ABSTRACT

Title of dissertation: HARDWARE ASSISTED SOLUTIONS FOR AUTOMOBILE SECURITY

Qian Wang
Doctor of Philosophy, 2019

Dissertation directed by: Professor Gang Qu
Electrical and Computer Engineering Department

In the past couple of decades, many in-vehicle features have been invented and deployed in order to make modern vehicles which not only safer and more reliable but also connected, smarter, and intelligent. Meanwhile, vehicular ad-hoc networks (VANETs) are proposed to provide communications between vehicles and road-side stations as the foundation of the intelligent transportation system to provide efficient and safe transportation. To support these updated functions, a large amount of electronic equipment has been integrated into the car system. Although these add-on functions around vehicles offer great help in driving assistance, they inevitably introduced new security vulnerabilities that threaten the safety of the on-board drivers, passengers and pedestrians. This has been demonstrated by many well-documented attacks either on the in-vehicle bus system or on the wireless vehicular network communications. In this dissertation, we design and implement several hardware-oriented solutions to the arousing security issues on vehicles. More specifically, we focus on three important and representative problems: (1) how to secure the in-vehicle Controller Area Network (CAN), (2) how to secure the communication between vehicle and outside, and (3) how to establish trust on VANETs.

Current approaches based on cryptographic algorithms to secure CAN bus violate the strict timing and limited resource constraints for CAN communications. We thus emphasize on the alternate solution of intrusion detection system (IDS) in this dissertation. We explore monitoring the changes of CAN message content or the physical delay of its transmission to detect on the CAN bus. We first propose a new entropy-based IDS following the observation that all the known CAN message injection attacks need to alter the CAN identifier bit. Thus, analyzing the entropy changes of such bits can be an effective way to detect those attacks. Next, we develop a delay-based IDS to protect the CAN network by identifying the location of the compromised Electronic Control Unit (ECU) from the transmission delay difference to two terminals connected to the CAN bus. We demonstrate that both approaches can protect the integrity of the messages on CAN bus leading to a further improve the security and safety of autonomous vehicles.

In the second part of this dissertation, we consider Plug-and-Secure, an industrial practice on key management for automotive CAN networks. It has been proven to be information theoretically secure. However, we discover side-channel attacks based on the physical properties of the CAN bus that can leak almost the entire secret key bits. We analyze the fundamental characteristics that lead to such attacks and propose techniques to minimize information leakage at the hardware level.

Next, we extend our study from in-vehicle secure CAN communication to the communication between vehicle and outside world. We take the example of the popular GPS spoofing attack and show how we can use the rich information from CAN bus to build a cross-validation system to detect such attacks. Our approach is based on the belief that the local driving data from the in-vehicle network can be authenticated and thus trusted by secure CAN networks mechanisms. Such data can be used to cross-validate the GPS signals from the satellite which are vulnerable to spoofing attacks. We conduct driving tests on real roads to show that our proposed approach can defend both GPS spoofing attacks and location-based attacks on the VANETs.

Finally, we propose a Blockchain based Anonymous Reputation System (BARS) to establish a privacy-preserving trust model for VANETs. The certificate and revocation transparency is implemented efficiently with the proofs of presence and absence based on the extended blockchain technology. To prevent the broadcast of forged messages, a reputation evaluation algorithm is presented relying on both direct historical interactions of that vehicle and indirect opinions from the other vehicles.

This dissertation features solutions to vehicle security problems based on hardware or physical characteristics, instead of cryptographic algorithms. We believe that given the critical timing requirement on vehicular systems and their very limited resource (such as the bandwidth on CAN bus), this will be a very promising direction to secure vehicles and vehicular network.

HARDWARE ASSISTED SOLUTIONS FOR AUTOMOBILE SECURITY

by

Qian Wang

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2019

Dedication

To my baby Angela Peggie

Acknowledgments

I owe my gratitude to all the people who have helped me to make this thesis possible and because of whom my graduate experience has been one that I will cherish forever.

First and foremost I would like to thank my advisor, Professor Gang Qu for his support and guidance throughout my journey of pursuing the doctoral degree. For the past five years, he has kindly advised me in the research on hardware security and supported me to explore new interesting ideas. In addition to his academic support, I greatly value his personal and emotional support for my life and future career. It has been a pleasure to work with and learn from such an extraordinary individual.

I would also like to extend my thanks to Professor Yasser Shoukry, Professor Charalampos Papamanthou, Professor Manoj Franklin and Professor Yang Tao for agreeing to serve on my dissertation committee and for sparing their invaluable time reviewing the manuscript. Especially, I would thank Professor Shoukry reviewing the draft of this dissertation and providing constructive feedback, who offered great counseling for my career.

My sincere thanks also go to Dr. Shalabh Jain and Dr. Jorge Guajardo, who provided me an opportunity to join their team at Bosch as an intern, and who gave me kind advice and supervision. I would also like to acknowledge Intel Internship Program and Mr. Manoj Sastry for his kind support during the internship at Intel. The work at Intel gave me an invaluable lesson on how in-vehicle network security is applied in an industrial setting.

My colleagues at Maryland Embedded Systems and Hardware Security Lab have enriched my graduate life in many ways and deserve a special mention. My collaboration with Zhaojun Lu, Mingze Gao, Tanvir Arafin, Xi Chen, Omid Aramoon, Qian Xu and Pengfei Qiu have been very fruitful. I would also like to acknowledge help and support from my roommates and friends especially Yanzhou Liu, Yitian Wang, Yuntao Liu, Yeming Hao, Yanxue Hong and Guowei Sun. They have all helped me to live a smooth life while focusing on my Ph.D.

My thanks also go out to the support funds for my Ph.D. study as my work was in part sponsored by the National Science Foundation and by the Air Force Office of Scientific Research.

Finally, I want to express my deepest thanks to my family, my parents who have always loved and supported me, and have pulled me through against impossible odds at times. I also owe thanks to a very special person, my husband, Hui Cai for his continued love, support and understanding during my pursuit of Ph.D. degree that made the completion of dissertation possible. Words cannot express the gratitude I owe them.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AD	Analog-to-digital
AR	Autoregression
BARS	Blockchain-based Anonymous Reputation System
CA	Certificate Authority
CAN	Controller Area Network
CAN-FD	CAN with Flexible Data Rate
CerBC	Blockchain for Certificates
CRC	Cyclic Redundancy Check
DDoS	Distributed Denial of Service
DLC	Data Length Code
DoS	Denial of Service
DSRC	Dedicated Short-range Communication
ECU	Electronic Control Unit
EDR	Event Data Recorder
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GW	Gateway
ID	Identifier
IDS	Intrusion Detection System
IMU	Inertial Measurement Unit
IoT	Internet of Things
ITS	Intelligent Transportation System
LEA	Law Enforcement Authority
LSTM	Long Short-term Memory
MA	Moving Average
MAC	Message Authentication Code
MesBC	Blockchain for Messages
NTP	Network Time Protocol
OBD-II	On-board Diagnostic

OBU	On Board Unit
OSM	Open Street Map
PKI	Public Key Infrastructure
PMU	Phasor Measurement Unit
PnS-CAN	Plug-and-Secure protocol in CAN
PoW	Proof of Work
RCP	Resource Command Processor
RevBC	Blockchain for Revoked Public Keys
RPM	Revolutions per Minute
RSU	Road Side Unit
SVM	Support Vector Machine
TA	Trusted Authority
TLS	Transport Layer Security
TPD	Tamper Proof Device
TRIP	Trust and Reputation Infrastructure-based Proposal
TTP	Trusted Third Party
VANETs	Vehicular ad-hoc Networks
VAR	Vector Autoregression
V2I	Vehicle-to-Infrastructure
V2V	Vehicle-to-Vehicle

List of Algorithms

Chapter 1: Introduction

1.1 Overview of Automobile Security

Over the past few years, an increase in the number of connectivity interfaces is added to the traditional automobile to improve safety and convenience. To assist those fantasy computerized functions, sensors with electronic control unit (ECU) are increasingly demanded in modern vehicles. For example, approximately 50-100 ECUs are deployed in a modern vehicle nowadays. Each ECU is assigned with some specific functions and will be communicated to each other through the well-known controller area network (CAN). CAN protocol is the de-facto standard for in-vehicle networks and this adoption has been mandated in all cars manufactured since 2008. Despite the high reliability and widespread implementation of the CAN protocol, it is not surprising that vehicles can be hacked owing to vulnerabilities on the CAN bus. Moreover, the potential for interference in such attacks is exacerbated by the increasing external connections with the in-vehicle control network, including USB ports, WiFi, Bluetooth, and the mandatory on-board diagnostic (OBD-II) port. As a result, the lack of security fundamental protections of the CAN makes the in-vehicle CAN bus networks vulnerable to attacks. In 2010, researchers provided the first evidence showing that it is possible to control the vehicle by simply delivering compromising CAN messages [3]. By injecting these malicious CAN messages, they could cause intentional malfunction of a wide of components including the engine, instrument panel, radio, heating lights brakes and locks. Later on, several high profile attacks have been demonstrated on modern car by academic and industrial researchers, e.g. [4, 5, 6]. These attacks are primarily facilitated by the lack of security (authentication, encryption) in the existing architecture of CAN.

On the other side, as the plenty of smart functions have been added on the vehicles, vehicles increasingly evolve into an intelligent system. The intelligent vehicles equipped with radio communications interfaces will exchange data with neighboring vehicles or the other entities through the vehicular ad-hoc networks (VANETs). However, some unique features such as high mobility, short connection times and short-range connection make conventional network security mechanisms are not always effective on VANETs. As a result, the vehicular network systems are much more vulnerable to security threats and restricted on the network conditions. As proved in paper [7], the recently aroused security attacks on the communication channel and sensors would cause significant instability in the communication of vehicle systems. Furthermore, those communication disabilities will influence the other applications as well, i.e, cooperative driving and autonomous driving [8]. Therefore, the security challenges need to be solved in order to make such networks safety usable in practice. To solve the security problem in the next generative vehicular networks, several frameworks to guarantee the security credential management for vehicular communications are proposed [9]. On the other hand, when communicating through the VANETs, the drivers' privacy needs to be protected as well. In particular, the other entities and casual observers should not be able to track down the vehicle's trajectory in the long term. It is also of great importance providing the anonymous authentication with preserved privacy for the VANETs. Several works have been published focusing on this problem [10, 11].

1.2 Inter-vehicular Network: CAN

ECUs installed in the vehicle need to communicate and exchange information through the in-vehicle bus system follows CAN protocol [12]. CAN is the most commonly used protocol in the vehicle industry and it has a fast reaction time and proper transmission reliability [13]. It is a multi-master message broadcast system with a maximum signal rate of 1Mbps. It allows the ECU to control the vehicle maneuvers by sending and sharing messages through its bus system with the guarantee of timing predictability and fault-tolerance for communication. As a result, CAN is commonly used for vehicles' core control systems like body systems, engine management, and transmission control. Compared to the traditional networks, such as USB and Ethernet, CAN does not send large data blocks under the supervision of a central master. In a CAN network, many short messages are broadcasted to the entire network, which provides data consistency in every node of the system [14].

1.2.1 CAN Frame

The CAN message, also known as CAN frame, contains fields as identifier (ID), data length code (DLC), data field, cyclic redundancy check (CRC) and other control bits as shown in Fig 1.1 [15]. The messages sent out on the CAN bus are distributed based on the identifier preceding the CAN frame. For the base frame format, the identifier contains 11 bits and it has 29 bits for the extended frame format. However, the ID in the CAN frame is neither the sender address nor the receiver address. The first function of the identifier is to represent the priority of the message. As we discussed before, CAN is a broadcast protocol and each ECU could request the transmission without any central arbitration. To make sure the transmissions are in sequence, the ID serves as the flag of the priority of that message. Instead of the priority, the ID also represents the function of the message. For example, the ID as 0x21 may represent the vehicle speed while the other ID such as 0x13 may represent the tire pressure information. This function-mapping is defined by the automobile manufacturer and is usually not disclosed to the public. In practice, automotive designers assign a set of IDs for each ECU and one ID could be assigned to several ECUs as well. The overlap in using message IDs makes CAN messages are easily forged and difficult to be traced [16].

The data field contains a maximum of 8 bytes to be transmitted within the message frame. Usually, the transmitted data is either the instructions of the ECUs or the readings from the installed sensors on the vehicles. Usually 8 bytes could meet the basic requirement of communication on the CAN bus and guarantee the real-time simultaneously. However, recently aroused vulnerabilities from the CAN bus may require security protections which implementing the cryptographic algorithms with longer bit-length, i.e, the message authentication code usually needs 256-bit payload. The payload of CAN frame seriously hinders the application of cryptographic algorithms [17] and protocols [18, 19] on the CAN. Those cryptographic protocols based methods that send additional messages for message authentication, will lead to a rapid increase in the bus load (to more than 50%). As a result, there are arousing discussions and proposals to replace the CAN network by the Ethernet [20] which mainly because the Ethernet packets have enough pay-load for complex cryptographic protocols. The control field contains some control flags of the frame, e.g, the length of the data field and the reserved bit. The fault tolerant of the CAN data frame is realized by the cyclic redundancy check (CRC) code which is appended after the data field. The CRC field is designed to detect any error in the data packet caused by transmission. And after CRC, two bits are assigned for acknowledgment. The acknowledgment field confirms the receipt of a valid CAN packet.

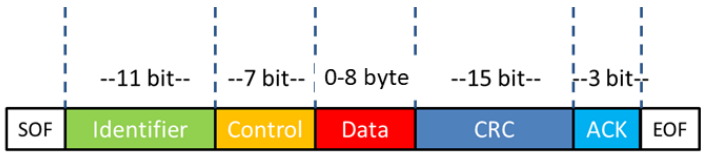

Figure 1.1: CAN-frame in base format

1.2.2 Arbitration Mechanism of CAN Bus

CAN bus is a broadcast medium consisting of a series of nodes connected via a twisted-pair cable with termination impedance at the end. It has two logical states, the dominant 0 state, where the bus is driven by a voltage, and the recessive 1 state, where the bus is grounded. CAN data transmission uses this bit-wise arbitration method to decide which message should be sent on the CAN bus when multiple messages request the bus simultaneously. The arbitration rule is that the dominant (logical 0) bits can always overwrite recessive bits (logical 1). More specifically, CAN protocol implements the arbitration relied on the identifier field of the frame with high-priority ID will win the bus arbitration while the node with low-priority ID will be grounded because of losing the capability in transmission. The node with lower priority who fails to transmit in the last time would automatically attempt to re-transmit six bit clocks after the end of the last message. This makes CAN bus system very suitable as a real-time prioritized communication system. Also, this arbitration method requires all nodes on the CAN network to be synchronized on the CAN network at the same time [21].

Since the CAN bus follows the bit-wise arbitration rule, the ID allocation is very important because the arbitration is decided by the first 11-bit identifier. As each frame starts with the 11-bit ID, the dominant bits overwrite the recessive priority bits of other frames transmitted at the same time. Only the highest priority frame is received by all the connected nodes, while a lower priority device automatically backs off when a recessive bit has been overwritten by somebody else. At first, message IDs must be unique when transmitted on the CAN bus, otherwise, two nodes would continue transmission beyond the end of the arbitration field (ID) causing an error. Next, the manufactures should consider the message priority when they allocate the IDs through the CAN bus. For example, they should assign those critical real-time messages with high priority (the lower numerical ID). As a result, bus usage can be achieved before any messages deadlines are missed [22]. A sender ECU may include its unique ID number in the packet, and a receiver ECU retrieves the packet by identifying the ID of the sender. Thus CAN packet has no explicit destination field.

The modern automobile may have up to 100 ECUs for diverse subsystems. An ECU controls the specific maneuvers for his subsystem by sending out the messages orderly and receiving the sensor data by decoding the messages through the CAN bus. The messages sent out on CAN bus are distributed based on the ID preceding the CAN frame. However, the ID is not the sender address nor the receiver address. The ID represents the function of the message, and this function-mapping is defined by the automobile manufacturer and is not disclosure to public. In practice, automotive designers assign a set of IDs for each ECU and one ID could be assigned to several ECUs as well. The overlaps in using message IDs would arouse a challenge for the detection system, even the detection mechanism could find out the malicious message based on the ID, he may fails in knowing who is actually sending this. Thus, it is difficult to identify the malicious attacker just with the knowledge of the message IDs.

1.2.3 CAN Bus Network Typology

CAN bus has different standardized speeds ranging from 5 to 1000 kBit/s. Most common are 500 kBit/s and 100 kBit/s, while 500 kBit/s networks have higher demands in terms of cables and CAN bus transceiver chips. Depending on the make and model, there are several CAN buses in a modern car. The most prominent reason for having more than one CAN bus is to guarantee the clear separation for safety in order to hold fault-tolerance in case one bus fails. The other benefit is that using multiple buses could reduce cost as deploying the lower speed CAN buses where high-speed CAN buses are not needed. Because of the difference in the speed, there are two types of CAN bus layout, the high-speed CAN and the low-speed CAN [5, 23]. Notice that for both high-speed and low-speed CAN, the speed of the transition is faster when a recessive to dominant transition occurs since the CAN wires are being actively driven.

ISO 11898-2 is the most used physical layer standard for high-speed CAN networks. The data rate is defined up to 1M bit/s with a required bus length of 40 m at 1M bit/s. As shown in Figure 1.2, the high-speed CAN uses a linear bus terminated at each end with 120Ω resistors. The high-speed standard specifies a two-wire differential bus whereby the number of nodes is limited by the electrical busload. The two wires are identified as CAN H and CAN L. The common mode voltage ranges are from -2 V on CAN L to +7 V on CAN H. The node can be a simple I/O device or an embedded controller with the CAN interface. The node may also be a gateway allowing a standard computer to communicate over the USB or Ethernet port to the devices on a CAN network. The linear topology CAN network is usually used as a high-speed CAN bus for the critical automotive applications, e.g, the engine and transmission.

The layout of low-speed CAN bus network is regulated in the ISO 11898-3 standard. It is also called fault tolerant CAN, uses a star bus and is terminated at each node by a fraction of the overall termination resistance. Fault tolerant CAN is often used where groups of nodes need to be connected together forming a sub-

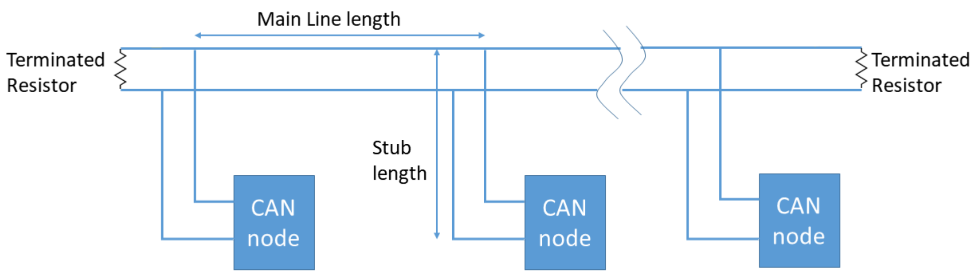

Figure 1.2: High-speed CAN Network ISO 11898-2

network. Since for this specification a short network was assumed, the problem of signal reflection is not as important as for long bus lines. This makes the use of an open bus line possible which indicates that low-speed drivers can be used for networks with very low power consumption and the bus topology is no longer limited to a linear structure. It is possible to transmit data asymmetrically over just one bus line in case of an electrical failure of one of the bus lines. ISO 11898-3 defines data rates up to 125 kbit/s with the maximum bus length and up to 32 nodes per network are specified. The common mode voltage ranges are between -2 V and +7 V. The power supply is defined at 5 V. The low-speed CAN bus is usually used on some non-critical applications, such as the entertainment system or display unit of the vehicle.

In a real vehicle, both the two types of bus layout will be applied. Moreover, the length of the bus network will be extended to 3-10 meters based on the scale and design of the real vehicle. The physical layout of ECUs will not be altered after manufacturing, even under cyber attacks where the attacker could easily achieve access on the bus but have no capability in changing the physical layout of the CAN bus.

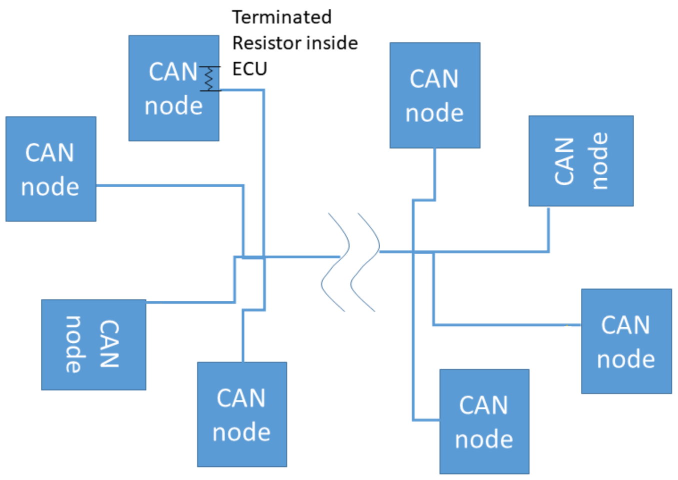

Figure 1.3: Low-speed CAN Network ISO 11898-3

1.2.4 CAN with Flexible Data Rate

CAN with Flexible Data Rate (CAN-FD) is an extension to the classic CAN bus protocol. It is an improved CAN-based communication protocol, with higher communication bandwidth (up to 8 Mbps for the payload) and increased payload size (up to 64 Bytes) [24]. The CAN-FD protocol is designed with the goal to increase the bandwidth while keeping unchanged most of the software and hardware (especially the physical layer).

The CAN-FD protocol realizes the requirement for "real time" by minimizing of delays between instruction and transfer of data (latency) with higher bandwidth. CAN-FD also allocates more data capacity in the CAN-frame. As aforementioned, the classic CAN has the capacity as 8 bytes in the data field, while CAN-FD can hold up to 64 bytes. This is accomplished through a decrease in relative overhead and improvements to software simplicity and efficiency when transmitting large data. CAN-FD also has decreased the number of undetected errors by improving the performance of the CRC algorithm. In addition, CAN-FD is compatible with the existing CAN networks, allowing the new protocol applied on the same network as classic CAN. CAN-FD has been estimated to transmit data up to 30 times faster than the classic CAN. Besides, CAN-FD protocol controllers can take part in normal CAN communication. This allows a gradual introduction of CAN-FD nodes into existing CAN systems.

1.3 Vehicular Ad-hoc Networks

Intelligent Transportation Systems (ITSs) [1, 25] have gained lots of popularity in both industry and academia. In addition, to provide entertainment services, the main motivation of ITS is to improve road safety and driving conditions [7]. For example, by enrolling in the ITS, some critical driving conditions detected by the in-vehicle embedded sensors could be shared with nearby vehicles or facility. Usually, VANETs are established with two types of communication, namely Vehicle-to-Vehicle (V2V) communication and Vehicle-to-Infrastructure (V2I) communication [26]. In V2V communication, vehicles communicate with nearby vehicles to exchange information. Similarity, vehicles communicate with the infrastructure installed along the road directly by the V2I communication. Dedicated short-range communication (DSRC) radio [27] is the commonly used communication protocol for V2V and V2I communications in VANETs.

1.3.1 System Structure of VANETs

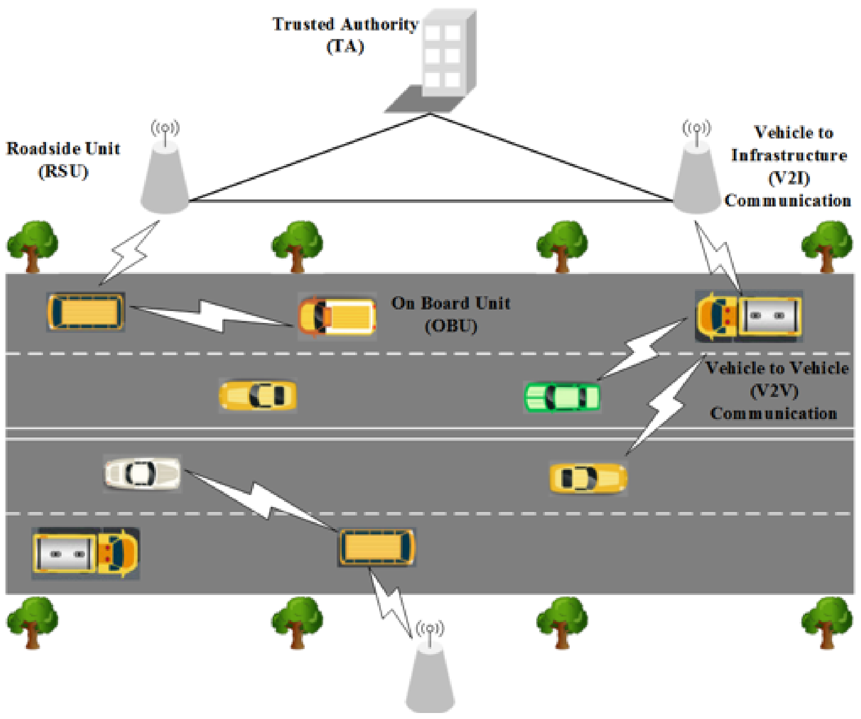

Figure 1.4: System model of VANETs

The architecture of VANETs involves various hardware and software components. As shown in Figure 1.4, the VANET system consists of three major components: OBU (On Board Unit), RSU (Road Side Unit) and trusted authority (TA). OBU of each vehicle is connected with a sensor network to gather information such as velocity, breaking information, etc. It can communicate with RSUs and OBUs of other vehicles [28]. All RSUs along the road are interconnected with each other. TA is connected to all the RSUs through a wired connection with the role in managing the entire VANETs.

1) On Board Unit: Each vehicle is equipped with an OBU as a transceiver to communicate with RSUs and the other vehicles. OBU consists of resource command processor (RCP), read/write storage, network device, and sensors. Sensors such as global positioning system (GPS), tamper proof device (TPD), event data recorder (EDR), speed sensor will collect information and send those readings back to the OBU. Then OBU monitored and gathered the information to form messages, which will be sent to neighboring vehicles and RSUs through wireless medium [28].

2) Roadside Unite: RSUs are generally stationary devices installed along the roadside or at dedicated locations such as at intersections or parking spaces [29]. The RSUs are equipped with network devices for dedicated short-range communication with the OSU on vehicles as well as communication with the infrastructural network. The main functions of RSUs include: (i) Extending the communication range of VANETs by relaying the messages to other OBUs and RSUs. (ii) Running safety applications such as traffic condition reporting or accident warning. (iii) Providing Internet connectivity to OBUs.

3) Trusted Authority: TA is responsible for the trust and security management of the entire VANETs including verifying the authenticity of vehicles and revoking nodes in the case of broadcasting fake messages or performing malicious behavior [28]. Thus, the TA has high computational capability and sufficient storage capability.

The ITS aims to provide solutions to road safety and the traffic congestion problems. It improves the driving conditions by integrating information technology in transport systems. Efficient communication channels play an essential role for the ITS. Here, we will distinguish two possible types of communications in the VANETs:

- **Vehicle To Vehicle:** V2V is communication between vehicles in the ad-hoc mode. In this mode, a vehicle can receive, transmit or exchange valuable traffic information such as traffic conditions and road accidents.

- **Vehicle To Infrastructure:** V2I is used to broadcast messages between the network infrastructure and vehicles. It is usually for exchange of useful information about road conditions and sensor measurements to be taken into account. In this mode, a vehicle first establishes a connection with the RSU and then communicates with external networks such as the Internet.

As a new emerging technology in transportation, VANETs differs from the other ad-hoc networks as the basic entity in this network is fast moving vehicles. To sum up, VANETs have the following unique characteristics compared to other types of networks, :

1. Mobility: Vehicles in VANETs are normally moving at high speed. Therefore, a little delay in V2V/V2I communication can result in many problems [30].

2. Dynamic Network Topology: The topology of VANETs is changing quickly due to high mobility of the vehicles. This makes the VANETs vulnerable to attacks and difficult to identify malicious vehicles.

3. Real-time Constraints: The transmission of information in VANETs has a particular time limit range. This is designed to give the receiver sufficient time to make decisions and take corresponding actions promptly.

4. Computing and Storage Capability: It is ordinary to process a large amount

of information among vehicles and infrastructures in VANETs. Thus, the computing and storage capability are absolutely a challenging issue.

5. Volatility: It is normal that the connections between two nodes in VANETs just occur once because of their mobility. The connections between nodes would remain for a limited period of time within a few wireless hops [31]. Thus, it would be difficult to ensure the security of personal contacts in VANETs.

1.3.2 Security Issues for VANETs

The driving force behind VANETs is to provide comfort and safety to drivers and passengers. Here, we outline the effective security mechanisms which should be designed to ensure the appropriate operations of VANETs. The key security services are availability, confidentiality, authenticity, data integrity and non-repudiation [28]. In Figure 1.5, the services and their corresponding threats are listed, which will be elaborated next in this section.

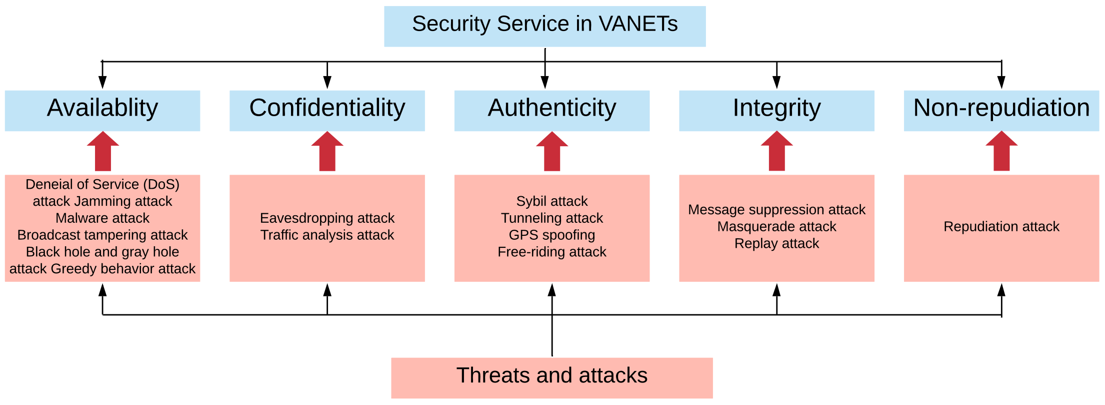

Figure 1.5: Security services in VANETs and the corresponding threats and attacks [1]

- **Availability:** Availability is a critical security requirement for VANETs. It ensures that the network and applications remain operational even in the presence of faulty or malicious conditions [32]. Several attacks are designed to break the availability of the VANETs. In this category, the most famous is the Denial of Service (DoS) attacks [33]. Inside or outside attackers perform the DoS by jamming the communication channel or overriding the resources in VANETs. The attackers may be distributed, which is the Distributed Denial of Service (DDoS) [34]. When malware is installed to operate the OBUs and RSUs, attackers can penetrate into the VANETs to disrupt the normal functionality.

- **Confidentiality:** Confidentiality guarantees that the data can be accessed only by designated recipients who are not able to understand confidential information that pertains to each entity [30]. If a mechanism lacks confidentiality for the exchanged data in a vehicular network, exchanged messages are particularly vulnerable to attacks such as the improper collection of clear information [35]. In these cases, the attacker can gather information about the location of the vehicle or its routes, which may affect the privacy of individuals. It is difficult to detect this kind of attack since it is virtually passive and user currently is not aware of the collection. The typical example which threatens confidentiality is the eavesdropping attack [36]. Eavesdropping is the intentional attempt to get confidential information about the protected data. The attacker could steal identity information or collect location data of a target vehicle in order to track that vehicle. The attacker may listen to the message transmission and then analyze the frequency and duration of messages to gather confidential information.

- **Authenticity:** Authentication is a mechanism to protect the VANETs against the malicious entities, which is considered to be the first line of defense against various attacks in VANETs [28]. All existing entities in the network must authenticate before accessing available services. Any violation or attack involving the process of identification or authentication exposes all the network to serious consequences. Ensure authenticity in a vehicular network is to protect the authentic nodes from outside or inside attackers infiltrating the network using a falsified identity [35]. The importance of identificationauthentication process comes from the fact that it is frequently used whenever a vehicle needs to join the network or service. There are several types of attacks in this category. For example, in Sybil attack, a malicious node forges a large number of fake identities in order to disrupt the proper functioning of applications in VANETs [37]. It makes an illusion to other vehicles that there is a traffic jam and enforces them to change their routes so that the road will be left clear. The authenticity is also threatened by the GPS spoofing attack. An attacker can generate false GPS signals stronger than the original signals from the trusted satellites to fool the vehicles to think that it is available in a different location [38]. In cooperative authentication scheme, vehicles may take advantages of others' authentication contributions and rarely make their own.

