# A Robust, Reliable and Deployable Framework for In-vehicle Security

by

Azeem Hafeez

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical and Computer Engineering)
in the University of Michigan-Dearborn
2020

Doctoral Committee:

Professor Selim Awad
Assistant Professor Anys Bacha
Professor Hafiz Malik, Chair
Professor Adnan Shaout

ORCID iD 0000-0001-8855-6158

Dedicated to my family, friends and colleagues

# ACKNOWLEDGEMENTS

All is well that ends well, and since every beginning has an ending, my Ph.D. research ended in this semester i.e. Winter 2020, and it ended well. Special thanks to my amazing Ph.D. Supervisor, Sir Hafiz Malik, it could have never been possible without his guidance, assistance and all-time available help. Thanks to the people who made this dissertation possible, I would like to pay special thanks to Professor Paul Richardson who was a source of motivation for me throughout my degree, I don't have words to thank Professor Paul. I will also like to thank my colleagues from ECE office, Mariann Brevoort, Jesse Cross, Amanda Donovan and Michael Hicks who always supported me and celebrated my success as a family. I would also like to thank especially my family, my friends, my grandfather (Mohammad Baksh Multani) who motivated me to do Ph.D., my dissertation committee members, my father (Hafeez Ur Rehman) who always believed in me that I can achieve this goal, my uncle (Ashraf Chaudry) who encouraged me during my high school studies, my wife (Shangraf Malik) who provided me her full support during my graduate studies. Last but not least, I like to dedicate this to my two lovely sons Ibraheem and Moosa who provided me determination to achieve this goal.

# TABLE OF CONTENTS

# LIST OF FIGURES

**Figure**

# LIST OF TABLES

**Table**

# LIST OF ABBREVIATIONS

**AI** artificial intelligence

**ANN** artificial neural network

**CAN** controller area network

**CANL** CAN Low

**CANH** CAN High

**CIDS** clock based intrusion detection system

**CPS** cyber-physical system

**CRC** cyclic redundancy code

**DAC** digital to analog converter

**CIDS** clock based intrusion detection system

**DIDS** distortion based intrusion detection system

**ECU** electronic control unit

**ECUs** electronic control units

**EMI** electromagnetic interference

**FIR** finite impulse response

**FN** false negative

**FP** false positive

**HMI** human machine interface

**IDS** intrusion detection system

**IVNs** in-vehicle networks

**LEM** lumped element model

**LIN** local interconnect network

**ML** machine learning

**MOST** media oriented system transport

**PDF** probability density function

**PM** performance matrix

**QoS** Quality of Service

**ROC** receiver operating characteristic

**SOA** state Of-the-art

**TN** true negative

**TP** true positive

# ABSTRACT

Cyber attacks on financial and government institutions, critical infrastructure, voting systems, businesses, modern vehicles, etc., are on the rise. Fully connected autonomous vehicles are more vulnerable than ever to hacking and data theft. This is due to the fact that the protocols used for in-vehicle communication i.e. controller area network (CAN), FlexRay, local interconnect network (LIN), etc., lack basic security features such as message authentication, which makes it vulnerable to a wide range of attacks including spoofing attacks. This research presents methods to protect the vehicle against spoofing attacks. The proposed methods exploit uniqueness in the electronic control unit electronic control unit (ECU) and the physical channel between transmitting and destination nodes for linking the received packet to the source. Impurities in the digital device, physical channel, imperfections in design, material, and length of the channel contribute to the uniqueness of artifacts. I propose novel techniques for electronic control unit (ECU) identification in this research to address security vulnerabilities of the in-vehicle communication. The reliable ECU-identification has the potential to prevent spoofing attacks launched over the CAN due to the inconsideration of the message authentication. In this regard, my techniques models the ECU-specific random distortion caused by the imperfections in digital-to-analog converter digital to analog converter (DAC), and semiconductor impurities in the transmitting ECU for fingerprinting. I also model the channel-specific random distortion, impurities in the physical channel, imperfections in design, material, and length of the channel are contributing factors behind physically unclonable artifacts. The lumped element model is used to characterize channel-specific distortions. This

research exploits the distortion of the device (ECU) and distortion due to the channel to identify the transmitter and hence authenticate the transmitter.

# CHAPTER I

# Introduction

The main objective of this chapter is to provide the introduction of the dissertation. Section 1.1 provides the motivation to implement security in modern electric vehicle, which is followed by the research objectives in section 1.2. Original contributions of this research are reported in section 1.3. Threat model is presented in section 1.4. Finally the complete outline of dissertation is provided in section 1.5.

## 1.1 Motivation

The modern vehicle is a cyber-physical system (CPS) equipped with many wireless and wired communication interfaces and a large number of microcontrollers and electronic control units (ECUs) networked via various in-vehicle networks (IVNs) (1; 2; 3; 4; 5; 6; 7; 8), such as CAN (8), LIN (1), media oriented system transport (MOST) (2), FlexRay (3), etc., that connect safety-critical components of the vehicle, including brakes, airbags, engine control, and active safety devices, e.g., electronic stability program, adaptive cruise control, and so on. Integration of wireless interfaces, e.g., Bluetooth, Wi-Fi, etc., with IVNs and use of the legacy CAN protocol for in-vehicle control communication pose serious security threats to connected autonomous vehicles (AVs) (9).

Advances in-vehicle technologies are unable to keep pace with the growing attack

surfaces and vectors, leaving millions of vehicles vulnerable to a wide range of attacks e.g. man-in-the-middle, and packet spoofing (10; 11; 12; 13). This is due to the fact that the automotive industry is still relying on the legacy controller area network (CAN) protocol for in-vehicle communication among ECUs. Whereas, CAN protocol lacks basic security features such as message authentication, confidentiality, and integrity, thus cause the CAN as an easy victim of attack through ECUs (14; 15; 16). The ECUs on the in-vehcile network therefore are vulnerable to various attacks, including packet spoofing attacks, data injection attacks, denial of service (DoS) attacks etc., that can defect the vehicle. Recently, researchers have successfully hijacked vehicles from a remote location, and killed the vehicle engine in middle of a highway (16).

Researchers have proposed various solutions to detect and prevent attacks on the CAN protocol for in-vehicle control networks. These methods can be classified into two categories: (i) message authentication code (MAC)-based approaches (17; 18; 19; 20; 21; 22; 23; 24), and (ii) intrusion detection-based approaches (25; 26; 27; 28; 29; 30; 31; 32; 33; 34; 35; 36; 37; 38; 39; 40; 41; 42; 43; 44; 45; 46).

The MAC-based methods, achieve security and privacy by encrypting the payload of the CAN-packet before transmission. For instance, in (17), Wang et. al. demonstrated a MAC-based framework VeCure for CAN security. In VeCure, 64-bit MAC for every 64-bit message was transmitted between the ECUs. Intuitively, the method exhibited high computational cost, 50% additional transmission overhead, and also required a higher data rate. In (21), Hiroshi et. al. designed an authentication mechanism for the CAN protocol against spoofing attacks. The monitoring node provided the authentication code for all ECUs and verified the code for all CAN messages. In (19), Hazem et. al. proposed a lightweight CAN authentication protocol (LCAP). The LCAP required to append a "magic number" that was generated by a one-way hash function employed on TESLA prototype (47). The protocol required 16-bits of

the data field to append the authentication code, which still creates 25% overhead. The MAC-based approaches have the intrinsic overhead that lowers the transmission performance, hence makes them unreliable for the CAN security (17; 46; 23).

To address the limitations of MAC-based solutions, researchers have proposed intrusion detection-based approaches for CAN network traffic analysis (26; 27; 21; 19; 48). The intrusion detection-based approaches have lower data rate requirements because no additional bits are transmitted during the message transmission; hence, it does not add additional network overhead. In (26), Cho and Shin demonstrated a clock-based intrusion detection system (CIDS) that used ECU fingerprinting. Each ECU contained a crystal oscillator known as a clock; the ECU fingerprinting measured the clock skewness against the received packets and detect the attack. However, Sang et. al. (28) and Tayyab et. al. (49) demonstrated that CIDS can be bypassed by estimating the clock parameters. In (46), message authentication was performed through ECU detection by applying higher-order moments of the CAN signal in both times- and frequency-domains. However, the approach was intolerant against the transmitter induction and the performance of the system seriously decays if the number of transmitters is increased. Therefore, I need an IDS-based approach that extracts unique fingerprints from the signal, works for a higher number of transmitters, and also exhibits low computational complexity.

To address the aforementioned limitations of existing in-vehicle security techniques, novel IDS-based message authentication approaches are presented in this research. My approach exploits the two types of distortions.

- Device-specific distortion
- Channel-specific distortion

### 1.1.1 Device-specific Distortion

In the first part of this research, my approach exploits the uniqueness in device-specific distortions e.g., semiconductor impurities, DC offset, aliasing error, the mismatch between the nominal and measured values of electric components in DAC, etc., for message fingerprint generation. I hypothesize that distortions due to digital-to-analog conversion operation at the ECU output are device-dependent that can be used to link the received packet to the transmitting ECU. Therefore, I associate the received packet through a specific ECU, and the ECU-pin responsible for message transmission.

### 1.1.2 Channel-specific Distortion

In the second part of this research, my approach exploits the uniqueness in channel-specific distortions. I also hypothesize that the distortion in the signal behaves like a unique signature, which can be used to link the signal to the channel and hence the transmitter. The proposed research exploits uniqueness in the channel-specific distortion for linking the received CAN packet to the transmitting source. The impurities in the physical channel, imperfections in design, material, and length of the channel contribute to the uniqueness. The lumped element model (LEM) for transmission lines is used to characterize channel artifacts.

## 1.2 Research Objectives

There are 2 main objectives of this research:

### 1.2.1 ECU and Pin Level Uniqueness

The first objective of this research is: (i) to investigate ECU-level uniqueness for a given network and (ii) to investigate pin-level uniqueness for a given ECU to authen-

ticate the message. The proposed method relies on distinctive physical artifacts of the $DAC$ of the transmitting ECU for device-level fingerprinting. The imperfections in material, design, fabrication of $DAC$ are contributing factors that create distortion in the ECU signal. I perform the statistical modeling of this distortion and use it as a feature vector for transmitter identification (i.e. transmitting ECU, and ECU-pin) through neural network architecture.

### 1.2.2 Channel Level Uniqueness

The second objective of this research is to investigate channel-level uniqueness for a given network. The proposed method relies on distinctive physical artifacts of the channel, for channel fingerprinting. Material and design imperfections in the physical channel and length of channel are the main contributing factors behind the channel-specific unique artifacts, are leveraged to link the received electrical signal to the transmitter. I performed the statistical modeling of this distortion, then there are two approaches used for channel identification.

- Non-parametric approach (Neural network based approach)
- Parametric approach (Maximum Likelihood Ratio Test)

## 1.3 Contributions

Thus, the main contributions of this dissertation are:

- I provide a mathematical model of the distortion sources i.e. imperfections in the material, design, fabrication of $DAC$.
- I propose a statistical model of the device (ECU)-level distortion for transmitter identification.

- I also propose that different transmitting pins in a single device have unique distortion and can be used for ECU-pin identification.
- I provide a mathematical model of the distortion sources i.e. manufacturing imperfections, design imperfections of channel
- Model the channel-specific distortion for CAN channel.
- Channel-specific distortion extraction and its uniqueness analysis
- Propose a reliable framework neural network based approach for linking received CAN packet to the true transmitting source.
- Propose a parametric approach which estimates multiple density functions of distortion, and adopt the best fit density function to identify the source transmitter. In my case, the gamma distribution function is the best fit density function that is used to compute the $\alpha$ and $\beta$ parameters of the gamma distribution. Through empirical analysis, I observed that the parameters $\alpha$ and $\beta$ have random nature thus can also be represented through the gamma function, which in this case is the Gaussian function. Afterward, for a test signal, $\alpha$ and $\beta$ are computed, that are then used to identify the channel for message authentication and attack prevention through a novel modified likelihood approach.

## 1.4 Threat Model

In the threat model, I have two types of threats. The first threat is from physical access, and the second threat is from wireless access as shown in Fig. 1.1. The physical access threat means that the adversary physically removes one of the ECUs from the vehicle and connects it's own ECU for injecting malicious messages. The second threat is from wireless access in which the adversary injects a message in the CAN using a wireless interface like radio, Wifi, vehicle to vehicle (V2V) communication etc. This

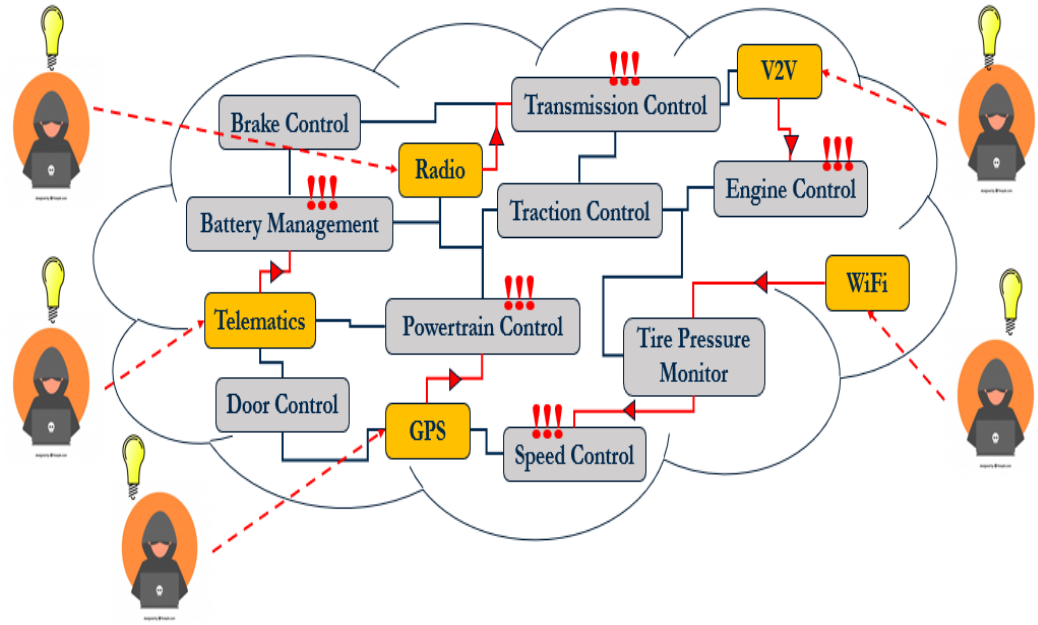


Figure 1.1: Threat model for in-vehicle communication

research is robust against ECU impersonation attacks that are launched through the physical access of the car e.g. the adversary attacks the CAN through physical access by deploying a new ECU. Since the distortion in the signal is dependent on *DAC* and material imperfections, the feature vector estimated will be different as well, the message will not be authenticated and attack will be identified. Moreover, my approach is also effective for attacks launched through wireless interface e.g. attack launched on the infotainment system to access CAN as done by Miller et. al. (50).

Given a vehicle network as shown in Fig. 1.2, suppose that an adversary tries to penetrate the vehicle network through the wireless interface of the infotainment system with the help of a CAN message, which is for the braking system. In CAN message, the information about the sender is missing, as the messages are functionality-based. However, the fingerprinting ECU will correctly recognize the ECU, it will not authenticate this message as it is coming from the wrong sender, and it will send a warning signal to braking unit ECU. Now, the ECU braking unit knows not to apply the brakes since the CAN message for applying brakes is not supposed to come from

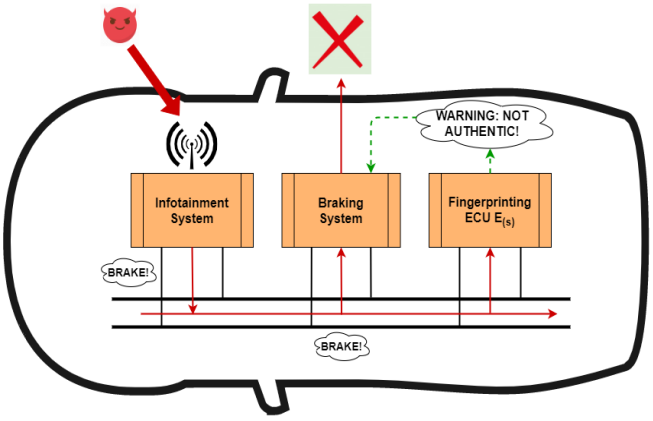


Figure 1.2: Attack detection for in-vehicle communication

the infotainment system.

