end-to-end solution will be an interesting extension to this research. In such a research, the performance of the components should be tested and the communication between the parties should be tested against several attacks.
Bibliography


[41] E. Bloustein, *Privacy as an Aspect of Human Dignity: An Answer to Dean Prosser*. New York University, School of Law, 1964. 23


[46] European Court of Human Rights, “Uzun v. Germany (35623/05),” 2010. 27


[54] R. D. Carmichael, “On the numerical factors of the arithmetic forms $\alpha n \pm \beta n$,” *Annals of Mathematics*, vol. 15, no. 1/4, pp. 30–48, 1913. 48


[64] A. K. Lenstra, Key Lengths, June 2004. 57, 58, 70


import java.math.*;
import java.util.*;
import java.security.SecureRandom;

public class Bresson {
    /* Public static parameters which are often used */
    public static BigInteger ONE = BigInteger.ONE;
    public static BigInteger TWO = new BigInteger("2");
    public static BigInteger FOUR = new BigInteger("4");
    public static BigInteger MINUS_ONE = new BigInteger("-1");

    private BigInteger x;
    private BigInteger p, q; // p and q are two large primes
    private BigInteger lambda; // lambda = ((p-1)*(q-1))/2.
    public BigInteger n; // n = p*q
    public BigInteger nsquare; // nsquare = n^2
    public BigInteger g; // g = a^((2n) \mod n^2) with random a in range [1, n^2]
    public BigInteger h; // h = g^x \mod n^2
    public int bitlength; // bitlength of n

    public Bresson(int bitLength) {
        bitlength = bitLength;

        /* Constructs two randomly generated k-bit probable primes */
        p = BigInteger.probablePrime(bitLength / 2, new SecureRandom());
        q = BigInteger.probablePrime(bitLength / 2, new SecureRandom());
        n = p.multiply(q);
        nsquare = n.multiply(n);

        /* Constructs random variable a in range [1, n^2] */
        BigInteger a = new BigInteger(bitLength, new SecureRandom());
        a = a.mod(nsquare.subtract(ONE)).add(ONE);

        /* Constructs generator g = a^(2n) \mod n^2 */
        g = MINUS_ONE.multiply(a.modPow(TWO.multiply(n).subtract(ONE)).divide(TWO));

        /* lambda = (p-1)(q-1)/2 */
        lambda = p.subtract(ONE).multiply(q.subtract(ONE)).divide(TWO);

        /* Pick a random x in the range [1, (n^2)/2] */
    }
}
```java
x = new BigInteger(bitlength, new SecureRandom());
mod(nsquare.subtract(ONE)).divide(TWO).add(ONE);

/* Compute h = g^x mod n^2 */
h = g.modPow(x, nsquare);

/**
 * Encrypts plaintext m. ciphertext (A,B) = (g^r, h^r * (1 + mn)) mod n^2.
 * This function explicitly requires random input r and x to help with encryption
 * @param m plaintext as a BigInteger
 * @return ciphertext (A,B) as a BigInteger array
 */
public BigInteger[] Encryption(BigInteger m) {
  /* Pick a random r in the range [1,n/4] */
  BigInteger r = new BigInteger(bitlength, new SecureRandom()).div(n.subtract(ONE).divide(FOUR)).add(ONE);
  return new BigInteger[] {
    g.modPow(r, nsquare),
    h.modPow(r, nsquare).multiply(ONE.add(m.multiply(n)))
  };
}

/**
 * Encrypts plaintext m. ciphertext (A,B) = (g^r, h^r * (1 + mn)) mod n^2.
 * This function explicitly requires random input r and x to help with encryption
 * @param m plaintext as a BigInteger
 * @param g generator as a BigInteger
 * @param h public key as a BigInteger
 * @param n public modulus as a BigInteger
 * @return ciphertext (A,B) as a BigInteger array
 */
public BigInteger[] Encryption(BigInteger m, BigInteger g, BigInteger h, BigInteger n) {
  BigInteger nsquare = n.multiply(n);
  /* Pick a random r in the range [1,n/4] */
  BigInteger r = new BigInteger(bitlength, new SecureRandom()).div(n.subtract(ONE).divide(FOUR)).add(ONE);
  return new BigInteger[] {
    g.modPow(r, nsquare),
    h.modPow(r, nsquare).multiply(ONE.add(m.multiply(n)))
  };
}

/**
 * Decrypts ciphertext c. plaintext m = L(B/(A^{-x}) mod n^2) mod n.
 * @param (A,B) ciphertext as a BigInteger
 * @return plaintext as a BigInteger
 */
public BigInteger Decryption(BigInteger A, BigInteger B) {
  BigInteger modInvA = A.modPow(x.multiply(MINUS_ONE), nsquare);
  return B.multiply(modInvA).subtract(ONE).divide(n).mod(n);
}

/**
 * Decrypts ciphertext B. plaintext m = L(B^lambda mod n^2) * lambda^{-1} mod n.
 * @param B ciphertext as a BigInteger
 * @return plaintext as a BigInteger
 */
public BigInteger Decryption(BigInteger B) {
  BigInteger u = B.modPow(lambda, nsquare);
  return u.subtract(ONE).divide(n).multiply(lambda.modInverse(nsquare)).mod(n);
}

/** main function */
public static void main(String[] args) {
  int bitlength = Integer.parseInt(args[0]);
  SecureRandom random = new SecureRandom();
```
// instantiating an object of bresson cryptosystem */
Bresson b = new Bresson(bithlength);