Such selfish behavior is referred to as free-riding attack [39], which will pose a serious threat to cooperative message authentication.

- **Integrity:** Integrity ensures that the content of a message is not modified during transmission, which protects against the unauthorized creation, destruction or modification of data. The integrity might be hindered by the following attacks. The most common is the message suppression/fabrication/alteration attack. The attacker alters some part of the transmitting message to bring about an unauthorized effect [40]. The masquerading attacker enters into the VANETs system as a valid user with stolen passwords and creates false messages [41]. The attackers may continuous re-inject previously received beacons and messages back on to the network, which will confuse the traffic authorities when identifying the vehicles in emergency incidents [42].

- **Non-repudiation:** Non-repudiation in security means the ability to verify that the sender and the receiver are the entities who claim to have respectively sent or received the message [43]. The non-repudiation of data origin proves that data has been sent and the non-repudiation of arrival proves that they were received. In the VANETs' context, since the manipulated data related to the safety and privacy of the users, it should be always possible to verify all hardware and software changes of security settings and applications (update, modification, addition, etc.) [44].

1.3.3 Trust Models for VANETs

Trust management is an inherent issue in VANETs. Various authentication methods are developed to ensure the messages are sent from legitimate vehicles. However, it cannot prevent a legitimate or internal vehicle from broadcasting bogus or altered messages malevolently. These bogus or altered messages may not only decrease transportation efficiency but also will cause accidents that may threaten human's life [45]. For example, vehicle A broadcasts a warning message to notice the vehicles behind it that vehicle A is out of control. When vehicle B receives this warning, it is crucial for B to determine the trustworthiness of the message and take a quick response. In this case, it is impractical to ask neighbor vehicles or trusted third party (TTP) for help due to the urgent time constraint. If this warning message is bogus, it is dangerous for vehicle B to brake hard. As a result, how to establish trust among authenticated vehicles becomes a serious issue. It is desirable that each vehicle in VANETs can detect dishonest vehicles and the malicious messages sent by them.

Because of the unique characteristics of VANETs, some challenges are presented for trust management such as decentralization and scalability [46]. Moreover, it is common that two vehicles may interact with each other just for one time and there is no guarantee to meet in the future [47]. Thus, it is impossible to depend on centralized systems such as TTP to build long-term relationships. Here, we summarized the popular trust models applied in VANETs.

- **Entity-centric Trust Models:** Entity-centric trust models focus on evaluating the trustworthiness of vehicles by establishing a reputation system or making a decision according to the opinions of neighbors. Several typical works based on this scheme have been proposed. Minhas et al. [48] develop a multifaceted trust modeling approach to detect entities that are generating malicious data. This method incorporates role-, experience-, priority-, and majority-based trust to make real-time decisions. Mrmol et al. [49] propose a trust and reputation infrastructure-based proposal (TRIP) for VANETs to quickly and accurately distinguish malicious or selfish nodes with the help of RSUs. TRIP takes into account three different sources of information when calculating the reputation score for each node: direct previous experiences with the targeting node, recommendations from other surrounding vehicles and, when available, the recommendation provided by a central authority. Haddadou et al. [50] propose a distributed trust model to allocate credits to nodes and securely manage these credits. It creates self-selection among the network's nodes and exhausts the credit for those nodes with bad behavior. Generally, the correctness of data from other vehicles can be guaranteed. Due to the high mobility of vehicles, it is difficult to collect enough information to calculate the reputation score of a specific node. Moreover, how to ensure the security of the reputation system itself is another critical issue that has not been resolved.

- **Data-centric Trust Models:** Data-centric trust models focus on proving the trustworthiness of the received data. In order to verify the trustworthiness of the received data accurately, the models need cooperative information from various sources such as neighbor vehicles or RSUs. Gurung et al. [51] propose a trust model to directly evaluate the trustworthiness of a message based on various factors such as content similarity, content conflict, and route similarity. Huang et al. [52] develop a voting system with different voting weights according to the distance from the event. The opinion from the vehicle closer to the event possesses higher weight when evaluating the trustworthiness of a message. Rawat et al. [53] propose a deterministic approach to evaluate the trust level of the received message by using received signal strength (RSS) for determining the vehicle's geolocation (position coordinate). Hussain et al. [54] suggest email-based trust and social network based trust to establish and manage data level trust. The main limitations of data-centric trust models are latency and data sparsity. Respectively, a large number of data from various sources may contain redundant information, which will increase latency or overwhelm the significant information. On the contrary, data sparsity is prevalent in VANETs. It is unrealistic for data-centric trust models to perform well without enough information.

- **Combined Trust Models:** Both entity and data based trustworthiness are considered in this category. Combined trust models not only evaluate the trust level of vehicles but also calculate the trustworthiness of data [55]. Thus, these models inherit both the benefits and drawbacks of entity-centric and data-centric trust models. Attack-resistant trust management scheme proposed by

Li et al. [55] evaluates the trustworthiness of both data and mobile nodes to cope with malicious attacks in VANETs. Trustworthiness of data is evaluated based on the received data from multiple vehicles. Trustworthiness of a node is determined based on functional trust and recommendation trust, which respectively indicates whether a node can fulfill its function and what is the trust level of it.

1.4 Hardware Assisted Solutions for Automobile Security

1.4.1 Motivations of the Dissertation Research

Nowadays, automobiles have evolved from mechanical devices to connected communication platforms. Approximately 50-100 ECUs are installed in vehicles nowadays and they are no longer isolated from the outside world. Indeed, smart features such us adaptive cruise control, autonomous steering/parking, lane keeping etc. are enabled by ECU controller and widely deployed in the vehicle system. Each ECU is assigned with several functions and should be able to connect and share messages through the bus system. CAN is the most commonly used protocol which allows the ECU to control the vehicle maneuvers by sending and sharing messages through its bus system with the guarantee of timing predictability and fault-tolerance for communication. Unfortunately, CAN has no mechanism for supporting security protocols or for ensuring message confidentiality and integrity by primary design. The lack of security in CAN bus has aroused much attention in both industry and research. Moreover, the potential for interference in such attacks is exacerbated by the increasing external connections with the in-vehicle control network, including USB ports, Bluetooth, the mandatory OBD-II port and the extended connection to the VANETs [56].

Come to the VANETs related applications, GPS spoofing attack on vehicle is a typical example. Nowadays, each vehicle of the VANETs is required to be equipped with a positioning system (receiver) to achieve the applications in communication and data sharing. GPS is typically used by the vehicle or the driver for self-localization and navigation but the technology is also used for remote traffic surveillance and collision avoidance on the VANETs. Despite for the in-vehicle threatens, The major goal of spoofing attacks on an automobile GPS system is to produce erroneous position signals to fool the navigation systems. Since many of the functions and applications in VANETs are dependent on the civilian GPS signals to locate and synchronize in the network, successful GPS spoofing attack can not only hinder the location based applications but also facilitates other cyber attacks as well (i.e., Sybil attack on VANETs). Besides, it is crucial for the VANETs to prevent internal vehicles from broadcasting forged messages while simultaneously preserving the privacy of vehicles against tracking attacks. However, there are some challenges aroused by the intrinsic features of the VANETs, as the high mobility makes it unpractical to build long-term interaction among vehicles. As a result, the topology always changes rapidly in VANETs as well, which require an efficient but low-cost verification for the enrolled vehicles of the network.

To mitigate those serious attacks aroused or to solve the bottlenecks of ensuring security around vehicles, several techniques have been proposed for integrating security into the current CAN architecture and the VANETs [57, 58]. Similar to traditional secure systems, these techniques rely on secure provisioning cryptography algorithms, e.g., adding MAC scheme on CAN bus, applying encryption on the GPS signal and establishing trust model based on cryptographic algorithms. However, using those standard and traditional cryptographic techniques to protect data originality and integrity may violate timing constraints for both CAN and VANETs communications. As a result, in this dissertation, we focus on searching some light-weight schemes to handle those security vulnerabilities around the vehicles.

Specifically, we start from investigating the features of CAN protocol and proposed automotive Intrusion Detection Systems (IDS) can be applied without causing computational and communication overhead in the CAN protocol. Because of its self-adaptive nature, IDS can be easily adapted in the automotive domain as a seamless extension to new vehicles. We then take use the CAN infrastructure to assist the originality in GPS spoofing detection without any further hardware costs on equipment. Besides, we also brought up a blockchain based trust management system. Blockchain is believed to be a secure and decentralized computational infrastructures, which could provide a disruptive solutions for the problems of centralization, privacy and security when storing, tracking, monitoring and sharing data. The verification process assisted by blockchain is light-weight thus it is a promising candidate for implementing the certification verification on VANETs. We exploit the features of block-chain to establish distributed trust management while simultaneously protect the privacy of vehicles for the VANETs.

1.4.2 Contributions of the Dissertation Research

To address these security issues aroused around the vehicles, we are aiming to find security solutions for the automobiles in this dissertation. Specifically, we are focusing on several security primitives: IDS, side channel attacks, spoofing attacks, trust models and blockchains.

In sum, the contributions of this work are highlighted as the following:

- First, we present two effective IDSs for the CAN network. The first IDS relies on monitoring the entropy changes on the identifier field of the CAN frame. This entropy-based IDS is designed to cover the common CAN messages injection attacks. The other one is based on the physical layout of CAN network which could detect compromised ECUs by identifying the changes of delay. These physical property-based IDS could identify injection attacks as well as impersonation attacks by fingerprinting the sender ECU.

- Then, we study the vulnerabilities along with the PnS-CAN protocol. We identify that the characteristics of CAN bus can be utilized to extract the secret key during execution of the protocol. To demonstrate this, we have launched successfully side-channel attacks on this protocol.

- Next, we move the scope from the internal vehicle CAN bus protocol to external vehicle applications. By making use of the informative CAN bus data, we proposed a low-cost GPS spoofing detection method. Besides, the data collected from the in-vehicle network is believed to be authenticated and trusted by the IDS we proposed, which does not need any interference by the third parties.

- Finally, we exploit the features of block-chain to establish distributed trust management while simultaneously protect the privacy of vehicles for the VANETs. All the messages are recorded in the blockchain as persistent evidence to further evaluate the reputation score. The reputation score provides an incentive for internal vehicles to prevent misbehaviors.

1.4.3 Organization of the Dissertation Research

In this dissertation, we first propose two kinds of IDS on the CAN bus system and demonstrate their functions to detect three types of regular attacks on bus. The details of this work are illustrated in Chapter 2. In Chapter 3, we study the new key exchange protocol, i.e, the Plug-and-Secure protocol with CAN bus (PnS-CAN), and investigate the physical leakage on this protocol. Besides, by utilizing the rich in-vehicle data, we build a GPS spoofing detector for automobile applications as illustrated in Chapter 4. In Chapter 5, We design a trust model to improve the trustworthiness of messages sent on the VANETs, which is based on the blockchain technology. At last, we conclude in Chapter 6, with future work directions in machine learning based IDS solutions and high-efficient blockchain solution for VANETs .

Chapter 2: Intrusion Detection Systems for CAN bus

2.1 Introduction

With the recent aggressive push of autonomous vehicles, many ECUs are introduced for data processing and communications between inside the vehicle network (through the CAN bus) and the VANETs [59]. Moreover, the applications of the vehicle network make the vehicle as a connected device to the Internet of Things (IoT), not just a traditional physical part. As vehicles adopt more connectivity functions to the external network, security threats on electronic controllers of vehicles are highly rising. With safety remains the most important concern for automakers, security is now gaining lots of attention because a compromised ECU or faked communication messages can cause fatal failures just like mechanical problems. It has been shown that attacks can be successfully launched on cars to cause severe impacts on the normal driving behaviors [3, 5]. Indeed, the security level of vehicles should be set higher than those for other smart devices because cars could become critical threats to human life once they are attacked. By injecting malicious CAN messages through those ports, the attacker could cause intentional malfunctions to a wide set of components including the engine, instrument panel, radio, heating, lights, brakes, etc. These attacks are primarily facilitated by injection from the connecting ports to the CAN bus network. Despite the injection attack on the CAN bus, another kind of attack as the masquerade attack has gained increasing attention nowadays. This kind of attack is more severe than the regular injection attack as the attacker hide his identity when launching the intrusion which is difficult for the system to discover. To efficiently defend this, one possible method is to find out the real sender by fingerprinting the sender ECU.

When the in-vehicle serial communication network was designed in the late 1980s, there was only about 1% electronic equipment in the vehicle system. Thus the original design requirements of such network at that time were merely to connect the limited in-vehicle ECUs. A simple message broadcasting based communication protocol, known as the CAN, was used for this purpose without any consideration of security. Not surprisingly, due to its lack of security concerns and protection, CAN's vulnerabilities are exposed to the attackers and become a major target for a variety of threat models. Those successful attacks demonstrate that without the intrusion detection or authentication methods, it becomes very challenging for the CAN-based communication network to defend emerging attacks such as message replays, injections, and modification [60, 61]. Nevertheless, the important role that CAN plays in the communication of vehicular systems has made it an irreplaceable component and thus the research community has continued its efforts in seeking secure and efficient protection method for the CAN network.

2.2 Countermeasures and Current Status of IDS

As discussed above, the simple design of CAN bus makes it vulnerable to cyber attacks from the outside and security has become the primary issue. Generally speaking, there are two kinds of countermeasures to defend against attacks on CAN bus: message authentication and intrusion detection.

Message authentication code (MAC) has been successfully used in other security applications, but it is highly restricted by the nature of CAN that was not designed for security. Several cryptographic techniques have been proposed for integrating security into the current architecture [57, 58, 62]. Similar to traditional secure systems, these techniques rely on secure provisioning of symmetric keys within the nodes on the CAN bus. However, secure and robust provisioning, maintenance and update of cryptographic keys within the automotive supply chain can incur significant overhead and may require changes to the automotive manufacturing and servicing facilities. The implementation is also limited by the payload of the CAN message which is only 64 bits. For example, if we want to implement SHA256 with CAN bus payload, it will need 8 messages to finish an authentication code. Obviously, it will introduce inevitable delays for the whole system. This stems from the fact that the CAN protocol allows only for 64 bits to be used for payload. That is, implementing the SHA256, for example, using the CAN protocol will lead to an overhead of eight messages to carry the computed SHA256 code; an 800% increase in the communication bandwidth. This large increase in the communication bandwidth will degrade the performance of the system in terms of its ability to satisfying the required hard timing constraints. When the message is safety critical, such as emergency stop, this could result in a life-threatening incident.

To ensure the security of the CAN bus and reduce the complexity introduced by cryptographic algorithms, intrusion detection systems are proposed to employed with the CAN protocol. The aim to develop IDS is to detect anomalies on the messages and alert the system when monitoring the bus transmissions. Existing automotive IDSs can be summarized into two categories: message-content (data) based IDS and physical-based IDS. First, data-based IDS usually focuses on the messages transmitted through the bus to find any abnormal in the contents or sequence. The data contents would be changed if the attacker presents includes the message ID [63], the payload [64] and the frequency [65] or message sequences [66]. By taking use of information theory models or machine learning models [67], the changes from the contents could be detected. The advantage of data-based IDS is that it is easy to detect data alteration/fabrication attacks and spoofing attacks with appropriate pre-learned models. But the drawbacks are that it could not efficiently identify the impersonation attacks. For example, if ECU A shuts down ECU B using bus-off attack, and then he could impersonate ECU B by sending malicious messages on the bus, which is difficult to be detected by the content-based IDSs. On the contrary, the other main kind of IDSs, the physical-based attacks, are monitoring the changes of physical related properties on the bus, e.g., the clock-skew, the voltage level and the transmission delays. By exploring the clock-skew of different ECUs, it could fingerprint the sender of the malicious messages [68]. The other voltage-based attacker identification takes use of the voltages changes on the network to identify the attacker ECU. The first proposed voltage based scheme is called Viden, which exploits the voltage measurements to construct and update the transmitter ECUs' voltage profiles as their fingerprints [69]. VoltageIDS is also a voltage-based IDS, it is the first automotive IDS capable of distinguishing between errors and the bus-off attack [70]. Those physical-based IDS would require attack samples about measurements to training its model. Hence, it ends up being unpractical since it is not feasible to apply with the vehicle system.

2.3 Adversary Models and Attack Scenarios

Before introducing the IDS framework, we first define and elaborate the adversary models in this section. Specifically, we would describe the attack interfaces which the adversary would take advantage use and also the specific attack scenarios we try to tackle for the proposed IDS work.

2.3.1 Attack Interfaces

The main goal of the adversary is to transmit malicious CAN messages intentionally causing malfunctions of the vehicle without being suspected by the detection mechanism. Car hacking experiments show that the attack interfaces the adversary could get access to the in-vehicle communication network are restricted to several locations. Usually, the adversary would invade the in-vehicle network through two common physical attack surfaces: either by directly plugging into the OBD-II port or using remote wireless communication interface. As shown in Figure 2.1, both the

OBD-II port under the driving panel and the antenna on the roof are circled. For the directly wired insertion, the attacker can insert his device to collect CAN log data through the OBD-II port. For the wireless connection, the attacker can use the wireless channel to inject messages through the ECU attached to the antenna. In most cases, the positions the adversary uses are fixed on the vehicle during the attack. Once set up the attack interface, it is unnecessary for the adversary to change it. For the accessibility of the adversary model, we assume that the adversary can compromise an in-vehicle ECU through numerous attack surfaces, and thus gain its control. Once an ECU is compromised, we consider the adversary to be capable of performing at least the following malicious actions. The adversary can inject message with forged IDs or his legal IDs, change the frequency of the messages. Also, since CAN is a broadcast bus, the adversary can sniff messages on the CAN bus.

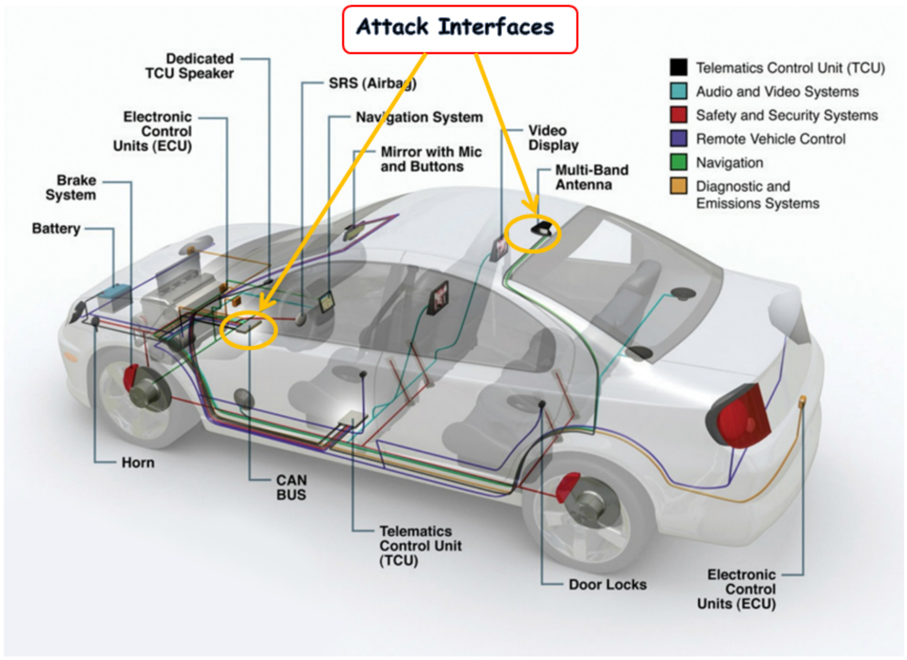
Attack Interfaces
Dedicated TCU Speaker
SRS (Airbag)
Electronic Control Units (ECU)
Navigation System
Mirror with Mic and Buttons
Video Display
Multi-Band Antenna
Brake System
Battery
Horn
CAN BUS
Telematics Control Unit (TCU)
Door Locks
Electronic Control Units (ECU)
■ Telematics Control Unit (TCU)
■ Audio and Video Systems
■ Safety and Security Systems
■ Remote Vehicle Control
■ Navigation
■ Diagnostic and Emissions Systems

Figure 2.1: The attack interfaces of CAN bus and the CAN bus layout in real vehicle [2]

All the facts discussed above are considered as the general adversary model for our proposed two kinds of IDS. For the entropy-based IDS which relies on monitoring the changes on the contents, we consider the attacker can take use of both the physical connection and wirelessly connected interfaces, there are no significant differences for which interface the attacker applied, since the behavior we try to monitor belongs to digital signal (i.e., whether this bit is 0 or 1).

For the delay-based physical IDS, which relies on the physical location to detect the adversary. We consider an adversary that is capable of injecting malicious messages on the CAN bus through the fixed ports. However, we assume that the adversary has no capability in modifying the physical layout of the CAN bus. The only method he could modify the layout is by breaking into the vehicle, which could make it discovered easily. Also, even though he could break into the layout and install the malicious ECU, he could not dynamically change the location of the malicious ECU. During the injection, the adversary can only modify the identifier and the contents of the messages, which is the same as the general injection assumption.

2.3.2 Specific Attack Types

2.3.2.1 Injection Attack

A straightforward yet quite threatening attack on the CAN bus is the message injection attack. To achieve this, the adversary first connects to the in-vehicle network through the interface and then starts to inject malicious messages onto the CAN bus. The proposed injection attack typically involves injecting fake packets with modified identifiers and phony data values. This injection attack is the main focus for the contents based IDS, which refers to our first proposed entropy-based IDS. Therefore, when discussing the content-based IDS, we would like to elaborate on this type of attack by two sub-groups: the strong injection attack and the weak injection attack. Meanwhile, we propose four specific injection attack scenarios to test the effects of our entropy-based IDS which naturally falls into these two categories. The practicability of such an adversary model has already been proved and demonstrated in [22, 68].

For the weak adversary model as shown in Figure 2.2, the comprised ECU can only send out messages with the specific IDs assigned to it. That is to say, the attacker is restricted in selecting CAN IDs for malicious messages. This mechanism relies on the transmitter filter installed outside of the ECU to stop the malicious messages without valid ID sending on the CAN bus. To hamper the communications on the CAN bus, a weak attacker can only send messages with limited IDs, which are legal to the filter.

Under this weak adversary injection model, we design one specific attack which would be tested in the entropy-based IDS.

(Weak Adversary Model - Messages Injection with Fixed CAN ID): This scenario is categorized as week model because the attackers can only control the ECU, but cannot control the transmitter filter. That is to say, the attacker can only transmit the fraudulent messages with assigned IDs. Even though the control power is limited in the weak model, the attacker can also break the availability of the CAN bus system by sending out useless massive messages into the bus system. That

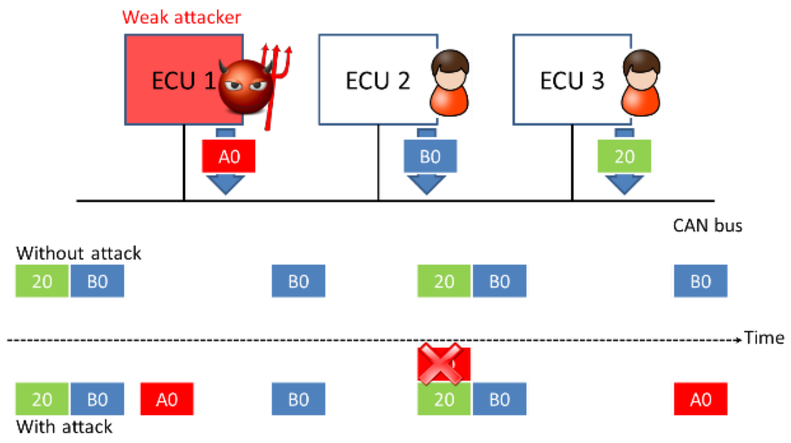

Figure 2.2: Weak attack model for the CAN bus. Here, the attacker can only send ID A0 on the bus

happens when the dummy messages sent by the attacker are dominant the original messages transmitted at this time. By manipulating the frequency of messages, the attacker can gain the bus control just using a limited number of CAN IDs.

In contrast, for the strong attack model, the attacker could send out messages with any IDs on the CAN bus, which is illustrated in Figure 2.3. Thus, the attacker with strong capabilities would realize in hampering the communication on CAN bus by two purposes. First, the adversary could send malicious messages with higher priority IDs (leading zeros are in the front, e.g. 0000XXX) to override any periodic messages sent by a legitimate safety-critical ECU. Those malicious messages might be targeted to some operational receivers as the victim ECUs. As a result, the receiver ECUs would get distracted, inoperable or executing errors. Second, by sending messages with higher priority IDs, the compromised ECU could win the bus arbitration for a period of time and stop the legal ECUs sending their regular messages, which indeed hinders the regular operations of the CAN bus. In other words, the adversary could maliciously stress the bus and compromise the usability of the other users just like flooding attacks.

By considering this strong attack capability, we designed three corresponding specific attack scenarios listed as follows,

(Strong Adversary Model - Flooding Message Injection): In this scenario, the attacker aims to break the availability of the CAN bus by sending out useless massive messages on the bus. The attacker could completely control a compromised ECU in this scenario. This kind of attack is usually accomplished by injecting CAN messages containing the most dominant identifier, i.e. 0x00. But this assumption is challenged by the CAN transceivers have the detection mechanism for zero overloads on the CAN bus. Specifically, if the transceiver sees the bus is always transmitting 0 (high voltage), it will automatically shut down the transmission by pulling the high voltage to low. A more efficient strategy for the attacker which avoids being exposed to the transceiver detection mechanism could be an attempt in sending messages with changeable IDs. This modified flooding attack mislead the detection mechanism and is difficult to be discovered by the transceivers. Besides, as there are no specific receivers for those malicious messages on the bus, which makes the abnormal traffics could not be reported by the victim receivers as well. As a result, the attacker should make a trade-off between the number of injected IDs and exposure to the detection system. For instance, if the attacker keeps sending flooding messages with changeable IDs, those messages will overwhelm the regular communications and cause significant changes on the CAN bus. On the other hand, the attacker will expose himself to the detection scheme by making obvious changes on the CAN bus. As a sequence, if the compromised ECU is easy to be found out and shut down forever, it cannot mount attacks on the CAN bus anymore.

(Strong Adversary Model - Message Injection with Single CAN ID): In contrast to the flooding scenario discussed above, the attacker could narrow down the number of IDs he uses to conceal himself from the abnormal detection schemes. Let us suppose in this scenario, the attacker injects the malicious messages with single CAN ID for two purposes: first, he wants to win the bus arbitration from the other low priority IDs thus stopping the transmission of useful messages on the bus; second, the malicious attacker could manipulate the contents of messages meaningfully and send them out with the fixed ID which would disturb the functions of the receiver ECUs. From the above discussion, we can find out that the second task is the following extension of the first one. The attacker can choose from those two tasks according to his purpose. If he only focuses on disturbing the CAN bus as the first task, he can arbitrarily select the injected ID from the whole ID set. Otherwise, if he wants to achieve both the first task and the second, he needs to carefully select the injected ID from the ID set and use it for a targeted function. As mentioned before, the distribution of CAN ID is assigned by the vehicle manufacturer and only around 10% to 20% IDs are used in the CAN network, e.g. for the 2016 Ford Fusion we used for testing, only 223 IDs are valid, which takes 10.88 % of the whole ID set (2048 in total). The other IDs are never used by any ECUs in this car. For this scenario, we assume that the injected IDs are picked from the valid CAN ID pool. From the other aspect, this attack can be categorized as the increasing frequency attack [22], which the specific IDs are sending out with higher rates. And the author proposed to counting the time interval of each ID to detect the injection. However, collecting and calculating the time interval of all ECUs would introduce delay overhead, it is infeasible for the device to respond in real-time.

(Strong Adversary Model - Message Injection with Multiple IDs): The same attack set-up as the second scenario, but this time we assume that there are multiple attackers with different injected IDs, or the attacker chooses to inject messages with multiple IDs. That is to say, the balance of the bus would be disturbed by two or more CAN IDs. This makes inferring the faked IDs more difficult as the multiple sources would influence the ID distribution together and it is not easy to figure out who is the malicious ID. But on the contrary, multiple participated IDs may also cause entropy changing tremendously on the CAN bus, which makes the alert detection more confident.

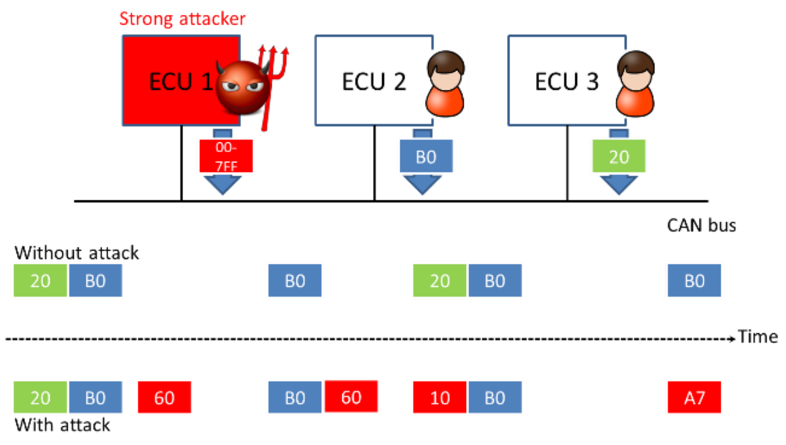

Figure 2.3: Strong attack model for the CAN bus. Here, the attacker can send arbitrary ID on the bus

The physical IDS will be able to detect those aggressive injection attack discussed above. However, since it could be also detected by the other contents based IDS instances as well, we would put this kind of attack on low priority and would emphasize the superior detection capability for the other, more sophisticated attacks, discussed as follows.

2.3.2.2 Masquerade Attack

Masquerade attack is a kind of sophisticated in-vehicle attack first proposed and demonstrated in [68]. The objective of a masquerade attack is to manipulate an ECU that is in charge of a safe-critical function while hiding the fact that the ECU is compromised. To mount a masquerade attack without being detected, an adversary needs to suspend a target ECU and then inject malicious CAN messages without notice by the other ECUs. When a legitimate ECU is incapacitated, the masquerading ECU would broadcast fake packets in its place at the same frequency or it takes other action to cover any broadcast time-shifts. In fact, for the masquerade attack, the CAN ID and message contents may not change based on the attack strategy, thus making it difficult to be detected by the contents based IDS. However, after taking over the transmission of the legitimate ECU, the attacker starts to transmit malicious messages at a different physical location. Therefore, another detection option considered in this chapter is proposed to detect the masquerade attack by utilizing the information from the physical layer of transmission.

2.3.2.3 Bus-off Attack

The bus-off attack is proposed by [22]. In this attack, an adversary who has an remote access to the in-vehicle network performs simultaneous transmission of bits in fields other than the identifier field. Due to this simultaneous transmission, a target ECU will enter the bus-off mode with loss of the bus transmission. As a result, the bus-off attacker can intentionally suspend the target ECU. The bus-off attack could be achieved with both the wireless injection and the direct attached injection through the OBD-II port. No matter which attack interface is applied, the location for the injection is fixed on the bus. Also to launch the bus-off attack, it usually requires several cycles (128 or 256) to make the counter overflow thus entering into the bus-off mode. To tackle this bus-off attack, one solution is to distinguish the transmitters even though they simultaneously transmit the message bits. The counter capacity in terms assist to accumulate the delay changes over the cycles. Our proposed delay-based plug-in-monitor IDS could achieve this by finding the delay difference of the two transmitters, as the bad ECU is supposed to transmit at a different location from the legitimate ECU. Accordingly, we could detect the malicious messages generated from the attack ECU in the bus-off attack as well.

2.4 Entropy-based Approach

2.4.1 Framework of Entropy-based Intrusion Detection System

Our proposed entropy-based IDS is deploying on the 11-bit identifier part of the CAN frame. We will discuss the implementation details in this section.