## 1.5 Dissertation Outline

The rest of this dissertation is organized into nine chapters. The brief summary of each of these chapters is as follows:

For in-vehicle communication, different protocols i.e. CAN, LIN, Flexray et. are used. The security systems proposed in this research are protocol independent, as physical signal is used for fingerprinting ECUs. In Chapter-II, an overview and comparison of CAN, Flexray and LIN protocols is presented.

Intrusion detection based approaches are further subdivided into: (a) parameter monitoring based approaches (b) information theory based approaches (c) machine learning based approaches and (d) fingerprinting based approaches. Chapter-III outlines the related work to my research.

Chapter-IV provides a novel technique for electronic control unit (ECU) identification based on ECU distortion address security vulnerabilities of the controller area network (CAN) protocol. In this regard, my technique models the ECU-specific random distortion caused by the imperfections in digital-to-analog converter, and semiconductor impurities in the transmitting ECU for fingerprinting.

Chapter-V is about using a the lumped element model to characterize the channel-specific step response. ECU and channel imperfections lead to a unique transfer function for each transmitter. Due to the unique transfer function, the step response for each transmitter is unique. In this section, control system parameters are used as a feature-set, afterward, a neural network is used transmitting node identification for message authentication.

Chapter-VI proposes a method exploits physical unclonable attributes in the physical channel between transmitting and destination nodes and uses them for linking the received packet to the source. Non-parametric modeling is used to estimate distortion distribution, which is used for transmitting node identification. A neural network is trained to identify the channel and hence transmitter.

Chapter-VII also exploits channel distortion for message authentication, a novel, computationally efficient parametric approach is developed to quantify the distortion in form of probability density function (PDF). The best fit PDF over histogram of distortion, is gamma distribution function. Afterwards, the $\alpha$ and $\beta$ parameters of gamma distribution function are computed for multiple records in each channel to obtain the joint density function. Finally, the likelihood ratio test is applied on joint density function to identify the channel and transmitter.

Chapter-VIII concludes this dissertation and describe the future perspectives of the research.

# CHAPTER II

# Overview of CAN, FlexRay and LIN Protocol

CAN, FlexRay and LIN protocols are commonly used for modern in-vehicle communication. A modern vehicle contains 70 to 100 ECUs. This chapter presents a literature review of these protocols. The communication cycle, process, message structure, and hardware elements are discussed for all three protocols. Performance is measured in terms of reliability and latency. In addition, a comparison between the CAN, FlexRay, and LIN protocols is made. Study shows that CAN protocol has advantages when it comes to real-time priority-based communication. However, if all the events have equal priority, then FlexRay works well. The LIN protocol is budget-friendly and has the lowest cost in all 3 protocols but at the same time, it is unreliable.

The rapid advancement in the automotive industry increases the demand for developing remarkably efficient communication techniques in order to provide higher bandwidth for better data rates. To examine this issue, the most recent interfacing systems for in-vehicle networking are introduced in this comparative study. The discussed methods are represented in CAN, FlexRay, and LIN protocols. Each protocol is suitable for a set of specific applications that perform a certain group of purposes and combined will form a complete communication network that connects the intra-vehicle control units in one fully functional system.

The distinctive and quick development in digital and computer hardware technology inspired the automotive world to advance in communication and networking for the in-vehicle network area. As a result of the technological advancement in communication systems, electronic systems in vehicles are becoming more diverse and complex structurally and functionally to cope with the new innovations, providing further convenient and efficient services for drivers to confidently monitor the internal and the external performances of the system. Ultrasonic sensors, radar, and cameras are utilized to control the internal performance of the vehicle (51; 52; 53). To maintain a high level of safety to drivers, passengers, and pedestrians, the systems are particularly constructed to monitor the surrounding environment of the vehicle (51). Furthermore, the infotainment system is being added due to its importance in connectivity such as Bluetooth, navigation, smartphones, and audio system (51; 52; 53).

Data traffic, latency, and jitter in the network may cause a delay in services. However, this issue can be controlled, and the bandwidth can also be maximized through the Quality of Service (QoS) measurements. This can vary with different functions of the in-vehicle communication, which can be mainly classified into time-dependent and independent functions, or real-time and non-real time functions for controlling message transmission between ECUs (51; 54). Examples of real-time applications, which are represented by using CAN, LIN, and FlexRay protocols, are the braking, steering, engine control, transmission, multimedia, human machine interface (HMI), telematics audio, and seating systems (51; 52; 53; 55; 54). One the other hand, FlexRay protocol is also considered more of a hybrid system technology that can be a time-triggered or an event-triggered protocol (51; 55; 54). All automotive networking methods are dedicated to communicating signals effectively among the control units to improve message transmission safety, lowering the cost of cabling, maximizing bandwidth, and saving package space (51; 56; 57).

The overview and a comparative study are discussed in detail. This chapter is

organized in the following fashion: Section 2.1 is an overview of CAN. Section 2.2 is an overview of FlexRay. Section 2.3 is an overview of LIN. Section 2.4 is a comparison study between CAN vs. FlexRay, LIN vs. CAN, and LIN vs. FlexRay.

## 2.1 CAN- an Overview

In the 1980s, the CAN bus protocol was proposed by Robert Bosch GmbH (58). This robust system has been broadly applicable in communication and networking areas such as automotive networking systems, medical appliances, entertainment realm, and domestic apparatus (58). In the pre-CAN bus era, the message exchange for point-to-point communication systems was mainly based on source and destination addresses. After the CAN bus system was introduced, broadcast communication was utilized, where every node via the bus can transmit and receive message. Unlike the traditional method, the CAN bus system added more flexibility to the networking technology. This makes adding further nodes to the network more possible and convenient, and it will not affect the main structure of the network system and the existing nodes as well. CAN bus uses a multi-master configuration (23) which allows the transmission or reception of the messages via any node when the bus is free. This also guarantees a fixed delay duration. CAN bus is an event-triggered protocol (23) in which the message is initiated as a response to a triggered event or request in the network (23; 46).

As a real-time system, for every message on CAN bus there is a unique priority depending on the message identifier (ID) that exists in each message frame (23; 42; 59; 49). If the message has the lowest ID, it will have the highest priority to pass through the bus and vice versa (46). Furthermore, via the message arbitration technique, the bus collision is avoided due to a message prioritization feature, where nodes on the CAN bus send messages consecutively and the message with the lowest identification value will have the priority to transmit on the bus (42; 46). CAN bus

utilizes broadcasting communication topology for sending messages by using differential signaling mode at the physical layer, which is represented by two communication twisted pair wires, CAN High (CANH) and CAN Low (CANL) (42; 46). The signal transmission methodology of CAN bus is represented by 1s, recessive bits, and 0s, dominant bits. In idle mode, where the recessive bit (1s) is transmitted, CANH and CANL are set to 2.5 volts, causing the voltage between the wires to be zero (42; 46). When the dominant bits (0s) is transmitted, CANH rises to 3.5 volts and CANL goes to 1.5 volts, where the voltage difference between the two wires is becoming 2 volts (42; 46; 60). CAN bus topology is illustrated in Fig. 2.1 (46). For example, three nodes in binary (23) (Node 1: 11001011111, Node 2:110011111111, and Node 3: 110010110010) attempt to transmit messages simultaneously. However, the message which is assigned the lowest identity (third node) will broadcast the information first. Hence, the bus collision is prevented. The scenario above is illustrated in Fig. 2.2, (23).

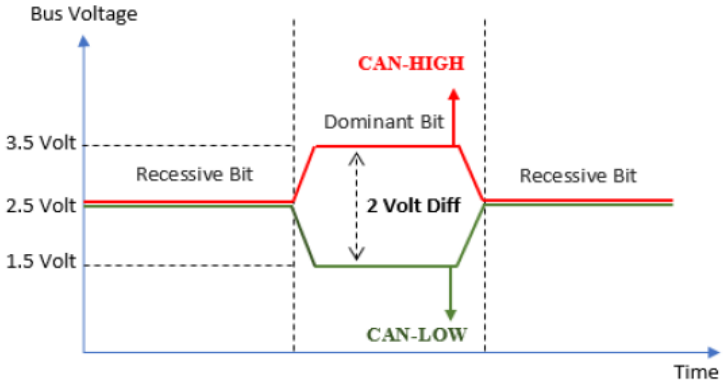


Figure 2.1: CAN signalling mode

There are two common CAN-bus formats which are the Standard format, where an 11-bit Identifier frame is used, and the Extended format, which implements a 29-bit Identifier frame (61; 49; 46). Other major frames include Data Frame, Remote Frame, Overload Frame, and Error Frame.

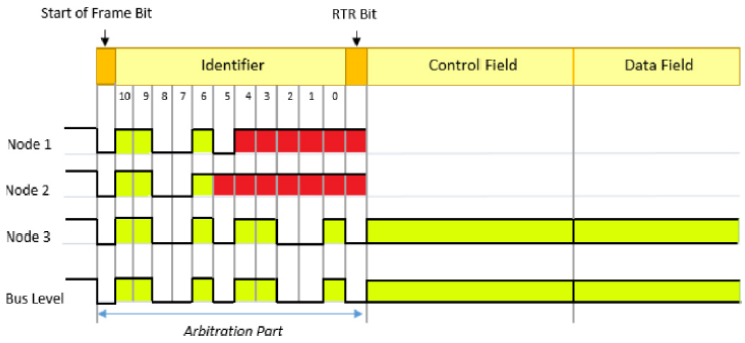


Figure 2.2: Arbitration field operation in CAN

1. Data Frame: The purpose of the data frame is to convey messages from sender to receiver. As illustrated in Fig. 2.3 (23) this frame comprises a 1-bit start frame, 12-bit Arbitration Field, 6-bit Control Field, a Data Field ranging from 0-64 bits, 16-bit CRC Field, 2-bit ACK field, and a 7-bit End of frame field, respectively. In the arbitration field, the message priority is defined and a single bit decides whether a data frame or a remote frame will be transmitted. The control field decides the size of the data frame. The CRC Field contains 15 bits for a Cyclic Redundant Checksum algorithm is implemented to detect errors, and uses one recessive bit delimiter (23). In ACK field acknowledgement occurs as part of the communication protocol; the message from the CRC field is being received and detected in ACK field for any possibility of error occurrence. If there is no error detected and the data matches the original message, then it will be reported to the transmitter that the sent message is being validated and received accurately. This process happens by replacing the recessive bit with a dominant bit in the ACK field (23; 60).

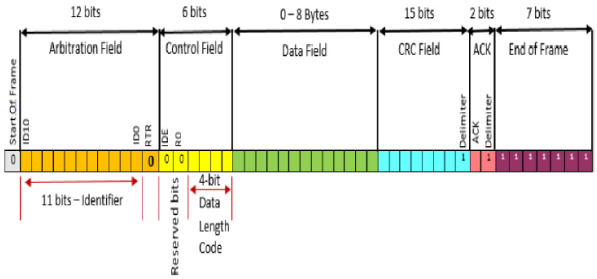


Figure 2.3: Data frame of CAN

2. Remote Frame: The remote frame in general is made of five fields which are a 13-bit Arbitration Field, 6-bit Control Field, 16-bit CRC Field, 2-bit ACK field, and 7-bit End of Frame Field. Fig. 2.4 (23) shows the full depiction of the Remote Frame structure, which closely resembles the Data Frame. The primary difference is that the Remote Frame does not transmit data; instead, the Remote Frame creates a sender response to the requested data from the receiver (with the same identifier) (23; 60) and a recessive RTR bit in the Arbitration Field identifies the message to be a remote frame (23; 60).

3. Error Frame: The error frame involves two parts which are a 6-bit Error Flag and an 8-bit Error Delimiter. When a node receives an error message, the Error Flag and Error Delimiter are sent. In case of any inaccuracy, the error flag will be raised. The Error Frame is shown in Fig. 2.5 (23).

4. Overload Frame: The Overload Frame is transmitted whenever the receiving node is extremely busy and cannot receive the message and a delay between the sent messages is activated. There are two conditions upon which an Overload flag is transmitted. The first condition is the internal condition of the receiver, which causes further latency in transmitting the next or remote frame or data frame (62). The second condition for transmitting the Overload frame is the detection of a dominant bit during the intermission. In addition, the Overload Frame contains a 6-bit Overload flag and an 8-bit Overload Delimiter. Fig. 2.6 (23) shows the layout of the Overload Frame.

5. Bit Stuffing Method: The bit stuffing rule is violated when six identical bits are sent consecutively via CAN bus, which indicates an error (62). Therefore, a complementary bit is added after transmitting five identical bits consecutively to some fields in CAN bus to avoid bit stuffing such as Arbitration Field, Control Field, and CRC Field (23). If bit stuffing occurs during transmission, then every node will transmit the error frame as an indication of violation of bit stuffing law. Fig. 2.7 (23)

shows how the bit stuffing is added to fields.

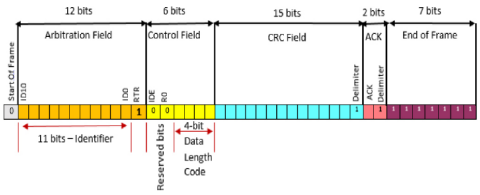


Figure 2.4: Remote frame of CAN

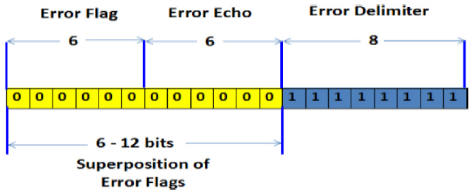


Figure 2.5: Error frame of CAN

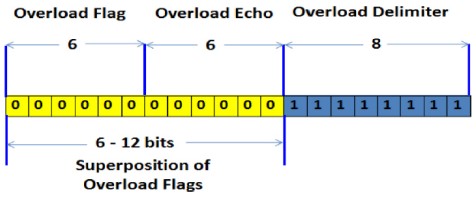


Figure 2.6: Overload frame of CAN

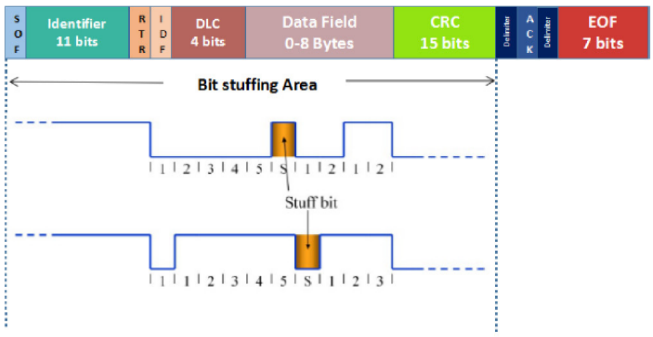


Figure 2.7: Frame bit stuffing of CAN

## 2.2 FlexRay- an Overview

In 2000, FlexRay Consortium introduced the FlexRay protocol especially for safety purposes in the automotive production, which requires a higher data rate. Therefore,

FlexRay technology can be considered safer and more advanced than CAN and LIN technologies. The most significant feature of FlexRay, that CAN and LIN busses lack, is network versatility by which FlexRay protocol adopts various topologies such as star, bus, and Hybrid topology (63). Like the functionality of HUB topology in the Ethernet network, the star topology is described as a central node that is connected to other nodes to expand the network and cover longer distances. The other benefit of the star topology is that a partial breakdown of the network will not affect the functionality of the entire network (23; 64). Star topology might be exposed to the ambient noise that could be caused by long distance wiring. To eliminate most of the noise and to increase the purity of the network, the star topology adopts the multiple legs concept instead of the one leg star topology (23). On the other hand, FlexRay can combine both the bus and the star topologies together as a hybrid system. Using the bus and the star topologies in parallel sounds more promising in the future of intra-vehicle networks due to high fault-tolerance and reliability of the star topology, despite the affordability and simplicity of use for the bus topology (23). In Fig. 2.8 (23), a basic diagram illustrates the hybrid topology of FlexRay. FlexRay supports event-triggered and time-triggered tactics for the in-vehicle communication network. The deterministic data reaches its destination in an expected time frame with the support of time-triggered protocol, making this type of network is suitable for hard real-time embedded systems (23).

FlexRay handles data collision by the mechanism of Time Division Multiple Access TDMA (23). The nodes in the FlexRay network connect to the network-synchronization clock, and by using TDMA the deterministic conditions are met and the consistency of the data is ensured. In other words, each node will write to the bus in periodic order due to the clock synchronization network that is supported by FlexRay protocol (23; 63). FlexRay offers robustness with its two-wire or four-wire differential signalling, the latter of which contains two separate twisted pairs tasked

with transmitting the same information. Comparison of the signals received from each twisted pair aids in determining the level of fidelity in the received data. The communication cycle is comprised of four frames which are the Static Segment, Dynamic Segment, Symbol Windows, and Network Idle Time.