// instantiating two plaintext msgs */
BigInteger m1 = new BigInteger("123");
BigInteger m2 = new BigInteger("42");

// Computing the ciphertexts c_i = (A_i, B_i) */
BigInteger[] c1 = b.Encryption(m1);
BigInteger[] c2 = b.Encryption(m2);

System.out.println("m1 : " + m1 + 
"m2 : " + m2 + 
"dec (A1, B1) : " + b.Decryption(c1[0], c1[1]) + 
"dec (B1) : " + b.Decryption(c1[1]) + 
"dec (A2, B2) : " + b.Decryption(c2[0], c2[1]) + 
"dec (B2) : " + b.Decryption(c2[1]) + 
D(E(m1) * E(m2) mod n^2) = (m1 * m2) mod n */
BigInteger product_B1B2 = c1[1].multiply(c2[1]).mod(b.nsquare);
BigInteger product_A1A2 = c1[0].multiply(c2[0]).mod(b.nsquare);
BigInteger sum_m1m2 = m1.add(m2).mod(b.n);
System.out.println("m1 * m2 : " + m1.multiply(m2) + 
"dec (B1 * B2) : " + b.Decryption(product_B1B2) + 
D(E(m1)^m2 mod n^2) = (m1^m2) mod n */
System.out.println("m1^m2 : " + m1.modPow(m2, b.nsquare) + 
"dec (B1^m2) : " + b.Decryption(c1[1].modPow(m2, b.nsquare)) + 
)}
Appendix B

Pseudo code algorithms for the proposed cryptosystems

\[ p \leftarrow \text{safePrime(bitLength/2)}; \]
\[ q \leftarrow \text{safePrime(bitLength/2)}; \]
\[ n \leftarrow p \cdot q; \]
\[ \text{nsquare} \leftarrow n \cdot n; \]
\[ a \leftarrow \text{rand}(1, \text{nsquare}); \]
\[ g \leftarrow -a^{2n} \mod \text{nsquare}; \]
\[ \text{lambda} \leftarrow \frac{(p - 1)(q - 1)}{2}; \]

**Algorithm 1:** Key generation Bresson et al.-based solution

\[ r \leftarrow \text{rand}(1, n/4); \]
\[ A \leftarrow g^r \mod n^2; \]
\[ B \leftarrow h^r \cdot (1 + m \cdot n) \mod n^2; \]

**Algorithm 2:** Encryption Bresson et al.-based solution

\[ u \leftarrow B \cdot A^{-x} \mod n^2; \]
\[ m \leftarrow (u - 1)/n \mod n; \]

**Algorithm 3:** Decryption algorithm using \( A^x \)
Bresson et al.-based solution

\[ u \leftarrow B^{\lambda(n)} \mod n^2; \]
\[ m \leftarrow (u - 1)/n \cdot \lambda(n)^{-1} \mod n; \]

**Algorithm 4:** Decryption algorithm using \( \lambda \)
Bresson et al.-based solution
APPENDIX B. PSEUDO CODE ALGORITHMS FOR THE PROPOSED CRYPTOSYSTEMS

$p \leftarrow \text{safePrime}(\text{bitLength});$
$q \leftarrow (p - 1)/2;$
$g \leftarrow \text{rand}(1, p);$  
$x \leftarrow \text{rand}(1, 2q);$  

Algorithm 5: Key generation Ateniese et al.-based solution
$k_{\alpha \rightarrow \beta} \leftarrow (h_3)^{\frac{1}{2}} \mod p;$

Algorithm 6: Re-Encryption key generation
Ateniese et al.-based solution
$r \leftarrow \text{rand}(1, 2q);$  
$A \leftarrow e(g, h)^r \mod p;$  
$B \leftarrow (Z^m \cdot Z^r) \mod p;$

Algorithm 7: Re-encryption Ateniese et al.-based solution

Algorithm 8: First level encryption Ateniese et al.-based solution
$r \leftarrow \text{rand}(1, 2q);$  
$A \leftarrow h^r \mod p;$  
$B \leftarrow (Z^m \cdot Z^r) \mod p;$

Algorithm 9: Second level encryption Ateniese et al.-based solution

Algorithm 10: First level decryption Ateniese et al.-based solution
$m \leftarrow \log_Z(B/A^\frac{1}{2} \mod p);$  

Algorithm 11: Second level decryption Ateniese et al.-based solution

$m \leftarrow \log_Z(B/e(A, g)^{\frac{1}{2}} \mod p);$