2.4.1.1 ID Binary Entropy Definition

The identifier (11 bits) in CAN frame represents the message's priority as well as its functions, but without the transmitter or receiver information. For example, a message containing wheel speed values might have ID=0x31, and the message which contains the engine temperature values has the other ID such as ID=0x20. Notice that the IDs are assigned by the manufacturers, and even are distinct among different models from the same manufacturer. Only one ECU is assigned to transmit a given ID at a time, but one ECU could have multiple assigned IDs for different functions. Besides, several ECUs might be allowed to transmit the same ID, but not at the same time. That is to say, a specific ID is not unique for one ECU and could be received by different receivers as well. Indeed, traffic in automotive networks is much more restricted. Every packet in a vehicular CAN network and its possible data content are specified. The permitted value range is defined as well as the length of every signal and the packet function. The basic frame format has an 11-bit ID, whereas an extended format has a 29-bit ID. We will use the basic CAN ID format (11-bit) because it is much more prevalent, however, the detection method proposed here is not dependent on the type of format, and it could also be applied to the extended format.

We use the probability of '1' occurrence at each bit to represent the distribution of that bit. As defined in the following,

Definition: p_i = # of messages where Bit i is 1/total number of messages. The range of p_i is from [0,1]. The binary entropy function H(p) is defined as the entropy of a Bernoulli process with probability p. If $\Pr(X == 1) = p$, then $\Pr(X == 0) = 1 - p$ and the entropy of X in Shannon is given by

$$\mathrm{H}(X) = \mathrm{H}_b(p) = -p\log_2 p - (1-p)\log_2(1-p) \tag{2.1}$$

Our proposed entropy-based IDS is deploying on the 11-bit identifier part of the CAN frame. We will discuss the implementation details in this section.

2.4.1.2 Entropy-based Detection Scheme

The first step of the ID entropy detection approach is to generate the golden template by calculating the binary entropy in regular communications. Data collected from the real CAN bus in a 2016 Ford Fusion vehicle shows that the entropy values are steady under normal driving maneuvers. The raw CAN data we collected are from different driving situations, e.g. turning the audio on, turning the light on, driving with cruise control and so on. We observe that the entropy distribution on each bit of ID field only changes slightly in these different testing scenarios. The variation is small (e.g., from the range $1e^{-8}$ to $9e^{-8}$) which is confident to prove that the entropy is steady under normal driving. Therefore, we build the golden model based on the normal CAN bus data which later can be used to compare with the CAN bus data under attack. If the entropy value changes significantly from the golden template, it will be classified as being attacked. The golden entropy value of the in-vehicle network is generated from 35 different real road tests on the 2016 Ford Fusion, including the road tests and the engine tests.

Let $\hat{\mathbf{H}} = \{\mathrm{H}_1\mathrm{H}_2\mathrm{H}_3 \cdots \mathrm{H}_{11}\}$ be the binary entropy vector measured from the real CAN log data. We get the golden template $\hat{\mathbf{H}}_{temp}$ by averaging 35 measurements from diverse driving behaviors. For each bit in the template as $\mathrm{H}_i \in \hat{\mathbf{H}}_{temp}$, we calculate the range of it as $\max(\mathrm{H}_i) - \min(\mathrm{H}_i)$. And the threshold for detection can be acquired as a positive multiple of the range, expressed as $Th = \kappa(\max(\mathrm{H}i) - \min(\mathrm{H}i))$, where κ is the coefficient which determines the margin of the threshold. The value of κ is chosen from $\kappa = [3, 10]$ empirically. In this experiment, we choose κ =5. The golden template extracted from the regular communications on CAN bus is shown in Figure 2.4. The template is an 11-bit vector as we measure the binary entropy for each bit.

In the detection procedure, we compare the binary entropy to the template and calculate the entropy change bit by bit. If the entropy change is larger than the threshold, we will suspect that the CAN bus is under intrusion attack, and the system will send an alert signal. If the entropy change is below the threshold, this change will be treated as a normal drift.

Figure 2.4 illustrates one example of binary entropy changes due to the injection attack. The golden template is represented in left side colored in red and the

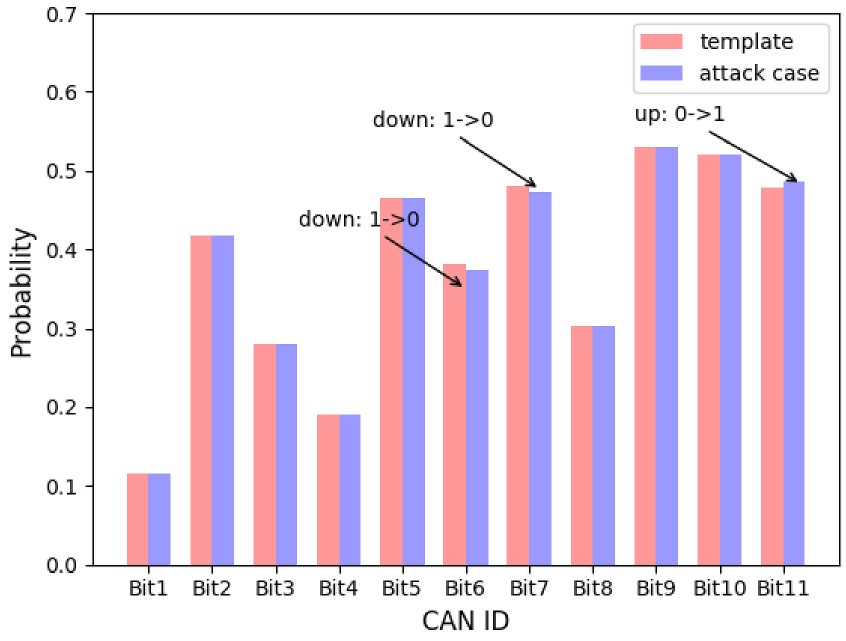

Figure 2.4: Golden template and an example when injection attack causes changes on the identifier bit

entropy result under attack is shown on the right side. First, it is evident to figure out some changes of bits, e.g., Bit 6, Bit 7 and Bit 11. On the occurrence changes at some specific bits, we attempt to infer the malicious ID from the difference between the test entropy and the golden template. As Figure 2.4 shows, Bit 11 changes in positive direction, i.e., from 0 to 1. As a result, we can infer that the injected ID would be 1 on Bit 11. On the contrary, negative values would indicate that this bit of malicious ID is probably to be 0. Therefore, we could eventually infer the malicious IDs by monitoring the bit changes in several positions. For example, the most possible injected ID, in this case, could be xxxxx00xxx1 (x means unknown for this bit). From this clue, we could narrow down the space for searching the malicious ID.

2.4.2 Evaluation and Experimental Results

In this section, we first discuss and define the metrics used to evaluate the detection system. And then we present our experimental results got from tested in the attack scenarios discussed before, which demonstrate that our entropy detection system is effective on the discussed types of attacks.

2.4.2.1 Experiment Set-up

For the application of the information-theoretic measures introduced above, access to the network traffic of a vehicle is required. In our set-up, we use the Vehicle Spy 3 Professional software tool connecting a 2016 Ford Fusion internal CAN network through the OBD-II port. And the baud rate is set as the standard for the CAN network: 125M bit/s for the middle-speed CAN and 500M bit/s for the high-speed CAN. We conduct our tests and experiments on the data acquired from the middle-speed CAN bus in this section, however, our detection method would also work for high-speed CAN bus.

2.4.2.2 Injection Rate and Detection Rate

We define the Injection rate I_r as the proportion of successfully injected messages on the bus over the total number of messages the malicious ECU sends to compete for the bus arbitration. The injection rate is relevant to the priority of the message which represents its capability to mount the CAN bus. As the CAN bus's arbitration relies on the numeric of ID, different IDs would win the bus arbitration with different probabilities. For example, if the regular message transmitting on the bus has a higher priority ID, the attacker would fail to send his malicious message on the CAN bus. The number of times the attacker win the bus over the total times he tries in a fixed period of time should be a good metric to evaluate the capability of the CAN ID, which is the injection rate defined here. On the other hand, if the attacker wants to accomplish his attack, he needs to select the proper ID in assisting him to win the bus arbitration. For instance, if the attacker chooses the IDs with higher priority (smaller value in decimal), he would have more chances to win the bus arbitration, which in turn results in increasing the number of injected messages overall.

We calculate the injection rate defined above and select some typical IDs from the CAN log data and the results are shown in Figure 2.5. The Injection rate is high when the numeric value of ID is smaller and will decrease with the numeric value of ID increases. That is due to the arbitration scheme of CAN bus as on bit level bit 0 always dominants bit 1. The injection rate is significant as it represents the destructive power of the attacker and also it highly relates to the detection rate which we will explain in the following.

The detection rate as D_r in our intrusion system is defined as the proportion of successfully detected attacks over the total number of attacks. In a period evaluation time (e.g., 5s), if the detection system observes the entropy changes is above the threshold, it will treat this attack as detected. That is to say, the detection rate is the proportion of attacks that are correctly identified.

Obviously, the detection rate is highly related to the entropy changes on the

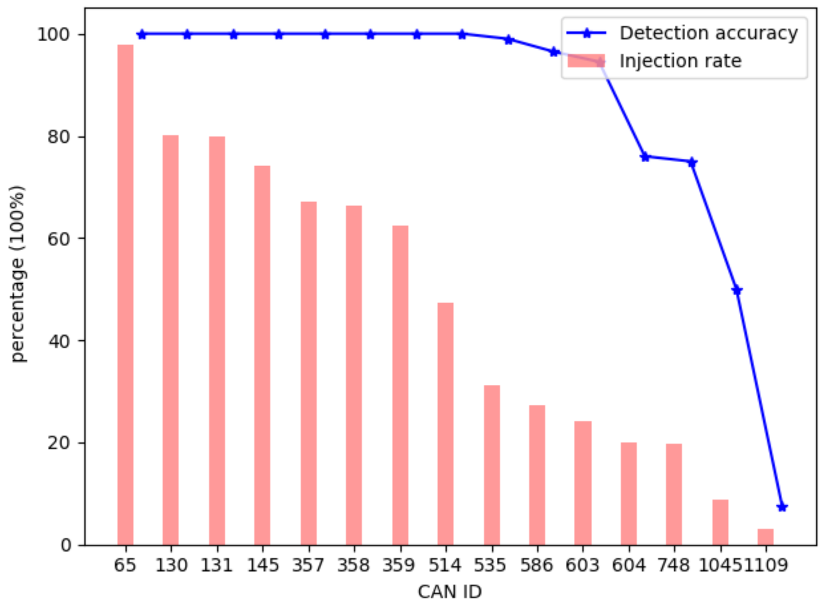

Figure 2.5: Injection and detection results for different identifiers

CAN bus. And the entropy change is influenced by the number of injected messages. Moreover, the number of injected messages depends on three parameters: the injection rate I_r of a specific ID, the injection frequency f. Thus, the total number of successfully injected messages can be expressed as a product shown in the formula

$$\#\text{of injected messages} = Ir \times f \tag{2.2}$$

If the malicious attacker attempts to increase the success rate of injection during a fixed period of time, he should either choose the ID with higher injection rate or increase the injection frequency. To demonstrate the relationship between injection rate and detection rate, we test 15 selected IDs under the same injection frequency f and evaluation period T_0.

As the results are shown in Figure 2.5, the detection rate (blue line) decreases along with the injection rate. That is because the total number of injected messages on the CAN bus would decrease which makes the entropy changes imperceptible. As a result of this, the detection rate of this scenario would decrease. The conclusion is that for just single ID injection attack, if we keep the inject frequency the same, the detection rate (sensitivity) is positively correlated to the injection rate.

2.4.2.3 Inferring Malicious IDs

The entropy-based IDS we designed will accomplish two tasks: 1) Detect the entropy changes and demonstrate the evidence of the attack; 2) Find out the injected IDs along the messages and alert to the system for those malicious IDs. Here, we will discuss how to infer the malicious IDs based on the changes of the entropy observed on CAN bus.

Continued with the example discussed before, if we observe the first bit of ID (Bit 1) changes, we can infer the corresponding bit based on the direction of changes. Assume the Bit 1 changes in negative direction (decreases), the corresponding bit of the ID will be 0 of high probability. Then we can select those IDs in the ID pool whose first bit is 0 as the candidates. By continuing this procedure and combined observed changes on the other the bits, the range of candidates would be narrow down. However, this inference is non-deterministic as we have no clues on the other unchanged bit, which could be either 0 or 1. As a result, we propose a rank selection method to evaluate the accuracy of the hit rate when searching the malicious ID. The procedure works as follows, we select the first n rank IDs as the candidates which meet the constraints derived from the entropy changes. If the malicious one is in the candidate set, we will mark this detection as a hit. We use the hit rate to evaluate the efficiency of the inferring. Due to the arbitration rule in CAN bus; the ID would be more powerful in the injection if the most significant bits are 0. As a result, we sort the IDs in ascending order, which means the preceding elements in the sorted list would have a greater probability to be the malicious target. Then, we search the target from the candidate set. If there is a hit, we will mark this as a successful detection. By calculating the hit rate, we can evaluate the success and effectiveness of our IDS system. The results of hit rate (inferring accuracy) for different attack scenarios are shown in Table 2.1. For all the tests in our experiment, we set rank=10, which indicates each time we select the first 10 identifiers as the candidates.

2.4.2.4 Results Analysis of Different Attack Scenarios

We evaluate the feasibility of the proposed IDS on the four attack scenarios discussed above. First, we evaluate the detection rate of different scenarios to verify the accuracy and efficiency, and then we discuss on the tractability for the malicious ID under different attack scenarios.

In the evaluation procedure, we test the detection rate under several circumstances, which is with randomly flooding injection, single message injection, and multiple messages injection, restricted ID message injection in weak mode (WI) with different injection frequencies (i.e., 100Hz, 50Hz, 20Hz and 10Hz). The detection rates averages from different frequencies for the 4 basic scenarios are summarized in Table 2.1.

Attack scenarios	**Detection rate**	**Inferring accuracy**
Flood	100%	—-
Single Injection	91%	97.2%
Multiple Injection_2	97%	91.8%
Multiple Injection_3	97.2%	88.5%
Multiple Injection_4	99.97%	69.7%
Weak Injection	93%	96.6%

Table 2.1: Results of detection rate and inferring accuracy from different injection attack scenarios for the entropy-based IDS

In the flooding attack scenario, the attacker's aim is to hamper the bus communication with many faked messages. It would cause dramatic entropy changes and thus this attack is the easiest one for detection. However, as the attacker arbitrary chooses the injected IDs without clear preference, it is infeasible to infer the injected IDs based on the random changeable entropy. But if the attacker keeps sending flooding messages with different IDs, it will be easily detected by the filter in the gateway. Besides, it will leak some side-channel information to the internal detection sensor of this system as it is mounted on the bus for such a long period of time. For example, the power consumption and the requests for communication will be distinct for this attacked ECU.

For the single message injection, the attacker injected one specific ID with the purpose of influencing the target ECU. First of all, this injection would cause entropy changes on the CAN bus and our intrusion system would have in average 91% rate to figure out this attack. Note this result is the average on every test CAN IDs including those with lower injection rate. For those CAN IDs with higher injection rate, the detection rate could be as high as 100%. In this scenario, it is easier to infer the malicious ID based on the entropy changes, so the inferring accuracy is as high as 97.2%.

Then, we test and evaluate for multiple ID injections where the attacker would inject messages with multiple IDs randomly selected from the ID pool. First, in this scenario, messages with multiple CAN IDs are injected into the CAN bus causing such complex changes than the single message injection. The basic approach to detect the multiple injection messages starts with extracting the clues for inferring the ID based on a combination of the entropy changes. To achieve the purpose, we need to analyze not only the change direction but also the changing rate of each bit. We modified the selected algorithm to derive new constraint for the selection. Compared with the result in the single ID injection, the average detection rate is higher as multiple IDs are used, which is due to the increase of the injection rate. However, the accuracy of inferring malicious IDs decreases when having multiple injected IDs. That is reasonable as the disturbance of multiple sources makes the algorithm difficult to select the exact combination of malicious IDs. Moreover, the inferring accuracy would decrease with adding more IDs which is also demonstrated in Table 2.1. We test the multiple injection attacks with 2, 3 and 4 injected IDs respectively and the injection rate keeps going up as we enlarge the number of IDs while the inference accuracy goes down because of the disturbance of multiple IDs. As we mentioned above, the multiple ID injection is restricted by the filter and in real applications, it is challengeable for the attacker to have a fully-compromised ECU on the CAN bus. With 4 and more injection IDs, the compromised ECU would be easily figured out by the gateway filter which is similar to the flood attack.

We also complete the same test on the weak adversary model, which the attacker is restricted by the sender filter. Thus he can only send the messages with the ID assigned to him. The detection result is same as the single message injection, however, the inferring accuracy is a bit lower than the single ID injection, because the attacker is not restricted to inject single ID on the bus, he can manipulatively change the entropy by using multiple IDs he could legally send.

From results discussed above, we could find out that the success detection rates are all above 90% for all the scenarios. And the inferring accuracy for several scenarios is all above 90% as well. Moreover, with our golden entropy model, we are able to successfully detect almost all the attacks (the 2nd column in Table 2.1) and locate the malicious message ID except the flooding attack (the 3rd column in Table 2.1).

2.5 Delay-based Approach

2.5.1 Framework of Delay-based Intrusion Detection System

To cope with the sophisticated attacks targeted on the in-vehicle CAN bus, we propose a delay-based plug-in-monitor IDS which takes use of the unaltered property of the physical topology of the CAN network. The fundamental observation from the monitor is that each ECU outputs a characteristic signal delay difference when monitoring on the two selected points. During the attack, the attacker would use the attack interface different from the original ECU on the CAN bus which will cause the measurement of delay changing from the original observations when messages are transmitted by the legitimate ECU. Therefore, we can identify those malicious messages sent from the transmitter out of the expected range, thus distinguishing the compromised ECU from the legitimate one.

2.5.1.1 The Physical Layout of the CAN bus Network

In this section, we will describe the physical layout of the CAN bus as well as the signal transmission properties on the bus, which are relative to the delay measurement in our set-up. As aforementioned, the electrical CAN signals fall into two states: a dominant state (logic 0) and a recessive state (logic 1). For both high-speed and low-speed CAN network frameworks, the transition is faster when a recessive to dominant transition occurs since the CAN wires are being actively driven. The speed of the dominant to recessive transition depends primarily on the length of wire and the capacitance of the wire used.

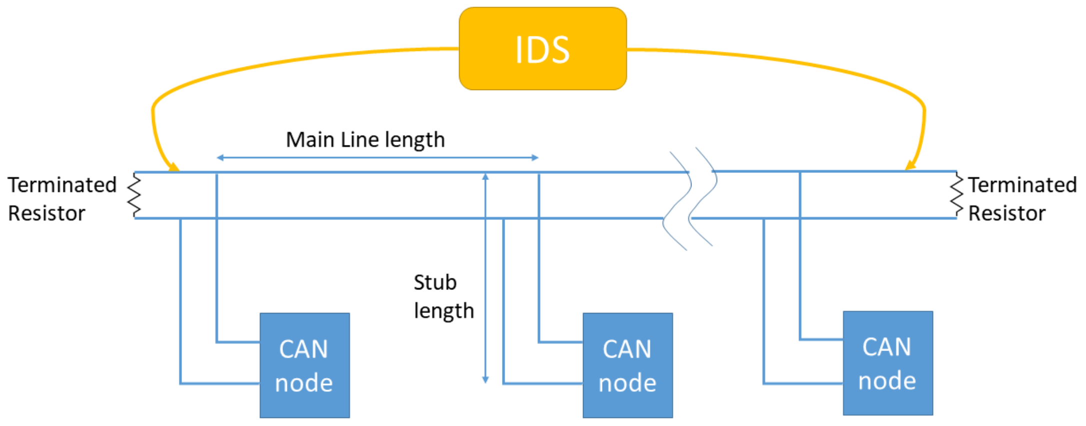

Figure 2.6: High Speed CAN Network ISO 11898-2 with the IDS monitor

CAN is a multi-master serial bus standard for connecting ECUs also known as nodes. The ISO 11898-2 standard provides guidelines for the layout of high-speed CAN network. As shown in Figure 2.6, the high speed CAN uses a linear bus terminated at each end with 120 Ω resistors. Two or more nodes join to the linear CAN network to communicate. The complexity of the node can range from a simple I/O device up to an embedded computer with a CAN interface and sophisticated software. The node may also be a gateway allowing a standard computer to communicate over a USB or Ethernet port to the devices on a CAN network. The linear topology CAN network is usually used as a high speed CAN bus for automotive applications.

The other layout of CAN bus network is regulated in the ISO 11898-3 standard shown as Figure 2.7. It is also called low speed or fault tolerant CAN, uses a star bus and is terminated at each node by a fraction of the overall termination resistance. Fault tolerant CAN is often used where groups of nodes need to be connected together forming a sub-network.

In a real vehicle, both the two types of the layout will be applied. Moreover, the length of the bus network will be extended to 3-10 meters based on the scale and design of the vehicle. The physical layout of ECUs will not be altered after manufacturing, even under cyber attacks where the attacker could easily achieve access on the bus but have no capability in changing the physical layout of the CAN bus.

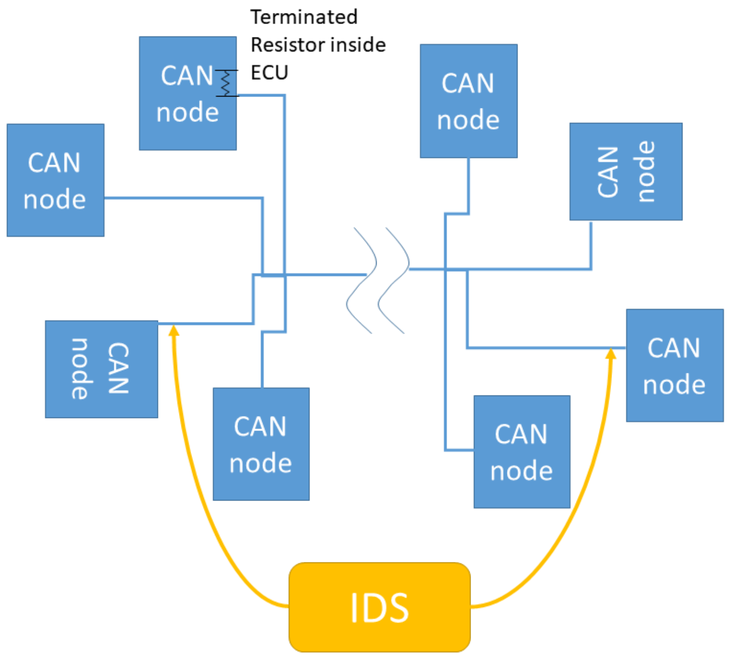

Figure 2.7: Low Speed CAN Network ISO 11898-3 with the IDS monitor

2.5.1.2 Effects of Transmission Delay

The cause for the latency of the message transmitting on CAN bus could be summarized as the speed of different CAN controllers and transceivers, gateway processing delays, propagation delay of wire, propagation delay of connectors and the clock drift or skew of the oscillators. In our work, all of the delays discussed above will count for the final delay difference captured by the monitor.

Implementing a CAN node requires a CAN transceiver and a CAN controller or processor with the appropriate protocol stack. In either case, the CAN controller must be configured to reconcile the data rate and timing on the bus with the hardware oscillator used for the controller. The different implementation of the CAN node might cause observable delay differences on the CAN bus. As cable length increases, the high-frequency content of the signal is attenuated, so data rates are limited for long distances. Propagation delay, which also increases with cable length, can interfere with the synchronization and arbitration between nodes. The typical propagation delay of a twisted pair cable for the CAN bus is 5 ns/m. Thus, for a traditional network of length up to 50 m (the maximum length for CAN bus), the difference in the time the transmitter drives (or releases) the bus and an observer observes a signal transition can be up to 250 ns. Though such delays are accommodated within the CAN bit timing specification for a correct sampling of the bit value, they can be exploited by the monitor to identify the transmitter. Except for the typical propagation delay caused by the cable length, the relative bit timing difference observed by a monitor for two transmitters can be augmented by the other delays along the transmission path as well.

The objective of our work is to fingerprint the transmitter ECU by the delay difference from the victim ECU to the attack ECU. The two kinds of bus topology with the IDS are shown as in Figure 2.6 for High-speed CAN and Figure 2.7 for Low-speed CAN. We place the monitor connecting to the CAN bus network through two wires directly attached to the bus. The monitor would watch the messages transmitted on the bus and calculate the delay difference from his two plug-in points. Thus, it is crucial to choose the monitor plug-in positions to enlarge the coverage of detection for the bus system. We propose the following specific rules based on the layouts of the CAN bus network.

1. The first rule is to maximize the delay difference of ECUs in the network in

order to distinguish them. For the high-speed CAN network, the best position to cover the maximum number of ECUs in supervision is at the end of the bus network. This is naturally because if the monitor is inserted not at the terminations, the two points will be fallen into one side of some ECUs, and the delay difference calculated accordingly will be constant. Those ECUs would not be covered in monitoring for this IDS. For the fault tolerant CAN network, the best strategy is to place the two monitor points in two separated clusters of the network as shown in Figure 2.7. If the network only has one central cluster, the two points can be placed in any two edges of the star network.

2. The second rule is to keep the distribution distinguishable. If we place the plug-in points much far away from the observation network, the monitor would not tell the delay difference for the sub-network. Thus, considering the sub-CAN network structure, we design the monitor which is responsible for a number of 5 to 20 ECUs in the sub-network.

2.5.1.3 Electrical CAN Signal Measurement and Reprocessing

More specifically, we design the procedure of the plug-in-monitor IDS into three phases, which are respectively the delay measurement, learning parameter settings, and the final intrusion detection.

- **Phase I : Profiling the delay differences for each ECU.** The first step is to measure the delay difference and register the legitimate ECUs based on the measurements when the vehicle is in normal mode. The delay difference from the transmitter ECU to the two plug-in points would be recorded with its ID into the system. Then, for each message ID transmitted on the bus, the system has the location of its legal transmitters. The profiling information will be used by the IDS to determine whether or not the message originated from the legitimate transmitters in the following step. Phase I runs as the initialization and registration step of IDS and will update the record of ECU when it is necessary.

- **Phase II: Exploiting the threshold of the delay differences.** Based on the data collected during the profiling stage, the IDS can build the distribution map for each message ID with its legitimate transmitters. Further analyzing the statistical distribution, the system will decide the threshold for each ID to distinguish between legitimate transmitters and malicious transmitters. The details in setting the threshold will be discussed in section 5.

- **Phase III: Attack detection.** For each transmitted message, only one ECU is assigned to its transmission in most cases and the position of the ECU is fixed unless the vehicle is broken into. Based on the timing delay difference, the IDS could estimate the transmission range of the ECU and further decide whether it is legitimate or not.

2.5.2 Evaluation and Experimental Results

We now evaluate the practicability and efficiency of the IDS in achieving an accurate attacker identification on the CAN bus. At first, we will demonstrate the delay difference existence on the prototype of CAN bus. Then, we will show how the IDS could identify the transmitter based on the delay difference and detect the attack consequently.

2.5.2.1 Experiment Set-up

Our experimental setup utilizes the Arduino UNO with a seeed studio CAN bus shield to prototype the ECU. The CAN bus shield consists of a microchip MCP 2515 CAN controller and a MCP 2551 CAN transceiver to provide the can bus communications. The micro-controllers are connected via the standard CAN twist pair cable with a terminal resistor.

The typical CAN architecture consists of several sub-networks of ECUs connected by one or more powerful nodes that act as gateway (GW). However, for our system, the nodes are connected in a single chain with the GW at one end. This setup closely emulates one subnet of the network found in most modern cars. We use another micro-controller as a monitor to probe the bus and record the delay time for both online and offline processing.

2.5.2.2 Delay Profiling

First, we use three Arduino UNO boards to prototype the propagation delay related to the distance. One Arduino UNO board is regarded as the master, and other two are slaves to it. A laptop is used to monitor the CAN bus communication and activities, so all three Arduino UNO boards are connected to the same laptop.

Regarding the communication part, we majorly use the communication library provided by Seeed studio to achieve the message delivery in the CAN bus system. The whole system starts by sending messages from the master with time-stamps (t_1). Then the slave who receives the message will echo a message back to the master to confirm it already received the message. When the master receives the echo-back message (time-stamp t_2), we calculate the difference of $t_2 - t_1$ which is the delay of the whole echo-back communication. During the time-stamp process, the timer (16MHz) inside the Arduino is used but we add the interrupt function to improve the time resolution. Hence, the time-stamp accuracy can be reached up to $\frac{1}{16}$ microsecond in our system. The distinguishable delay difference is shown in the Figure 2.8, we measure it at length from 0.5m, 1m, 2m, and 4m. Notice that this delay contains the initial processing delay of the micro-controller which is estimated as 190ns in this case. The real prorogation delay should be the absolute value shown in the figure minus the baseline delay. As the resolution for the timer of the micro-controller is limited, we need to accumulate multiple tests in order to generate the concrete results. The results shown in the figure are averaged by 1000 echo-back traces.

In practice, there is enough statistical dispersion for two ECUs, as depicted in the Figure 2.9. From this histogram, we can see the delay of two slaves ($t_2 - t_1$) is distinguishable. Based on this, it is possible to build a mapping between the observed characteristics and their associated IDs. And later we could classify the messages to the transmitter ECUs based on the delay difference. We also test the delay when the message content is modified. According to the data, the message

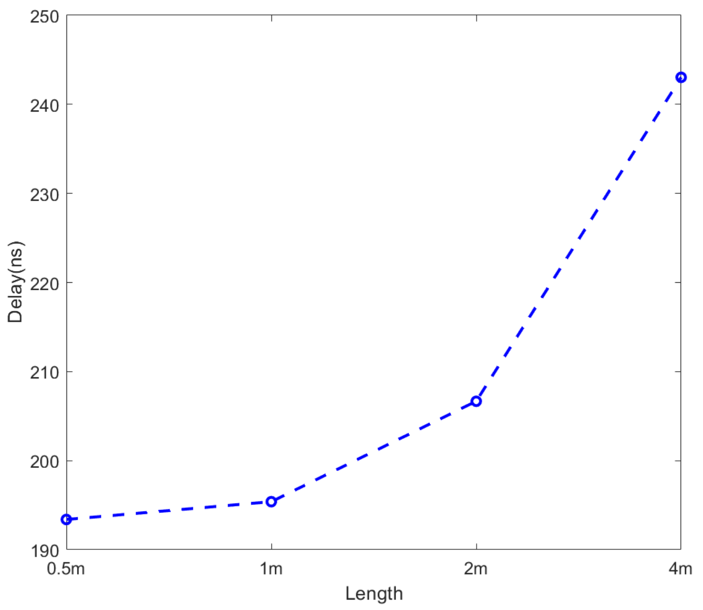

Figure 2.8: Delay changes over length of the cable

with all 0 has the largest delay. The reason for this is because zero padding messages tend to have more stuffing bits. In our future work, we will test the delay difference between two slaves under different temperatures, and also nd which specic range is the most distinguishable.

2.5.2.3 Attack Detection

A distance profile represents the ECU behavior of the message transmission. The IDS would exploit every newly derived message to construct the range of message transmitter. Although the monitor is triggered on each message and associated with message ID, if messages originate from the same transmitter, their results are near-equivalent. Thus, the IDS could detect the presence of malicious attackers by monitoring the delay difference in the network, where the delay difference indicates the location of the transmitter.

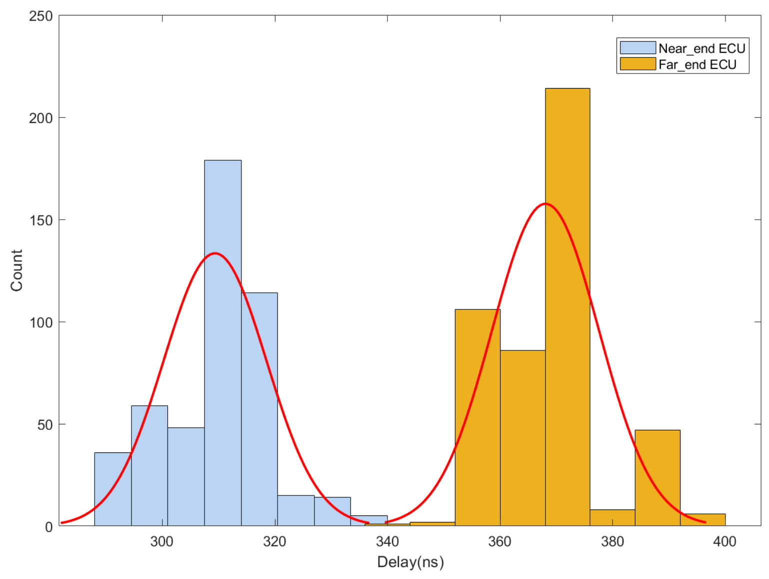

Figure 2.9: Histogram of delay difference from Near-end ECU and Far-end ECU

In the attack experimental set-up, we continually used the delay profiling acquired from the first stage. Specifically, we have two ECUs and one is deployed in the far-end and the other is in the near-end. First we assign the ECU1 with ID 0x01 and ECU2 with ID 0x02. In the clean stage, they are transmitting their messages with their assigned IDs normally. After the delay profiling procedure discussed before, the IDS is accomplishing in collecting the difference of the delay to form the range of transmitters for each message. Then we design the attack as the ECU1 would send with his assigned ID as normal but ECU2 would try to impersonate ECU1 by sending messages with IDs as both 0x01 and 0x02. When the messages are transmitted on the can bus, the IDS would refer to the profile stored and make a comparison on the record. If the delay difference falls outside the legitimate range, it will trigger the system an injection attack. As the monitor finds out that some messages with ID 0x01 are not sent from the registered location, so it will trigger an alert to the system.