1. Static Segment: The Static Segment of transmission supports deterministic data, ensuring that certain data reaches its destination in a set period of time.

2. Dynamic Segment: The Dynamic Segment acts as the data frame in the CAN network for messages that necessitate an event-triggered protocol, in which data determinism approach is not required.

3. Symbol Windows: The Symbol Windows section is used for network maintenance (23).

4. Network Idle Time: The Network Time Idle, known as "silent time", preserves clock synchronization in the network (23). The ECUs use this frame to regulate any drift that might affect the network timing from the previous cycle. The FlexRay hybrid topology is depicted in Fig. 2.8 (23).

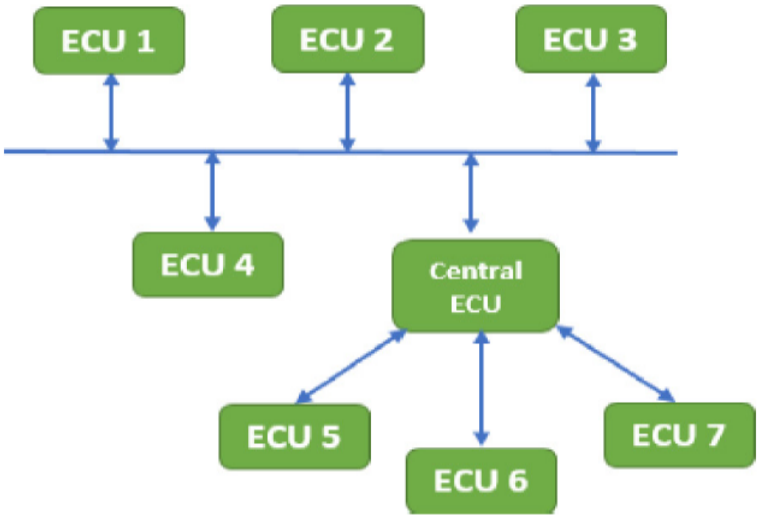


Figure 2.8: FlexRay hybrid topology

## 2.3 LIN- an Overview

One common protocol in modern vehicles is the Local Interconnect Network, otherwise referred to as LIN. A LIN bus generally connects non-critical systems that do not require direct connection to the CAN (Controller Area Network) bus, which helps to offload bus traffic from the CAN bus or reduce the total number of CAN buses required in a vehicle (65). Typical non-critical vehicle systems which use LIN include seating systems, the windshield wiper system, lighting systems on the interior of the vehicle, temperature control systems, and door modules (66). None of these applications require high reliability or robustness, so these systems do not require the high reliability and performance provided by CAN. The LIN protocol uses a master-slave configuration, with only one master and up to 16 slaves on each LIN bus (61; 66). The master is responsible for controlling the data transfer cycle among all devices on the LIN bus. LIN is schedule-based, or deterministic, and thus requests are made by the master in a predetermined order. This schedule of requests makes it possible to know exactly when a specific request will be sent. The master sends a request for specific data by sending what is called a header, which is broadcast to the LIN bus. Each slave is designated to respond only to a specific set of protected identifiers, which are part of each header transmitted by the master (65; 66). When responding to a protected identifier, a device will publish data to the LIN bus. Note that only one node is assigned to respond to any given message, so no two slaves should ever simultaneously send a response. Although only one node may respond by transmitting to the LIN bus, other nodes may listen to the response and read the data being transmitted (65). This scheduling system mandates that systems wait their turn to share data, and no single system's transmission is valued more highly than another. Another consequence of scheduling is that the maximum latency between a slave becoming ready to transmit data and its actual turn to broadcast data can be determined using the schedule (66). The LIN bus is comprised of a single wire, and the master and

slave devices can be daisy-chained along this bus. The bus is pulled up to a logical high level, and when a LIN transmitter wants to send a dominant bit, it drives the bus to zero volts by providing a connection to ground. A low bit is considered to be dominant because only one node has to provide a connection to ground for the bus to become low. Generally, the high voltage level will be equivalent to that of the car battery in vehicles, or roughly twelve volts, for recessive bits. Again, zero volts are measured for dominant bits. Note that these are ideal values; in reality, these voltages can shift up or down by as much as ten percent of the battery voltage (66).

In LIN communication, there are two main components of a single communication cycle: a header, also called a token or a request, and data. Only the master may send a header.

1. Header: The first part of a transmission sequence is the header. This begins with a break, which contains 13 consecutive dominant bits and ends with one delimiting bit, which is recessive. In LIN, dominant bits have 0 volts, while recessive bits typically assume a value between 9 and 18 volts (65), although there are some exceptions. This break field indicates to the slaves that transmission is going to begin. Next, there is a sync period. In LIN's master-slave configuration, only the master is required to possess a crystal oscillator, whereas slave nodes may use much less accurate oscillators such as low-cost RC circuits. Because slave devices use these cheaper, less accurate oscillators, the sync period is necessary so slave devices can determine the baud rate of the master and match that rate. The sync period consists of a dominant start bit, then 8 consecutive alternating bits (0x55) and lastly a recessive stop bit. In essence, this signal is equivalent to transmitting the master's clock signal. Lastly, the protected identifier is sent. This consists of one dominant start bit, a six-bit identification number for the specific task requested by the master, two parity bits, and one stop bit (65). It should be noted that in LIN 2.0 the two most significant bits of the protected ID, or bits four and five of the header byte, are

used to indicate the length of the expected data transmitted in response, which can be two, four, or eight bytes long (61).

2. Data: The data sent in response to a specific protected identifier may contain either two, four, or eight frames. Each frame begins with a dominant start bit, then contains one byte of data, and terminates with a stop bit. Between each start and stop bit, each byte of data begins with the least-significant bit and concludes with the most-significant bit (66). After the data bytes are sent, a one-byte checksum is sent, also beginning with a start bit and ending with a stop bit. A checksum is calculated by the slave and then transmitted. This transmitted checksum should match the checksum calculated by the master. If checksums do not match, this indicates an erroneous data transmission (61).

3. Parity: One check for message fidelity occurs during the transmission of a header. Each header frame sends a dominant start bit, the six-bit protected identifier, two parity bits, and a stop bit (65). The parity bits are computed by perform a specified operation on the protected identifier in the header. The identifier bits and the parity bits as a whole should follow an accepted rule when transmitted. If this rule is violated in the received message then it is known that the transmitted message was altered and corrupted. For LIN 2.0, the first bit is calculated by taking the logical XOR operation of bits 0, 1, 2, and 4 of the protected identifier. The resulting parity bit, P0, will be equal to one if the number of ones in these four bits is an odd number, and zero otherwise. The second parity bit, P1, is the result of the logical XOR operation with bits 1, 3, 4, and 5 of the protected identifier. P1 will be one if the number of ones in these four bits is an odd number, and P1 will be zero otherwise. If one of the protected identifier bits is corrupted during transmission, the reproduction of these parity bits will yield different results than those transmitted, indicating transmission failure (67).

4. Checksum: Another fidelity check occurs after all data frames are transmitted

in a response, ensuring valid transmission of data. A checksum is transmitted, which is the output of an algorithm that produces a one-byte result in this case. The checksum is calculated by the transmitting device and is broadcast on the LIN bus after transmitting all of its data. The recipient of the data then runs the same checksum algorithm on all received data bytes, and compares its own output to the checksum transmitted on the bus to determine if the data sent was truly error-free. Older LIN protocols calculated a checksum by summing only the data bytes that were transmitted, but LIN 2.0 also incorporates the protected identifier sent by the master in the header into this computation (68).

## 2.4 Comparison of CAN, FlexRay and LIN Protocol

### 2.4.1 CAN vs. FlexRay

The CAN bus offers a robust, simple and affordable solution for the in-vehicle network where the ECUs are all connected with each other and with a bus. This architecture avoids complicated bi-directional communication, and the number of wires required is just two. On the other hand, FlexRay is another well-structured basic in-vehicle network system that promises a higher speed of transmission than the CAN bus can support. One of the most significant features of the FlexRay that services the automotive industry is providing higher rate data (10 Mbps) than CAN and LIN communication (1 Mbps) (23), which expands the bandwidth for data transmission. As a multi-master network, the CAN bus broadcast communication mechanism allows transmission of the messages with higher priority first, which makes it suitable for hard real-time systems (62). However, FlexRay is suitable for deterministic data communication that prevents prioritization of messages in all situations (23). This deterministic communication is given to vehicle dynamic controls and chassis (51).

Regarding reliability, the CAN system supports error detection techniques in-

cluding Cyclic Redundancy Code (CRC) which allows all nodes in the bus to detect and avoid messages that carry errors (23). Meanwhile, FlexRay offers redundant communication capability (dual-channel communication) (23), which makes FlexRay technology more secure than LIN and CAN networks, which totally lack this feature. Another significant feature of FlexRay is network versatility: FlexRay supports star, bus, and hybrid topologies, which are not supported by the CAN protocol (23; 64). The maximum bus length for 1Mbit/s CAN protocol is 40-meters, while FlexRay allows at most 24-meter bus length. CAN bus cable is made of two wires, while FlexRay requests two to four wires. Fault-tolerance in FlexRay is ideal in-vehicle applications like braking and steering control systems (51). The CAN bus is an event-triggered protocol (23), whereas FlexRay is both a time-trigger and an event-trigger protocol and it has the tolerance to event-triggered and time-triggered data in the same cycle (51). Furthermore, the message transmitted via FlexRay can be both synchronous and asynchronous, which satisfies the requirements for vehicle components that depend on hard real-time systems such as an anti-lock braking system, or ABS, brake control, and engine control (23). In the CAN protocol, the transmitted messages are only asynchronous.

Both methods have several disadvantages. Although the broadcasting nature of transmitting messages via CAN bus offers simplicity, the standard CAN bus system commonly used in the automotive industry today lacks a message authentication technique for both the source and the destination. This might increase the chance of hackers to break through the system, send malicious messages and then fully attack the system, where the receiver cannot realize if it is the original or a spoof message (23). In the CAN protocol, data collision and collapse would be very possible whenever two nodes try to transmit data via the same bus at the same instance, so an arbitration method is used to avoid the data collision since it is relatively easy to apply and implement. However, this method is not practical for high rate data trans-

Table 2.1: CAN vs. FlexRay

| **Comparison Criteria** | **CAN** | **FlexRay** |
|---|---|---|
| Applications | Safety-Critical Systems | Safety-Critical Systems |
| Configuration | Multiple-Master | Multiple-Master |
| Cost and Complexity | Relatively High | Very High |
| Transmission Speed | 1 Mbit/s | 10 Mbit/s |
| Signalling Method | 2-Wire Twisted Pair Differential Signalling | 2-Wire/Four-Wire Twisted Pair Differential Signalling |
| Number of ID Bits | 11 or 29 | 11 |
| Failure Management | If Node Fails, That Node Stops Transmitting | Topology Dependent; In Star Topology, Failing Node is Suppressed |
| Transmission Process | Priority-Based and Event-Driven | Time/Schedule-Based and Priority Based/Event Driven |
| Topology | Two-Wire Bus | Two-Wire Bus or Star |
| Error-Checking | 15-Bit CRC, Acknowledgement Bit | 24-Bit CRC and 16-Bit Header CRC(68) |

mission (23). However, FlexRay has disadvantages as well, such as implementation cost, and difficulty of the protocol relative to the CAN protocol. Also, transmitting time-triggered and event-triggered data can impact the efficiency of data transmission (51). Table 2.1 summarizes the comparison of these protocols.

### 2.4.2 LIN vs. CAN

Because of the master-slave configuration, LIN slaves are not required to have their own crystal oscillators, but rather can have cheaper RC oscillator circuits with a tolerance of 15%. This is the most important factor in what makes LIN a low-cost communication protocol in comparison to other protocols like CAN (66). Furthermore, the LIN bus only requires a single wire as opposed to the shielded twisted pair wire required for CAN, which makes LIN more cost effective (61). The single-master configuration of LIN is considered to be relatively simple to implement compared to CAN, which uses a multiple-master architecture (65).

Due to the transmit-by-schedule nature of LIN, it does not support priority-based

transmission, whereas CAN communication will serve nodes with high priority first as they become ready to broadcast data. However, this means that LIN allows for deterministic communication through scheduling, allowing one to know precisely when a specific message will be sent. This is unlike CAN, in which one can approximately predict when a node will transmit data based on its frequency of transmissions, but this cycle will be irregular due to bus arbitration (68).

LIN has low transmission speed in the range of 1 to approximately 20 kbit/s, which is slower than typical CAN transmission rates, which reach up to 1 Mbit/s. Due to LIN's scheduling nature, when a transmission error is detected, the schedule continues as planned. However, in CAN, if an error is detected in a transmission, the same message will be re-transmitted the next time the bus is available (68).

Both LIN and CAN combat electromagnetic interference (EMI) in different ways. LIN has reasonable immunity to EMI due to its low transmission speeds, whereas CAN uses a much higher transmission speed. Low speeds also prevent excessive EMI radiation from the LIN bus. Unlike LIN, however, CAN combat EMI by using differential signaling, which works well because both CAN line likely receive the same disturbances from EMI, so the potential difference between the two lines theoretically remains unaffected. CAN also use a twisted pair and shielding to increase immunity and decrease radiation of EMI (61). Another reliability issue with LIN is that when the master fails all systems connected to the bus fail, due to the master orchestrating all communication (65). In CAN, this is not a problem because it uses a multiple-master topology. When one CAN node has generated too many transmission errors, the node will shut itself down (69).

Lastly, CAN is able to accommodate more systems and/or more unique messages on a single bus. LIN's protected identifiers are only six bits long, providing a total of 64 different identifiers that can be transmitted by the master. CAN networks either use 11 bits or 29 bits, enabling a much larger variety of unique IDs to be transmitted

Table 2.2: LIN vs.CAN

| **Comparison Criteria** | **LIN** | **CAN** |
|---|---|---|
| Applications | Lower-Performance Systems | Safety-Critical Systems |
| Configuration | Single-Master, Slaves | Multiple-Master |
| Cost and Complexity | Low | Relatively High |
| Transmission Speed | 20 kbit/s | 1 Mbit/s |
| Signalling Method | Single Line Signalling | 2-Wire Twisted Pair Differential Signalling |
| Number of ID Bits | 6 | 11 or 29 |
| Failure Management | If Master Fails, Whole LIN Bus Fails | If Node Fails, That Node Stops Transmitting |
| Transmission Process | Time/Schedule-Based | Priority-Based and Event-Driven |
| Topology | Single-Wire Bus | Two-Wire Bus |
| Error-Checking | 1-Byte Checksum | 15-Bit CRC, Acknowledgement Bit(68) |

(61; 49). The main differences between LIN and CAN are summarized in Table 2.2.

### 2.4.3 LIN vs. FlexRay

Again, LIN's single-master configuration demands that only the master has a highly-accurate crystal oscillator, which makes this configuration relatively cheap, as opposed to FlexRay, which does not use a single master-slave configuration and requires high accuracy in all nodes' oscillators. LIN communication is all schedule-based, whereas FlexRay allows for both schedule-based and event-driven communication by allocating time slots for both static and dynamic frames (70).

The bandwidth of LIN is much lower than that of FlexRay; LIN's 20 kbit/s bandwidth is much smaller than FlexRay's 10 Mbit/s bandwidth (68). Furthermore, data transmission sizes are very different. A LIN message can transmit two, four, or eight bytes onto the LIN bus in response to a request from the master, while a FlexRay node may transmit up to 254 bytes of data in one data frame (70).

The FlexRay star topology allows for higher reliability than LIN communication in

Table 2.3: LIN vs. FlexRay

| **Comparison Criteria** | **LIN** | **FlexRay** |
|---|---|---|
| Applications | Lower-Performance Systems | Safety-Critical Systems |
| Configuration | Single-Master, Slaves | Multiple-Master |
| Cost and Complexity | Low | Very High |
| Transmission Speed | 20 kbit/s | 10 Mbit/s |
| Signalling Method | Single Line Signalling | Two-Wire/Four-Wire Twisted Pair Differential Signalling |
| Number of ID Bits | 6 | 11 |
| Failure Management | If Master Fails, Whole LIN Bus Fails | Topology-Dependent; In Star Topology, Failing Node is Suppressed |
| Transmission Process | Time/Schedule-Based | Time/Schedule-Based and Priority-Based/Event-Driven |
| Topology | Single-Wire Bus | Two-Wire Bus or Star |
| Error-Checking | 1-Byte Checksum | 24-Bit CRC and 16-Bit Header CRC(70) |

the sense that it gives the ability to disconnect nodes experiencing failure, preventing these faulty messages from transmitting to all other nodes (71). LIN uses only a single wireline while FlexRay generally uses two-wire differential signaling. FlexRay also offers a redundant four-wire protocol, where one transmitted message is sent over two buses. The received messages from each bus can then be compared to ensure that they are identical and no corruption has occurred. This kind of security feature is used only for systems that are safety-critical (70). Also, because in FlexRay up to 254 bytes of data can be transmitted by one node, compared to a maximum of 8 data bytes in LIN, FlexRay uses three 8-bit CRCs (a common type of checksum) as opposed to LIN's single byte checksum (70). The main comparisons between LIN and FlexRay are shown in Table 2.3.