The precision, recall and the F-1 score of the tests are shown in the Table2.2. Precision measures the percentage of true positive over the sum of true positive and false positive. Precision gives us the clue that the proportional of real labeled attack to some falsely diagnosed attack. While recall measure the corresponding true positive over the sum of true positive plus false negative. The false negative of attack alert means that the attack do exists but it is not discovered by the IDS. F-1 score is the average of precision and recall. To make a confident decision, we could take a threshold as the mismatch rate to generate a confidential alert of the system.

	Precision	**Recall**	**F1-score**
Benign	0.98	0.97	0.97
Attack	0.97	0.98	0.97
avg/total	0.97	0.97	0.97

Table 2.2: Results on attack detection from physical-based IDS

2.5.2.4 Detection Error Rate Analysis

The detection rate of the IDS discussed above is based on the physical layout of the ECU testbed. Eventually, the result relies on the distance of the attack ECU from the victim ECU. Here we will use a statistical method to analyze the detection error rate of the IDS.

First, based on the experimental results, we figure out the delay variances obey the Gaussian distribution as $\mathcal{N}(\mu,\, \sigma^2)$. Assume there are two ECUs on the bus, and each would have the distribution as $\mathcal{N}(\mu_0,\, \sigma_0^2)$, $\mathcal{N}(\mu_1,\, \sigma_1^2)$. First, the decision is made by the threshold th, and the delay is represented as variable d. As a result, the decision rule for two ECUs could be summarized as

- If $d > th$, then the message is sent by ECU 0
- If $d < th$, then the message is sent by ECU 1

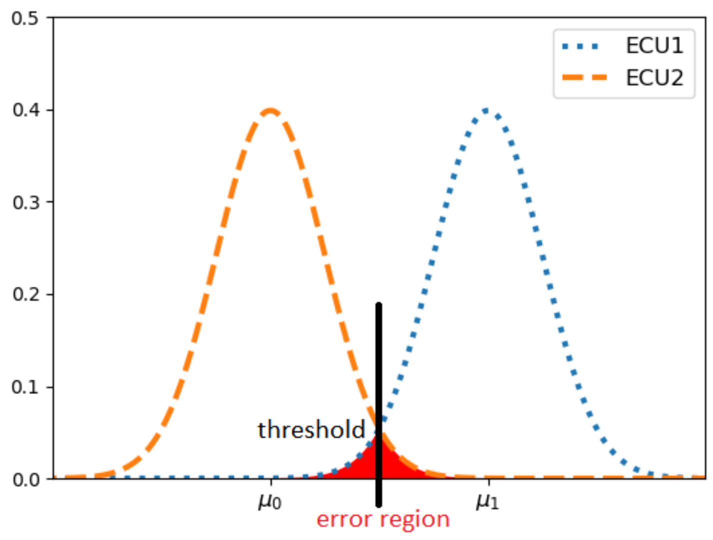

Figure 2.10: An example for delay distribution and the error region

If the means are close to each other $\mu_0 \approx \mu_1$, there will be an overlap of the decision boundary as the Figure 2.10 shows, the threshold is drawn in the vertical line. The decision boundary is used to determine the source of the message follows the rule described above. The error region is drawn in red, based on the distribution we can derive the error probability is

$$BER = \Pr(b = 0) \int_{-\infty}^{Th} p_0(x)dx + \Pr(b = 1) \int_{Th}^{\infty} p_1(x)dx \tag{2.3}$$

We need to choose the threshold such that the error rate is minimized.

$$\frac{\partial BER}{\partial Th} = 0 \Rightarrow p_0(Th) = p_1(Th) \tag{2.4}$$

where, the probability density is $p(x) = \frac{1}{\sigma\sqrt{2\pi}}e^{\frac{-(x-\mu)^2}{2\sigma^2}}$. If the distribution of ECUs is identical, the best threshold line would be in the middle of the cross area. The best case is that the two ECUs are apart to each other and there is no overlap for densities. In phase II, the decision boundary should be chosen as the optimum threshold and record it in the system.

2.6 Summary

More and more stealthy attacks targeted on in-vehicle network threaten the confidentiality of the regular messages transmitting on the bus. In this chapter, we propose two novel IDS to defend the cyber attacks on the CAN bus. The first IDS is based on monitoring the entropy changes on the bus, which is a lightweight and non-intrusive security framework that can protect CAN bus without modifying its protocols and implementation. Our proposed IDS is capable of restricting attackers from injecting a large number of malicious messages with higher priority identifiers, as well as finding out the malicious identifiers. Further, we use the measurement of delay difference between two monitor points when transmitting messages to fingerprint the ECUs on bus. We successfully demonstrated our proposed idea on CAN bus prototype built on Arduino micro-controllers and CAN bus shields. This proposed method only requires the installation of a monitor on the CAN bus network without any modification of the protocol and ECU functions.

Chapter 3: Probing Attacks on Physical Layer Key Agreement for CAN Network

3.1 Introduction

Over the past few years, an increase in the number of connectivity interfaces on a traditional automobile has drawn the attention of the security community. Several high profile attacks have been demonstrated on the modern car by academic and industrial researchers, e.g. [4, 71, 72]. These attacks are primarily facilitated by the lack of security (authentication, encryption) in the existing architecture of the CAN. Concurrently, several techniques have been proposed for integrating security into the current architecture [57, 62, 73, 74].

Similar to traditional secure systems, several of these techniques rely on secure provisioning of symmetric keys within the nodes on the CAN bus. However, secure and robust provisioning, maintenance and update of cryptographic keys within the automotive supply chain can incur significant overhead, and may require changes to the automotive manufacturing and servicing facilities. Several commercial systems to enable such a process have been proposed, e.g. in [75].

Dynamic generation and distribution of keys in a secure manner can provide an alternative (or reduce the functional requirements) of traditional provisioning systems. One promising approach, Plug-and-Secure for CAN, has been recently proposed towards this goal, in [76, 77]. A key advantage of the Plug-and-Secure scheme is the utilization of inherent physical layer properties of the CAN bus to provide security guarantees for key agreement between groups of nodes.

Cryptographic systems that are provably secure in the computational model have often been compromised by exploiting characteristics of their physical implementation. Physical characteristics such as timing differences, power leakage, and other features can provide a covert communication medium, side-channels, leaking system information to an adversary. Several attacks have been demonstrated on traditional systems using side channels [78, 79]. Comprehensive analysis of a system to identify and exploit such side-channels can be difficult. It has been observed from traditional systems that prevention of such attacks can add significant overhead to system design and negatively impact performance.

In this chapter, we demonstrate that the properties of the physical layer of the CAN bus can be utilized to violate the security of the PnS-CAN system by an adversary capable of probing the CAN bus. Here we discuss several voltages, timing and transient characteristics based side-channels that can be used to attack the system and partially extract the secret keys. We demonstrate those attacks on CAN bus test-bed and also discuss several solutions as countermeasures for those side channel attacks.

We investigate the side-channels for the two-party PnS-CAN protocol proposed in [76]. Our contributions are as follows,

1. We identify characteristics of the CAN bus that can be utilized to extract the secret key during execution of the PnS-CAN protocol. We outline a general methodology to map existing CAN identification techniques to attack our scheme. We further demonstrate the attack of the system due to one of the non-trivial feature, i.e. timing characteristics.

2. We propose countermeasures that can be included in the transceiver hardware or the CAN controller to minimize such leakage.

3.2 System Models

3.2.1 Property of CAN bus Physical Layer

CAN bus, the primary communication network for most modern cars, is a broadcast medium consisting of a series of nodes connected via a twisted-pair cable with termination impedance at either end. It has two logical states, the dominant '0' state, where the bus is driven by a voltage, and the recessive '1' state, where the bus is grounded. If two nodes transmit a bit simultaneously, the effective state of the bus is '0' if any of the nodes transmits the dominant bit. Thus, the bus acts as a logical AND gate between inputs from the nodes.

The CAN bus utilizes differential signaling to transmit the data. In the CAN standard, when transmitting the dominant bit 0 on the bus, the output pins of the nodes, CANH and CANL, are driven to different voltage levels, and the difference from CANH to CANL is the output of the CAN bus. Similarly, transmission of a recessive bit 1 occurs when CANH and CANL are not driven, and will have similar voltage levels.

As this work focuses on attacking the PnS-CAN protocol, it inherits the system requirements for successfully building that system, as enumerated in [77]. We expect the typical automotive network (CAN) to be comprised of heterogeneous nodes, i.e. nodes from different manufacturers or families. Since the PnS-CAN protocols are based on simultaneous transmission by two nodes, all write operations on the bus during the key agreement phase use ECU pairs. Thus, the ECUs require modified CAN controllers that allow simultaneous transmission of data during this phase. However, during regular operation, the network operates in a CAN-compliant manner, and hence has only single node transmitting during a frame.

3.2.2 PnS-CAN Scheme

The PnS-CAN scheme described in [76, 77], enables key agreement between multiple nodes connected to the CAN bus. For completeness, here we present a brief overview of the fundamental operations of the two-party key exchange scheme. We include specific implementation aspects, that aid our attacks, in the protocol definition. For detailed discussion about the security aspects of the original scheme, the reader is referred to [77].

The PnS-CAN scheme between two nodes utilizes the wired AND property of the CAN bus to mask the bits simultaneously transmitted by the nodes. The security of this scheme is based on the inability of an eavesdropper to differentiate between transmissions that result in the same logical output on the bus,i.e. combinations that result in the dominant (0) output. The operation of the basic two party protocol between node_{N_1} and node_{N_2}, using random seeds r_1, r_2. In this, a secret bit is generated between node_{N_1} and node_{N_2} when one of the nodes transmits a logical 0 (dominant bit) while the other transmits a logical 1 (recessive bit). Note that an ideal adversary is unable to identify which of the two nodes, node_{N_1} or node_{N_2} transmitted the dominant bit. The group key scheme described in [77] utilizes the **PnS-TwoParty** protocol between successive nodes to form a PnS chain. It was demonstrated that pairwise interactions between consecutive ECUs are sufficient to share the key with the group. We note that in systems, where the adversaries only have high-level (software) access to the nodes, they can only observe the logical output of the bus as determined by a single transceiver samples. Thus, though perfect obfuscation of the bit values in **PnS-TwoParty** is theoretically possible, it cannot be guaranteed for most practical systems where the adversaries may observe multiple high-resolution bus samples.

3.2.3 Adversary Model for PnS-CAN

We consider an arbitrarily powerful adversary that is capable of observing the variation of CAN bus signals with high voltage precision and timing resolution. Such an adversary can be simply realized by an eavesdropper who accesses the wires directly using a high precision oscilloscope. An alternate means could be through a regular ECU connected to the CAN bus with a high precision analog-to-digital

(A/D) converter at the input and a modified CAN controller capable of sampling the bus at a high frequency. In a car, such nodes can be connected to the OBD-II diagnostics port. A representation of the CAN bus with adversarial presence is illustrated in Figure 3.1.

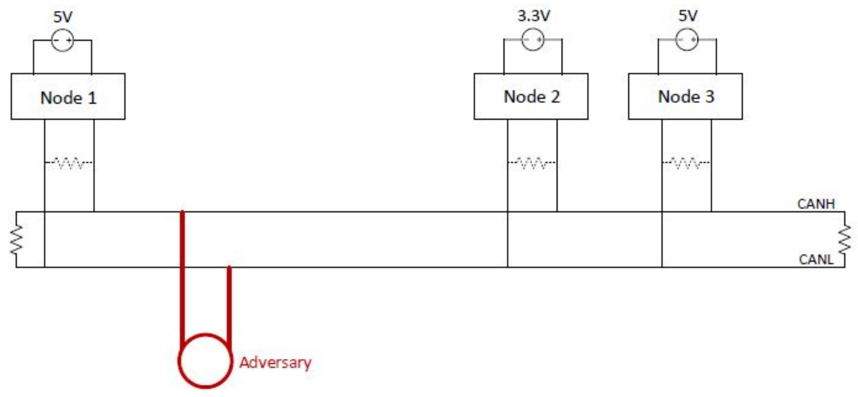

Figure 3.1: Representation of a typical CAN bus with 3 nodes and an eavesdropping adversary

The goal of this work is to demonstrate the existence of side-channels. Even though such powerful adversaries are capable of actively injecting or modifying signals, we consider the adversarial behavior to be restricted to passive observations. However, we assume that the adversary is capable of observing the system for an arbitrarily long time. During regular system operation, we assume that the adversary is capable triggering the ECU of its choice to transmit. However, we assume that such selective modifications cannot be made during execution of the PnS-CAN protocol. Further, we assume that the adversary cannot control the content of the transmissions.

Here we limit the adversarial access to a single point on the network. It should be noted that multiple points of observation can undoubtedly increase the leakage.

However, since in an unmodified system, a single adversary is sufficient for complete compromise, we leave the consideration of multiple probes to future analysis. For our attacks, we assume the location of the adversary to be carefully selected and fixed. An adversary can theoretically improve its observations by adaptive changing its observation point.

3.3 Attacks on PnS-CAN

To illustrate the properties that can be exploited to mount a successful attack, consider the **PnS-TwoParty** system. As described before, a secret bit will be generated when one of the nodes transmits the dominant bit 0, i.e. drives the bus, and the other node transmits the recessive bit 1, i.e. performs no action. Even though, in the PnS-CAN system, nodes transmit messages as full frames, for each bit of significance (secret bit), only a single node is driving the bus. Thus identification of the transmitting node effectively leaks the bit, as the bit is 0 if the driving node is primary participant and 1 otherwise.

For example, consider a PnS interaction between nodeN_1 and nodeN_2 using the random sequences $\{0,1,1,0\}$ and $\{1,0,0,1\}$ respectively. This results in 4 shared secret bits, i.e. key = 0110. The bit observations on the bus corresponding to these would comprise of 8 bits (random bits interleaved with the complement). $(b_1, b_2); \cdots ; (b_7, b_8)$, $b_i = 0$. Here, b_1 results from nodeN_1 transmitting a dominant bit and nodeN_2 a recessive bit. Thus if the adversary can identify nodeN_1 as the active node during b_1, it can learn that the first secret bit is 0. Next, we describe the phenomenon that can be used to differentiate between various transmitters based on physical properties of the CAN bus.

3.3.1 Experiment Set-up

Our experimental setup utilizes an FPGA (Altera Cyclone-V) based implementation of **PnS-TwoParty** scheme, implemented into a modified CAN controller. The modified controller generates fully compliant CAN 2.0 frames that can be accepted by any traditional CAN controller. The test network consists of 16 nodes, using custom designed transceiver boards that utilize the FPGA digital outputs, connected via the standard CAN twisted pair cable.

The typical CAN architecture, consists of several sub-networks of ECUs connected by one or more powerful nodes that act as gateway nodes. However, for our system, the nodes are connected in a single chain with the GW at one end. This setup closely emulates one subnet of the network found in most modern cars. We use a commercial USB-CAN module to monitor the bus traffic. We use an Agilent 6012A oscilloscope to probe the bus and record the samples for offline processing.

3.3.2 Generalized Attack

Each PnS interaction occurs between two nodes in the network. As discussed earlier, the attack on the PnS-CAN system simply reduces to identification of the transmitting node for each individual bit of the frame. Since there are just two nodes in each PnS-CAN round, the adversary simply needs to distinguish between the signals transmitted from the two nodes (binary hypothesis testing). Here, we present a general outline of the attack methodology that the an attacker may follow.

(Data Collection) The adversary can observe the signal transmission on the bus for a long duration prior to the attack. Note that the any CAN-compatible node can synchronize to the transmitter and identify and sample individual bits in a transmitted frame. Consider that the adversary samples each bit k times. We denote these bit observations by $\mathcal{T} = x_{-\infty}, \cdots, x_0$, where $x_i \in \mathcal{X}_k = \mathbb{R}^k$ is a vector of samples of the bit from the bus. We assume that the system has M ECUs connected to the bus. Thus the observations x_i correspond to transmissions from any of the M ECUs over a period of time. For traditional CAN networks, such data is not labeled as the packets do not contain identifying information. However, an adversary can generate labels for the data by activating known ECUs and observing their output.

(Classifier Training) An adversary trains classification functions, $\mathcal{D}_{N_i,N_j}$:$\mathcal{X}_k^n \rightarrow \{0, \bar{0}\}^n$, for each pair of labels N_i, N_j, that estimate the the sequence of transmitters based on the input observations. We note that using $\{0, \bar{0}\}$ represents the process of differentiating between the transmitters, rather than identification. This represents a larger class of classifiers, as any classifier that can correctly assign the labels, can also differentiate between the transmitters. The adversary can train the classification functions directly from $\mathcal{T}$, using a variety of supervised or unsupervised techniques, e.g. binary support vector machines (SVM), Long short-term memory (LSTM), Convolutional Neural Networks [80].

For scenarios where the node does not have knowledge cannot acquire knowledge of the participating nodes or where the training data is unlabeled, the node may train a generalized decision function $\mathcal{D}_{0,\bar{0}}$:$\mathcal{X}_k^n \rightarrow \{0, \bar{0}\}$, that uses a classification function preceded by a selection approach (such as maximum likelihood based estimator) select the best function. Clearly, the performance of a generalized classifier is sub optimal compared to the classifier for a particular pair of nodes. Thus an adversary, without priory knowledge of the transmitters has a significant disadvantage.

(PnS-CAN compromise) During PnS-CAN operation, for an adversary to extract the key, it does not require perfect identification of the node identity. Since the PnS system involves just two ECUs at any time, its task is to simply distinguish between the transmissions from the two participating ECUs. Consider the scenario where two nodes N_i, N_j execute the PnS protocol to produce n bits. The adversary observes the k sample values for each of the n bits as $x_i = \{s_1(i), s_2(i), ..., s_k(i)\}, 1 \leq i \leq n$ and uses the classifier $\mathcal{D}_{N_i,N_j}$ (or general classifier in scenarios where N_i, N_j is unknown) and obtains a sequence of estimates $(\hat{T}_1, \hat{T}_2, \cdots \hat{T}_n) \in \{0, \bar{0}\}^n$.

For PnS-CAN, classification without correct labeling of the nodes still reveals the complete secret (or its inverse). Thus the disadvantage of the weaker adversary is simply 1 bit of entropy, which is insignificant and results in violation of system security. We note that methods from CAN identification literature, e.g. from [67, 68, 81, 82], can be directly mapped to this generalized outline to attack the PnS-CAN scheme. However, the accuracy may be lower since the identification is for a single bit rather than a group of bits (frame).

3.3.3 Attack Procedure

The detailed attack is be described as *Attack PnS-TwoParty*. We utilized this to identify the secret key for 12 pairs of nodes in our setup. With minor modification of the threshold parameters between different iterations, we were able to successfully identify all the secret bits exchanged between each pair.

Attack: PnS-TwoParty

1. The adversary synchronizes to the first 1 to 0 transition, i.e. start of frame bit (SOF). The transmitting node is the referred to as the primary node.

2. The adversary utilizes the first 19 bits of the header to estimate the expected variation parameters of transition times (μ_p, σ_p).

Start of the PnS data frame

3. If the bit value has changed, compute the transition time. Compute the bit triggering the transition by comparing the transition time to a threshold τ, where τ is a function of (μ_p, σ_p).

4. If bit value has not changed,

   (a) If this is corresponding to the first bit (non-inverted), then the possible transitions are $0-0 \rightarrow 0-1$, $0-1 \rightarrow 1-0$, $0-0 \rightarrow 1-0$, and vice versa. If the current voltage level is higher than the previous bit ((b) in Figure 3.2), both nodes transmitted a dominant value for the current bit.

Otherwise, if the level decreases, nodes transitioned to a $0 - 1$ or $1 - 0$ configuration. Utilize the next bit to compute the current value.

(b) If this is corresponding to the second bit, it could have only resulted from a $0 - 1 \rightarrow 1 - 0$ transition or vice versa. If a dip is detected at the start of the frame ((a) in Figure 3.2), the primary node was transmitting a dominant bit in the previous frame. Otherwise, if an increase is detected ((c) in Figure 3.2), the secondary node was transmitting the 0.

Soft synchronization

5. If the secondary node ever triggers a recessive to dominant transition, resynchronize to the secondary node and switch the roles of the primary and secondary nodes. This is depicted by (r) in Figure 3.2.

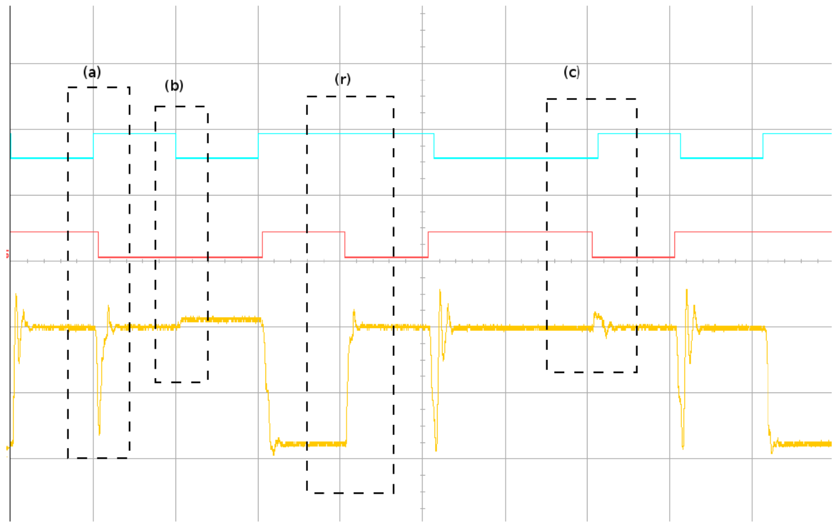

Figure 3.2: Snapshot of bus observations for PnS-CAN protocol

3.3.4 Observations on Physical Layer of CAN Bus

Similar to other electrical systems, in an automotive network will have differences in characteristics of the driver and network between an observer and the transmitters. We illustrate 3 phenomena that can be utilized to differentiate between bits transmitted by different nodes. One of the advantages of using differential signaling for CAN transmissions, is that it enables devices with varying electrical characteristics to be utilized on the same bus without any additional compensation circuitry. While this improves the design and robustness of the bus, the different characteristics also enables identification of the transmitter, leading to leakage of the bits. For a transmitted bit, differences in steady state characteristics can be attributed to several factors.

3.3.4.1 Difference in Driver Circuits

Transceivers from different manufacturers (or even different models of the same manufacturer) can have different drive characteristics and output voltage range of the CANH and CANL pins. This can be due to different circuits, components or load impedance. Thus an adversary measuring the absolute voltage on CANH and CANL lines with respect to a common ground reference can distinguish between the dominant transmissions from different nodes.

For example, consider a realization of the CAN network using Microchip (MCP2551) for node_{N_1} and NXP (TJA1040) for node_{N_2}. The ideal specified range for the CANH pin for MCP2551 is between 2.75V and 4.5V , while the same range

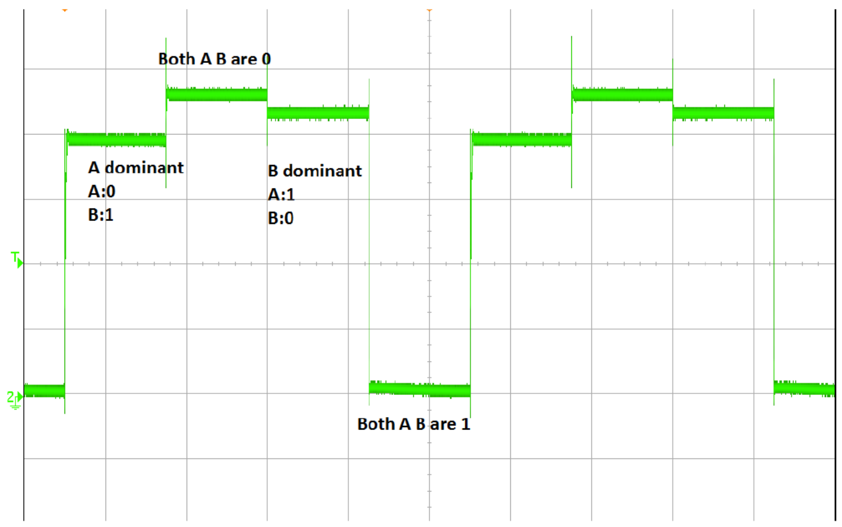

Figure 3.3: Observations for scenario of 2 nodes, A:MCP2551, B:TJA1040

for TJA1040 is between 3V and 4.25V . In Figure 3.3, we illustrate the voltage observations of the adversary for a sequence of bits corresponding to the generation of a secret bit in the PnS-CAN protocol, i.e. transition from a 0-1 scenario to a 1-0 scenario. The adversary can clearly distinguish between the dominant transmission by nodeN_1 and nodeN_2.

3.3.4.2 Different Operating Voltages

The differential signaling of the CAN bus allows interoperability of nodes with different supply voltage, without scaling circuits. For example, typical automotive networks contain ECUs with both 5V and 3.3V operating voltage. Though these nodes have similar differential voltage between the CANH and CANL lines, the absolute voltage level on each line, during bit transmission, is different. An adversary can utilize this identify the node transmitting the dominant bit during PnS-CAN.

3.3.4.3 Different Physical Locations

Even nodes with identical drivers and operating voltages can seem different from the view of an intermediate observer in the network. This is due to the differences in the effective impedance of the network segment between the two transceivers and the observer point. For a typical CAN bus scenario, several factors can contribute to these differences, e.g. different length of wires between the nodes and the observer, or different number of intermediate nodes. In Figure 3.4, we illustrate the differential voltage of two nodes at varying distances (over 1m difference) from the observer. In a typical CAN network, the difference in distance can be over 30m. Though this difference appears small in comparison to other phenomenon, it can be useful in many scenarios.

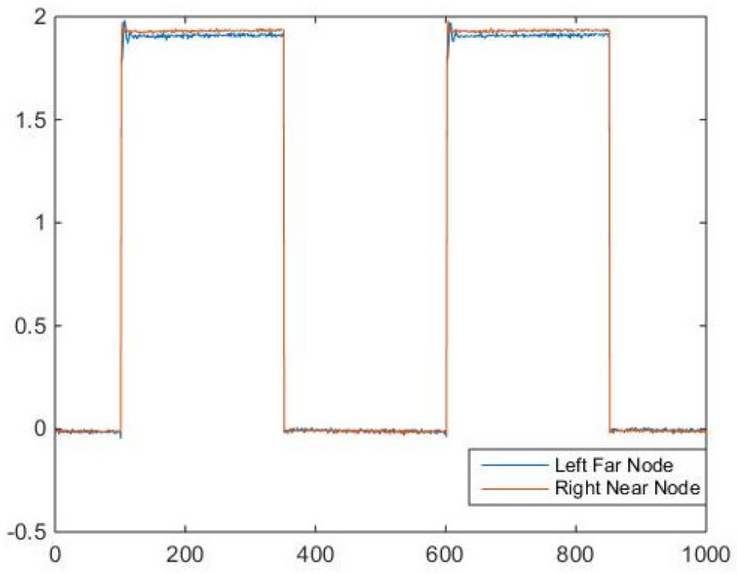

Figure 3.4: Observations of two identical transmitters located at different distances from the adversary

3.3.5 Evaluation and Experimental Results

We demonstrate an attack on the PnS-CAN scheme using the characteristics described before. We utilize the oscilloscope to obtain samples of the differential bus voltage at $125 Msamp/s$. The transitions are identified by as points of large change in bus voltage (greater than the CAN trigger) followed by a steady state over at least half the bit width. We utilize the 50% point of the transition to compute the rise time, fall time and latency.

Node	Delay (ns)				Node	Delay (ns)			
ID	Min	Max	Mean	Std	ID	Min	Max	Mean	Std
1	138	166	151.8	12.6	9	118	154	135.2	15.7
2	140	168	153.4	12.5	10	118	154	135.0	15.3
3	140	168	153.8	12.6	11	122	156	139.1	14.7
4	140	172	156.2	12.9	12	124	158	140.9	14.6
5	130	162	144.4	14.2	13	116	146	130.6	12.6
6	130	160	144.8	14.0	14	118	146	131.2	12.3
7	132	164	147.1	13.7	15	118	152	135.4	14.8
8	136	164	148.7	12.4	16	122	154	137.2	13.3

Table 3.1: Signal delays between nodes and observer

First, we investigate the separability of the nodes by measuring the propagation delays of the nodes synced with respect to the observation point. In Table 3.1, we enumerate the propagation delays from each node. Intuitively, nodes that have similar propagation delay would be difficult to differentiate (in the perfectly synchronized scenario). Further, in Table 3.2, we enumerate combinations of nodes have the least (and maximum) overlap. Such nodes pairs correspond to the nodes with the largest (and smallest) adversarial advantage.

In Figure 3.2, we illustrate a snapshot of the CAN protocol between two nodes as observed by an adversary. Intuitively, the attack exploits the difference in propa-

Interval overlap	N_1	N_2
6.00	2	13
6.00	2	14
6.00	3	13
32.00	11	12
34.00	9	15
34.00	10	15
36.00	9	10

Table 3.2: Maximum and minimum overlaps of propagation delay

gation delays by synchronizing to one of the transceivers and estimating the transmitter based on the rise and fall time offset. There are two key features of the PnS-CAN implementation that aid our analysis.

1. The initiating node transmits the PnS header and the secondary node synchronizes to the initiating node. Since only a single node is transmitting during the header phase, it is easy to estimate the timing variation for the synchronous bits.

2. As described before, the random bits are interleaved with the inverted bits. This introduces dependence in successive transmissions as it enables only certain transitions during the PnS phase. This can be utilized to estimate the bits in some cases. The detailed attack is be described as **Attack PnS-TwoParty**. We utilized this to identify the secret key for 12 pairs of nodes in our setup. With minor modification of the threshold parameters between different iterations, we were able to successfully identify all the secret bits exchanged between each pair.

3.4 Countermeasures for PnS-CAN

As discussed, the PnS-CAN module is susceptible to a range of characterization attacks due to consistency of the features of the physical signals from individual nodes. While this is good for robustness of the system, the CAN standard allows for significant variation of these properties. Thus, intuitively, we design the countermeasures to add controlled noise to the physical signals, such that signals from different nodes are indistinguishable for an adversary. The goal of the noise addition is to minimize the adversarial advantage between nodes.

Here, we propose mitigation at different levels of abstractions, namely the transceiver hardware level, CAN controller level and the system level. At the hardware and controller level, the goal is to minimize the adversarial advantage between all ECUs either by adding noise or improving cooperation between the nodes. At the system level, the goal is to minimize adversarial advantage by restricting interaction between highly identifiable nodes. It should be noted that the performance of the countermeasures proposed here is highly dependent on the CAN properties. Thus, they are proposed with the goal of minimizing the leakage, rather than provably eliminating it. We leave as an open problem, the design of countermeasures that can formally shown to be effective against the adversaries treated here.

The voltage observed by an adversary is a function of strength of the current source and impedance of the network between the observer and the current source. For a non-uniform network, this directly relates to the physical positions of source and the observer. To prevent identification by the adversary, voltage can be varied for each transmission of the dominant bit by using a random number of current sources, located at different points in the network.

Our experimental results demonstrate that the observations at a single point due to dominant bits from multiple transceivers is different. Such a phenomenon can also be observed in box (c) of Figure 3.2. In Table 3.3, we enumerate the output at the observer due to simultaneous transmissions from different transceivers connected to the bus. For increased variations, we performed experiments using 3 different transceiver families, i.e. nodeN$_1$ (MCP2551), nodeN$_2$ (TJA1040) and nodeN$_3$ (TJA1041).

nodeN$_1$	nodeN$_2$	nodeN$_3$	V_{out}
0	0	0	2.4230
0	0	1	2.1281
0	1	0	2.1197
0	1	1	1.8208
1	0	0	2.3400
1	0	1	1.7710
1	1	0	1.7629
1	1	1	0.0000

Table 3.3: V_{out} for dominant t_x by multiple nodes

Similar to variation in sources, adding a random number of sink nodes may vary the observation by an adversary. In the regular setting, each node on the bus that is not transmitting acts as a sink. Thus, we can vary the number of sinks on the bus by adding the capability of electrical isolation of non-participating nodes. Thus any transceiver with such a capability can be in 3 distinct modes, i.e. transmitting a dominant signal, transmitting a recessive bit (or just passive), or electrically isolated. In Table 3.4, we outline the voltage observations of the adversary for different states of the tristate nodes. The 'X' denotes a transceiver that is electrically isolated. Ideally, with N transceivers, we can obtain $3^N - 2^N$ different voltage levels. The variation of voltage levels with a bit can decrease the probability of its identification. We outline a few architectures to utilize this feature to prevent adversarial information leakage.

nodeN$_1$	nodeN$_2$	nodeN$_3$	V_{out}
X	0	0	2.5842
X	0	1	2.1174
X	1	0	2.0923
0	0	X	2.3159
0	1	X	1.9647
1	0	X	2.1493
0	X	0	2.2957
0	X	1	1.9599
1	X	0	2.1415

Table 3.4: V_{out} for dominant t_x by multiple nodes with isolation

3.5 Summary

While the PnS-CAN scheme is a promising mechanism for generating group keys, it is highly susceptible to probing attacks by even simple adversaries. We presented a general methodology for attacking the system and demonstrated a simple timing based attack. We discussed several sources of physical identifiers that can be used by an adversary. We proposed several mechanisms to mask these identifiers at the hardware, CAN controller and system level. This work serves as a proof-of-concept for the existence of attacks on the PnS-CAN schemes and the the ability of the system designer to prevent them.

Chapter 4: GPS Spoofing Detection Methods on the Edge

4.1 Introduction

Nowadays, most of the modern automobiles are equipped with GPS module to assist location relative functions such as smart navigation, emergency assistance, and traffic information service. With the rapid developments of autonomous driving, the algorithms for self-driving rely on the GPS system as a critical fallback to ensure accurate timing and location. Moreover, as more and more vehicles becoming connected to join the VANETs in assistance of driving, GPS signals have also been widely applied in VANETs for precise positioning, message time stamping as well as other real-time network applications [56, 59]. However, from the security perspective, the civilian GPS signals are very vulnerable to outside attackers because they are not encrypted and can easily be spoofed [83]. The GPS spoofing attack could deceive a receiver by broadcasting counterfeit GPS signals that are stronger than the real GPS signals. The same kind of GPS spoofing attacks could also take place on vehicles and this kind of spoofing attacks will cause severe risks on the vehicles because the locations of vehicles are critical and supposed to be reliable and integral for the navigation systems. The situation would become more severe for this attack on the self-driving vehicles because the safety or security-critical decisions and processes of autonomous driving highly rely on the sensor data including the position data from the GPS.

Once the GPS signal gets spoofed, the central control system for navigation would be the first to be misled by the malicious attackers. The urgent task for the reliability of the navigation system is to detect the spoofing attacks as soon as possible after the attack took place. There have been several authentications and anti-spoofing techniques for this propose. One of the straightforward methods is to use multiple receivers, i.e, the antennas to cross-check the signal [84, 85]. When adding multiple receivers to check the incoming signals, the slight changes introduced by the spoofed source could be captured by the out of receivers. However, sophisticated attackers could replicate the spoofed signal with phase-aligned to two and more GPS receivers. These sophisticated spoofing attacks are hard to be detected by multi-receivers because it synthesizes spoofing signals for multiple satellites in a way that initially overlays them on top of the true signals. The other methods to detect GPS spoofing leverage on the analysis of the signals as a time of arrival or received strength [86]. While these detection methods would require equipment to capture the characteristics of the signals and may not be flexible on a vehicular system.

Since data is increasingly produced and generated at the edge of the network, it becomes natural to consider processing the data at the edge. Under this circumstance, edge computing is proposed to be an efficient method to support lots of applications. With the push from the edge computing, our proposed GPS validation could be applied at the edge of the network, i.e, the vehicle. Besides, edge computing is also believed to have the potential to address the concerns of data safety and privacy.

Driven by the increasing threat and the lack of realistic short-term solutions, we develop a low-cost validation mechanism for detecting GPS spoofing attacks on vehicles based on the driving information acquired through the in-vehicle CAN bus. Our proposed method relies on the key insight that the data from the inner vehicle network are trusted and demonstrated by the in-vehicle authentication methods [16, 21, 23, 69]. Unlike previous attempts to secure GPS, our proposed method neither requires any updates of the GPS infrastructure nor of the vehicle's receivers. In contrast, our proposed mechanism would detect the spoofing attacks by reconstructing the GPS position from continuously analyzes the information recorded on the CAN bus, such as the vehicle speed and the steering angle. Moreover, we proposed this cross-check GPS spoofing detection method as a paradigm of edge computing, since all the computations are processed at the edge of the network.

This chapter makes the following contributions:

1. **A low-cost method on the edge**

   Compared to several other approaches proposed to detect the spoofing attacks such as the cross-correlation of encrypted signals [84], calibration with multiple antennas [87] and the methods rely on inertial high-stability clocks [86], our cross-validation method has significant advantages on the cost. First, our method does not require any extra devices such as the additional antenna or receiver which might be too heavy to carry on and not practical in vehicles.

Second, our method focuses on the plain GPS signals and does not need any encryptions and decryptions on the GPS signals. Therefore, our method could avoid aligning with military signals for civilian vehicles which may arouse permission issues on the commercial vehicles. That is because the United States GPS also includes the military signals that have encrypted on the spreading codes and the legacy code. And those codes can be predicted only with a secret encryption key. Spreading the secret key to civilian vehicles is not applicable. At last, to solve the bottleneck as data transportation through the VANETs, our detection method would take place at the edge side (i.e., the vehicle) and take use of the already deployed micro-controllers in the vehicle. Those controllers have comparable computing power to process the data from numerous sensors and fulfill the GPS spoofing detection.

2. **Trust and Privacy are validated by the in-vehicle network**

   Some methods for positioning rely on the collaboration of the neighbor vehicles in the network, thus arouse the problem of privacy risk for the locations [88]. First, the assisting car might not want to share position to the lost car due to the privacy consideration. Moreover, the assisting car might misbehave to fake false signals to cheat the lost car. Those aroused problems might be solving by the proposed privacy-preserving location, however, need much more efforts to set-up the communications, which will inevitably add influential delays to the system [89]. However, our proposed method uses the local on-board signal which does not need any extra communications from the outside of vehicles. Moreover, the signals collected on board are proved by the in-vehicle authentication communications which are believed to be trusted without any interference by the third-parties [90]. Besides, since the data are collected and processed on the edge side, the privacy of the vehicle and the driving information could be guaranteed as the interactions to the cloud vulnerabilities are cut-off.

3. **Applicable on autonomous vehicles**

   As cars become increasingly autonomous, and self-driving vehicles move from the test markets to the main streets, security will become the primary challenge for automakers. The more connected a vehicle is, the more susceptible it becomes to potentially deadly cyber attacks. Our proposed cross-validate method could be applied to autonomous vehicles with trivial efforts. Besides, autonomous cars take use of a variety of sensors to perceive their surroundings, including radar, laser light, GPS and computer vision modules, which could provide our system with rich sensory data as free of charge.

4.2 Preliminaries

4.2.1 GPS Signal and Spoofing

GPS is the world's premier Global Navigation Satellite System (GNSS), consisting of 31 satellites launched by the U.S. military and made available for civilian and commercial use as well. GPS-capable receivers can determine their position and time by picking up time-stamped signal data from a minimum of four overhead satellites. By using the time stamps to calculate the time of arrival, a receiver can calculate a triangulated position. The time of arrival measurements are affected by a range of errors resulting in typical localization uncertainty of $\sigma = 4$m (mean error is about 7m) [91]. Ordinarily, GPS is believed to be precise and reliable, so that lots of geographic positioning systems including the navigation system for vehicles are dependent on GPS signals. However, problems and attacks with this system have become increasingly evident. The reason is that civilian GPS signal is unencrypted, and it has no proof-of-origin or authentication features, so the system remains extremely susceptible to fraud, spoofing, jamming, and cyberattack. As a result, civilian (public) GPS signals can be decoded by everyone, including airplanes, vehicles, ships and the other systems required location services. Attacks on the GPS signal are usually performed by an adversary by either jamming or spoofing one or both radio channels. For the jamming attack, an adversary transmits overwhelming radio interference over the band. For spoofing attacks, the adversary mimics the legitimate signals by intentionally alter data received by a victim receiver with the purpose of modifying the localization or time result of the victim. Since civilian GPS signals are not encrypted and the structure of the signal is well known, spoofing attacks are relatively straightforward to execute using a commercial signal generator and RF transmitter. Over the years, the price to perform GPS spoofing attacks has dramatically dropped. Nowadays, mobile commercial GPS spoofing devices are available for less than \$1000 with publicly software tools installed which enables any parties to generate faked GPS signals.

To tackle down those spoofing attacks on GPS, several authentication and anti-spoofing techniques for GNSS signals have been developed in recent years. These techniques can be broadly categorized as signal and data-level authentication. For signal authentication, received signal characteristics of the civilian signal may be verified against encrypted GPS transmissions. Additionally, the plane-of-polarization and angle-of-arrival can be measured to validate the signal as well. Paper [92] proposed to justify the directional characteristics and polarization of the received signal. The strength of a received GPS signal is typically less than -150 dBW and presence of substantially stronger signals for a single satellite or over the entire frequency band can be a sign of attack in progress. However, it is possible for a spoofing attacker to adjust power levels to evade detection limiting the usefulness of the threshold detection mechanism in practice [92]. Similarity, using directional characteristics of the receiver antenna to cross-validate received signals from each satellite requires specialized phase tracking hardware to detect directional variations. Similarly, signal polarization characteristics have been shown to be an effective authentication aid [93]. However, dedicated receiver front ends and signal processing is required to implement the approach efficiently.

Data-level spoofing detection uses demodulated GPS data to detect spoofing. GPS data can be validated with known position data or otherwise obtained time to detect attacks [94]. For example, a stationary receiver can check its known position with the position-solution of the receiver. Since trilateration position error can be 10 meters, the position solution is a weak measure of credibility. It has been recently demonstrated by Jiang et al. [95] that attackers can use this uncertainty to induce a phase angle error of 52 degrees in a Phasor Measurement Unit (PMU) receiver by using simple optimization based evasion algorithms. Data-level spoofing detection methods can also be applied with assistance by cryptographic techniques [96]. Those methods propose to authenticate signals from satellites with additional signals encrypted by the secret key. However, these techniques are not resistant to replay attacks and would require a costly upgrade of the GPS infrastructure.

Monitoring jumps in the time reported by the GPS signals is another possible GPS spoofing countermeasure. One can deploy accurate clocks to measure time deviation between the reported GPS clock and the onboard clock. However, precise clocks (such as atomic clocks etc.) are expensive and not used in practice for commercial purposes. For IoT components, one can also compare the GPS time with networked time protocols such as Network Time Protocol (NTP). However, the approach can suffer when the network is down. Moreover, the resolution of attack detection is limited by the accuracy of the networked clocks.

4.2.2 GPS Usage in Vehicles

Nowadays, each vehicle of the VANETs is required to be equipped with a positioning system (receiver) to achieve the applications in communication and data sharing [1]. GPS is typically used by the vehicle or the driver for self-localization and navigation but the technology is also used for remote traffic surveillance and collision avoidance on the VANETs. In the latter use-cases, vehicles are required to periodically broadcast position and velocity to inform the neighboring vehicles and the road assistant controllers through the VANETs. VANETs can achieve effective communication between moving node by using different ad-hoc networking short-range radio technologies such as Wifi IEEE 802.11 b/g, WiMAX IEEE 802.10, Bluetooth, IRA. Irrespective of any transmission interfaces used, these transmitted messages contain a position that is directly derived from the on-vehicle GPS receivers. Although the GPS receiver has widely been used for conventional automotive applications, it does not guarantee reliability and continuity of position data.

The major goal of spoofing attacks on an automobile GPS system is to produce erroneous position signals to fool the navigation systems. The vehicles under attack may result in deviate to wrong places or loss of GPS signals. Since the GPS signals used by vehicles are not encrypted, an attacker can easily spoof the RF antenna input of a GPS receiver using a signal generator and RF transmitter. Consequently, the on-road vehicles are vulnerable to GPS spoofing attacks, where an attacker transmits fake signals that imitate signals from satellites but higher power and at different time delays. The false signal is designed to cause the GPS receiver to report an incorrect position solution. As vehicles become more autonomous, robust GPS positioning will be an invaluable resource if threats like spoofing can be successfully abated. Since many of the functions and applications in VANETs are dependent on the civilian GPS signals to locate and synchronize in the network, successful GPS spoofing attack can facilitate other cyber attacks as well (i.e., Sybil attack on VANETs).

Some methods of spoofing detection on vehicles have been proposed which concentrate on the raw signal observations like signal power and clock offsets in identifying an attack. Since receiver spoofing detection methods to be able to detect some attacks, but most sophisticated spoofing techniques are extremely difficult to be detected without external information. An inertial measurement may assist in calibrating the GPS signals using filters benefit [87]. Even though these detection approaches do not require changes to the GPS infrastructure, they still have a crucial drawback. They assume more specialized GPS receivers increasing, e.g., the complexity and power requirements. The major problem is that it is urgent to find a low-cost GPS spoofing detection method without disclosing the privacy of the vehicle. In paper [89], the author proposed a spoofing detection and mitigation system relies on an existing cooperative adaptive cruise control system to provide inter-vehicle ranging and data sharing. However, this work takes use of the cruise control system and requires multiple vehicles to collaborate on the detection.

Instead of threats from spoofing attacks, lost of GPS signal can also cause malfunctions of the navigation system. GPS receiver depends on weak satellite signals from about 22000 km away in space, which is very easy to interfere with the other signals. Besides, The GPS requires at least four beacon signals to be overhead and the urban density and skyscrapers also cause difficulties in receiving four messages and the issue of multi-path signals occurs within the vicinity of high rise buildings. Further, the mobility of the vehicles makes it even worse to receive precise and continuous GPS signals. As a result, some vehicles may occur temporary loss of GPS signals due to the cover of the terrains.

4.2.3 Edge Computing

The development of IoT and the success of cloud services have pushed the horizon of a new computing paradigm, edge computing, which calls for processing the data at the edge of the network [97]. Now with the arrival in the edge computing era, where there will be a large quality of data generated by the end devices, and a lot of applications will also be deployed at the edge to consume these data [98]. Some applications would require very short response time, some may involve private data, and some might produce a large quantity of data which could be a heavy load for network communications. Under those circumstances, cloud computing is not always efficient enough to support these applications. As a result, edge computing is believed to have the potential to meet the requirements as the response time requirement, battery life constraint, bandwidth cost saving, as well as data safety and privacy [99].

In summary, there are mainly two reasons for the growing demand on the edge computing. First and foremost, since data is increasingly produced at the edge of the network, it would be more efficient to process and filter the data at the edge where also allocated enough computing power nowadays. The rationale of edge computing is that computing should happen at the proximity of data sources. Here we define "edge" as any computing and network resources along the path between data sources and cloud data centers. In our application, a smart vehicle is an edge between the vehicle sensors and the network cloud (i.e., VANETs). Modern vehicles are usually equipped with 50-100 micro-controllers to process the maneuvers of vehicles. One of the main function of those controllers is to process the data generated or collected by the sensors. Those controllers have the comparable computing power to process the data at the edge which generate the “decision” from the numerous sensor data. That is why the edge computing paradigm worked on modern vehicles.

Secondly, with the growing quantity of data generated at the edge, speed of data transportation is becoming the critical bottleneck for the cloud-based computing paradigm. For example, about 5 Gigabyte data will be generated by a Boeing 787 every second, but the bandwidth between the airplane and either satellite or base station on the ground is not large enough for data transmission. Consider an autonomous vehicle as another example. One Gigabyte data will be generated by the car every second and it requires real-time processing for the vehicle to make correct decisions. If all the data needs to be sent to the cloud for processing, the response time would be too long [72]. Not to mention that current network bandwidth and reliability would be challenged for its capability of supporting a large number of vehicles in one area. In this case, the data needs to be processed at the edge for shorter response time, more efficient processing and smaller network pressure. As an example, cameras in an autonomous vehicle capture a huge amount of video data, which the system must process in real time to yield good driving decisions. If the vehicle must send the data to the cloud for processing, the response time would be too long. And a large number of autonomous vehicles in one area would further strain network bandwidth and reliability. Processing data at the network edge would yield shorter response times, more efficient processing, and less pressure on the network.

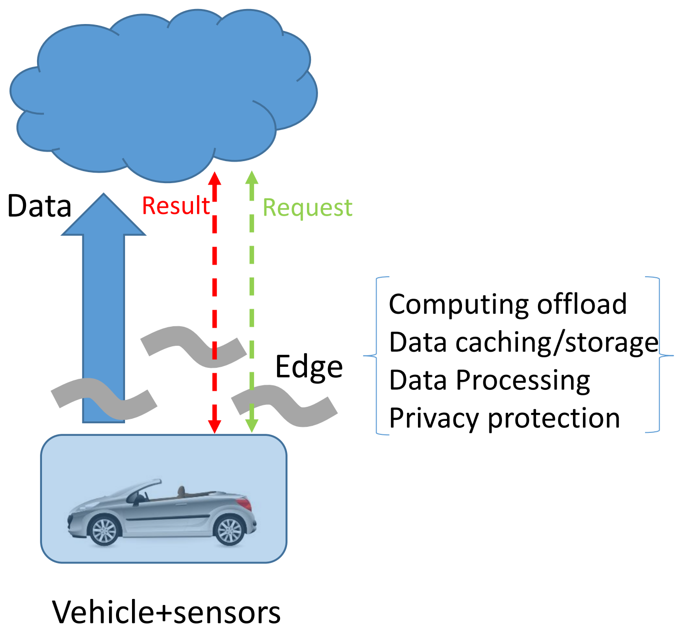

Figure 4.1: Edge computing paradigm for connected vehicles

In the cloud computing paradigm, the end devices at the edge usually play as a data consumer. As the example shown in Figure 4.1, the end vehicle usually receive the GPS signal from the cloud which is either the satellite network or the central VANETs facilities [100]. Nowadays, smart vehicles deployed with multiple sensors can also produce data. The change from a data consumer to data producer/consumer requires more function placement at the edge. For example, it is very normal for the vehicle to report the location or share traffic data through a cloud service every single minute. However, the volume of data could be fairly large and it would occupy a lot of bandwidth for uploading. In this case, there is a possibility that the data could be processed or calculated adjusted to suitable resolution at the edge before uploading to the cloud network. Another benefit from edge computing is privacy preserving. Since the physical data collected by the things at the edge of the network is usually private, processing the data at the edge could protect user privacy better than uploading raw data to cloud [101].

4.3 Threat Model in GPS Spoofing

In the past, incidents were reported where spoofs successfully interfered with the integrity of the GPS system on vehicles. Based on common assumptions on the attacker's capabilities, we assess the threat model in this section. First, we clarify our considered adversary model. Second, we reason about key assumptions that our proposed GPS spoofing detectors.

4.3.1 Threat Model

The motivation of the attacker is to interfere the vehicle safety by injecting false positioning information into the GPS system. An attacker may first want to stop the navigation system or the other position based function of the vehicle to hinder the vehicle's system. He can achieve to divert the vehicle to somewhere else without notice by the vehicle. Moreover, the attacker can destroy VANETs communication by manipulating the GPS signal of the victim vehicle. For those VANETs protocols which use the position as authentication proofs, the victim vehicle will result in failure to prove the location, thus cannot get any service or partition into the VANETs.

In our adversary model, the attacker is able to use specially crafted signals with a higher power to the victim receiver. The attacker aims at spoofing a moving vehicle from a position on the ground. Since the availability and cost of the GPS transponders, we decide to use the simulation for GPS spoofing validation in this chapter of dissertation. We assume the GPS spoof changed the receiver's signal by using higher power at the target location, thus resulting in changing the decoded GPS signal. Here, we assume the attacker's aim is to divert the vehicle to the other location, so he will continuously inject false GPS signals into the navigation system.

4.3.2 Validation of Assumptions

Our proposed method relies on two key assumptions which we validate in this section. First and foremost, the vehicle would know correct start GPS location when the vehicle's engine starts. We assume this would be achieved by other mechanism such as cryptographic location authentication but cannot be authenticated with high frequency. While this is considered as one-time per journey authentication, it would not engage too much cost in communication and calculation. The second assumption is that the spoofing attacker will only affect the GPS signal but has no capability in changing the other signals in the vehicle, i.e, the vehicle speed or the steering wheel angle. The confidential of those signals are guaranteed by the in-vehicle security mechanism. We validate these two assumptions with controlled lab experiments and simulations with real-world vehicle data.

4.4 Reconstruction of GPS Signals on the Edge

We propose a mechanism to cross-validate the GPS signals received by the automobiles with an edge computing process on the trusted data from the in-vehicle network. Our proposed mechanism would serve as an independent system infrastructure on the vehicle which continuously analyzes the contents generated through the vehicle behaviors. Furthermore, our proposed method could also be applied in predicting the location of the vehicle when the on-board GPS signal is unavailable. The data used in the GPS spoofing detection system are mainly from the vast majority of sensors installed and operated on the modern vehicles, especially the autonomous vehicles, are equipped with a variety of sensors for driving assistance. Rich information such as vehicle speeds, locations, and time stamps are commonly recorded by the central control system of the vehicle. It is almost no cost by using the existing rich data from the in-vehicle sensors. As a result, our proposed method is considered to be a lightweight method to detect GPS spoofing. We proposed to apply our two developed algorithms to realize the purpose on the vehicle, the computing edge, which would save the communication times and overload. Also, we use downloaded map information to further improve the accuracy on reconstruction GPS signals, which is also operated on the edge side.

As discussed previously, detecting GPS spoofing attacks with extra hardware extensions is of high-cost and not feasible for vehicles. It is obviously unnecessary to find a place to install extra antennas or receivers in a compact vehicle.

4.4.1 Method I: Non-holonomic Car-like Robot Demo (Kinematic Model)

As previously stated, the main aim of the attacker is to trick the GPS receiver into reporting an inaccurate position without identifying the failure. To defeat the attacker's purpose, the first proposed spoofing detection approach leverages the driving information data and the physical model of vehicle to reconstruct the location information. Specifically, the method takes use of the non-holonomic kinematic model to calculate the trajectory of the vehicle and reports the spoofing attacks if any inconsistencies found.

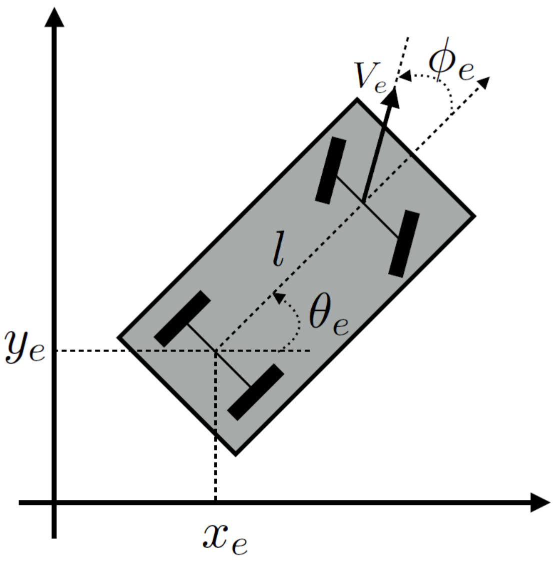

Figure 4.2: Generalized Coordinates of a car-like robot

The movements of the vehicle can be modeled by the kinematic equations of a car-like robot [102]. The kinematic model is derived from the physical feature of a mobile wheel with the presence of non-holonomic constraints due to the rolling without slipping condition between the wheels and the ground. The symbol of the kinematic model of vehicle is shown in Figure 4.2. For simplicity, we assume that two wheels on each axle collapse into a single wheel located at the mid-point of the axle. For a front-drive vehicle, we assume that the front wheel can be steered while the rear wheel orientation is fixed. The generalized coordinates are $q = (x, y, \theta, \phi)$, where x and y are Cartesian coordinates of the rear wheel, θ measure the orientation of the car body with respect to the x axis and ϕ is the steering angle. The derived front-wheel driving model is obtained as

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\phi \\ \sin\theta\cos\phi \\ \sin\phi/l \\ 0 \end{bmatrix} v_1 + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} v_2 \tag{4.1}$$

Where v_1 is the driving velocity refers to the front wheel and v_2 is the steering velocity input.

Note that the input v_2 in the dynamics is the angular velocity of the wheel, however, our dataset does not provide this measurement. But the dataset has recorded the steering wheel angle status with time stamps which could be used to calculate the steering velocity. Specifically, the steering wheel angle is recorded in degree and the range apart from the normal reference is from -900 degrees to 900 degrees. Based on the mechanical structure of the vehicle, the turning of the steering is proportional to the actual turning of the wheels. Based on vehicle manufacture books, the ratio is varying among different models and it usually falls in between 12:1 to 24:1. Thus, to calculate the dynamics of the vehicle with the steering velocity, we should derive the appropriate steering ratio of the test vehicle. For example, the steering ratio is r and the turning of the steering wheel is Φ, the actual change of the wheel would be $\phi = \Phi/r$. And the angle velocity discussed above can be modeled as $v_2 = \frac{d\phi}{dt}$. The other input of the dynamic functions $v1$ could be acquired directly from the dataset. Each state of vehicle is represented as $q = (x, y, \theta, \phi)$ and the coordinates of the position are calculated by the inputs and the transit state of the function. Based on all the transit states derived from the dynamics equation, the transitory of the vehicle is achieved.

In the kinematic model discussed above, we make approximations when simulating the dynamics of the model. For example, we treat the two wheels as a single axis in the middle but a practical model should also consider the rotation angles of each wheel as generalized coordinates. Furthermore, an accurate model should also account for the presence of actuators and sensors on the wheel axis as well as for typical non-ideal such as tire deformation. For example, at higher speeds or with sharp steering, the no-wheel-slip assumption breaks down due to the lateral force. Besides, the above system is assumed to be drift less, i.e, no motion takes place under zero input that there are fewer control inputs than generalized coordinates. But the real road test would be totally different and it is not trivial to establish a complicated mathematical model for the real road condition. Moreover, some parameters to model the dynamics of the vehicle is confidential of the manufacture and not disclosed to the public. As a result, we choose to use a simple one. However, we shall admit the existence of errors when using an approximate model.

4.4.2 Method II: Regression Model

As an alternative to mitigate the error generates by approximate modeling, we derive a regressive algorithm to describe the relationship of the steering wheel angle and heading of the vehicle. More specifically, we calculate the next coordinates based on the current location, current velocity, and the steering-wheel angle. For example, we calculate the next coordinate based on the length of the route (l) and the heading angle (θ) from the previous coordinate. The length of routes can be achieved easily by assuming uniform motion in a small period of time, i.e, $l = v \cdot t$, where v is the sampled velocity of this period and t stands for the time interval. Thus, we could acquire the changes on the coordinated from the following equations,

$$\begin{cases} \Delta x = l \cos \theta_h \\ \Delta y = l \sin \theta_h \end{cases} \tag{4.2}$$

with δ_x represents the changes from the X coordinate and δ_y denotes the changes on the Y coordinate. Since the distance l is calculated by the speed and time, the only unknown variable is the moving direction (heading) as the θ_h. Since then, the problem can be formulated as a regression problem to find the relationship of the heading θ_h based on the current and past steering wheel angles. Intuitively, we would think that the steering wheel controls the heading of the vehicle and it should follow an equation as $\theta_h = f(\theta_w)$. As a result, the key aim of this method is to use regression algorithm in finding the best fit function f and then applying the function to reconstruct the vehicle trajectory. Figure 4.3 shows the regression results on the heading versus the change of the steering wheel angle $\theta_{w(t_n)} - \theta_{w(t_{n-1})}$. The strict line indicates the linear relationship between the steering angle changes and the heading. The spots in the figure are the sampled data from the test drive. Using the derived function f to calculate the heading of the vehicle from the known steering wheel angles, we can construct the next coordinates when plugging into Equation 4.2. One evident advantage of Method II is that it avoids the complex states and parameters calculated in Method I. The exact model as to whether a front wheel is driven or rear wheel driven is not necessary to be defined with Method II in reconstructing. As a result, we could skip the approximated assumption in the mechanical model. Besides, the result from the second method should be more accurate than Method I as it is calculated on from real data which is not from the approximate formulation. One of limitation of Method II is that as a regression process it would require lots of data points for a training process to derive the exact function f, which might be restricted on the edge device. As a result, we propose the training phase (regression analysis) should be an off-line procedure, which is implemented on the other devices and the final output, function f, is reported to the vehicle later.

4.4.3 Combined with Map Information

Not limited to the data collected by the sensors in the vehicle, the public map data can also be used to locate the vehicle, which in terms validates the GPS

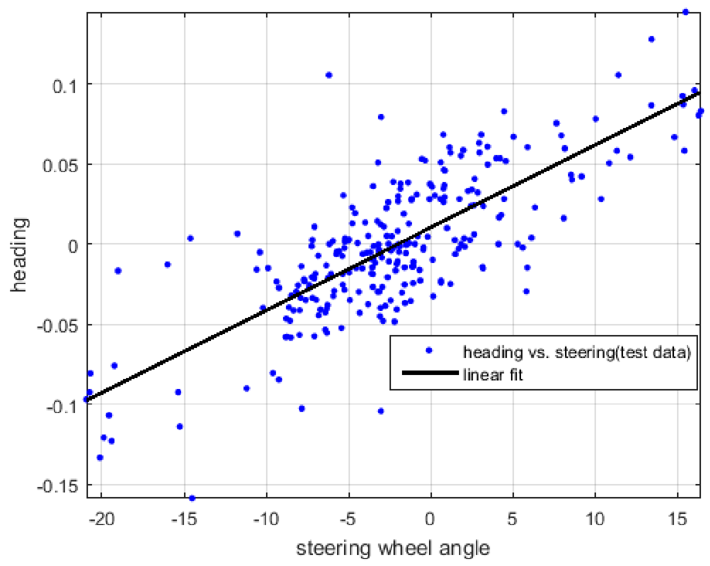

Figure 4.3: Regression results on the heading versus steering wheel angle

signal. In this scenario, we consider the GPS validation is processed on the edge part and the map information is considered as off-line data which would not add extra communications on the edge. The third method makes use of the information from the map to match the constructing coordinates acquired previously to a real location along the road. In particular, we perform a comparison from the calculated positions to the stored map positions. When the differences from those two positions are small, we will assume this vehicle is on those routes. By continuously monitoring several locations, we can infer the trip of the vehicle and compare the physical GPS locations of the route with the received GPS signal.

In this experiment, we collect the road information from the Open Street Map (OSM) which is a collaborative free editable map. The OSM contains the data from GPS receivers, which makes it proper for assisting the GPS validation scheme. Open Street Map uses a topological data structure, with four core elements listed as follows.

- **Nodes** are points with a geographic position, stored as coordinates (pairs of a latitude and a longitude). A set of nodes are used to represent a way, besides they are used to represent map features without a size, such as points of interest or mountain peaks.

- **Ways** are ordered lists of nodes, representing a polyline, or possibly a polygon if they form a closed loop. They are used both for representing linear features such as streets and rivers, and areas, like forests, parks, parking areas, and lakes.

- **Relations** are ordered lists of nodes, ways, and relations, where each member can optionally have a "role". Relations are used for representing the relationship between existing nodes and ways. Examples include turn restrictions on roads, routes that span several existing ways (for instance, a long-distance motorway), and areas with holes.

- **Tags** are key-value pairs. They are used to store metadata about the map objects (such as their type, their name, and their physical properties). Tags are not free-standing but are always attached to an object: to a node, a way or a relation.