In this chapter, CAN, FlexRay, and LIN technologies are discussed and compared to each other to provide an idea of how the combined interconnected systems of automobiles work together proficiently to prevent critical issues with transmitting

messages. A comparison is made to demonstrate the differences, topologies and specifications of each protocol and as a complete interconnected communication system in each vehicle. In general, the number of ID bits for CAN is 11 or 29, which gives it more flexibility in data transmission with different IDs (61; 49). FlexRay is 11 bits, and LIN is 6 bits. The comparison of CAN, FlexRay and LIN shows that the systems are well structured and suitable for signal transmission in a real-time driven fashion. However, FlexRay has a higher speed in transmitting signals due to its hybrid nature of time-triggered and event-triggered approaches, which gives it higher bandwidth than CAN and LIN. FlexRay is considered safer than CAN and LIN due to higher data rate requirements. Error detection methods differ among discussed protocols. FlexRay offers redundant communication capability while CAN and LIN offer cyclic redundancy code (CRC). Unlike FlexRay and LIN, CAN priorities signal transmission due to the bus arbitration mechanism. However, FlexRay and LIN do not allow message arbitration due to the deterministic nature of their data communication. LIN is more cost-effective compared to CAN and FlexRay because it does not require a high-accuracy crystal oscillator. Instead, it only needs low-cost RC oscillator circuits in slave devices for performance. In terms of applications, CAN and FlexRay are used for safety-critical systems while LIN is more suitable for low-performance applications that are not time-critical such as window lift, wipers and mirrors.

CAN and FlexRay are multiple-Master communication models, yet LIN is a Single-Master Multi-Slave model. This configuration makes LIN cheaper than CAN and FlexRay, which are more expensive due to higher accuracy needed in all node oscillators. However, the entire LIN bus fails if the Master fails. In CAN bus, the failure mode is restrained by node failure where the transmission stops. On the other hand, FlexRay failure management is topology dependent. Moreover, the transmission speed of FlexRay is the fastest, which is 20 Mbit/s, CAN is 1 Mbit/s, and lastly, the transmission speed of LIN is 20 Kbit/s. The signaling method of CAN is based

on two twisted-pair wiring, FlexRay is based on two or four twisted-pair wiring. Nevertheless, LIN is based on single-line signaling. This makes LIN more available to be used in a wide range of vehicles for a low price. Eventually, a comparative study was carefully implemented on three essential and contemporary communication methods for in-vehicle networking. It shows the importance and specialty of each system individually, and then in one whole system collectively.

# CHAPTER III

# Related Work

In 2015, Miller et. al. (50) hijacked a jeep in the middle of the highway, which led to a recall of 1.4 million vehicles. They were able to control the main functions of the vehicle like engine and brakes etc., hence making a case to improve security in modern electric vehicles for the safety of passengers. The existing state Of-the-art (SOA) on countermeasure against spoofing and impersonation attacks on in-vehicle networks can be divided into (i) message authentication based approaches (19; 18; 20; 21; 22; 23; 24), (ii) intrusion detection based approaches (29; 30; 31; 32; 33; 34; 35; 36; 37; 38; 39; 40; 41; 44; 25; 59; 45; 43; 26; 46; 27; 42) as shown in Fig. 3.1.

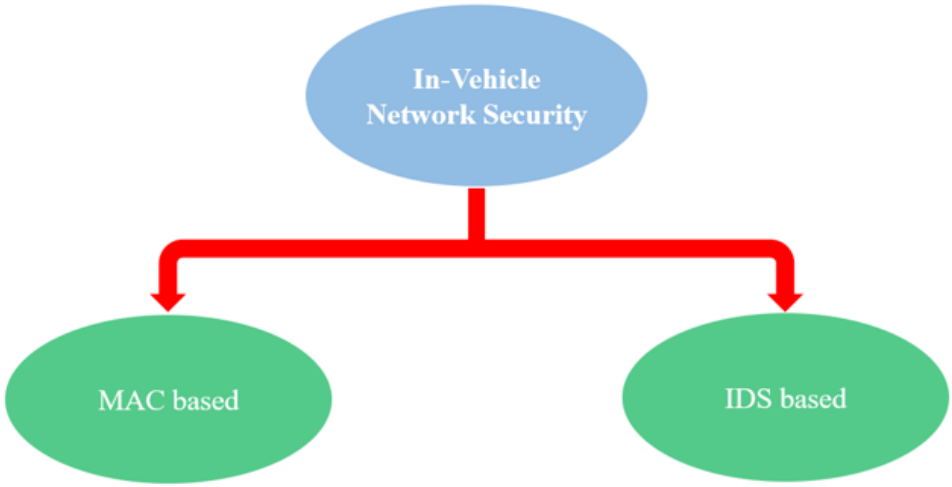


Figure 3.1: Existing SOA approaches for IVN attacks

Most of the work done for in-vehicle network security relies on the message authentication code (MAC) - a traditional computer network security framework. In (17), Wang et. al. proposed a framework for security in-vehicle systems (VeCure) which relies on the message authentication code (MAC) for secure communication. In VeCure, the transmitting node sends a 64-bits of MAC for every 64-bit message. This method has high computational cost considering the computational capacity of the ECU and 50% additional transmission overhead. Hiroshi et. al. designed an authentication mechanism for the CAN protocol against spoofing attacks. The monitoring node provided the authentication code for all ECUs and verified the code for all CAN messages. Hazem et. al. proposed a lightweight CAN authentication protocol (LCAP). The LCAP required to append a "magic number" that was generated by a one-way hash function employed on TESLA prototype (47). The protocol required 16-bits of the data field to append the authentication code, which still creates 25% overhead. MAC-based approaches work well, however, they have 4 major limitations. First, the MAC-based approaches typically add extra overhead in the network because more bits are required for encryption. The second limitation is that the MAC-based approach typically needs a higher data rate. The third limitation is that if the encryption key is estimated by the adversary, it can easily make an attack. The fourth limitation is that additional hardware is required for the MAC-based approach; this adds extra cost to the system. Moreover, due to the centralized monitoring node, the whole network will become compromised if this node is eliminated/compromised.

To overcome the shortcomings of MAC based approaches, the intrusion detection system (IDS) based approaches are used. Intrusion detection based approaches are further subdivided into: (a) parameter monitoring based approaches (29; 30; 31), (b) information theory based approaches (32; 33; 34), (c) machine learning based approaches (35; 36; 37; 38; 39; 40; 41), and (d) fingerprinting based approaches (44; 25; 45; 43; 26; 46; 27; 42) as shown in Fig. 3.2. The intrusion detection sys-

tems (IDSs) are being used persistently in computer network for many years (72). Cybersecurity professionals for automotive security have also started using IDSs for protecting connected vehicles against cyberattacks (27).

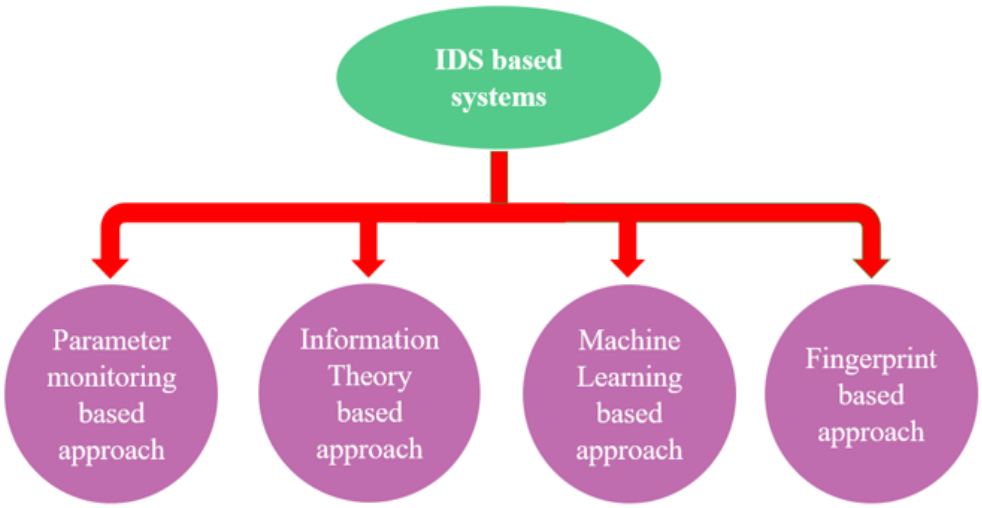


Figure 3.2: Intrusion detection based approaches

## 3.1 Parametric Monitoring based IDS

The $1^{st}$ type of IDS is based on parameter monitoring. Parametric monitoring based approaches can be subdivided into ($a_1$) frequency-based technique and ($a_2$) remote frame-based technique (73). In frequency-based technique, the frequency/periodicity of message frames is monitored (30). In this technique, transmission intervals of CAN messages can be detected and compared against the established baseline (74). Researchers have shown that during spoofing attacks and denial of service (DoS) attacks, the frequency of the message frames is changed (31; 50; 73). There are-*two* limitations of the frequency-based technique, the first limitation is that it works only for periodic messages, the second limitation is that if the adversary has knowledge of the period, it can launch a spoofing attack. The second type of parametric monitoring based approach is the remote frame-based technique. Whenever a node in CAN

receives a remote frame, it responds to the sender with a message within a certain amount of time. Offset in this response time can reflect a suspicious activity (29). There are-*two* limitations of this technique, the first limitation is that it requires an additional node, the second limitation is that if the adversary has knowledge of the response time, it can launch a spoofing attack.

## 3.2 Information Theory-based IDS

The $2^{nd}$ type of IDS is information theory-based. The internal communication of an ECU is in order (73). The set of sequences in which different subroutines/functions are called during a certain task are stored and if the sequence is violated during a task, it indicates the occurrence of an attack. For instance, Wang et. al. collected 6.673 million CAN packets for different vehicles (75), their experimental results showed that CAN message have low entropy of an average of 11.915 bits (75). The idea of entropy-based attack detection was first introduced by Muter et. al., the issue with this type of approach is that if an adversary injects a small number of malicious messages, the attack is difficult to be recognized (76). In another study, Marchetti et. al. showed that if an attacker launches an attack in the vehicular network, an entropy-based attack detector is able to detect the attack only if the attack is made in a high volume of forged CAN messages (32).

## 3.3 Machine Learning-based IDS

The $3^{rd}$ type of IDS is the machine learning (ML) and artificial intelligence (AI) based approach. ML and AI-based classification, regression and clustering are used to provide to implement security in vehicular networks at different levels (73). For instance, Taylor et. al. proposed a Long ShortTerm Memory (LSTM) neural network to detect CAN bus attacks (35). Typically nodes in CAN protocol do not produce a

variety of data. In this method, the recurrent neural network (RNN) for CAN bus anomaly detection were used. Further, it required the neural network to be trained for each node. Another ML-based algorithm for in-vehicle security Ternary Content Addressable Memory (TCAM) was introduced by Markowitz et. al. (37), TCAM is a special type of high-speed memory usually used by modern switches and routers for fast look-up tables and packet classification. In its training phase, this system used the classifier to characterize the fields and build a model for the messages, based on their field types. In the testing phase, the system detected deviations from the model (37). Kang et. al. proposed a deep learning-based ML technique (36). Deep learning is a machine learning technique that used a number of hierarchical layers of non-linear processing stages. The 64-bit data field of valid CAN messages was used as an input in the deep neural network. In the validation phase, the output of the deep learning-based model is '1' or '0'. Here, '1' refers to a normal CAN packet and '0' refers to an anomaly/attack. ML-based approaches are typically used to prevent attacks at the application layer. ML-based models work well, however, they require considerable computing and storage resources.

## 3.4 Fingerprinting-based IDS

The $4^{th}$ type of IDS is the fingerprint-based approach. The idea of fingerprinting the source from physical signal is not only used for fingerprinting ECUs, but is also used for fingerprinting other electronic devices such as microphones (77; 78). This approach is used to avoid attacks at the physical layer. In this type, the inimitable characteristics in physical signal are used to identify the legitimate transmitter. In (26), Cho and Shin proposed a clock based intrusion detection system (CIDS) which used an approach of ECU fingerprinting. This system used deviation from the basic characteristics of the clock based digital systems, that is, "the tiny timing error known as clock skew." For identification of an ECU, the clock skew and clock offset are

primarily used. Thus, the proposed system identification deviates from the basic clock-based ECU. Cho and Shin (26) conjointly made a paradigm of planned IDS and showed the incontestable efficiency of planned CIDS in real automobiles. CIDS leveraged the fact that the clock skew is a physical property of each ECU that cannot be changed by the adversary (26). However, Tayyab et. al. (49) and Sang et. al. (28) demonstrated that CIDS can be bypassed by estimating the clock parameters. In (46), higher-order moments of CAN signal both in time- and frequency-domain are used for transmitter identification. It has been demonstrated that the method demonstrated in (46) achieved detection accuracy of 98.3%. Another prominent work in IDS is done by Sang et. al. (28) in which the authors demonstrated "cloaking attack", in which an adversary modified the timing of transmitted messages in order to match the clock skew of a targeted ECU, hence bypassing the security of the system. They (28) showed a new IDS that is developed based on the widely-used Network Time Protocol (NTP). This IDS used two parameters, clock skewness and cumulative sum, to identify the transmitting ECU for message authentication.

### 3.4.1 Scission

Kneib and Huth proposed a method 'Scission' to implement security in CAN. Scission uses a sampling rate of $20 Msamples/sec$. After sampling the signal from each ECU, the statistical features of the signal from each transmitter are calculated in addition to energy of the signal. These features proved to be unique for each transmitter because of variations in supply voltages, variations in grounding, variations in resistors, termination and cables, and imperfections in bus topology causing reflections. This feature-set is used to link the message to the source ECU to authenticate the message (45). Scission detected the valid message with a 99.85% accuracy (45). The feature-set is shown in Table 3.1. Viden uses only acknowledgement bits to identify the transmitter. The advantage of Scission is that it can prevent attacks

using any part of the frame. The strength of a signal is determined by energy, which is expressed by eq. (3.1).

$$Energy = \frac{1}{N}\sum_{i=1}^{N} x(i)^2 \tag{3.1}$$

Table 3.1: Time domain feature-set

| **Feature** Name | **Equation** |
|---|---|
| Maximum | $\mathrm{m}_{ij} = (Min(y_{ij}(i))\|i = 1....N)$ |
| Minimum | $\mathrm{M}_{ij} = (Max(y_{ij}(i))\|i = 1....N)$ |
| Mean | $\mu_{ij} = 1/N \sum_{i=1}^{N} y_{ij}(i)$ |
| Variance | $\sigma_{ij}^2 = \sqrt{(1/N-1)(\sum_{i=1}^{N} y_{ij}(i) - \mu_{ij})}$ |
| Skewness | $\rho_{ij} = (1/N) \sum_{i=1}^{N} ((y_{ij}(i) - \mu_{ij})/\rho_{ij})^3$ |
| Kurtosis | $\mathrm{k}_{ij} = (1/N) \sum_{i=1}^{N} ((y_{ij}(i) - \mu_{ij})/\rho_{ij})^4 - 3$ |

### 3.4.2 Using Inimitable Characteristics of Signals

In (26), Choi et al. used inimitable characteristics of signals using unique signatures from each ECU fingerprinting (26). This method adds a monitoring unit to the CAN-protocol, in which messages are functionality-based. This monitoring unit supervised the messages transmitted by ECU and ensured that it sent only the authentic messages (26). If an ECU is supposed to control the sensors of the car, it should not send messages to the engine. If the ECU does send messages to the engine, then the system alarms of an attack (26). The monitoring unit used a sampling rate of $2.5 Gsamples/sec$. After sampling the signal from each ECU, this method used moment generating functions of signals in time domain as well as frequency domain as a feature-set. Further, transmitters are classified for message authentication by

Table 3.2: Frequency domain feature-set

| **Feature** Name | **Equation** |
|---|---|
| Spectral Std-Dev | $\sigma_s = \sqrt{(\sum_{i=1}^{N}(y_f(i))^2 * (y_m(i)))/\sum_{i=1}^{N}(y_m(i))}$ |
| Spectral Skewness | $\rho_s = \sum_{i=1}^{N} y_f(i)y_m(i)/\sigma_s^3$ |
| Kurtosis Std-Dev | $\mathrm{k}_s = (\sum_{i=1}^{N}(y_m(i) - C_s)^4 * y_m(i))/\sigma_s^4 - 3$ |
| Spectral Centroid | $\mathrm{C}_s = (\sum_{i=1}^{N} y_f(i)y_m(i))/(\sum_{i=1}^{N} y_m(i))$ |
| Irregularity K | $\mathrm{IK}_s = \sum_{i=2}^{N-1} \lvert y_m(i) - (y_m(i-1) + y_m(i) + y_m(i+1))/3\rvert$ |

"supervised learning". The feature-set used was the same as in Scission as seen in Table 3.1. In addition, the method used frequency domain features as shown in Table 3.2. This method detected the ECU with 96.48% accuracy (26). Table 3.3 shows the differences between each method.

### 3.4.3 VoltageIDS

Choi et al. used VoltageIDS (44) for security measures against masquerade attacks from an outside source. The signal was acquired at a sampling rate of $2.5 Gsamples/sec$ at the monitoring unit. After acquiring the signal. some additional features in addition to the features used in (26) were used both in time domain as well as in frequency domain. This technique is done by "supervised learning" (44). Supervised learning is done in two steps: a training step and testing step. The training step creates a multi-class classifier, which has the classes and ECUs equal in number (44). This step uses the sets created in the multi-class classifier and labels them with CAN IDs (44). During the testing step, the VoltageIDS detects a masquerade attack if the multi-class classifier has predictions of probable classes that do not match the CAN ID (44). Since this method used frequent learning, the system

is an easy target for injection attacks, meaning it can adapt to the attacking messages (44). Fig. 3.3 shows the multiple steps VoltageIDS uses to make sure there are no irregularities/invaders in the system. This method has a transmitter detection accuracy of 93.54% on real vehicle.

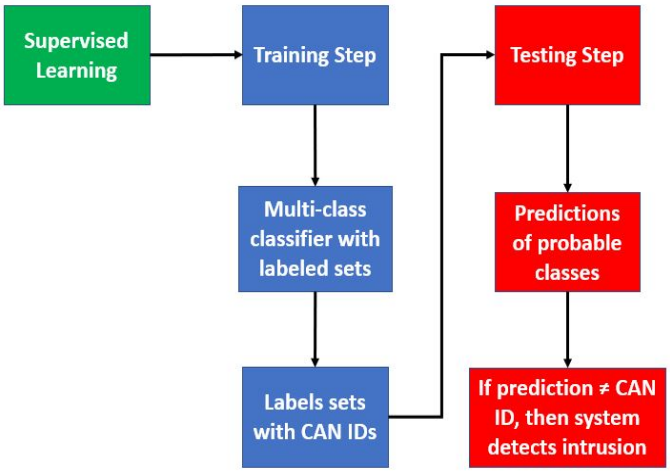


Figure 3.3: VoltageIDS steps

Table 3.3: Comparison of fingerprinting methods

| Research Work | **Method** | Contribution | Accuracy | Advantages | Disadvantages |
|---|---|---|---|---|---|
| Kneib et. al. (45) | **Scission** | It allows the localization of the transmitter which is ECU. | 99.85% | Localizes the location of the attack. | Addition of an extra ECU is needed. Adds cost. |
| Choi et. al. (43) | **Using Inimitable Characteristics of Signals** | Identifies inimitable ECUs | 96.48% | No change in the ECU system is needed. | Requires installing an extra unit. |
| Choi et. al. (44) | **Voltage IDS** | Distinguishes between errors and the bus-off attacks. | 93.54% | Addresses the attack properly. | Extra unit needed. |
| Cho et. al. (26) | **CIDS** | The ECU identification is independent of voltage characteristics. | 99.95% | Cannot be bypassed by burning chip. | Does not identify the location of the attacker. |
| Cho et. al. (25) | **Viden** | Fingerprinting of ECUs based upon voltage-based signatures. | 99.8% | Identities the ECU. | Burning out the signature detection ECU by high voltage will allow the attacker to invade the system. |

### 3.4.4 Clock based Intrusion Detection System (CIDS)

The CIDS (26) exploits uniqueness in the clock parameters, such as clock offset and clock skew for CAN node fingerprinting. Previously, various researchers have used this concept for wireless access point fingerprinting (79; 80) and wired Ethernet device fingerprinting (81). These researchers (79; 80; 81) rely on the presence of time-stamp in the packet headers. The absence of time-stamp field in the CAN protocol makes it unfit for direct extension of clock-based intrusion detection systems (79; 80; 81). Cho and Shin (26) proposed the methodology to extract the sender node's clock characteristics in CAN network to design an intrusion detection, which is discussed in the subsequent sections.

**Clock parameters:** The clock parameters used for modelling the clock physical attributes are defined below (26):

- **Clock offset:** The difference between the time reported by the clock *Ci* and the true clock is called *clock offset*. The difference between the reported times of non-true clocks is known as the *relative offset*.
- **Clock frequency:** The rate at which the clock *Ci* advances.
- **Skew:** It is the difference between the frequencies of the true clock and the local clock *Ci*. The difference between the frequencies of two non-true clocks is known as the *relative skew*.

The clock offset and clock skew of each node in an asynchronous network depends solely upon the local clocks of the nodes. Estimates of these clock parameters can be used for node fingerprinting. As discussed earlier, the estimation of these parameters is easy to achieve in the presence of time-stamps inside the message packets, but the absence of time-stamp information in embedded networks (e.g., CAN protocol) makes it difficult to achieve it. The CIDS framework (26) estimates these parameters for periodic messages. Before outlining the CIDS method to estimate the parameters,

Table 3.4: Message IDs and the period

| ID | Period (ms) | DLC |
|---|---|---|
| 0xAA | 50 | 3 |
| 0xBB | 100 | 3 |
| 0xCC | 150 | 3 |
| 0xDD | 500 | 3 |

brief overview of my experimental setup and CAN network configuration is provided next.

**Experimental setup and implementation:** To validate and implement the CIDS, a CAN network is realized with four nodes. Shown in Fig. 3.4 is the snapshot of my realization of CAN network. Each node here consists of an Arduino UNO board as ECU and CAN Shield as the CAN transceiver. The node M (Monitoring Node) in the network is used for monitoring the traffic and is running the CIDS. It listens to the network traffic and computes clock parameters. These parameters are then sent via serial link to the MATLAB for plotting the graphs and visually analyzing the clock behaviors of the sender nodes. The remaining three nodes are labelled as E3, E5 and E6.

The speed of the CAN bus is set to 500 Kbps. The CAN traffic consists of messages of four different IDs at different time periods. The details of the contents of the CAN network traffic are listed in Table 3.4. To analyze the behavior of clock parameters for each node, each node (E3, E5 and E6) is used to transmit messages with four different IDs in such a way that each node is sending unique messages from a particular ID at any given time. This resulted in the following configurations {E3: 0xAA, E5: 0xBB, E6:0xCC}, {E3: 0xBB, E5: 0xCC, E6:0xDD} or {E3: 0xCC, E5: 0xDD, E6:0xAA}. Experimental data is collected for each configuration. The architecture of the CIDS which is running on node M is explained below:

**Parameter estimation and clock behavior modelling:** In in-vehicle CAN, most of the messages are periodic, and each node transmits messages with a specific

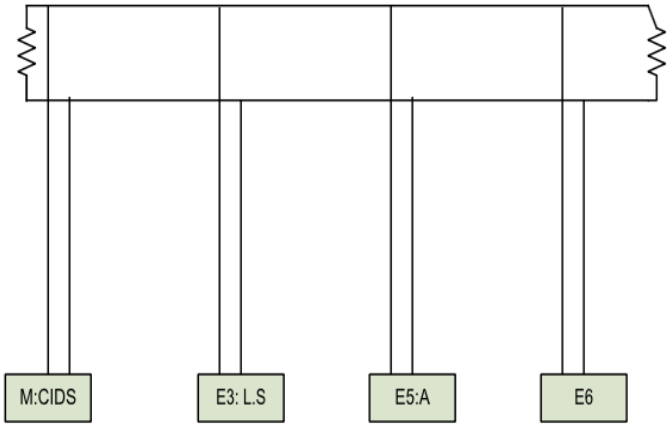


Figure 3.4: Block diagram of the CAN realization used for experiments

ID. These properties are exploited to estimate the clock parameters (e.g. clock offset and clock skew) of the message sender node.

**Periodicity estimation:** To estimate parameters of a node with a particular message ID, the period of the message is estimated first (49). The estimated period of a given message at node M node is a function of the sender node's clock crystal. Changing the message sender is expected to result in a slightly different value. This is due to the fact that the CAN protocol is asynchronous. Thus, the timing calculations at each node depends on the local clocks only.

**Estimation of offset:** Calculated time period is $T$ from the previous step while $t$ is the continuous time at the receiver side (node M) measured by the local clock of the receiver. Let the first message receives at $t{=}t_1$ and the predicted arrival time (with no offset) of the succeeding $i$ messages $t_i{=}t_1{+}iT$. In case of two asynchronous clocks, the expected arrival time with clock offset ($O_i$), subsequent messages' arrival time-stamps can be expressed as $t_i{=}t_1{+}O_i{+}iT$. To measure clock offset, predicted arrival time-stamp is subtracted from the actual time-stamp. The change in clock offset between two consecutive messages is negligible and expected to be close to zero.

**Clock skew estimation:** The average value of the clock offset is non-zero, but the change of offset value between consecutive messages will be negligible. Therefore,

to estimate the change in clock offset, the absolute average clock offset, computed for N messages are added and tracked. The resulting cumulative clock offset, called the CUMSUM, $O_{acc}$, is tracked for anomaly detection. The CUMSUM value is expected to grow linearly. The slope of the CUMSUM represents the clock skew of the message sender. The clock skew is unique and used for ECU fingerprinting.

**Clock behavior modelling and intrusion detection:** The linear behavior of the CUMSUM is approximated using Recursive Least Square (RLS) algorithm (26). The accumulative clock offset $O_acc$ can be expressed as:

$$O_{acc}[k] = S[k] * t[k] + e[k] \tag{3.2}$$

where $S[k]$ denotes the regression parameter and represents the clock skew, $t[k]$ denotes the elapsed time and $e[k]$ represents the identification error. The identification error is also known as the residual error.

The $O_{acc}$, $S$ and $e$ are calculated for every $N$ messages. Therefore, a total of $k*N$ messages are needed for parameter estimation at $k^{th}$ step. The value of $S$ is computed using RLS algorithm, while residual, $e$, is computed as:

$$e[k] = O_{acc}[k] - S[k-1] * t[k] \tag{3.3}$$

As long as the messages are being sent by the original ECU, the $S[k]$ is expected to remain unchanged, resulting in residual $e \approx 0$. This intrusion system is designed to solve the problem of sender authentication absence in the CAN protocol and, hence, is expected to detect the spoofing attacks. Whenever the source of a message is changed, the clock behavior associated with that ID also changes, resulting in the large value of the residual error that is tracked continuously.

### 3.4.5 Viden

Cho et al. introduced a technique, "voltage based attacker identification (Viden)" (25), which used voltage outputs of the ECUs to fingerprint the system. Viden is based on voltage profiles from different ECU outputs (25). It characterizes the voltage profile in acknowledgement bits and uses voltage signatures coming from each ECU at relative lower sampling rate of $50Ksamples/sec$ as compared to other methods. If the voltage differs from the signatures of authentic ECU, it triggers an alarm to report an attacker is in the system (25). Viden also localized the location of adversary device from which the attack was launched, which most methods cannot do (25). The results from Viden showed that it safeguarded vehicles from attackers with a 99.8% accuracy rate. The voltage profile of ECU also changes with environmental conditions like temperature, humidity, etc. The voltage profile also changes with other conditions, including noise, electromagnetic interference. Viden uses adaptive signal processing to adapt to different inside and outside conditions. The voltage profile was characterized in each condition and then used to identify transmitting ECU with a very high accuracy of 99.8%.

### 3.4.6 Implementing In-Vehicle Security Using Higher Order Statistics

Researchers have proposed various methods to link a CAN packet to its source by using physical characteristics of signal. For instance Cho and Shin have proposed a method clock intrusion detection system (CIDS) which uses clock skewness to identify the transmitter. CIDS works well, however it can be bypassed if the attacker has knowledge of clock characteristics of sender and receiver. This research (46) proposed a method to bypass CIDS, then it presents a new method by developing a framework to link a CAN packet to its source. Physical signal attributes of the received packet consisting of channel and node (or device) contains specific unique artifacts are used to achieve this goal. Material and design imperfections in the physical channel and

digital device are the main contributing factors behind the device and channel specific unique artifacts. Uniqueness of the channel-device specific attributes are also investigated for time- and frequency-domain. Feature vector is made up of both time and frequency domain physical attributes and then employed to train an artificial neural network (ANN)-based classifier. Performance of the proposed fingerprinting method was evaluated by using a dataset collected from six different channels and four identical ECUs transmitting same message.

The contribution of this research was to propose a method to bypass clock intrusion detection system (CIDS), to present a novel non-crypto-based approach for message authentication for the CAN protocol that exploits uniqueness in the channel and device-specific distortions to link the received CAN packets to 'the' transmitting ECU. The robustness of the proposed method in terms of *channel-level* as well as *device-level* variability is evaluated using bench-testing. The transmitter identification model that depends on the proved claim that any electronic device that has various components like the micro-controller, DAC, clock etc.. like the ECU and the impulse response of the CAN channel are unique in nature and thus these unique artifacts and distinguishable statistical features can be extracted from the received signal and were used to link them to its source ECU and identify the sender. The variations from different channels were used for various feature extraction tools. The uniqueness was evaluated both in time and frequency domain. The effectiveness of method was validated using the feature extraction method proposed in (82). A 40-Dimensional Scalar features in both time and spectral domain are extrapolated using LibXtract. LibXtract is a library for feature extraction (83). It uses a FEAST toolbox that works on the joint mutual information criterion to rank the feature. Using this feature extraction 11-D feature vector for ECU and channel identification was used for transmitter identification.

The main reason for the channel specific artifacts or ECU specific artifacts is

because of the material and design imperfections that are embedded into them at various stages of its development. In the first part of this research, data was recorded data for the 3 types of cable family with 6 different lengths each using same ECU. The input message used was same for all the channel types and the ECU used to generate this message is also same. The only variable for this experiment was channel class and the length. As the neural network is first trained (training phase), in this phase it is classified for three different cable families and six corresponding lengths (e.g., GXL: 0.5 meter, GXL: 1 meter, GXL: 2 meter, GXL: 3 meter, GXL: 4 meter, and GXL: 5 meter and so on). First the neural network is trained with multi-layer "scaled conjugate gradient back propagation" training algorithm. In this 11 input variables (both in frequency and time domain), 6 outputs to represent 6 different channel lengths of GXL cable. The stopping criteria of Epochs=2000, gradient = $10^{-7}$, and 3 hidden layers with 50, 40, and 40 hidden nodes respectively. In the second part of this research, channel was kept same and data was transmitted using 4 ECUs, 11 input variables (both in frequency and time domain), 4 outputs to represent 4 different ECUs. The stopping criteria of Epochs=2000, gradient = $10^{-7}$, and 3 hidden layers with 50, 40, and 40 hidden nodes respectively. With this set-up and idea that was proposed, a correction rate of 95.2% and 98.3% for both CAN-Bus channel identification and ECU identification was achieved respectively. The CAN-Bus and the ECU are from the same manufacturer and are identical, it leaves an inimitable artifacts in the output of the signal at the physical layer and the channel output. These characteristic artifacts are different for varying channel lengths and ECUs. Thus, this uniqueness was used to link the received physical signal to the actual transmitter.