After GPS signals are reconstructed by the previously proposed algorithms, we use map information to locate the route. This is the following step to confirm the GPS signal spoofing in order to take appropriate action. The procedure to extract

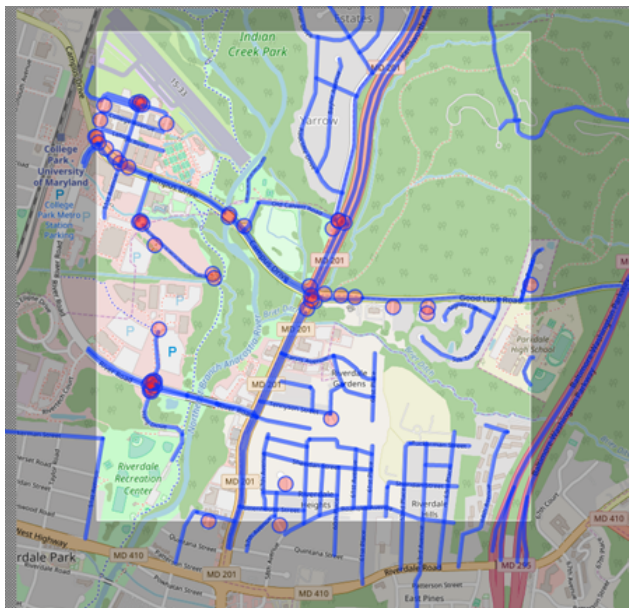
Figure 4.4: OSM of College Park metro area, the selected area is 5.56 km (horizontal) × 6.92 km (vertical)

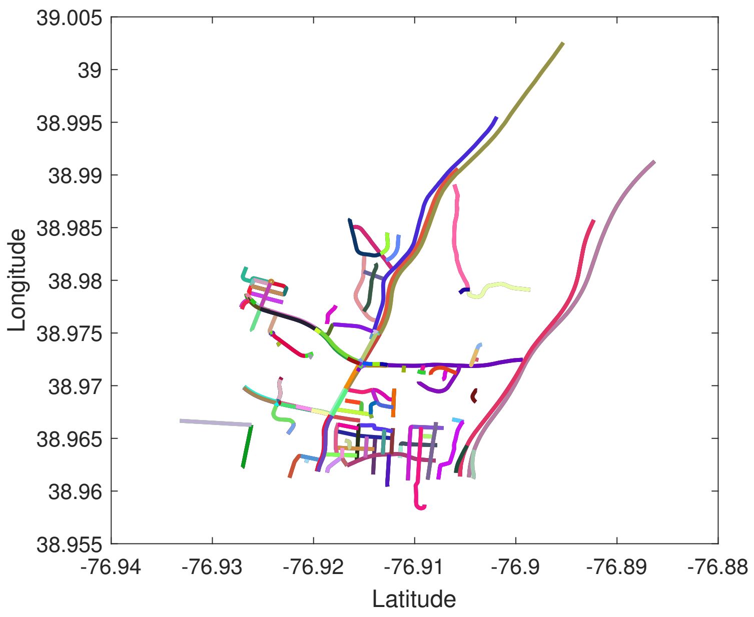

Figure 4.5: Extracted roads from OSM of College Park metro area, the total number road extracted is 252

the useful roads information and combine them to validate the GPS signal is listed as follows,

1. In the first step, the map including all the data (e.g, the roads as well as the other geographic features such as the parks, forests and lakes) in the nearest area is downloaded from the cloud, as shown in Figure 4.4. The example map shown represents an area of 5.56 km $\times$ 6.92 km.

2. Next, we extract the valid roads from the raw map data where the vehicles could present. The extracted lanes are drawn as in Figure 4.5 and the other useless regions or markers are removed from the map. After this filtering, the remaining ways only consist of sets of nodes stored as a pair of latitude and longitude $g(lat, long)$. We will use these sets of GPS positions from the map to validate the GPS constructed signals from the previous procedure. For the example, there are in total 252 valid road segments are extracted from the map area and drawn with different colors as shown in Figure 4.5. This step can also be implemented off-line or parallel with the procedure as constructing the GPS signals. Once we know a rough location of the vehicle from the range of GPS, we could extract the roads information in advance.

3. The following step is to locate the vehicle on the extracted routes, the specific calibration process is described in Algorithm 1. Based on the calculated GPS signal from the former methods, we calibrate the exact GPS positions along the route, which could lower the error rate and quickly detect any spoofing on the GPS signal. The specific details are described as, first, we acquire a list

of GPS signals as input $\mathbf{g}(g_0, g_1, \cdots, g_t)$. We iterate on this list, for each node g_i, we try to find the nearest routes along this node based on the threshold τ as $dist(g_i, r_j) \leq \tau$. We collect those routes r_j in the selected set as R. Keep updating R, we will find the final route R which consists of a set of GPS signals. Comparing the GPS location from the routes R as g_r with the GPS signal from the receiver $\hat{g}$, we can detect the spoofing if big differences exist.

Algorithm 1: Mapping calibration process

```
Input: $(r_0, r_1 \cdots, r_k) \leftarrow$ extracted roads and reconstructed GPS signals $g(lat; long)$
Output: a set of routes $\mathbf{r}$
    for a GPS signal $g_i$ from the set $\mathbf{g}(g_0, g_1, \cdots, g_n)$ do
        for $r_j$ from the road set do
            calculate the distance from $g_i$ to $r_j$
        end for
        find the nearest route $r_*$ for the signal $g_i$
        updated data for corresponding route $\mathbf{r} = \mathbf{r} \bigcup \{r_*\}$
    end for
```

4.5 Evaluation and Experimental Results

Have discussed the above methods that use driving information to cross-validate the GPS signal and detect any spoofing attacks, we now design a series of experiments to validate our initial hypothesis. First, we collect the driving information as well as the GPS signals from real road tests. With the OpenXC module assisted, we collect 15 driving trips from the same vehicle. The testing routes are designed as 10-20 minutes long, and tried to cover all kinds of driving scenarios, including left turn, right turn, driving around curved road and etc. With the OpenXC hardware demo plug into the vehicle, we collect the data from the OBD-II board and record it on the cellphone App. Then, we construct the route based on the driving information using the two methods we discussed above. At last, we need to establish whether the cross-validation method could detect the GPS spoofing attacks. This can be accomplished by emulating an attack with misleading GPS signals with respect to spoofing scenarios. To evaluate the applicability of our proposed GPS detection method to real world vehicle data, we assess its performance in terms of spoofing detection and detection time.

4.5.1 Driving Information Data Set

The driving data set we collected from the in-vehicle bus is extremely important to our proposed GPS spoofing detection methods. Here we will shortly introduce the data acquirement methods and the type of data from the set. We acquire the driving data by using the OpenXC platform which is a pluggable module have hardware and software that lets the customer extend the vehicle's applications. It uses a standard and well-known tool to open up a wealth of data from the vehicle to developers, even beyond OBD-II. OpenXC allows consumer devices, such as smartphones, to access data from any vehicles. Using OpenXC, users can monitor and read out data from many of the sensors on a vehicle, enabling new and innovative vehicle-centric applications. The OpenXC provides a rich dataset including the speed, steering positions, brake positions and the GPS signals from the sensor. This plentiful dataset assists the GPS spoofing detection scheme proposed in our paper. Here, we mainly use the OpenXC vehicle interface, which is plugged into a Ford 2017 vehicle through the OBD-II board to collect the in-vehicle information needed. The signal format and specification used to construct the GPS signal are listed in the below Table 4.1.

Name	Range	Frequency
steering_wheel_angle	-600 to +600 degrees	10 Hz
vehicle_speed	0 to 655 km/h	10 Hz
accelerator_pedal_position	percentage	10 Hz
latitude	-89.0 to 89.0 degrees	1 Hz
longitude	-179.0 to 179.0 degrees	1 Hz

Table 4.1: OpenXC dataset

4.5.2 Model Validation

From the above discussions, the key parameters to model the driving behavior of the vehicle should be determined before we reconstruct the GPS signal. As a result, the first step is to validate the default parameter of the test vehicle based on the driving data as well as the GPS signals without attacks for reference. For Method I, we need to yield the steering ratio r from the training routes in advance. Since lack of design details to calculate the ratio, we try to find the fitted steering ratio for the test vehicle by sweeping mechanical reasonable range, i.e, from 11 to 20. Figure 4.6 displays an example of the constructed routes with a set of ratio values and the corresponding true GPS signals (marked in a star) for reference. The figure shows visually the third curve with $r = 13$ is the closest trajectory to the real route. We further narrow down the range and verify the best-fit steering ratio of the test vehicle is 12.4. We will use $r = 12.4$ in the following experiments on GPS spoofing detection.

Similar for Method II, the preliminary step is to model the relationship between the steering θ_w and the heading θ_h. We use the Matlab curve fitting toolbox automatically find the best fit for the relationship. By trying different regression methods (e.g, linear, polynomial and Fourier), we derive a linear relationship between the steering and the heading from 5000 single data points for training, as shown in the Figure 4.3.

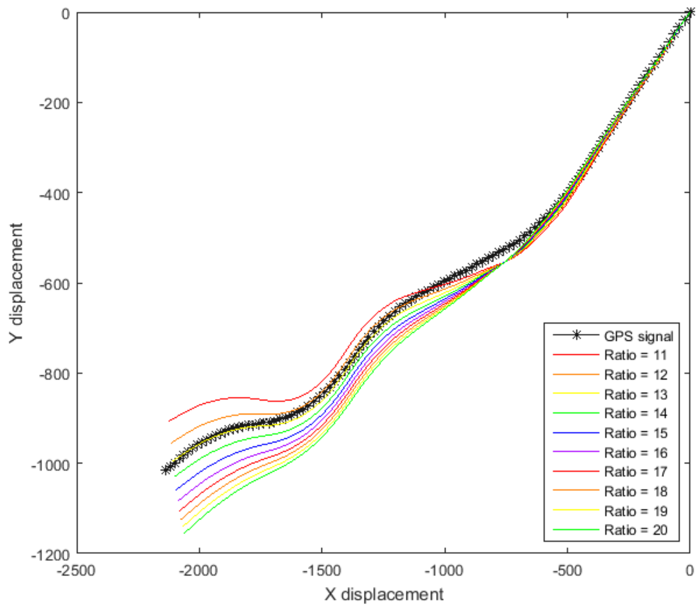

Figure 4.6: Reconstructed GPS Routes with different ratio, range of r is from 11 to 20 and the best fit is $r = 13$

4.5.3 Reconstruction of GPS Signals

We propose the implementation of two methods to reconstruct the GPS signal from the in-vehicle dataset. The test performs a cross-check between the reported GPS positions from the receiver and the estimated positions from our reconstruction methods. While we have confirmed that the two methods could construct the trajectory based on the driving information with an accepting error range, we now focusing on constructing our detector. The most important factor of a well-defined detector is the threshold for detection.

We check for each incoming position report whether $dist(g_i, \hat{g}_i) < \tau$ holds, where g_i is the calculated GPS position and $\hat{g}_i$ is the position reported by the receiver. $dist()$ is the Euclidean distance function, and τ denotes a predefined threshold which tolerates measurement errors. Choosing the right threshold τ depends on the accuracy of the underlying reconstruction method. Smaller τ lead to higher false positive rates, while larger τ create more space for the spoofer.

More specifically, to make the signals consistent in formal representation, we map the GPS signal from degrees to relative displacement by calculating the great-circle distance from the latitude and longitude. Therefore, the coordinates of GPS location can be expressed as $P_g(x_g, y_g)$ in meter. Correspondingly, the reconstructed location is expressed as $P_r(x_r, y_r)$. Then, the errors can be defined as $dist(P_r - P_g) = \sqrt{(x_r - x_g)^2 + (y_r - y_g)^2}$, which is represented in meter. We evaluate the error (m) for route reconstruction by applying the two methods discussed above as shown in Table 4.2. For all the test routes, the average error is 10.13m for Method I and 6.25m for Method II. By considering the route distance from the real road situation, a thresh hold as small as 15m is appropriate to detect the spoofing attacks. Note that the maximum error from Method I is above the threshold. To cope with the false alert for Method I, we consider a sequence of 10 consecutive test points above the threshold as a valid detection of spoofing. In addition, the average error of

Method II is 6.25 m which achieves 38% decrease from the Method I. Therefore, with lower error, Method II will achieve better performance in spoofing detection and positioning.

	Method 1		*Method* 2	
Applications	**Max error**	**Average error**	**Max error**	**Average error**
route#1	18.03	10.49	13.90	6.95
route#2	19.01	10.67	13.28	6.75
route#3	19.02	10.71	11.73	5.90
route#4	18.68	9.54	12.91	5.58
average	18.57	10.13	12.92	6.25

Table 4.2: Statistical results from route reconstruction

4.5.4 Calibration with Map Information

With the two GPS reconstruction methods discussed above, we achieve the best errors from Method II as 6.25 m. This accuracy is acceptable when the road network is sparse in some rural areas. However, the result needs to be improved when coming to the dense area in the city, where the distance between two roads might be less than 5 meters. To improve GPS spoofing detection accuracy, we identify an additional optimization technique that helps to lower the error rate. As the open source maps contain GPS signals as fixed locations, this allows to better predict the vehicle's location by incorporating the discussed construction methods with routes information extracted from maps.

Based on the former calculation, the maximum errors usually occurred when the vehicle takes turns. This is because the steering signal changed frequently and sharply when taking turns and the errors from the real signal are large when taking use of our heading estimation method. As shown in the Figure 4.7 (a), the vehicle

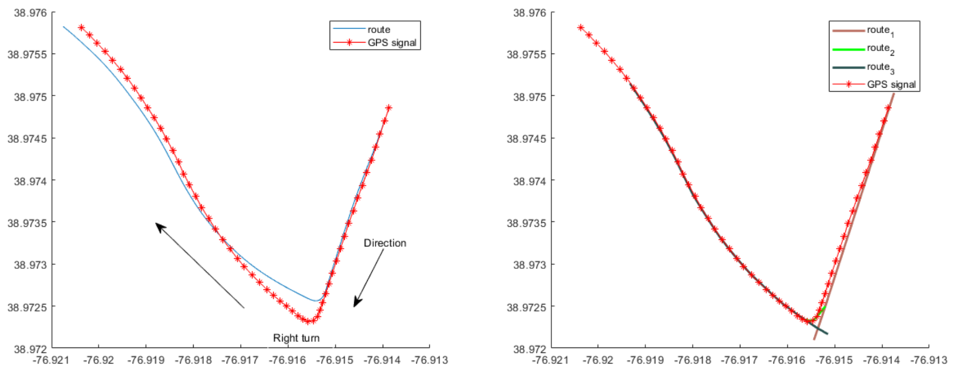

Figure 4.7: (a) Reconstructed GPS signal using Method II (b) Reconstructed GPS signal with map information

is making right turns along the route, the reconstructed GPS signal aligns the route well before the turn. However, the calculated path deviates from real GPS coordinates and the mismatches accumulated with time. As clearly shown after the right turn, a large margin is seen from the reconstructed positions to the real positions. Since this margin will definitely influence accuracy as well as the threshold set for spoofing detection, we consider making use of the map information to further calibrate the GPS signals. For example, we extracted the route information at that intersection and figured out the vehicle is taking right turn based on several observations after the turn. As a result, the reconstructed GPS signal aligns the path with the real signal, which is shown in Figure 4.7 (b). We conduct the same map calibration on the training routes. Results also demonstrated that the errors drop from 6-10 meters to 1.21 meters with the map information, which achieves 10 times accuracy. With these achievements, the threshold for detection could be narrow down and the false negatives for detection would decrease.

4.5.5 GPS Spoofing Detection

With the definition of the valid detector discussed above, we now seek to emulate a GPS spoofing attack and measure the effects. We compared our two spoofing detection tests with regard to the detection rate and detection delay.

As we stated before, conducting legitimate GPS spoofing attacks in the wild on the road test is challenging. We, therefore, use the simulated GPS spoofing attacks to test the baseline of our expectations. The goal of these experiments is to demonstrate that the vehicle could be misled form the route by emulated GPS attacks consisted of wrong GPS signals. To simulate how real GPS spoofing attacks impact the GPS signals on board, we designed several routes deviate from the real location to simulate the GPS receiver is under spoofing. The speed of the fake routes keeps the same as the one for the real route. The attacks would take place at the intersections. For example, when the vehicle turns right, however, the designed spoofing signals indicate it turning left.

From Table 4.3, we could find out that the detection accuracy for the simulated spoofing attacks is 100%. And the response time of Method II is a bit faster than Method I because it does not evaluate consecutive points to make a decision. Overall, our results confirm that the faked GPS signal can be detected using construction methods from in-vehicle measurements.

Applications	**Detection rate**	**Detection time(s)**
Method I	100%	18
Method II	100%	15

Table 4.3: Detection results from GPS spoofing

4.6 Summary

GPS spoofing attacks threaten the confidentiality of the navigation and position applications in vehicles. In this chapter, we propose a cross-validate a low-cost method for detecting the spoofing of civilian GPS signals received by vehicles. Using driving information collected from the real routes, we demonstrate the ability to detect such GPS spoofing attacks with high accuracy. We believe that our proposed add-ons can easily be included in the future vehicles, and represents the best means of reliability detect such attacks in the short and medium terms.

Chapter 5: Blockchain-based Anonymous Reputation System for Trust Management in VANETs

5.1 Introduction

Recently, VANETs are suggested as the foundation of intelligent transportation system to improve the efficiency of transportation and ensure the safety of both vehicles and pedestrians. Two types of communications, namely vehicle-to-vehicle and vehicles-to-infrastructure are established in VANETs to assist in sharing the valuable driving information through the network [26]. Through dedicated short range communication radio, vehicles exchange messages with nearby vehicles as V2V and communicate directly with roadside units as V2I.

However, some unique characteristics of VANETs such as high mobility and volatility [31] make it vulnerable to various kinds of attacks. Security, privacy, and trust are becoming more and more critical when designing the VANETs. Although some security services have been well-studied in other fields can provide secure communication channels against external attackers, trust management and privacy protection for vehicles are still open issues for VANETs. Specifically, it is difficult to deal with mis-behaviors and forged messages from authenticated vehicles. These forged messages could not only decrease the transportation efficiency, but also cause traffic accidents which threaten human life [45]. And in other case, internal attackers can easily track other vehicles or profile the drivers' actions by analyzing all the broadcast messages in VANETs.

To mitigate those attacks from both internal and outside, a trust communication environment should be proposed to evaluate the trustworthiness of messages based on both direct historical interactions and indirect opinions about the senders [103]. An effective trust model should have the following properties:

1. Efficiency. It should determine the trustworthiness of a warning message in both congestion and sparsity situations.

2. Privacy. It should not reveal any sensitive information about the senders without permission, i.e, the identity of the senders.

3. Robustness. It should be resistant to those attacks aiming at deceiving the trustworthiness evaluation or disabling the trust model.

Blockchain is the underlying technology of Bitcoin protocol emerged in 2008 [104]. It is a distributed and public ledger scheme encrypted with Merkel tree and hash function. It reaches the consensus based on the proof of work (PoW) algorithm [105]. These significant features of blockchain make it potential for constructing the desirable trust model in VANETs. All the broadcast messages and actions of vehicles will be written into the immutable and unforgeable record, which can be verified and audited by every entity in the network later. However, the essential transparency of blockchain makes the privacy not protected. By checking the ledger, any actions made with any public key are traceable.

In this chapter, we propose a blockchain-based anonymous reputation system (BARS) to establish distributed trust management while simultaneously protect the privacy of vehicles. First, we exploit the features of blockchain and extend conventional public key infrastructure (PKI) with an efficient privacy-preserving authentication mechanism. The linkability between the public key and the real identity of a vehicle is eliminated when certificate authority (CA) operates the certificate issuance and revocation. All the actions of CA are recorded in blockchain transparently without sensitive information about vehicles so that the public key can be used as an authenticated pseudonym. Law enforcement authority (LEA) is responsible for managing BARS and keeping the pairs of public key and real identity in case of dispute. Second, we design a reputation management algorithm to evaluate the trustworthy of each vehicle from the authenticity of broadcast messages and opinions from the other vehicles. Since all the messages are recorded in blockchain as persistent evidence, it is easy for LEA to evaluate the reputation score for each vehicle. The reputation score could provide incentive for internal vehicles to prevent misbehaviors and mitigate the usage of forged messages.

5.2 Preliminaries

Before elaborating how BARS works, we will first introduce the concept of blockchain and how it works to maintain the security. Then we investigate the problems of conventional PKI and explain the conception of certificate transparency.

5.2.1 Blockchain

Blockchain is a computational paradigm as the core component for the Bitcoin [104]. It is a distributed ledger containing all transactions ever executed within the network. The ledger is enforced with cryptography and carried out collectively in a peer-to-peer system. As a secure and decentralized computational infrastructure, it is widely acknowledged as a disruptive solution for the problems of centralization, privacy and security when storing, tracking, monitoring, managing and sharing data [106].

In blockchain, asymmetric cryptography is exploited to guarantee the security of transactions through shared channels. The digital signature generated with the sender's private key provides authentication, integrity, and non-repudiation. The transactions can be accessed and verified by all the participant nodes in the network, which is called mining. The participant nodes in the network are given incentives in the form of Bitcoins for performing the mining operations. Miners compete to solve a complex mathematical computation and the winner will earn Bitcoins as a reward. Figure 5.1 presents the structure of a blockchain, each block contains the hash value of the previous block, the timestamp, the nonce, and the root of a Merkle tree. Miners compete to generate a new block by solving a complex mathematical computation. A block is valid only if it contains PoW with a given difficulty. In the PoW algorithm, miners should find a nounce so that the result is below a certain difficulty threshold. The time consumption for finding a new block can be controlled by manipulating the difficulty. The block of successful PoW miner is selected to be next block in the blockchain. Once a winner's block is selected, all other nodes update to that new block [107].

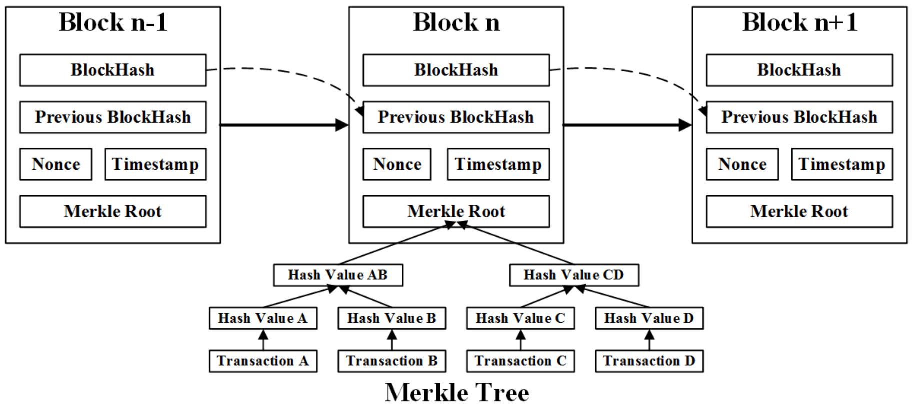

Figure 5.1: Structure of a blockchain, each block contains current and pervious blockhash, nonce, timestamp and the Merkle root

Blockchain realizes a secure network with untrusted parties which is desirable for VANETs with numerous nodes. PoW is a key component of Bitcoin, which is difficult to solve, but trivial to verify. It often boils down to a random process of trying to find a solution to a mathematical puzzle, like a partial hash collision [108]. In VANETs, the RSUs with relatively high computing power can behave as the miners. First, RSUs are responsible for verifying the messages to prevent Sybil attack. Second, RSUs will collect those verified messages and record them in blockchain. In this dissertation, we assume that attackers cannot control the majority of the RSUs.

RSUs implement the PoW scheme by finding a nonce that gives the block's hash the required zero bits. Once the computing effort has been expended to make it satisfy the PoW, the block cannot be changed without redoing the work [104]. Whoever attempts to change one block has to redo all the blocks after it. If the majority of computing power is controlled by normal RSUs, the honest chain will grow the fastest and outpace any competing chains. To modify a past block, an attacker would have to redo the PoW of all blocks after it and surpass the work of normal RSUs, which is not practical in VANETs with thousands of RSUs. Therefore, all the broadcast messages in VANETs are tamper-proof and cannot be repudiated.

The lack of central control of blockchain ensures scalability and robustness by using resources of all participating nodes and eliminating many-to-one traffic flows, which in turn decreases delay and overcomes the problem of a single point of failure [109]. The block header incorporates a Merkle root [110], which is the root of a hash tree containing hash values of all messages in VANETS. It allows nodes to verify that a message is part of a block by starting at the respective leaf and traversing the branches up to the root [108]. If the final hash and the Merkle root are identical, the message must be part of the block. For the verification, distributed nodes request just a list of intermediary hashes instead of the whole block with all messages. Since the block headers containing the Merkle root are secured, valid intermediate hashes cannot be faked easily. Thus, the existence of a message can be efficiently and reliably verified with this approach.

5.2.2 Certificate Transparency

Certificate transparency [111] is invented by Google aiming to prevent transport layer security (TLS) certificate authorities from issuing public key certificates for a domain without being visible to the owner of the domain. The technology is being built into Google Chrome aiming at website certificates. The core idea of certificate transparency is that a public log is maintained to show all the certificates that have been issued [112]. The log is append-only. Anyone can append a certificate to the log. Auditors can obtain two types of proofs: (a) a proof that the log contains a given certificate, and (b) a proof that a snapshot of the log is an extension of another snapshot (i.e., only appends have taken place between the two snapshots).

Abstractly, a certificate is a signed pair (sub_j, pk_{sub_j}), asserting that a subject sub_j's public key is pk_{sub_j}. In certificate transparency, the CA's database of certificates is maintained as a Merkle tree in which these pairs are stored left-to-right in chronological order at the leaves of the tree. Items are added chronologically, by extending the tree to the right. A certificate is accepted by a browser only if it is accompanied by a proof that the subject's key pair has been inserted into the log. Observers can verify that the log is maintained append-only. To perform such a verification, the observer submits to the CA the hash value of the log at two different times. The CA returns a proof that the log corresponding to the later hash value is an extension of the log at the earlier time. The properties of Merkle trees ensure the insertion into the log, and both proofs can be done in time/space $O(logN)$.

It is vital for the log to be a single linear record. If the log maintainer can create different versions of the log to different users, the security is broken. Linearity is guaranteed in two ways. First, whenever a user interacts with the log, it requests proof that the current snapshot is an extension of the previously cached snapshot. In order to avoid that a version constructed for a particular user is being used before, it should be performed before authentication. Second, gossip protocols can be used to disseminate values of the log, which means that the users of the log need to find a method to exchange with other users the hash value of the log that they have received. At any time, a user can request proof that the snapshot currently offered by the log is an extension of a previous snapshot received through direct communication with other users.

In certificate transparency, one can prove that a certificate is in the log, but there is no proof of whether it is still current. Revocation transparency [113] is an extension of certificate transparency aiming to deal with revocation. Two related methods for revocation transparency are proposed. The first method stores revocations in a data structure called a sparse Merkle tree, which is a transformed Merkle tree in which most of the leaves are zero. A 256 bit-length path in this tree is used to represent the hash of a certificate. The path ends in a 1 or 0 according to whether the certificate is revoked or not. Thus, the tree is a kind of binary tree with 2^{256} leaves. However, because it is sparse, these leaves do not have to be stored individually. To revoke a certificate, one alters the sparse Merkle tree so that the relevant path terminates in 1, and one enters a record of this action in the certificate-transparency append-only log. Unfortunately, checking whether a certificate has been revoked is inefficient. The second method is to track the revocations by a separate mechanism, which is related to the number of revocations that in turn can be assumed proportional to the number of issued certificates. Alternatively, one can check the entire certificate-transparency log for revocation records. However, this is again linear in the number of issued certificates that proofs require linear space and time with data sizes measured in tens or hundreds of gigabytes. Therefore, this method is impractical.

5.3 Framework of BARS and Anonymous Authentication

Anonymous authentication is fundamental for trust communication and privacy protection. We first introduce the main components of BARS along with their corresponding functions of anonymous authentication. In BARS, CA and LEA can help in realizing the anonymous authentication by implementing three major tasks: system initialization, certificate update, and public key revocation. We then respectively present the three tasks and elaborate the process of privacy-preserving authentication.

5.3.1 Components of BARS

Law Enforcement Authority (LEA). The functions of LEA include registration, monitoring behaviors of vehicles, and evaluating the reputation scores of each vehicle. LEA authorizes CA for certificates issuance and revocation and keeps the database that contains correlation between vehicles' public keys and the real identities with high-level security.

Certificate Authority (CA). CA issues and revokes certificates only if it gets the warrant from LEA. All the actions of CA will be recorded transparently in the blockchain and can be verified by every entity in VANETs.

Certificate. The certificate contains the expiration date, the public key, and the reputation score but no real identity so that privacy of the vehicle is protected.

Blockchain for certificates (CerBC). CerBC acts as the public ledgers for all the issued certificates. It provides efficient proof of presence for received certificates.

Blockchain for revoked public keys (RevBC). RevBC acts as the public ledgers for all revoked public keys. It provides efficient proof of absence for the senders public key.

Blockchain for messages (MesBC). All the broadcast messages will be recorded in MesBC as persistent evidence in case of disputes.

Roadside Unit (RSU). The global consensus is based on the proof of work (PoW) provided by RSUs. As long as more than half of the RSUs are not compromised, the security of BARS can be guaranteed.

Vehicle. On one hand, vehicles can check the CA and LEA by verifying all the messages recorded in blockchains. On the other hand, vehicles can also verify each other to prevent misbehaviors and forged messages.

5.3.2 System Initialization

At beginning, each entity generates a pair of private and public keys. When vehicle A entering the network, it uses the secure channel to submit LEA its initial public key and materials to prove its legal identity. LEA will send a signed warrant to CA if the materials are valid. Next, CA will issue an initial certificate to vehicle A.

Note that the submitted material contains the vehicle A's private information. Thus, only LEA preserves them in the database with high-security level, which will be used for verifying the vehicle's real identity when a dispute happens.

5.3.3 Certificate Update

Vehicle A will send a request to LEA for updating certificate in the following situations: a) before the current certificate expires. b) if the security of its private key is threatened. c) if it requests to replace its public key for privacy consideration. The public key, reputation score, and expiry date are updated in a new certificate. Figure 5.2 illustrates the steps to update certificate anonymously.

We also elaborate the detailed steps for certificate updating as following.

Step 1. Vehicle A generates a new pair of public key and private key $\{pk_A^n, sk_A^n\}$.

Step 2. Vehicle A sends LEA certificate update request encrypted with LEA's public key pk_{LEA}. The request includes the vehicle A's current public key pk_A^{n-1}, updated public key pk_A^n, proofs of real identity, and the signature Sig_A generated by A's current private key sk_A^{n-1}.

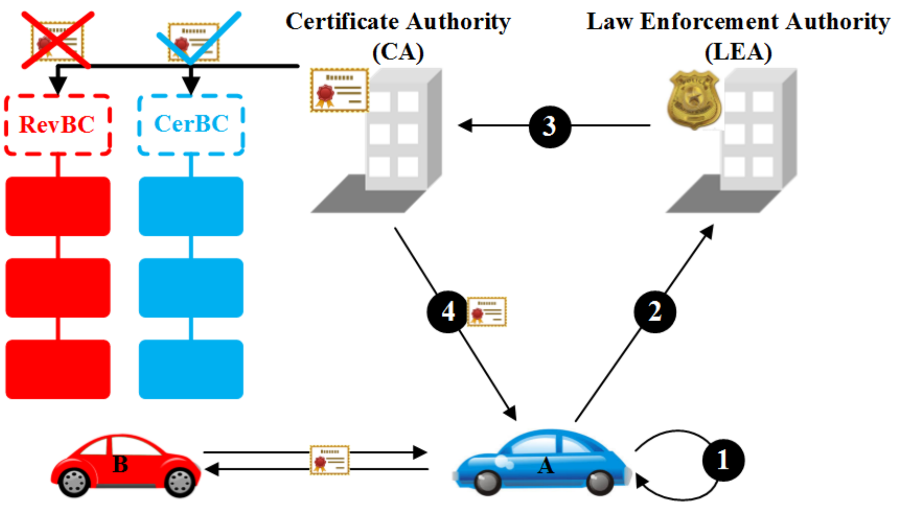

Figure 5.2: Certificate update process. CA, LEA and vehicle involve in this process

Step 3. If vehicle A's request is verified, LEA will send CA a signed warrant. For the purpose of privacy protection, the linkability between A's current and updated public keys is unknown by CA.

Step 4. CA will verify the signature in the warrant. Then, an updated certificate containing the updated public key pk_A^n, A's reputation score Rpt_A, and the expiration time T_A will be issued to vehicle A publicly and will be recorded into CerBC. The message contains the following components.

$$C_A^n = \langle pk_{CA}, Sig_{CA}, pk_{LEA}, Sig_{LEA}, pk_A^n, Rpt_A, T_A \rangle \tag{5.1}$$

5.3.4 Certificate Revocation

In general, Vehicle A's public key should be revoked before its expiry date or A's misbehavior is discovered by LEA. In order to provide revocation transparency, LEA sends a signed revocation instruction to CA that contains the revoked public key pk_{rev} and the revocation time T_{rev}. Then CA broadcasts the revocation messages that contains the revoked public key, the timestamp, signatures of CA and LEA: $Rev = \langle pk_{CA}, Sig_{CA}, pk_{LEA}, Sig_{LEA}, pk_{rev}, T_{rev} \rangle$.

The RSUs will verify all the revocation messages in a predefined interval, delete the expired public key, and lexicographically insert revoked public keys into RevBC. RevBC is constructed with a lexicographical Merkle tree [114] also called LexTree that can provide efficient proof of absence. The idea is that we can group all the information about a subject into a single node of the binary search tree, and while being able to efficiently generate and verify the proof of absence. We consider an order on bitstrings denoted $\leq$. This order could be the lexicographic order in the ASCII representations but it could also be defined anything else. A LexTree LT is a binary search tree of pairs of bitstrings:

- For two pairs (d, h) and (d', h') of bitstrings in LT, (d, h) is left of the occurrence of (d', h') if and only if $d \leq d'$ lexicographically;
- For all nodes $n \in LT$, n is labeled with the pair $(d, H(d, h_l, h_r))$, where d denotes the bistring and (d_l, h_l) (resp. (d_r, h_r)) is the label of its left child (resp. right child) if it exists; otherwise it would be the constant null.

In this way, it prevents CA or LEA from arbitrarily revoking any public keys because RevBC keeps the persistent evidence of all the revoking operations. Additionally, LexTree makes it very efficient for a vehicle to prove its public key is not in the RevBC, which means this public key is valid (if the associated certificate is in CerBC).

5.3.5 Authentication Process

Vehicle A's certificate C_A is used for authentication. As shown in Figure 5.3, when vehicle B receives C_A, it first checks whether the certificate is expired. If not, B will look up in the CerBC and RevBC to make sure C_A is present in CerBC but pk_A is absent in RevBC, which means pk_A is issued and not revoked by C_A. Then we will give the details of proof of presence and proof of absence in the authentication process.

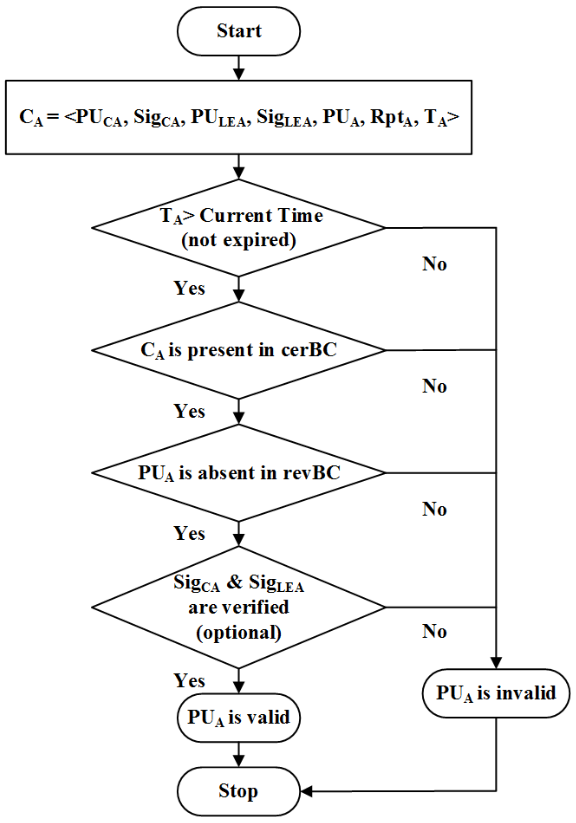
Start
C_A = <PU_CA, Sig_CA, PU_LEA, Sig_LEA, PU_A, Rpt_A, T_A>
T_A > Current Time (not expired)
No
Yes
C_A is present in cerBC
No
Yes
PU_A is absent in revBC
No
Yes
Sig_CA & Sig_LEA are verified (optional)
No
Yes
PU_A is valid
PU_A is invalid
Stop

Figure 5.3: The process of anonymous authentication in the BARS

5.3.5.1 Proof of Presence in CerBC

Figure 5.4 illustrates how to prove a certificate C_4 is present in the CerBC. A tuple $(dir, hash)$ is used to prove the presence for C_4 in the CerBC, in which $dir = left, left, right$ and $hash = h_3, h_{12}, h_{58}$. The receiver can get the root hash value using the tuple. If this root hash value is equal to the root recorded in CerBC, it indicates that C_4 and the associated public key is valid.

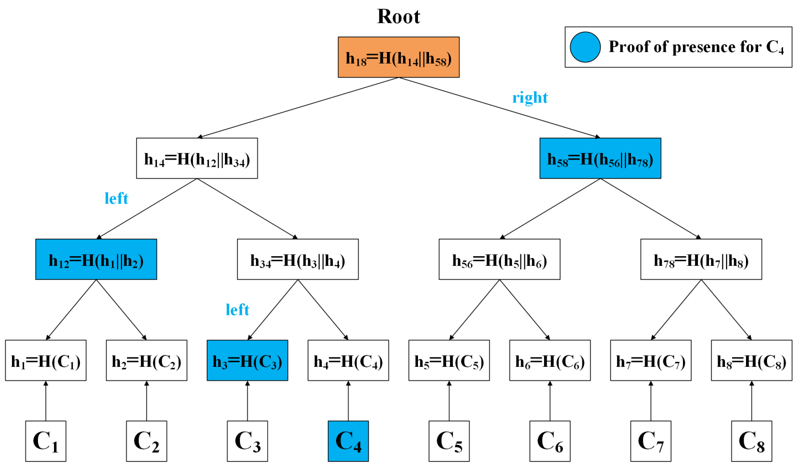

Figure 5.4: An example to show the proof of presence for a certificate

5.3.5.2 Proof of Absence in RevBC

It is unreasonable to force a vehicle to demonstrate that its public key is revoked. Thus, a proof of absence is necessary for a vehicle to convince others its public key is valid. As shown in Figure 5.5, all the revoked public keys (not expired) are recorded in the data structure based on lexicographical Merkle tree. Vehicle A should prove that two adjacent public keys (pk_7, pk_8) exist in the left-right traversal of the tree meanwhile $pk_7 \leq pk_B \leq pk_8$ lexicographically.

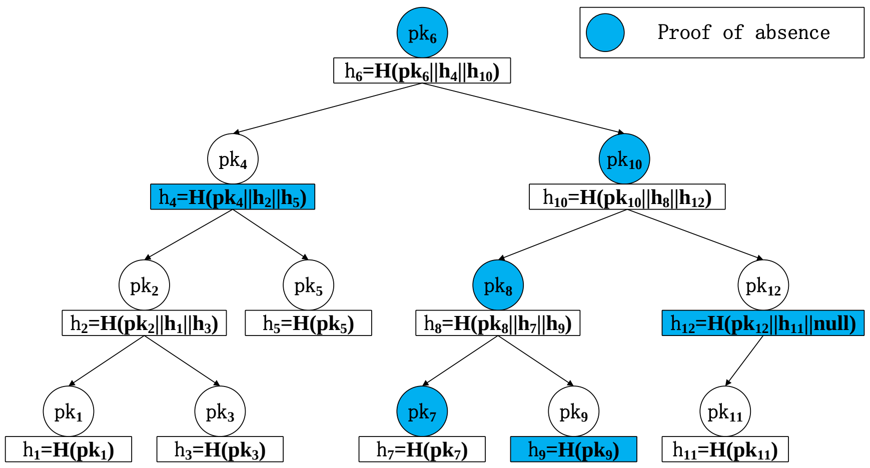

Figure 5.5: An example of proof of absence for a public key

A tuple $(pk, hash)$ is used for proof of absence, in which $pk = pk_7, pk_8, pk_{10}, pk_6$ and $hash = h_9, h_{12}, h_4$. Similarly, if the hash value calculated using the tuple is same as h_6 that is recorded in RevBC, it means that A's public key is absent in RevBC.

5.4 Reputation Management

We present the BARS as a decentralized, efficient and robust trust model in VANETs. BARS relies on the reputation score of a vehicle to determine the trust level of broadcast messages. The reputation score is the incentives for vehicles to share safety information and monitor each other so that misbehavior can be prevented and the spread of forged messages from internal vehicles can be mitigated. It is an essential issue but out of the scope of this dissertation to associate the reputation score with a vehicle's actual benefit.

BARS evaluates the reputation of a vehicle based on the authenticity of broadcast messages and opinions of other vehicles. The reputation evaluation mechanism consists of penalty and reward algorithms. Sharing authentic safety messages and disclosing misbehavior or spread of forged messages will increase a vehicle's reputation score. Contrarily, some bad operations such as performing misbehavior, broadcasting forged messages, and maliciously accusing other vehicles will decrease its reputation score.

In this section, we will elaborate how BARS can provide a trust communication environment meanwhile protect privacy of the vehicles.

5.4.1 Different Types of Messages

There are three types of messages involved in the VANETs communications: beacon messages, alert messages, and disclosure messages. Periodically, vehicles broadcast beacon messages containing driving status for traffic management. Alert messages will be broadcast when an emergency happens such as hard braking or losing of control. If any vehicles dispute the authenticity of the received message or witnesses misbehavior, they can send disclosure messages to LEA. Next, LEA will make a judgment on whether this will affect the reputation scores of related vehicles. According to the critically of emergency, alert messages have three levels.

Level 1. When vehicle A loses control, it will broadcast level 1 alert message to avoid potential collisions automatically.

Level 2. This level of alert message is used for forewarning nearby vehicles before the sender changes its driving status, including braking, lane changing and etc.

Level 3. In case of poor road conditions such as obstruction or road damage, passing vehicles will broadcast level 3 alert messages to alert vehicles behind to keep caution.

5.4.2 Reputation Evaluation Algorithm

The reputation evaluation algorithm consists of two aspects: reward mechanism and punishment mechanism. Based on our observations, there are two kinds of behaviors of the VANETs will be rewarded. First, the vehicle broadcasts alter messages honesty and actively. Second, the vehicle sends disclosure messages to LEA when it witnesses misbehavior or receives forged messages. On the contrary, there are also two actions will be punished. First, vehicle will get punished if it is disclosed for misbehavior or broadcasting forged messages. Second, the vehicle abuses disclosure messages to slander other vehicles.

There are several factors affecting the evaluation of reputation scores as follows. For example, if an alert message is life-critical and is received by more vehicles, the earlier one to report and broadcast is considered to given more contribution. Thus, the more reward the sender will get in terms of reputation scores. We list the factors would influence the reputation score as follows.

L: The level of alert messages, e.g., $L = 1, 2, 3$.

D_r: The relative density of vehicles which denotes as $D_r = D/D_{aver}$. In this chapter, D_{aver} is set to 20 vehicles per km.

S: The sequence of the senders, $S = 0, 1, ..., n$. The first vehicle to broadcast alert message will be set to $S = 0$.

In addition, we set reward coefficient α and penalty coefficient β in formula 5.2 and 5.3 to implement reward mechanism and penalty mechanism.

$$R(L, S, D_r) = \alpha \cdot D_r \cdot \frac{1}{e^S \cdot L} \tag{5.2}$$

$$P(L, S, D_r) = (-1) \cdot \beta \cdot D_r \cdot \frac{1}{e^S \cdot L} \tag{5.3}$$

As illustrated in Algorithm 2, if no receiver disputes the authenticity of an alert message, the reputation score will increase base on reward mechanism. On the contrary, if any receivers send disclosure messages to dispute the authenticity of the alert message, LEA will collect evidence to make a judgment. The vehicles who broadcast forged alert messages will be punished heavily. Whereas, the vehicles who abuse disclosure messages will also get punished.

Algorithm 2: Reputation Management Algorithm

Input: MA: Alert message broadcast by vehicle $V_i (i = 1, 2...n)$; MD: Denouncement message broadcast by vehicle $V_j (j = 0, 1...m)$; R'_i, R'_j: Current reputation of V_i and V_j; S_i, S_j: The sequence in which messages are sent; D_r: The relative traffic density at the location of the event.

Output: R_i, R_j: Updated reputation of V_i and V_j.

```
1: if $j = 0$ then
2:     for each $V_i$ do
3:         $R_i.honesty \leftarrow R'_i.honesty + (100 - R'_i.honesty) \cdot \mathrm{R}(MA.type, S_i, D_r)$
4:         $R_i.cooperation \leftarrow R'_i.cooperation + 100 \cdot \mathrm{R}(MA.type, S_i, D_r)$
5:     end for
6: else
7:     if $MA$ is true then
8:         for each $V_i$ do
9:             $R_i.honesty \leftarrow R'_i.honesty + (100 - R'_i.honesty) \cdot \mathrm{R}(MA.type, S_i, D_r)$
10:            $R_i.cooperation \leftarrow R'_i.cooperation + 100 \cdot \mathrm{R}(MA.type, S_i, D_r)$
11:        end for
12:        for each $V_j$ do
13:            $R_j.surveillance \leftarrow R'_j.surveillance + 25 \cdot \mathrm{P}(MA.type, S_i, D_r)$
14:        end for
15:    else
16:        for each $V_i$ do
17:            $R_i.honesty \leftarrow R'_i.honesty \cdot (1 + \mathrm{P}(MA.type, S_i, D_r))$
18:            $R_i.cooperation \leftarrow R'_i.cooperation \cdot (1 + \mathrm{P}(MA.type, S_i, D_r))$
19:        end for
20:        for each $V_j$ do
21:            $R_j.surveillance \leftarrow R'_j.surveillance + 50 \cdot \mathrm{R}(MA.type, S_i, D_r)$
22:        end for
23:    end if
24: end if
25: return $R_i, R_j$
```

5.5 Results and Analysis

5.5.1 Security Analysis

1. Security of Certificates. RSUs verify the signatures of CA and LEA in the certificate issuance or revocation messages and record them into CerBC and RevBC respectively. The global consensus is provided by the PoW of RSUs to guarantee that each vehicle has the identical public ledgers that consist of authenticated certificates and revoked public keys. As long as more than half of all the RSUs are not compromised, the CerBC and RevBC are unforgeable and immutable.

2. Security of Broadcast Messages. Proof of presence in CerBC and proof of absence in RevBC ensure the authentication of the public key of vehicle A. Then, vehicle A will use its private key sk_A to generate a signature for each broadcast message and receivers can use A's public key pk_A to verify the signature. RSUs and vehicles cooperatively recorded all the broadcast messages into the chronological MesBC, which is the persistent evidence when a dispute happens.

3. Privacy of Vehicles: Vehicle A uses public key as the pseudonym to hide the linkability between the public key and real identity. For the trade-off between security and privacy, the database of identity-public key pairs is stored with high-security level in LEA. It means only LEA knows the real identity of the public keys so that LEA is able to track the malicious vehicle when it

performs misbehavior or broadcasts forged messages. A vehicle can get several certificates in one request and change its public key at particular locations to enhance its privacy.

5.5.2 Validation of Reputation Evaluation

We consider three vehicles perform different behaviors in 100 hours. As Figure 5.6 presents, from T_1 to T_2 and T_3 to T_4, vehicle A and B actively broadcasts authentic messages and their reputation scores increase correspondingly. From T_2 to T_3, vehicle B broadcasts five forged messages and is disclosed by A. Thus, A's reputation score increases whereas B gets punishment. From T_4 to T_5, vehicle B abuses five disclosure messages to slander other vehicles. As a result, B's reputation score decreases. Vehicle C refuses to participate in the BARS. The results show that the reputation score effectively reflects vehicles' behaviors.

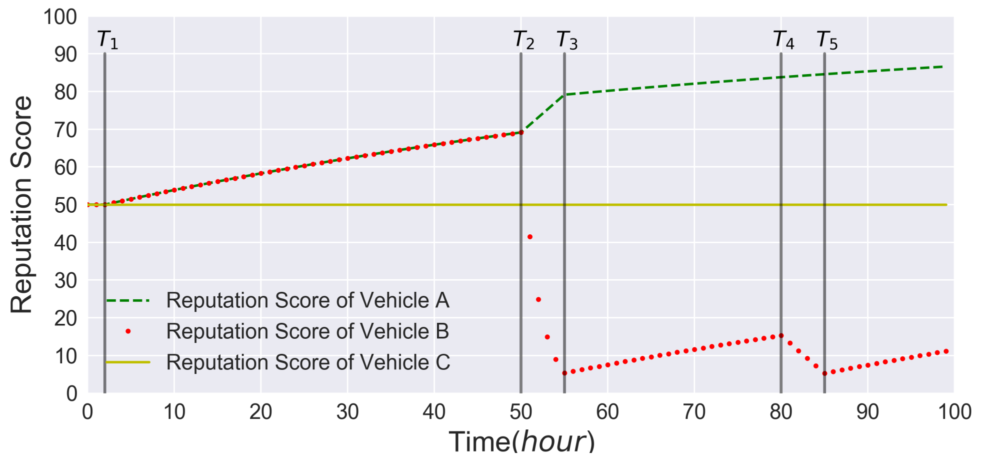

Figure 5.6: Reputation score changes of three vehicles when they behave differently in the BARS

5.5.3 Performance Evaluation

In BARS, receivers authenticate the public key of the sender based on proof of presence in CerBC and proof of absence in RevBC. We first evaluate the performance of anonymous authentication for vehicles in terms of storage and transmission overhead and computational overhead. The implementation of anonymous authentication is based on SHA-256. All the experiments are conducted using 2.5 GHz Intel Core i5 and 8 GB 1600 MHz DDR3. The time consumption to compute a SHA-256 is less than $t_1 = 0.01$ ms per 1 KB of input.

Storage and Transmission Overhead: A block header would take about 80 bytes [104]. Suppose that blocks are generated every 10 minutes, the storage overhead for one blockchain is 80 bytes $\times\ 6 \times 24 \times 365 = 4.2$ MB per year.

An authentication packet for a public key consists of the associated certificate (about 100 bytes), the tuple for proof of presence in CerBC, and the tuple for proof of absence in RevBC. Suppose there are n issued certificates and m revoked public keys, the total storage overhead of an authentication packet is $S = 100$ bytes $+$ 32 bytes $\times\, log2^n + (32$ bytes $+\, 8$ bytes$) \times log2^m$. After authentication, only the valid public keys (8 bytes for each) will be stored for the future communication.

As mentioned before, alert messages are broadcast only when an emergency happens. The main source of transmission overhead is the authentication packets and beacon messages (about 100 bytes for each). Suppose that vehicle A receives i authentication packets per second from nearby vehicles, in which there are

j new public keys. A will automatically discard the rest of authentication packets if the public key is in the list of authenticated vehicles. Thus, the total transmission overhead is $Tran = 100 \times (i - j) + S \times j$ bytes per second. If we set $i = 100, j = i \times 10\%, n = 1,000,000, m = n \times 10\%$, the total transmission overhead is about $0.177M/s$. The result shows that the storage and transmission overhead of anonymous authentication is acceptable.

Computation Overhead: The proof of presence and proof of absence are based on SHA-256 and can be done in time and space $O(logn)$ (n is the number of elements in the Merkle tree). Theoretically, the time consumption to authenticate one public key is $T = t_1 \times (log2^n + log2^m)$. We assume that $m = n \times 10\%$,i.e. 10% of the public keys are revoked. Figure 5.7 illustrates the time consumption of anonymous authentication in different scales of VANETs. The logarithmic-size proofs of presence and absence provide efficient authentication in large scale network.

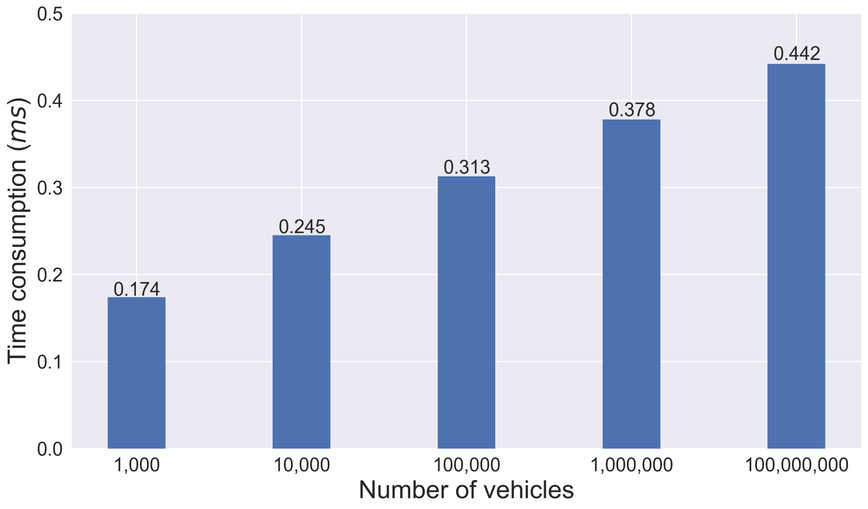

Figure 5.7: Time consumption of authentication. The time increases with the number of vehicles of the VANETs

5.6 Summary

In this chapter, we address the issues of trust and privacy in VANETs. In order to prevent the distribution of forged messages from authenticated vehicles meanwhile protecting the identity privacy of vehicles, a blockchain-based anonymous reputation system (BARS) is proposed for anonymous authentication and trust communication for VANETs. Vehicles use two blockchains (CerBC and RevBC) for authentication, which is based on proofs of presence and absence. Additionally, public keys act as the pseudonyms for anonymous communication and the linkability between real identity and the public key is broken to protect vehicles' privacy. On the other hand, all the broadcast messages are recorded in MesBC as persistent evidence to evaluate each vehicle's reputation. A reputation management algorithm is designed for trust communication to prevent the spread of forged messages and incent vehicles to expose misbehavior. Finally, we analyze the security and validity of BARS and evaluate the performance. The results show that BARS effectively improve the trustworthiness of broadcast messages and protect vehicle privacy with high efficiency.

Chapter 6: Conclusion and Future Work

6.1 Conclusion

In this dissertation, we proposed solutions to deal with the emerging security concerns on vehicles. Specifically, we started from the in-vehicle network bus communications and proposed two IDSs correspondingly to protect the in-vehicle CAN bus by hardware-dependent techniques. Those two IDSs combined could discover both content changes on the messages identifiers and the physical voltage changes observed on the bus.

Then we have investigated a commercial key agreement system for CAN bus and proposed physical side-channel attacks to break this key sharing protocol efficiently. Meanwhile, we have demonstrated that the in-vehicle information could be applied to detect spoofing on the GPS signal which is critical to the localization system of VANETs. At last, we have established a trust management mechanism among vehicles while protecting the privacy meantime which could be realized by blockchain techniques. The details in conclusion for each aspect are summarized as follows.

- We first developed a novel IDS scheme relied on checking the entropy changes of the CAN ID, which can be used to alert the CAN bus system when under attacks. We implemented the proposed IDS and validated its effectiveness on real bus data collected from a Ford Fusions internal CAN network through the OBD-II port. Our approach does not require modification on the message contents or frame structures. Thus, it is applicable to any implementation of CAN protocols, including CAN-FD. Besides, our proposed IDS does not require much computation overhead which could be implemented on the ECU.

- We then proposed an idea to fingerprint the ECUs on bus for solving those impersonate attacks. Our proposed scheme overcomes the blind spots of content-based IDS by monitoring the physical layer and is feasible to be deployed on a CAN network by adding minimal hardware to the existing CAN bus. We have demonstrated our proposed idea on CAN-bus prototype built on Arduino micro-controllers.

- We have demonstrated that several physical features including voltages, timing and so on can be used to probe the PnS-CAN scheme. We identify the fundamental characteristics that lead to such attacks and propose countermeasures to minimize the information leakage at the hardware level.

- We have demonstrated that a low-cost method to validate the GPS signal on the edge using the in-vehicle dataset. Specifically, we proposed two methods (i.e., the Kinematic model and Regression model) respectively to reconstruct the GPS signal from the driving maneuvers and we also take use of the off-line map information to further calibrate the GPS signals.

- We addressed the issue of trust and privacy in VANETs by proposing the BARS: A Blockchain-based Anonymous Reputation System. Two blockchains, CerBC and RevBC, make the activities of authority transparent for all entities in the VANETs. The blockchain implemented proof of presence and proof of absence provide anonymous authentication with high efficiency. Moreover, BARS also provided an effective reputation evaluation algorithm as an incentive for the vehicles participating in the trust management system.

6.2 Future Work

We also envision the implications and possible future directions of this dissertation work.

6.2.1 Security of CAN

6.2.1.1 Futher Discussions on Attacks and Countermeasures

The goal of the security discussion in this dissertation on CAN bus is twofold. Firstly, it demonstrates that even simple physical features can be utilized to compromise the CAN bus system, i.e., the PnS-CAN scheme. Secondly, it illustrates that detection techniques can be applied by monitoring the changes to protect the CAN bus from attacks. We emphasize that neither of these investigations is intended to represent the comprehensive attack vectors or defense mechanisms.

For example, given the robustness of the CAN protocol, one can imagine that there are a series of fundamental component variations that can be utilized to add noise to the characteristics extracted which may influence the detection rate of the IDS. However, future results in classification and learning theory can improve the performance of the detection method beyond what has been discussed. As a result, several open questions remain with the IDS discussed above. Further, improved detection countermeasures can be developed by digging and combining more physical features. While we have studied certain empirical dependencies, a comprehensive analysis can yield better insight into the identification of dominant physical characteristics. For our attacks and countermeasures, we utilized a simple laboratory set-up to emulate the CAN system. A real deployment of the system is subject to far more noisy conditions and many subtle electrical effects, e.g. transmission line phenomenon, that is not accurate in our setup.

Moreover, there are fundamental trade-offs that can be investigated between the IDS and the efficiency of the communications on the bus. For example, the addition of monitors or IDS for detection, while maintaining featured samples between the farthest nodes may require the operation of the system at a lower speed. However, this will decrease the bus efficiency also increase the observability of the adversary.

6.2.1.2 Immigration from CAN to Ethernet

Except for the security discussion around CAN bus, there is an aroused debate to replace the CAN network by the Ethernet, which has higher bandwidth, more flexible and already has security protocols (i.e., the MAC code) [20]. The motivation for this replacement is first intrigued by the communication and bandwidth requirements increase as more and more new and complex applications appear in the car for the use-cases as autonomous driving and entertainments. However, not only for the bandwidth consideration, existing vehicle control networks like the LIN, CAN and FlexRay standards are not designed to cover these increasing demands in terms of security. However, it is non-trivial to push the transfer as future networking technology should re-use as much as possible from the previous deployment while taking into account the automotive-specific requirements [115]. This includes hardware components as well as software stacks updated, which require huge efforts. Specifically, in-vehicle networking architecture appears as a heterogeneous system as a result of its historically grown nature. The usage of Ethernet in the car means a paradigm shift in the design of next-generation in-vehicle networking systems: connecting different domain networks, transporting different kinds of data (control data, streaming, etc.) and fulfilling the stringent robustness demands in terms of extended temperature range and EMC performance. It is necessary to decrease the heterogeneity of today's in-vehicle communication systems, that consist of several CAN, FlexRay or MOST busses connected via gateways, by aggregating several systems into one Ethernet-based communication network [116].

Come back to our proposed IDS schemes for the in-vehicle network, it is curious to find out the transferable property of them from the CAN to the Ethernet. For the content-based IDS by checking the 11-bit identifier, which takes use of the unique characteristics of the CAN bus arbitration, it is not directly applicable on the Ethernet. However, the general idea about the content-based IDS may also be applied with the Ethernet frame, but need to investigate the details of the Ethernet format. For the physical-based IDS, no matter what kind of protocols are used for the in-vehicle network, the physical probability would always exist and could be utilized. Thus, the applicability of physical IDS could immigrate to the Ethernet protocol as well. The only concerns left for this open question is the method to analyze the physical feature extracted may need modification due to a different arbitration protocol.

6.2.2 Cross-check Validation among Sensor Data

Various kinds of sensors are installed on the modern vehicle to guarantee the functionality and automation, as vehicles heavily rely on their sensor readings. Attackers can compromise one or more of the in-vehicle network sensors by changing the sensor readings to confuse the driver. Hence, anomalous sensor readings caused by either malicious cyber-attacks or faulty malfunction of sensors can result in disruptive consequences, and possibly lead to fatal crashes. Redundancy can be used to address the compromised sensors but adding extra sensors will increase the cost per vehicle and is therefore less attractive.

To balance the need for security and cost-efficiency, the natural redundancy in the sensor readings could be used to tackle the attacks on the sensor. By definition, natural redundancy occurs when the same physical phenomenon causes symptoms in multiple sensors. For instance, pressing the accelerator pedal will cause the engine to pump faster and increase the speed of the vehicle simultaneously. Engine revolutions per minute (RPM) and vehicle speed are multiple sensors which respond in a related fashion to the same cause of the accelerator pedal. The challenge is identifying the relationship between similar but different sensors under normal operation and detecting anomalous behavior accurately. It is similar to the cross-check method we proposed for detecting the GPS spoofing. We begin with the observation that sensors within the vehicle are naturally correlated.

As stated before, falsified sensor values have a significant impact on the safety and security of the vehicle as it can confuse the driver, and even cause the vehicle to misbehave. Several gaps are apparent in the literature. First, most of the detection method focuses on the pairwise correlation of two signals [117], while ignoring the other facts from the other signals as well. Some correlations between variables are evident as the fundamental similarity. For example, vehicle speed from the inertial measurement unit (IMU) sensor and from the GPS sensor are identical speed calculation. It is almost the same as using redundant sensors [118]. However, a lack of studying the other correlated values may give the chance for colluding sensor attacks. Second, most of the validation work focuses on the correlation at a specific time, without taking into the previous sensor value for consideration. The previous value might also correlate to the present value. To correlate the lag window data with the current value could also provide more evidence when detecting the anomaly. Last, to the best of our knowledge, there is a lack of deep learning implementations in anomaly detection and identification for multiple variants. Data are becoming more readily available and deep learning models are renowned for their performance using large datasets.

Here, we present a similar cross-check system as the GPS spoofing detection which targets on cross-checking the readings of sensors. In our approach, we proposed to use regression models to integrate correlations from multiple sensors and estimate a targeted sensor value using other correlated parameters in real time. The difference between the estimated value and observed value of this sensor data is used as a signal for detecting anomalies. First, we use a VAR model as a regression learning approach which estimates certain parameters by using correlated/redundant data. The estimated values are compared to observe ones to identify abnormal contexts that would indicate an intrusion. Then we try to use a deep learning technique called long short-term memory network (LSTM) to detect and identify anomalous behaviors of sensors in the vehicle. In this approach, a deep recurrent network model is trained to integrate correlations from multiple sensors with a time window and estimate a targeted sensor value later for anomaly detection. The integration of multiple heterogeneous sensors increases the attacker's difficulty of bypassing this detection scheme, while making the detection more stable under different scenarios. In addition, the major components (e.g., anomaly detection module) are embedded in edge computing devices, which make the anomaly detection be more efficient and privacy-preserving.

In order to detect anomalies, it is first necessary to estimate a targeted sensor value using other correlated parameters. It can be formulated as a prediction problem and modeled using regression models in machine learning. In the training phase, a set of correlated signals values are used to train a regression model M. Given a trained regression model M and a set of time-series parameters at time t, expressed as $V(t) = v_1(t), v_2(t), \cdots v_n(t)$ (e.g., v_1 = speed, v_2 = acceleration, v_3 = engine speed) extracted from the CAN bus messages. After learning the relationship among the multi-variable, we can predict the upcoming variable $V(t_m)'$ at time t_m. For the n variables, we select the targeted value for forecast as $v_i(t_m)'$ which is being estimated (e.g, speed). Meantime, the real measurement from the sensor is recorded as $v_i(t_m)$. Then anomalies can be flagged by comparing $v_i(t_m)'$ with observed $v_i(t_m)$.