# CHAPTER IV

# Study of ECU Specific Distortion and ECU Identification by Using Neural Networks

## 4.1 Introduction

A novel technique for electronic control unit (ECU) identification is proposed in this chapter to address security vulnerabilities of the controller area network (CAN) protocol. The reliable ECU-identification has the potential to prevent spoofing attacks launched over the CAN due to the inconsideration of the message authentication. In this regard, my technique models the ECU-specific random distortion caused by the imperfections in digital-to-analog converter, and semiconductor impurities in the transmitting ECU for fingerprinting. Afterwards, a 4-layered double Neural network architecture is trained on the feature-set to identify the transmitting ECU and the corresponding ECU-pin. The ECU-pin identification is also a novel contribution of this work, and can be used to avoid the voltage-based attacks. I have evaluated my method over a dataset generated through 7 ECUs with 6 pins having 185 records for each ECU and 40 records for each pin. The performance evaluation against state-of-the-art methods revealed that the proposed method achieved 99.4% accuracy for ECU-identification and 96.7% accuracy for pin-identification that signifies the reliability of the proposed approach.

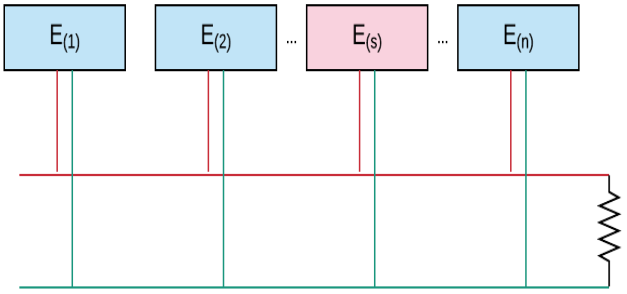


Figure 4.1: Generalized architecture of CAN

This approach exploits the uniqueness in device-specific distortions e.g., semiconductor impurities, DC offset, aliasing error, the mismatch between the nominal and measured values of electric components in digital to analog converter (DAC), etc., for message fingerprint generation. This research hypothesizes that distortions due to digital-to-analog conversion operation at the ECU output are device-dependent that can be used to link the received packet to the transmitting ECU. Therefore, I associate the received packet through a specific ECU, and the ECU-pin responsible for message transmission through a double neural network approach.

### 4.1.1 Research Objectives

The main objective of this research is: (i) to investigate ECU-level uniqueness for a given network and (ii) to investigate pin-level uniqueness for a given ECU to authenticate the message. The proposed method relies on distinctive physical artifacts of the $DAC$ of the transmitting ECU for device-level fingerprinting. The imperfections in material, design, fabrication of $DAC$ are contributing factors that create distortion in the ECU signal. I perform the statistical modeling of this distortion and use it as a feature vector for transmitter identification (i.e. transmitting ECU, and ECU-Pin) through neural network architecture. Thus, the main contributions of the chapter are:

- I provide a mathematical model of the distortion sources i.e. imperfections in

the material, design, fabrication of $DAC$.

- I propose a statistical model of the device (ECU)-level distortion for transmitter identification.
- I also propose that different transmitting pins in a single device have unique distortion and can be used for ECU-pin identification.

This chapter is organized as follows: Section 4.2 describes the system model, and outlines sources of device-specific distortion in the CAN signal. Experimental setup, dataset, performance measures, results, and analysis are provided in section 4.3, this section also provides the comparison of my method with the current state of the art research.

## 4.2 System Model

Fig. 4.1 shows a *subnet* in CAN that contains $\gamma = \{1, 2, \ldots, n\}$ ECUs represented as $E_{(i)}$, with a fingerprinting unit $E_{(s)}$ that sniffs the transmission of analog signal $y^{(a)}_{(i)}(t)$ by the ECU $E_{(i)}$, where $i \in \gamma$. The $E_{(s)}$ converts the signal $y^{(a)}_{(i)}(t)$ to a digital signal $y^{(a)}_{(i)}(n)$ with a sampling rate of 20Msa/sec. Afterwards, the $E_{(s)}$ computes the expected signal $y^{(e)}_{(i)}(n)$ that is used for distortion computation i.e. $d_{(i)}(n)$ for $E_{(i)}$, such that $\{d_{(i)}(n) \in n_{(i)} \mid n_{(i)} : [L \rightarrow U]\}$, where $L = -0.10$ and $U = 0.10$ represent the *lower* and *upper* distortion values respectively. The $d_{(i)}(n)$ is then used for feature extraction to generate the feature vector $\mathbf{x_{(r)}} = \{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$ for $E_{(i)}$, where $m$ represents the number of features, $r \in R$ and $R$ is the total number of records. The feature vector is then passed to a double neural network architecture, which is pre-trained on $R$ records for message authentication. Fig. 4.2 shows the architecture of the proposed method and is described in detail in following subsections.

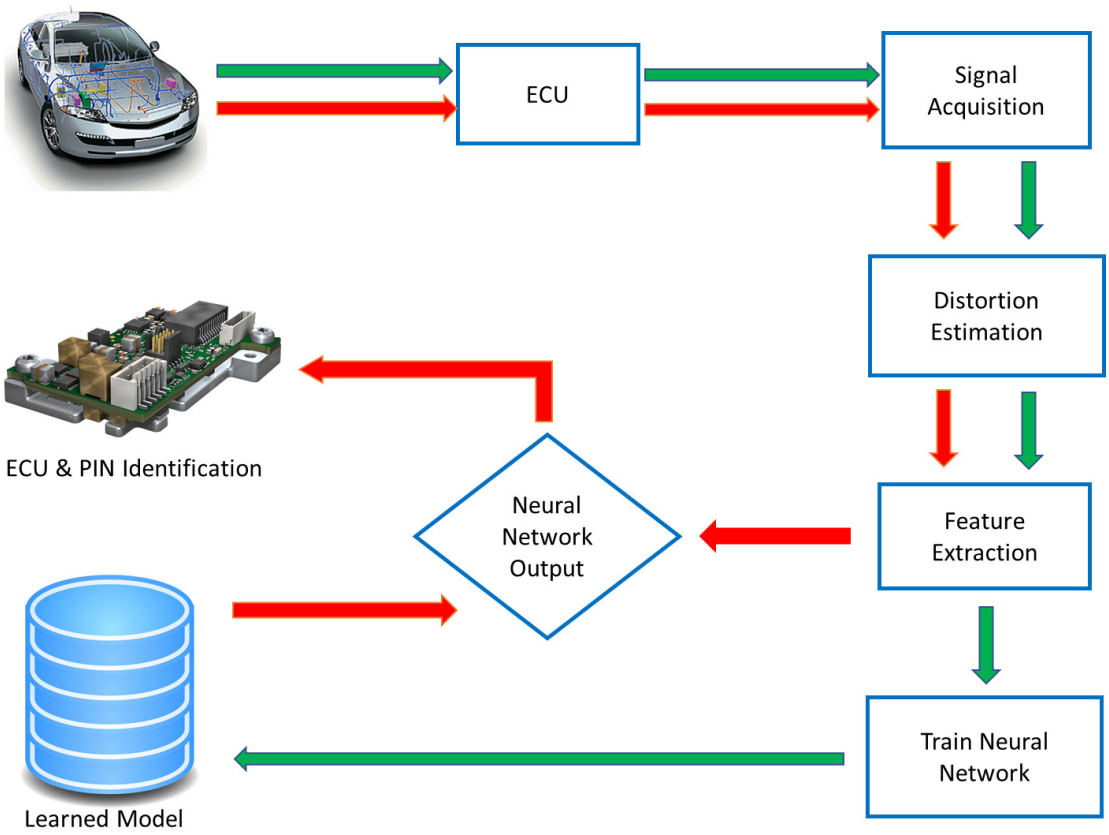


Figure 4.2: Block diagram of transmitter identification system

### 4.2.1 Signal Acquisition

The $E_{(s)}$ acquires the analog signal $y^{(a)}_{(i)}(t)$ generated by $E_{(i)}$ and converts this signal into digital signal $y^{(a)}_{(i)}(n)$ as represented in eq. (4.1).

$$y^{(a)}_{(i)}(n) = y^{(a)}_{(i)}(t)|_{t=nT_s}, T_s = 50 \times 10^{-9} \tag{4.1}$$

The $T_s = 50 \times 10^{-9}$ represents sampling time of $50\,nsec$ and sampling rate of $20\,MSa/sec$ for the signal. The reason to generate $y^{(a)}_{(i)}(n)$ is that the $y^{(a)}_{(i)}(t)$ occurs at infinite instants of time, thus demands large memory to get stored. However, as the $E_{(s)}$ has limited memory, therefore, the analog-to-digital conversion is performed. The number of bits required to store each sample are $n$, in my case, $n = 8\,bits$. In order to extract distortion, I compute the expected signal $y^{(e)}_{(i)}(n)$ from $y^{(a)}_{(i)}(n)$ using

eq. (4.2):

$$y_{(i)}^{(e)}(n) = \begin{cases} 3.5v & : \ 3.3 < y_{(i)}^{(e)}(n) < 3.7 \\ 2.5v & : \ Otherwise \end{cases} \tag{4.2}$$

The signal $y_{(i)}^{(e)}(n)$ is mapped to $3.5\,v$ if $y_{(i)}^{(a)}(n)$ is between $3.3\,v$ and $3.7\,v$ else will be $2.5\,v$. The signal $y_{(i)}^{(e)}(n)$ represents the ideal signal from $E_{(i)}$; however, it has been observed through extensive analysis of CAN communication signals, that the actual signal levels differ from the expected signal levels. As shown in Fig. 4.3, the waveform of the CAN signal captured using a $DS1012A$ oscilloscope for the $y_{(i)}^{(a)}(n)$, significantly differs from the expected signal $y_{(i)}^{(e)}(n)$ due to the distortion mainly attributed to semiconductor impurities, mismatch between nominal values and measured values of electric components, aliasing error of finite impulse response (FIR) filter and DC offset of $DAC$. Moreover, these imperfections are device-specific, hence, they can be used for fingerprinting of the ECUs.

### 4.2.2 Distortion Extraction

The imperfections observed in the signal acquisition stage are used for fingerprinting the ECUs. The fingerprinting is quantified in the form of distortion modeling which is acquired in *Density Estimation* stage. Before distortion modeling, I acquire distortion as shown in Fig. 4.4, which is represented in eq. (4.3).

$$d_{(i)}(n) = y_{(i)}^{(a)}(n) - y_{(i)}^{(e)}(n). \tag{4.3}$$

There are four main reasons for distortions which are discussed as follows:

**Mismatch of nominal and measured values of electric components:**

Imperfections in the electric components are one of the sources of $d_{(i)}(n)$. These imperfections can be described as a deviation of the measured values of electric components from their nominal values. Let $R_{o_{(i)}}$ be the value of the feedback resistor of

the $E_{(i)}$ and $\delta_{R_{(i)}}$ represents the deviation from the nominal value, commonly known as the tolerance level. The actual resistance $R_{a_{(i)}}$ can then be expressed as eq. (4.4) which is as follows:

$$R_{a_{(i)}} = R_{o_{(i)}} + \delta_{R_{(i)}}. \tag{4.4}$$

Let $d^{(R)}_{(i)}(n)$ represent the distortion due to $\delta_{R_{(i)}}$, which is the first cause of distortion at the $DAC$ output. The purpose of $DAC$ in ECU is to convert bits into a physical signal, which is in the form of voltage as shown in Fig. 4.5. The reason for this conversion is that the signal propagates through a channel in the form of the physical signal.

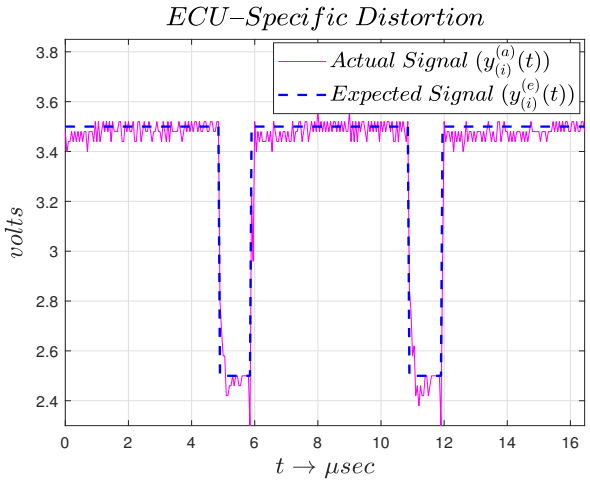


Figure 4.3: Screenshots of the actual and expected waveforms

**Semiconductor impurities:**

During the manufacturing process, the amount of impurities in the semiconductors cannot be completely removed. The imperfections in the device material and fabrication process are the other contributing factors to the observed distortion. Impurities in the semiconductor cause flicker distortion (also known as 1/f noise) at the $DAC$ output (84). I represent the distortion due to semiconductor impurities as $d^{(f)}_{(i)}(n)$, which is another cause of distortion at the $DAC$ output.

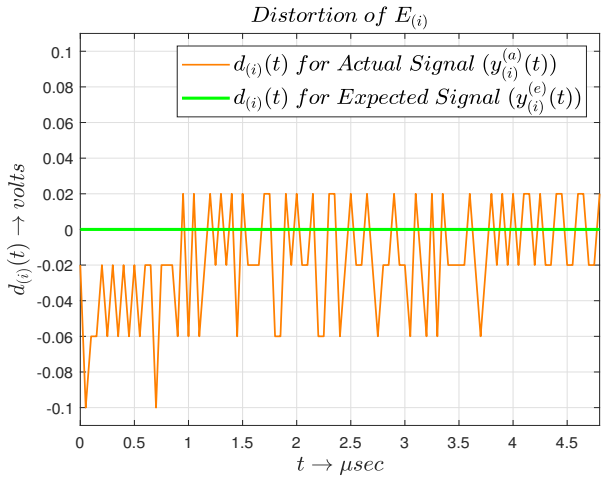


Figure 4.4: Distortion in the ECU signal

**Non-ideal behavior of low pass filter:**

Another contributing factor in distortion is aliasing error that occurs due to non-ideal behavior of *low pass filter* used in digital to analog conversion. The digital signal, which is acquired through the sampling operation on a continuous signal introduces the periodic repetition of its spectra. Let $Y^{(e)}_{(i)_D}(f)$ represent the Fourier transform of $y^{(e)}_{(i)}(n)$, which is input in $DAC$ and $Y^{(e)}_{(i)}(f)$ represent the Fourier transform in form of continuous signal. Low pass filtering is used to filter out $Y^{(e)}_{(i)}(f)$ from $Y^{(e)}_{(i)_D}(f)$ to avoid unwanted copies, which is commonly implemented using a finite impulse response (FIR) filter realization. Eq. (4.5) represents the relationship between input and output of DAC,

$$Y^{(e)}_{(i)_D}(f) = \sum_{n=-\infty}^{\infty} Y^{(e)}_{(i)}(f - nf_s) \tag{4.5}$$

Let $H(f)$ represent the transfer function of the FIR filter, the output of the FIR filter can be expressed through eq. (4.6).

$$Y_{(i)_{FIR}}(f) = Y^{(e)}_{(i)_D}(f).H(f) \tag{4.6}$$

Ideally $Y_{(i)_{FIR}}(f)$ should be same as $Y^{(e)}_{(i)}(f)$, but due to aliasing error these values

differs. The non-ideal behavior of FIR realization introduces aliasing at the $DAC$ output. Let $d_{(i)}^{(a)}(n)$ denote distortion due to the non-ideal behavior of the low-pass filter realization for the $E_{(i)}$. This is the third cause of distortion at the output of $DAC$.

**DC offset error:**

The DC offset error in the $DAC$ is another source of distortion (85). Ideally, the dominant bit level $= 3.5\,V$ and ideal recessive bit level $= 2.5\,V$ as shown in Fig. 4.5, but, DC offset $d_{(i)}^{(o)}(n)$ is added to the ideal voltage value due to grounding issues in $DAC$. The total distortion $d_{(i)}(n)$ due to $DAC$ for the $E_{(i)}$ can be expressed as eq. (4.7),

$$d_{(i)}(n) = d_{(i)}^{(R)}(n) + d_{(i)}^{(f)}(n) + d_{(i)}^{(a)}(n) + d_{(i)}^{(o)}(n). \tag{4.7}$$

Similarly, we can derive from eq. (4.3) that:

$$y_{(i)}^{(a)}(n) = y_{(i)}^{(e)}(n) + d_{(i)}(n). \tag{4.8}$$

Therefore, eq. (4.8) validates my hypothesis that distortion added in the received signal is dynamic, hence it can be an effective measure for fingerprinting the ECUs.

The device-specific distortion is also unique for each pin within the $E_{(i)}$ that can be represented as $E_{(i,l)}$; where $\{\ l \in \zeta \mid \zeta = 1, 2, \ldots, \lambda\ \}$ are total No. of pins within $E_{(i)}$. This also elaborates that for attack modeling I need to determine the affected ECU and the relevant pin. Additionally, pin-level artifacts are effective to detect spoofing attacks launched from a different pin of the same ECU. Recently, Sang et. al. (86) demonstrated voltage-based attack to permanently damage a pin of the target ECU. A voltage-based attack is launched by an adversary, which sends a high voltage through a pin (which has the maximum capacity of $5\,V$) to damage the pin permanently. However, by fingerprinting the $E_{(i,l)}$ we can avoid these attacks.

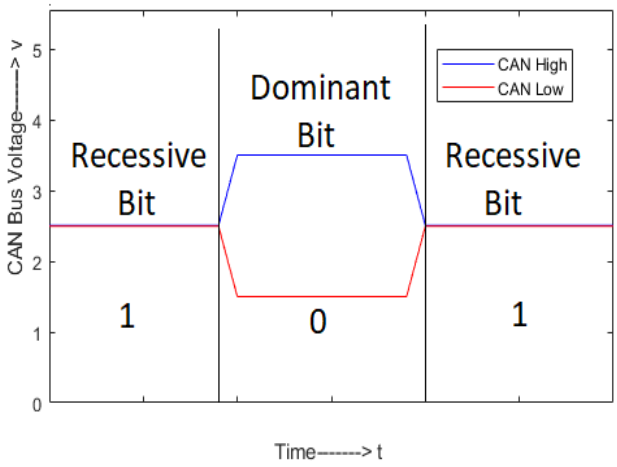


Figure 4.5: Ideal voltage levels for CAN

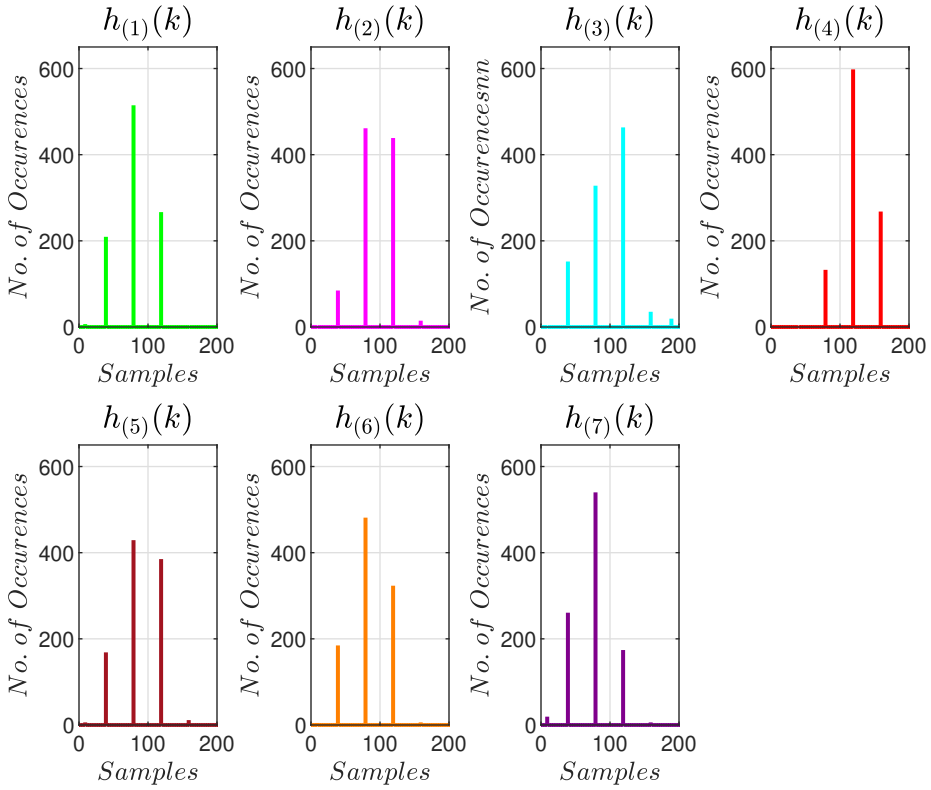


Figure 4.6: Histograms of $d_{(i)}(n)$ for ECU $E_{(1)}$-$E_{(7)}$

### 4.2.3 Density Estimation

After distortion modeling, I use $d_{(i)}(n)$ for histogram generation. The histogram will then be used as fingerprints for $E_{(i)}$ and $E_{(i,l)}$. In order to make histogram I need

to group the distortion values in $m$ histogram bins with step size $\beta$, which can be computed as in eq. (4.9).

$$\beta = \left(\frac{U - L}{m}\right) \tag{4.9}$$

Where U, and L represents the lower and upper values for distortion, and m represents the number of bins. In my case $m = 200$, thus, step size $\beta$ becomes $10^{-3}$. The histogram $h_{(i)}(k)$ for $N = 1500$ samples of $d_{(i)}(n)$ is computed using eq. (4.10). Here, $k = \{0, 1, 2, \ldots, m\}$.

$$h_{(i)}(k) = \sum_{n=1}^{N} \left[\delta\left(\frac{d_{(i)}(n)}{\beta} + 100\right) + h_{(i)}(k)\right] \tag{4.10}$$

Where $\delta(.)$ denotes Kronecker delta function (87), that can be computed through eq. (4.11).

$$\delta(n–k) = \begin{cases} 1 & : \quad n = k \\ 0 & : \quad Otherwise \end{cases} \tag{4.11}$$

Fig. 4.6 shows the histogram $h_{(i)}(k)$ of $E_{(1)}$–$E_{(7)}$. Afterwards $h_{(i)}(k)$ is used as feature set $X_{(i)} = \{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$.

### 4.2.4 ANN based Model Learning

A double artificial neural network (ANN)-based model is used to identify the source ECU and the corresponding pin. For this the ANN gets $X_{(r)} = \{\mathbf{x}_{(1)}, \mathbf{x}_{(2)}, \ldots, \mathbf{x}_{(R)}\}$ as input-set, and corresponding ECU-, and pin-labels $Y_{(r,e)} = \{y_{(1,e)}, y_{(2,e)}, \ldots, y_{(R,e)}\}$ $Y_{(r,p)} = \{y_{(1,p)}, y_{(2,p)}, \ldots, y_{(R,p)}\}$ respectively; and it predicts the ECU-labels and corresponding pin-labels as $\hat{Y}_{(r,e)} = \{\hat{y}_{(1,e)}, \hat{y}_{(2,e)}, \ldots, \hat{y}_{(R,e)}\}$ and $\hat{Y}_{(r,p)} = \{\hat{y}_{(1,p)}, \hat{y}_{(2,p)}, \ldots, \hat{y}_{(R,p)}\}$ respectively, where $e \in \gamma$, $p \in \zeta$, and $r \in R$. The classifier is trained on the dataset with three hidden layers having $v$ neurons (in my case $v = 10$) and "scaled conjugate

gradient back propagation" method for weight optimization. In the training phase, this model learns the weight vector represented by: $\omega_{\mathbf{e}} = \{w^{(1)}, w^{(2)}, \ldots, w^{(m)}\}$ for all $E_{(i)}$ and $\omega_{\mathbf{p}} = \{w^{(1)}, w^{(2)}, \ldots, w^{(m)}\}$ for all $E_{(i,l)}$. The output of both networks are then merged as final output in the testing phase (Fig. 4.7). The ANN-architecture for ECU recognition is presented in Table 4.1, and same architecture is used for ECU-pin recognition as well.

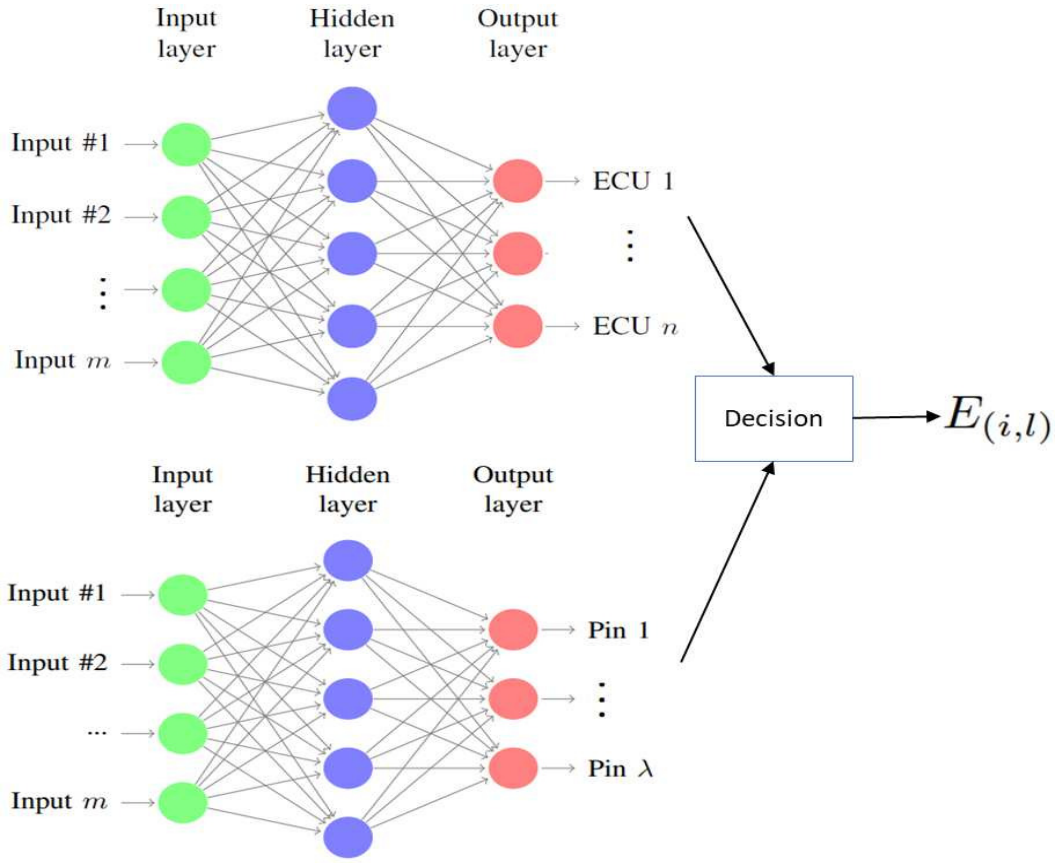


Figure 4.7: Merged neural network structure

Table 4.1: Summary of neural network structure

| | | | |
|---|---|---|---|
| | INPUT | | |
| Input: | $\overrightarrow{x_{(r)}} = (x^{(1)}, x^{(2)}, \ldots, x^{(m)})$ | $\dim(\overrightarrow{x}) = 1 \times m$ | (I.1) |
| | MIDDLE (HIDDEN) LAYERS (1–3) | | |
| Input: | $\overrightarrow{b} = \mathrm{U}\overrightarrow{X}$ | $\dim(\overrightarrow{b}) = 1 \times v$ | (I.2) |
| Output: | $\overrightarrow{c} = \mathrm{f(b)}$ | $\dim(\overrightarrow{c}) = 1 \times v$ | (I.3) |
| U: | $m \times v$ weight matrix | | |
| f: | $\frac{1}{1+e^{-b}}$ | | |
| | OUTPUT LAYER | | |
| Input: | $\overrightarrow{d} = w_e \overrightarrow{c}$ | $\dim(\overrightarrow{d}) = 1 \times v$ | (I.4) |
| Output: | $\overrightarrow{e} = \mathrm{g(d)}$ | $\dim(\overrightarrow{e}) = 1 \times v$ | (I.5) |
| $w_e$ : | $v \times m$ weight matrix | | |
| g: | $\frac{1}{1+e^{-d}}$ | | |
| | ERROR CORRECTION | | |
| Cost: | $\mathrm{E} = -\sum_{i=1}^{n}[y_{(r,e)} log(c)]$ | | (I.6) |
| $\Delta W_{ij}$ | $= -\alpha \partial E / \partial W_{vm} = \alpha \delta_i c_j$ | | (I.7) |
| $\Delta U_{mv}$ | $= -\beta \partial E / \partial U_{mv}$ | | (I.8) |

## 4.3 Experiments and Results

### 4.3.1 Experimental Setup

The proposed approach evaluates inter-class- (amongst ECUs), and intra-class- (amongst ECU-pins) variability, for message authentication. For inter-class variability, seven ECUs (transmitters) of the same make and model was used in this study and data was recorded through the CANH pin. For intra-class variability, six $DAC$ pins of the same ECU were analyzed to determine the pin-level characteristics.

The hardware comprised of seven Arduino UNO-R2 micro-controller kits; 7 CAN-Bus shield boards with $MCP\,2515$ CAN-bus controllers, $MPC\,2551$ CAN transceivers; and a $DSO1012A$ oscilloscope to record the voltage samples with a sampling rate of 20Msa/s, with 100 $MHz$ bandwidth. $Matlab\,R\,2018a$ software was used for statistical data analysis of the sampled signals. A computer simulation was written that con-

tinuously transmitted the messages from different ECUs and pins. Afterward, these messages were then used as the dataset for model training and evaluation.

### 4.3.2 Dataset Description

The ECU identification dataset comprised of 1295 ($7 \times 185$) records with 1500 samples in each record. Whereas, for pin-level identification, a dataset was collected for six different pins of each transmitting ECU with 40 records for each pin. The dataset used here was collected in the same environment, i.e., under the same temperature and using an identical message to observe the unique variations of the digital signals. For performance evaluation, 70% of a randomly selected dataset was used for training and the 30% of the data for testing purposes.

### 4.3.3 Performance Evaluation Measures

For performance evaluation, I used true positive (TP), true negative (TN), false positive (FP), false negative (FN), precision, recall, $F_1\,score$, accuracy, and error rate as performance evaluation measures. To evaluate the effectiveness of the proposed method, I determined how many ECUs were correctly identified in the response to messages sniffed by $E_{(s)}$. Let $TP$ represents true positive rate, $FP$ represents false positive rate, $TN$ represents true negative rate, and $FN$ represents false-negative rate, then precision can be defined as follows:

$$Precision = \left(\frac{TP}{TP + FP}\right) \tag{4.12}$$

Precision was used to measure the ratio of the true instances against the retrieved instances for a particular class. To measure the sensitivity I used the recall rates that can be computed as follows:

$$Recall = \left( \frac{TP}{TP + FN} \right) \tag{4.13}$$

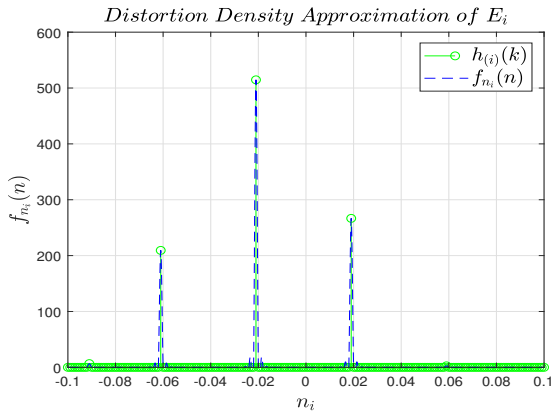


Figure 4.8: Density approximation of distortion

The recall was computed to measure the total number of relevant instances that were actually retrieved. In order to combine both measures i.e. precision, and recall I used $F_1\, Score$ that was computed as:

$$F_1\, score = 2 \times \left( \frac{Precision \times Recall}{Precision + Recall} \right) \tag{4.14}$$

The higher $F_1\, Score$ signifies the robustness of the classification approach. In order to evaluate the overall performance by considering all the classes together, I computed the accuracy of the method as follows:

$$Accuracy = \left( \frac{TP + TN}{TP + TN + FP + FN} \right) \tag{4.15}$$

Accuracy was computed to measure all instances that were correctly classified, despite the fact, whatever class they belong to. Moreover, by using accuracy value I also computed the overall error rate of the method as follows:

$$Error\text{–}rate = 1 - Accuracy \tag{4.16}$$

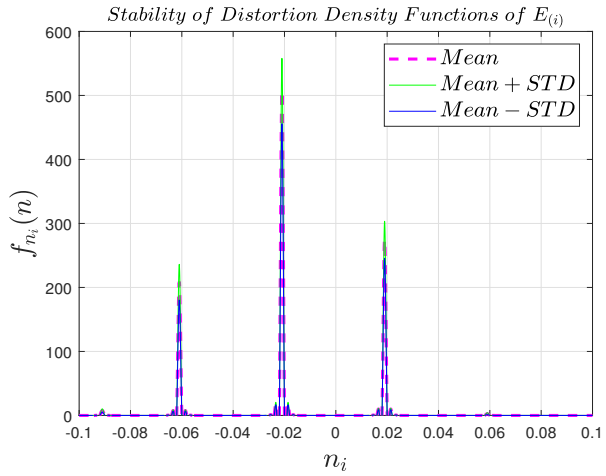


Figure 4.9: Effectiveness of feature-set

### 4.3.4 Feature Stability

The purpose of this experiment is to validate that different ECUs, even of the same make and model, introduce different artifacts while transmitting an identical message; and this uniqueness can be exploited to counter the spoofing attacks. To achieve this goal, all ECUs transmitted the same messages over the same channel with constant settings regarding temperature, and environment. To validate the claim of ECU-specific distortion, data were recorded for each ECU with identical channel inputs and transmission parameters. To verify uniqueness, I estimated distortion density function by applying Spline function (88) over histogram $h_{(i)}(k)$ to get $f_{n_i}(n)$ as shown in Fig. 4.8. Estimated distortion distribution represents the physical characteristics of each ECU. In order to find the stability, I generated $f_{n_i}(n)$ of each ECU for 100 times and computed mean and standard deviation (STD). From Fig. 4.9, it can be observed that the difference between the mean and mean $\pm$ STD is negligible, which shows that the feature-set remains constant over time for each ECU. Hence, it is proved that the proposed feature extraction approach induces the unique attributes for ECU representation that makes it effective for ECU identification.