We propose to perform correlation modeling among continuous time-series variables from Table 6.1. The variables are divided into two classes sensors within the vehicle which are broadcast on the CAN bus, and sensors from an external GPS system. Correlating both internal and external sensors gives us additional redundancy and robustness of the system. In order to successfully fool the system, the attacker has to compromise both internal and external systems, thus increasing the difficulty for a successful attack.

Name	Range	Frequency
steering_wheel_angle	-600 to +600 degrees	10 Hz
torque_at_transmission	-500 to 1500 Nm	10 Hz
engine_speed	0 to 16382 RPM	10 Hz
vehicle_speed	0 to 655 km/h	10 Hz
accelerator_pedal_position	percentage	10 Hz
odometer	0 to 16777214.000 km	10 Hz
fuel_consumed_since_restart	0 - 4294967295.0 L	10 Hz
latitude	-89.0 to 89.0 degrees	1 Hz
longitude	-179.0 to 179.0 degrees	1 Hz

Table 6.1: Time-series dataset

6.2.3 Efficient Blockchain System for Trust Management

For the trust management, we proposed in this dissertation, all the certificates and transactions are requiared to be recorded permanently and immutably on the blockchain to make the activities of the authorities transparent and verifiable. However, the experimental results show that to implement the anonymous authentication it inevitably adds the storage and transmission overhead for the VANETs. Therefore, it remains a challenge on how to use such blockchain effectively for authentication in real driving scenarios (e.g., high speed or a large number of messages during congestion) [119].

Another problem of the existing BARS system is the privacy-preserving suffers from several drawbacks. First, the activities of TA are not transparent to all the vehicles. TA may arbitrarily authorize some vehicles to join in the VANETs without being monitored. Second, the receivers need to query a Certificate Revocation prior to the message authentication to check whether the sender's public key has been revoked or not. With the rapid increase in the number of vehicles, this certificate updates requires a large amount of storage space and such query incurs high computation overhead. Those unsolved problems push the updates of the blockchain-based privacy-preserving authentication scheme to integrate the semi-trusted authorities with the emerging blockchain technology using carefully designed data structures. A novel data structure named the Merkle Patricia Tree (MPT) might be a future direction as it extends the conventional blockchain structure to provide a distributed authentication scheme without the revocation list. Meanwhile IBM's Hyperledger

Fabric (HLF) platform v1.1, which is a permissioned blockchain that securely tracks the execution history in an append-only replicated data structure without build-in cryptocurrency is a good candidate to implement the trust system on VANETs.

Appendix A: List of Publications

The following publications further encompasses the contribution of this dissertation work.

Journal Publications

1. **Qian Wang** and Gang Qu, "A Silicon PUF Based Entropy Pump ", *IEEE Transactions on Dependable and Secure Computing*, Volume 16, Issue 3, pp.402–414, 2018.

2. Zhaojun Lu, Wenchao Liu, **Qian Wang**, Gang Qu, and Zhenglin Liu, "A privacy-preserving trust model based on blockchain for vanets ", *IEEE Access*, Volume 6, pp.45655–45664, 2018.

3. An Wang, Yu Zhang, Weina Tian, **Qian Wang**, Guoshuang Zhang, and Liehuang Zhu, "Right or wrong collision rate analysis without profiling: full-automatic collision fault attack ", *Science China Information Sciences*, Volume 61, Issue 3, pp.032101, 2018.

4. Mingze Gao, **Qian Wang**, Md Tanvir Arafin, Yongqiang Lyu, and Gang Qu, "Novel Approximate Data Formats for Low Power and Security in the Internet

of Things ", *IEEE Computer*, Volume 50, Issue 6, pp.27–34, 2017.

5. **Qian Wang**, An Wang, Gang Qu, and Guoshuang Zhang, "New methods of template attack based on fault sensitivity analysis ", *IEEE transactions on multi-scale computing systems*, Volume 3, Issue 2, pp.113–123, 2017.

6. **Qian Wang**, An Wang, Liji Wu, and Jiliang Zhang, "A new zero value attack combined fault sensitivity analysis on masked AES ", *Microprocessors and Microsystems*, Volume 45, pp.355–362, 2016.

7. Zhaojun Lu, **Qian Wang**, Gang Qu, Haichun Zhang, and Zhenglin Liu, "A Blockchain-based Privacy-Preserving Authentication Scheme for VANETs ", *IEEE Transactions on VLSI Systems*, (accepted, to appear).

8. **Qian Wang**, Zhaojun Lu, Mingze Gao, and Gang Qu. "Edge Computing based GPS Signal Spoofing Detection in Vehicles ", *IEEE Transactions on Network Science and Engineering*, (under submission).

Conference Publications

1. **Qian Wang**, Mingze Gao, and Gang Qu, "PUF-PassSE: A PUF based Password Strength Enhancer for IoT Applications ", In Proceedings of the *20th International Symposium on Quality Electronic Design (ISQED)*, pp.198–203, 2019.

2. **Qian Wang**, Yiming Qian, Zhaojun Lu, Yasser Shoukry, and Gang Qu, "A Delay based Plug-in-Monitor for Intrusion Detection in Controller Area Net-

work ", In Proceedings of the *2018 Asian Hardware Oriented Security and Trust Symposium (AsianHOST)*, pp. 89–91, 2018.

3. Shalabh Jain, **Qian Wang**, Md Tanvir Arafin, and Jorge Guajardo, "Probing Attacks on Key Agreement for Automotive Controller Area Networks ", In Proceedings of the *2018 Asian Hardware Oriented Security and Trust Symposium (AsianHOST)*, pp. 7–12, 2018. (best paper award)

4. **Qian Wang**, Zhaojun Lu, Mingze Gao, and Gang Qu, "Edge Computing based GPS Spoofing Detection Methods ", In Proceedings of the *2018 IEEE 23rd International Conference on Digital Signal Processing (DSP)*, pp.1–5, 2018.

5. **Qian Wang**, Zhaojun Lu, and Gang Qu, "An entropy analysis based intrusion detection system for controller area network in vehicles ", In Proceedings of the *2018 31st IEEE International System-on-Chip Conference (SOCC)*, pp.90–95, 2018.

6. Zhaojun Lu, **Qian Wang**, Gang Qu, and Zhenglin Liu, "Bars: a blockchain-based anonymous reputation system for trust management in vanets ", In Proceedings of the *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pp.98–103, 2018.

7. **Qian Wang**, Mingze Gao, and Gang Qu, "A machine learning attack resistant

dual-mode PUF ", In Proceedings of the *2018 on Great Lakes Symposium on VLSI*, pp.177–182, 2018.

8. **Qian Wang**, Timothy Dunlap, Youngho Cho, and Gang Qu, "DoS attacks and countermeasures on network devices ", In Proceedings of the *2017 26th Wireless and Optical Communication Conference (WOCC)*, pp.1–6, 2017.

9. Mingze Gao, **Qian Wang**, Akshaya S Kankanhalli Nagendra, and Gang Qu, "A novel data format for approximate arithmetic computing ", In Proceedings of the *2017 22nd Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp.390–395, 2017.

10. **Qian Wang**, An Wang, Liji Wu, Gang Qu, and Guoshuang Zhang, "Template attack on masking AES based on fault sensitivity analysis ", In Proceedings of the *2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp.96–99, 2015.

11. Zhaojun Lu, **Qian Wang**, Xi Chen, Gang Qu, Yongqiang Lyu, and Zhenglin Liu, "LEAP: A Lightweight Encryption and Authentication Protocol for In-Vehicle Communications ", to appear in *2019 IEEE Intelligent Transportation Systems Conference.*

U.S. Patent and Invention Disclosure

1. Shalabh Jain, Qian Wang, Tanvir Arafin, Jorge Guajardo Merchan, **Method to Mitigate Transients Based Attacks on Key Agreement Schemes**

over Controller Area Network, Patent application number: 62/468,669.

2. Shalabh Jain, Qian Wang, Tanvir Arafin, Jorge Guajardo Merchan, **Method to Mitigate Voltage Based Attacks on Key Agreement over Controller Area Network**, Patent application number 62/468,705.

Bibliography

[1] Zhaojun Lu, Gang Qu, and Zhenglin Liu. A survey on recent advances in vehicular network security, trust, and privacy. *IEEE Transactions on Intelligent Transportation Systems*, 2018.

[2] Bala Deshpande. Predictive analytics in manufacturing: recycling value from telematics. `http://www.simafore.com/blog/bid/177716`, 2013. [Online, Accessed Dec 17, 2013].

[3] Karl Koscher, Alexei Czeskis, Franziska Roesner, Shwetak Patel, Tadayoshi Kohno, Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, et al. Experimental security analysis of a modern automobile. In *Security and Privacy (SP), 2010 IEEE Symposium on*, pages 447–462. IEEE, 2010.

[4] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, Tadayoshi Kohno, et al. Comprehensive experimental analyses of automotive attack surfaces. In *USENIX Security Symposium*. San Francisco, 2011.

[5] Charlie Miller and Chris Valasek. Adventures in automotive networks and control units. *DEF CON*, 21:260–264, 2013.

[6] Charlie Miller and Chris Valasek. Remote exploitation of an unaltered passenger vehicle. *Black Hat USA*, 2015, 2015.

[7] Mohamed Nidhal Mejri, Jalel Ben-Othman, and Mohamed Hamdi. Survey on vanet security challenges and possible cryptographic solutions. *Vehicular Communications*, 1(2):53–66, 2014.

[8] Mani Amoozadeh, Arun Raghuramu, Chen-Nee Chuah, Dipak Ghosal, H Michael Zhang, Jeff Rowe, and Karl Levitt. Security vulnerabilities of connected vehicle streams and their impact on cooperative driving. *IEEE Communications Magazine*, 53(6):126–132, 2015.

[9] Zhou Su, Yilong Hui, and Qing Yang. The next generation vehicular networks: A content-centric framework. *IEEE Wireless Communications*, 24(1):60–66, 2017.

[10] Huang Lu and Jie Li. Privacy-preserving authentication schemes for vehicular ad hoc networks: a survey. *Wireless Communications and Mobile Computing*, 16(6):643–655, 2016.

[11] Siam Umar Hussain and Farinaz Koushanfar. P3: Privacy preserving positioning for smart automotive systems. *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, 23(6):79, 2018.

[12] CAN Specification. Version 2.0. *Robert Bosch GmbH*, 1991.

[13] Karl Henrik Johansson, Martin Törngren, and Lars Nielsen. Vehicle applications of controller area network. In *Handbook of networked and embedded control systems*, pages 741–765. Springer, 2005.

[14] Nicolas Navet, Yeqiong Song, Francoise Simonot-Lion, and Cédric Wilwert. Trends in automotive communication systems. *Proceedings of the IEEE*, 93 (6):1204–1223, 2005.

[15] Mohammad Farsi, Karl Ratcliff, and Manuel Barbosa. An overview of controller area network. *Computing & Control Engineering Journal*, 10(3):113–120, 1999.

[16] Qian Wang, Zhaojun Lu, and Gang Qu. An entropy analysis based intrusion detection system for controller area network in vehicles. In *2018 31st IEEE International System-on-Chip Conference (SOCC)*, pages 90–95. IEEE, 2018.

[17] Stefan Nürnberger and Christian Rossow. –vatican–vetted, authenticated can bus. In *International Conference on Cryptographic Hardware and Embedded Systems*, pages 106–124. Springer, 2016.

[18] Chung-Wei Lin and Alberto Sangiovanni-Vincentelli. Cyber-security for the controller area network (can) communication protocol. In *Cyber Security (CyberSecurity), 2012 International Conference on*, pages 1–7. IEEE, 2012.

[19] Bogdan Groza and Stefan Murvay. Efficient protocols for secure broadcast in controller area networks. *IEEE Transactions on Industrial Informatics*, 9(4): 2034–2042, 2013.

[20] Chung-Wei Lin and Huafeng Yu. Cooperation or competition? coexistence of safety and security in next-generation ethernet-based automotive networks. In *Design Automation Conference (DAC), 2016 53nd ACM/EDAC/IEEE*, pages 1–6. IEEE, 2016.

[21] Shalabh Jain, Qian Wang, Md Tanvir Arafin, and Jorge Guajardo. Probing attacks on physical layer key agreement for automotive controller area networks. In *2018 Asian Hardware Oriented Security and Trust Symposium (AsianHOST)*, pages 7–12. IEEE, 2018.

[22] Kyong-Tak Cho and Kang G Shin. Error handling of in-vehicle networks makes them vulnerable. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1044–1055. ACM, 2016.

[23] Qian Wang, Yiming Qian, Zhaojun Lu, Yasser Shoukry, and Gang Qu. A delay based plug-in-monitor for intrusion detection in controller area network. In *2018 Asian Hardware Oriented Security and Trust Symposium (AsianHOST)*, pages 86–91. IEEE, 2018.

[24] Ricardo De Andrade, Kleber N Hodel, João Francisco Justo, Armando M Laganá, Max Mauro Santos, and Zonghua Gu. Analytical and experimental performance evaluations of can-fd bus. *IEEE Access*, 6:21287–21295, 2018.

[25] Muhammad Alam, Joaquim Ferreira, and José Fonseca. Introduction to intelligent transportation systems. In *Intelligent Transportation Systems*, pages 1–17. Springer, 2016.

[26] Amit Dua, Neeraj Kumar, and Seema Bawa. A systematic review on routing protocols for vehicular ad hoc networks. *Vehicular Communications*, 1(1): 33–52, 2014.

[27] Daniel Jiang and Luca Delgrossi. Ieee 802.11 p: Towards an international standard for wireless access in vehicular environments. In *VTC Spring 2008-IEEE Vehicular Technology Conference*, pages 2036–2040. IEEE, 2008.

[28] Maria Azees, Pandi Vijayakumar, and Lazarus Jegatha Deborah. Comprehensive survey on security services in vehicular ad-hoc networks. *IET Intelligent Transport Systems*, 10(6):379–388, 2016.

[29] Saif Al-Sultan, Moath M Al-Doori, Ali H Al-Bayatti, and Hussien Zedan. A comprehensive survey on vehicular ad hoc network. *Journal of network and computer applications*, 37:380–392, 2014.

[30] Anup Dhamgaye and Nekita Chavhan. Survey on security challenges in vanet 1. 2013.

[31] Richard Gilles Engoulou, Martine Bellaïche, Samuel Pierre, and Alejandro Quintero. Vanet security surveys. *Computer Communications*, 44:1–13, 2014.

[32] Yi Qian and Nader Moayeri. Design of secure and application-oriented vanets. In *VTC spring*, pages 2794–2799, 2008.

[33] Karan Verma, Halabi Hasbullah, and Ashok Kumar. An efficient defense method against udp spoofed flooding traffic of denial of service (dos) attacks in vanet. In *2013 3rd IEEE International Advance Computing Conference (IACC)*, pages 550–555. IEEE, 2013.

[34] Irshad Ahmed Sumra, Halabi Bin Hasbullah, and Jamalul-lail Bin AbManan. Attacks on security goals (confidentiality, integrity, availability) in vanet: a survey. In *Vehicular Ad-Hoc Networks for Smart Cities*, pages 51–61. Springer, 2015.

[35] Sherali Zeadally, Ray Hunt, Yuh-Shyan Chen, Angela Irwin, and Aamir Hassan. Vehicular ad hoc networks (vanets): status, results, and challenges. *Telecommunication Systems*, 50(4):217–241, 2012.

[36] Rongxing Lu, Xiaodong Lin, Tom H Luan, Xiaohui Liang, and Xuemin Shen. Pseudonym changing at social spots: An effective strategy for location privacy in vanets. *IEEE transactions on vehicular technology*, 61(1):86–96, 2012.

[37] Jyoti Grover, Vijay Laxmi, and Manoj Singh Gaur. Sybil attack detection in vanet using neighbouring vehicles. *International Journal of Security and Networks*, 9(4):222–233, 2014.

[38] Hengqing Wen, Peter Yih-Ru Huang, John Dyer, Andy Archinal, and John Fagan. Countermeasures for gps signal spoofing. In *ION GNSS*, volume 5, pages 13–16, 2005.

[39] Xiaodong Lin and Xu Li. Achieving efficient cooperative message authentication in vehicular ad hoc networks. *IEEE Transactions on Vehicular Technology*, 62(7):3339–3348, 2013.

[40] Ajay Rawat, Santosh Sharma, and Rama Sushil. Vanet: Security attacks and its possible solutions. *Journal of Information and Operations Management*, 3 (1):301, 2012.

[41] Mohammed Saeed Al-Kahtani. Survey on security attacks in vehicular ad hoc networks (vanets). In *2012 6th International Conference on Signal Processing and Communication Systems*, pages 1–9. IEEE, 2012.

[42] BG Premasudha, V RAVI RAM, J Miller, and R Suma. A review of security threats, solutions and trust management in vanets. *International Journal of Next-Generation Computing*, 7(1), 2016.

[43] Faisal Al-Hawi, Chan Yeob Yeun, and Mahmoud Al-Qutayti. Security challenges for emerging vanets. In *The 4th International Conference on Information Technology (ICIT09)*, pages 3–5, 2009.

[44] Dave Singelee and Bart Preneel. Location verification using secure distance bounding protocols. In *IEEE International Conference on Mobile Adhoc and Sensor Systems Conference, 2005.*, pages 7–pp. IEEE, 2005.

[45] Yu-Chih Wei and Yi-Ming Chen. Efficient self-organized trust management in location privacy enhanced vanets. In *International Workshop on Information Security Applications*, pages 328–344. Springer, 2012.

[46] Jie Zhang. Trust management for vanets: challenges, desired properties and future directions. *International Journal of Distributed Systems and Technologies (IJDST)*, 3(1):48–62, 2012.

[47] Stephan Eichler, Christoph Schroth, and Jörg Eberspächer. Car-to-car communication. 2006.

[48] Umar Farooq Minhas, Jie Zhang, Thomas Tran, and Robin Cohen. A multifaceted approach to modeling agent trust for effective communication in the application of mobile ad hoc vehicular networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 41(3): 407–420, 2011.

[49] Félix Gómez Mármol and Gregorio Martínez Pérez. Trip, a trust and reputation infrastructure-based proposal for vehicular ad hoc networks. *Journal of network and computer applications*, 35(3):934–941, 2012.

[50] Nadia Haddadou, Abderrezak Rachedi, and Yacine Ghamri-Doudane. A job market signaling scheme for incentive and trust management in vehicular ad hoc networks. *IEEE transactions on vehicular technology*, 64(8):3657–3674, 2015.

[51] Sashi Gurung, Dan Lin, Anna Squicciarini, and Elisa Bertino. Information-oriented trustworthiness evaluation in vehicular ad-hoc networks. In *International Conference on Network and System Security*, pages 94–108. Springer, 2013.

[52] Zhen Huang, Sushmita Ruj, Marcos A Cavenaghi, Milos Stojmenovic, and Amiya Nayak. A social network approach to trust management in vanets. *Peer-to-Peer Networking and Applications*, 7(3):229–242, 2014.

[53] Danda B Rawat, Gongjun Yan, Bhed Bahadur Bista, and Michele C Weigle. Trust on the security of wireless vehicular ad-hoc networking. *Ad Hoc & Sensor Wireless Networks*, 24(3-4):283–305, 2015.

[54] Rasheed Hussain, Waqas Nawaz, JooYoung Lee, Junggab Son, and Jung Taek Seo. A hybrid trust management framework for vehicular social networks. In *International Conference on Computational Social Networks*, pages 214–225. Springer, 2016.

[55] Merrihan Monir, Ayman Abdel-Hamid, and Mohammed Abd El Aziz. A categorized trust-based message reporting scheme for vanets. In *International Conference on Security of Information and Communication Networks*, pages 65–83. Springer, 2013.

[56] Zhaojun Lu, Wenchao Liu, Qian Wang, Gang Qu, and Zhenglin Liu. A privacy-preserving trust model based on blockchain for vanets. *IEEE Access*, 2018.

[57] Kyusuk Han, André Weimerskirch, and Kang G Shin. Automotive cybersecurity for in-vehicle communication. In *IQT QUARTERLY*, volume 6, pages 22–25, 2014.

[58] Weiying Zeng, Mohammed AS Khalid, and Sazzadur Chowdhury. In-vehicle networks outlook: Achievements and challenges. *IEEE Communications Surveys & Tutorials*, 18(3):1552–1571, 2016.

[59] Zhaojun Lu, Qian Wang, Gang Qu, and Zhenglin Liu. Bars: a blockchain-based anonymous reputation system for trust management in vanets. In *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 98–103. IEEE, 2018.

[60] Samuel Woo, Hyo Jin Jo, and Dong Hoon Lee. A practical wireless attack on the connected car and security protocol for in-vehicle can. *IEEE Transactions on Intelligent Transportation Systems*, 16(2):993–1006, 2015.

[61] Jonathan Petit and Steven E Shladover. Potential cyberattacks on automated vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 16(2): 546–556, 2015.

[62] Anthony Van Herrewege, Dave Singelee, and Ingrid Verbauwhede. Canauth-a simple, backward compatible broadcast authentication protocol for can bus. In *ECRYPT Workshop on Lightweight Cryptography*, volume 2011, 2011.

[63] Michael Müter and Naim Asaj. Entropy-based anomaly detection for in-vehicle networks. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 1110–1115. IEEE, 2011.

[64] Hyun Min Song, Ha Rang Kim, and Huy Kang Kim. Intrusion detection system based on the analysis of time intervals of can messages for in-vehicle network. In *Information Networking (ICOIN), 2016 International Conference on*, pages 63–68. IEEE, 2016.

[65] Michael R Moore, Robert A Bridges, Frank L Combs, Michael S Starr, and Stacy J Prowell. Modeling inter-signal arrival times for accurate detection of can bus signal injection attacks: a data-driven approach to in-vehicle intrusion detection. In *Proceedings of the 12th Annual Conference on Cyber and Information Security Research*, page 11. ACM, 2017.

[66] Mirco Marchetti and Dario Stabili. Anomaly detection of can bus messages through analysis of id sequences. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 1577–1583. IEEE, 2017.

[67] Min-Joo Kang and Je-Won Kang. Intrusion detection system using deep neural network for in-vehicle network security. *PloS one*, 11(6):e0155781, 2016.

[68] Kyong-Tak Cho and Kang G Shin. Fingerprinting electronic control units for vehicle intrusion detection. In *USENIX Security Symposium*, pages 911–927, 2016.

[69] Kyong-Tak Cho and Kang G Shin. Viden: Attacker identification on in-vehicle networks. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1109–1123. ACM, 2017.

[70] Wonsuk Choi, Kyungho Joo, Hyo Jin Jo, Moon Chan Park, and Dong Hoon Lee. Voltageids: Low-level communication characteristics for automotive intrusion detection system. *IEEE Transactions on Information Forensics and Security*, 13(8):2114–2129, 2018.

[71] Tobias Hoppe, Stefan Kiltz, and Jana Dittmann. Security threats to automotive can networks–practical examples and selected short-term countermeasures. In *International Conference on Computer Safety, Reliability, and Security*, pages 235–248. Springer, 2008.

[72] Blesson Varghese, Nan Wang, Sakil Barbhuiya, Peter Kilpatrick, and Dimitrios S Nikolopoulos. Challenges and opportunities in edge computing. In *2016 IEEE International Conference on Smart Cloud (SmartCloud)*, pages 20–26. IEEE, 2016.

[73] Benjamin Glas, Jorge Guajardo, Hamit Hacioglu, Markus Ihle, Karsten Wehefritz, and Attila Yavuz. Signal-based automotive communication security and its interplay with safety requirements. In *Proceedings of Embedded Security in Cars Conference*. Citeseer, 2012.

[74] David Brown, Geoffrey Cooper, Ian Gilvarry, Anand Rajan, Alan Tatourian, Ramnath Venugopalan, David Wheeler, and Meiyuan Zhao. Automotive security best practices. *White Paper*, pages 1–17, 2015.

[75] Norbert Bißmeyer. Security in ecu production. *ETAS White Paper*, 2016.

[76] Andreas Mueller and Timo Lothspeich. Plug and secure communication for can. *CAN Newsletter*, pages 10–14, 2015.

[77] Shalabh Jain and Jorge Guajardo. Physical layer group key agreement for automotive controller area networks. In *International Conference on Cryptographic Hardware and Embedded Systems*, pages 85–105. Springer, 2016.

[78] Qian Wang, An Wang, Liji Wu, and Jiliang Zhang. A new zero value attack combined fault sensitivity analysis on masked aes. *Microprocessors and Microsystems*, 45:355–362, 2016.

[79] Qian Wang, An Wang, Gang Qu, and Guoshuang Zhang. New methods of template attack based on fault sensitivity analysis. *IEEE transactions on multi-scale computing systems*, 3(2):113–123, 2017.

[80] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[81] Pal-Stefan Murvay and Bogdan Groza. Source identification using signal characteristics in controller area networks. *IEEE Signal Processing Letters*, 21(4): 395–399, 2014.

[82] Roar Elias Georgsen. Machine learning based intrusion detection in controller area networks. B.S. thesis, 2016.

[83] Mark L Psiaki and Todd E Humphreys. Gnss spoofing and detection. *Proceedings of the IEEE*, 104(6):1258–1270, 2016.

[84] Brady W O'Hanlon, Mark L Psiaki, Jahshan A Bhatti, Daniel P Shepard, and Todd E Humphreys. Real-time gps spoofing detection via correlation of encrypted signals. *Navigation*, 60(4):267–278, 2013.

[85] Mark L Psiaki, Brady W O'Hanlon, Jahshan A Bhatti, Daniel P Shepard, and Todd E Humphreys. Civilian gps spoofing detection based on dualreceiver correlation of military signals. In *Radionavigation Laboratory Conference Proceedings*, 2011.

[86] Md Tanvir Arafin, Dhananjay Anand, and Gang Qu. A low-cost gps spoofing detector design for internet of things (iot) applications. In *Proceedings of the on Great Lakes Symposium on VLSI 2017*, pages 161–166. ACM, 2017.

[87] Jung-Hoon Lee, Keum-Cheol Kwon, Dae-Sung An, and Duk-Sun Shim. Gps spoofing detection using accelerometers and performance analysis with probability of detection. *International Journal of Control, Automation and Systems*, 13(4):951–959, 2015.

[88] Siam U Hussain and Farinaz Koushanfar. Privacy preserving localization for smart automotive systems. In *Design Automation Conference (DAC), 2016 53nd ACM/EDAC/IEEE*, pages 1–6. IEEE, 2016.

[89] Nathaniel Carson, Scott M Martin, Joshua Starling, and David M Bevly. Gps spoofing detection and mitigation using cooperative adaptive cruise control system. In *Intelligent Vehicles Symposium (IV), 2016 IEEE*, pages 1091–1096. IEEE, 2016.

[90] Qian Wang, Zhaojun Lu, Mingze Gao, and Gang Qu. Edge computing based gps spoofing detection methods. In *2018 IEEE 23rd International Conference on Digital Signal Processing (DSP)*, pages 1–5. IEEE, 2018.

[91] Kai Jansen, Matthias Schäfer, Daniel Moser, Vincent Lenders, Christina Pöpper, and Jens Schmitt. Crowd-gps-sec: Leveraging crowdsourcing to detect and localize gps spoofing attacks. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 1018–1031. IEEE, 2018.

[92] Markus G Kuhn. Signal authentication in trusted satellite navigation receivers. In *Towards Hardware-Intrinsic Security*, pages 331–348. Springer, 2010.

[93] Todd E Humphreys, Brent M Ledvina, Mark L Psiaki, Brady W O'Hanlon, and Paul M Kintner. Assessing the spoofing threat: Development of a portable gps civilian spoofer. In *Radionavigation laboratory conference proceedings*, 2008.

[94] Todd E Humphreys. Detection strategy for cryptographic gnss anti-spoofing. *IEEE Transactions on Aerospace and Electronic Systems*, 49(2):1073–1090, 2013.

[95] Xichen Jiang, Jiangmeng Zhang, Brian J Harding, Jonathan J Makela, Alejandro D Domı, et al. Spoofing gps receiver clock offset of phasor measurement units. *IEEE Transactions on Power Systems*, 28(3):3253–3262, 2013.

[96] Kyle Wesson, Mark Rothlisberger, and Todd Humphreys. Practical cryptographic civil gps signal authentication. *Navigation: Journal of The Institute of Navigation*, 59(3):177–193, 2012.

[97] Xiang Sun and Nirwan Ansari. Edgeiot: Mobile edge computing for the internet of things. *IEEE Communications Magazine*, 54(12):22–29, 2016.

[98] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5):637–646, 2016.

[99] Weisong Shi and Schahram Dustdar. The promise of edge computing. *Computer*, 49(5):78–81, 2016.

[100] Yuyi Mao, Changsheng You, Jun Zhang, Kaibin Huang, and Khaled B Letaief. A survey on mobile edge computing: The communication perspective. *IEEE Communications Surveys & Tutorials*, 19(4):2322–2358, 2017.

[101] Nasir Abbas, Yan Zhang, Amir Taherkordi, and Tor Skeie. Mobile edge computing: A survey. *IEEE Internet of Things Journal*, 5(1):450–465, 2018.

[102] Alessandro De Luca, Giuseppe Oriolo, and Claude Samson. Feedback control of a nonholonomic car-like robot. In *Robot motion planning and control*, pages 171–253. Springer, 1998.

[103] Chaker Abdelaziz Kerrache, Carlos T Calafate, Juan-Carlos Cano, Nasreddine Lagraa, and Pietro Manzoni. Trust management for vehicular networks: An adversary-oriented overview. *IEEE Access*, 4:9293–9307, 2016.

[104] Satoshi Nakamoto et al. Bitcoin: A peer-to-peer electronic cash system. 2008.

[105] Madhusudan Singh and Shiho Kim. Intelligent vehicle-trust point: Reward based intelligent vehicle communication using blockchain. *arXiv preprint arXiv:1707.07442*, 2017.

[106] Marcella Atzori. Blockchain-based architectures for the internet of things: A survey. *Available at SSRN 2846810*, 2017.

[107] Arshdeep Bahga and Vijay K Madisetti. Blockchain platform for industrial internet of things. *Journal of Software Engineering and Applications*, 9(10): 533, 2016.

[108] Florian Tschorsch and Björn Scheuermann. Bitcoin and beyond: A technical survey on decentralized digital currencies. *IEEE Communications Surveys & Tutorials*, 18(3):2084–2123, 2016.

[109] Ali Dorri, Salil S Kanhere, and Raja Jurdak. Blockchain in internet of things: challenges and solutions. *arXiv preprint arXiv:1608.05187*, 2016.

[110] Ralph C Merkle. A digital signature based on a conventional encryption function. In *Conference on the theory and application of cryptographic techniques*, pages 369–378. Springer, 1987.

[111] Ben Laurie, Adam Langley, and Emilia Kasper. Certificate transparency. Technical report, 2013.

[112] Mark Dermot Ryan. Enhanced certificate transparency and end-to-end encrypted mail. In *NDSS*, pages 1–14, 2014.

[113] Ben Laurie and Emilia Kasper. Revocation transparency. *Google Research, September*, 2012.

[114] Vincent Cheval, Mark Ryan, and Jiangshan Yu. Dtki: a new formalized pki with no trusted parties. *arXiv preprint arXiv:1408.1023*, 2014.

[115] Till Steinbach, Franz Korf, and Thomas C Schmidt. Comparing time-triggered ethernet with flexray: An evaluation of competing approaches to real-time for in-vehicle networks. In *2010 IEEE International Workshop on Factory Communication Systems Proceedings*, pages 199–202. IEEE, 2010.

[116] Peter Hank, Thomas Suermann, and Steffen Müller. Automotive ethernet, a holistic approach for a next generation in-vehicle networking standard. In *Advanced Microsystems for Automotive Applications 2012*, pages 79–89. Springer, 2012.

[117] Arun Ganesan, Jayanthi Rao, and Kang Shin. Exploiting consistency among heterogeneous sensors for vehicle anomaly detection. Technical report, SAE Technical Paper, 2017.

[118] Huaxin Li, Li Zhao, Marcio Juliato, Shabbir Ahmed, Manoj R Sastry, and Lily L Yang. Poster: Intrusion detection system for in-vehicle networks using sensor correlation and integration. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 2531–2533. ACM, 2017.

[119] Zhaojun Lu, Qian Wang, Gang Qu, Haichun Zhang, and Zhenglin Liu. A blockchain-based privacy-preserving authentication scheme for vanets. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems.*