To further validate the unique attribute of the proposed method I plotted $f_{n_i}(n)$

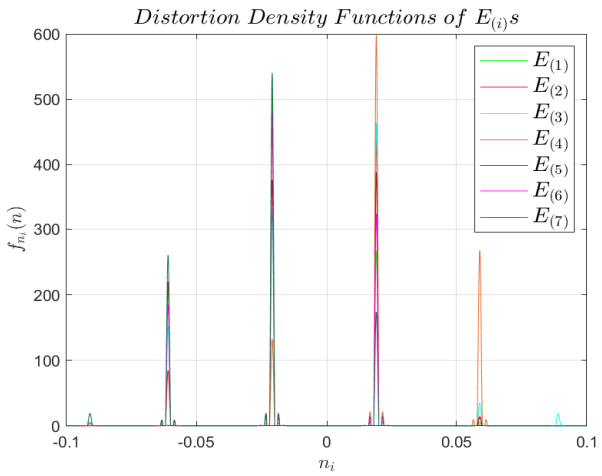


Figure 4.10: Estimated distortion distributions for all 7 ECUs

for seven ECUs as shown in Fig. 4.10, which clearly shows that each ECU has a unique representation. The benefit of the uniqueness is that the attacker cannot replicate an ECU's profile, thus signifies that my approach is robust against spoofing attacks.

### 4.3.5 ECU-Level Identification

In this experiment, I have evaluated the performance of the proposed method in terms of ECU identification. From Table 4.2 it can be observed that the proposed method achieves very high accuracy for ECU classification. The high accuracy signifies that the distortion introduced in each ECU due to *DAC* imperfections and semiconductor impurities is unique thus results in high accuracy for ECU identification. Moreover, it also justifies my hypothesis that the distortion due to *DAC* and semi-conductor impurities has the potential for ECU fingerprinting for attack detection. It can be observed from the Table 4.2 that $E_{(1)}$, $E_{(4)}$ and $E_{(7)}$ have 100% detection rates, that is mainly associated with the high distortion values appeared in form of high peaks as shown in Fig. 4.6. Moreover, by analyzing Table 4.2 in the perspective of Fig. 4.6, it can also be observed that if distortion is concentrated in a certain region, it increases the inter-class variability, which is one of the targets of

this research.

Table 4.2: Confusion matrix for ECU classifier

| | | *Target Class* | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| - | - | $E_{(1)}$ | $E_{(2)}$ | $E_{(3)}$ | $E_{(4)}$ | $E_{(5)}$ | $E_{(6)}$ | $E_{(7)}$ | *Total %* |
| *Predicted Class* | $E_{(1)}$ | 185 | 0 | 0 | 0 | 0 | 0 | 0 | 100 |
| | $E_{(2)}$ | 0 | 184 | 2 | 0 | 0 | 4 | 0 | 98.9 |
| | $E_{(3)}$ | 0 | 1 | 183 | 0 | 0 | 0 | 0 | 99.5 |
| | $E_{(4)}$ | 0 | 0 | 0 | 185 | 0 | 0 | 0 | 100 |
| | $E_{(5)}$ | 0 | 0 | 0 | 0 | 183 | 3 | 0 | 98.4 |
| | $E_{(6)}$ | 0 | 0 | 0 | 0 | 2 | 182 | 0 | 98.9 |
| | $E_{(7)}$ | 0 | 0 | 0 | 0 | 0 | 0 | 185 | 100 |
| | *Total %* | 100 | 99.5 | 98.9 | 100 | 98.9 | 98.4 | 100 | 99.4 |

Similarly, performance matrix (PM) Table 4.3 shows that $E_{(1)}$, $E_{(4)}$ and $E_{(7)}$ has 100% precision, recall, accuracy and $F_1\,Score$ rates. Furthermore, $E_{(2)}$ has 99.5% recall, but as it has only one false negative record, therefore, other performance evaluation rate slightly drops. On the other hand, $E_{(3)}$ has 99.5% precision, it has only one false positive, therefore slightly affects the precision.

Table 4.3: Performance matrix of ECU classifier

| - | *Precision* | *Recall* | *Accuracy* | $F_1\,Score$ | *ERR* |
|---|---|---|---|---|---|
| $E_{(1)}$ | 100% | 100% | 100% | 100% | 0% |
| $E_{(2)}$ | 98.9% | 99.5% | 99.8% | 99.2% | 0.2% |
| $E_{(3)}$ | 99.5% | 98.9% | 99.8% | 99.2% | 0.2% |
| $E_{(4)}$ | 100% | 100% | 100% | 100% | 0% |
| $E_{(5)}$ | 98.4% | 98.9% | 99.6% | 98.6% | 0.4% |
| $E_{(6)}$ | 98.9% | 98.4% | 99.6% | 98.6% | 0.4% |
| $E_{(7)}$ | 100% | 100% | 100% | 100% | 0% |

Fig. 4.11 shows the graphical representation of the ECU identification results. The high correlation amongst the performance evaluation measures for all ECUs clearly signifies the reliability of the proposed method. Furthermore, the area under the curve analysis of the receiver operating characteristic (ROC) plots as shown in Fig. 4.12

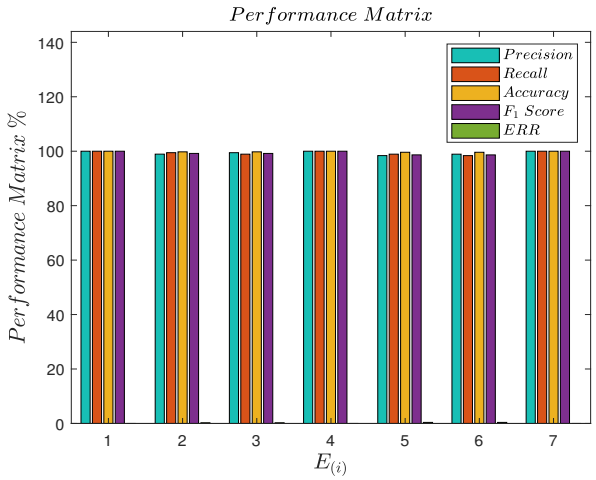


Figure 4.11: Bar graph of PM for ECU classifier

also confirms my claim that distortion is an effective measure for ECU fingerprinting, hence attack detection.

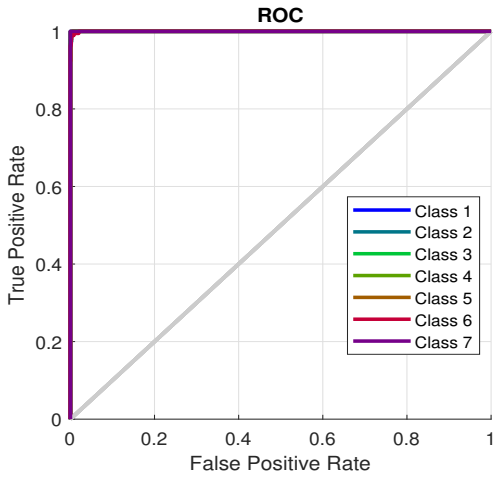


Figure 4.12: ROC of $E_{(i)}$

### 4.3.6 Pin-Level Identification

The purpose of this experiment is to validate that different pins, even from the same transmitter, introduce different artifacts into the transmitted signal, and the pin-level fingerprinting can be utilized for a reliable attack profile generation. To achieve this objective, 6 pins of the same transmitter were used to transmit the same

message over the same channel with constant settings e.g. temperature, environment, etc. Table 4.4 shows the classification performance of the proposed method in terms of a number of samples per class.

Table 4.4: Confusion matrix for pin classification

| | | *Target Class* | | | | | | |
|---|---|---|---|---|---|---|---|---|
| - | - | $Pin_{(1)}$ | $Pin_{(2)}$ | $Pin_{(3)}$ | $Pin_{(4)}$ | $Pin_{(5)}$ | $Pin_{(6)}$ | $Total\,\%$ |
| *Predicted Class* | $Pin_{(1)}$ | 38 | 2 | 0 | 0 | 0 | 1 | 92.7 |
| | $Pin_{(2)}$ | 2 | 37 | 0 | 0 | 0 | 0 | 94.9 |
| | $Pin_{(3)}$ | 0 | 0 | 40 | 0 | 0 | 0 | 100 |
| | $Pin_{(4)}$ | 0 | 0 | 0 | 40 | 0 | 0 | 100 |
| | $Pin_{(5)}$ | 0 | 0 | 0 | 0 | 39 | 1 | 97.5 |
| | $Pin_{(6)}$ | 0 | 1 | 0 | 0 | 1 | 38 | 95 |
| | $Total\,\%$ | 95 | 92.5 | 100 | 100 | 97.5 | 95 | 96.7 |

Table 4.5: Performance matrix of pin classification

| - | *Precision* | *Recall* | *Accuracy* | $F_1\,Score$ | *ERR* |
|---|---|---|---|---|---|
| $Pin_{(1)}$ | 92.7% | 95% | 97.9% | 93.8% | 2.1% |
| $Pin_{(2)}$ | 94.9% | 92.5% | 97.9% | 93.7% | 2.1% |
| $Pin_{(3)}$ | 100% | 100% | 100% | 100% | 0% |
| $Pin_{(4)}$ | 100% | 100% | 100% | 100% | 0% |
| $Pin_{(5)}$ | 97.5% | 97.5% | 99.1% | 97.5% | 0.9% |
| $Pin_{(6)}$ | 95% | 95% | 98.3% | 95% | 1.7% |

Table 4.5 shows the performance in terms of different performance evaluation measures. The performance of the pin-classifier is quantified in terms of precision, recall, $F_1\,score$, accuracy, and error rate. In 2 out of 6 cases, my method achieved 100 % precision, recall rate, accuracy and $F_1\,score$. whereas, overall 96.7% pin detection rate. Although, the pin detection rates are slightly lower than ECU detection rates, as the pin-detection is a novel concept, so the research efforts can be done in this area to further generate more interesting findings. The same results are graphically presented in Fig. 4.13. The area under the curve results as presented in Fig. 4.14 shows that still the pin level detection of the proposed approach is satisfactory.

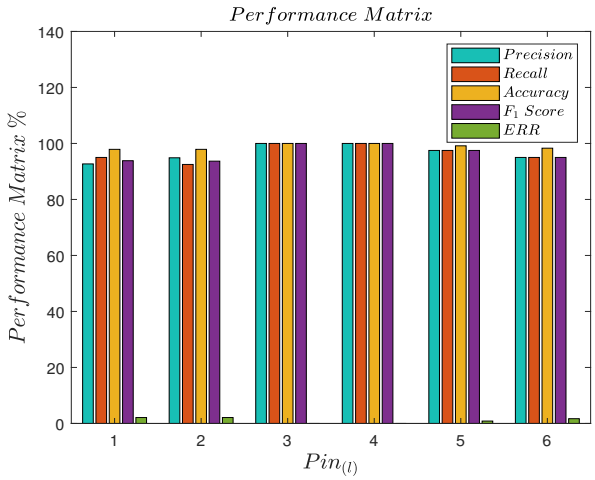


Figure 4.13: Bar graph of PM for pin classifier

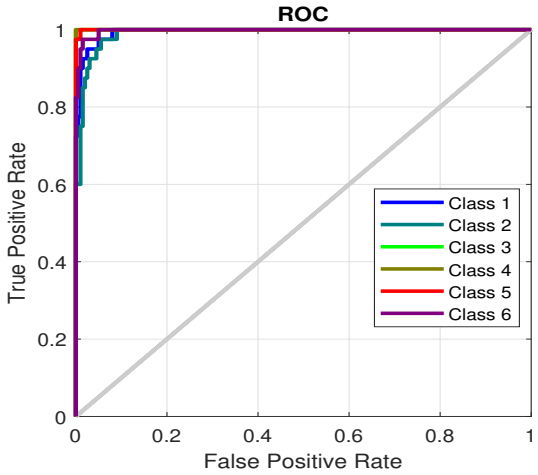


Figure 4.14: ROC of pin classifier

### 4.3.7 Comparison Against State-of-the-art

In this section proposed method is compared against state-of-the-art methods that are also doing the ECU identification. The performance is compared against ECU detection using Viden (25), Inimitable characteristics of CAN signal (43) and VoltageIDS (44).

In (25), Cho et. al. proposed a method named *Viden* that used voltage profile of acknowledgment (*Ack*) bits for transmitter identification. In the first phase, ACK bit was used to measure the message was originated from the genuine transmitter or not.

Afterward, voltage measurements were used to generate ECU fingerprints. Based on these fingerprints, the attacker ECU was identified. In (43), a monitoring unit was installed in the vehicle that analyzes the electrical CAN signals and computes the statistical features. These features were then classified to identify the ECU. In (44), ECU detection based on inimitable voltage characteristics technique was proposed. The feature vectors proposed in (43) were extended both in time- and frequency domains, and were classified for ECU identification in (44).

Table 4.6: Comparison with other methods

| **Research Work** | **Method** | **Accuracy** |
|---|---|---|
| Cho et. al. (25) | *Viden* | 99.57% |
| Choi et. al. (43) | *Inimitable Char. of CAN Signal* | 96.48% |
| Choi et. al. (44) | *VoltageIDS* | 95.54% |
| This method | *Distortion based IDS* | 99.4% |

Table 4.6 shows the performance comparison of my method against (44; 43; 25). From the results, it can be observed that my method is giving higher accuracy as compared to (44; 43) whereas, it is giving almost the same performance as (25). However, the main advantage of my method is that feature extraction and message authentication can be done in any part of the signal without the latency. Whereas in case of Viden, voltage profile is estimated for message authentication during the reception of the ACK bit but it also introduces the latency. Hence, from the aspect of latency, my method is more robust than the Viden.

# CHAPTER V

# Control System Parameters Based Transmitter Identification

## 5.1 Introduction

Fully connected autonomous vehicles are more vulnerable than ever to hacking and data theft. The controller area network (CAN) protocol is used for communication between in-vehicle control networks (IVN). The absence of basic security features of this protocol, like message authentication, makes it quite vulnerable to a wide range of attacks including spoofing attacks. As traditional cybersecurity methods impose limitations in ensuring confidentiality and integrity of transmitted messages via CAN, a new technique has emerged among others to approve its reliability in fully authenticating the CAN messages. At the physical layer of the communication system, the method of fingerprinting the messages is implemented to link the received signal to the transmitting electronic control unit (ECU). This chapter introduces a new method to implement the security of modern electric vehicles. The lumped element model is used to characterize the channel-specific step response. ECU and channel imperfections lead to a unique transfer function for each transmitter. Due to the unique transfer function, the step response for each transmitter is unique. In this chapter, control system parameters are used as a feature-set, afterward, a

neural network is used transmitting node identification for message authentication. A dataset collected from a CAN network with eight-channel lengths and eight ECUs to evaluate the performance of the suggested method. Detection results show that the proposed method achieves an accuracy of 97.4% of transmitter detection.

In the modern electric vehicle, CAN is used as a reliable, robust, and simple network protocol for in-vehicle communication. CAN is widely applicable in automotive industry applications, and it makes the transmission of the messages between the ECUs and the devices more organized and controllable. This protocol lacks message authentication, because the sender information in the packet is missing, which makes it vulnerable to spoofing attacks. In the past few years, security and safety are some of the major concerns for the vehicle industry. In 2015, Miller et. al. (50) killed the vehicle engine in the middle of the highway, and they were able to take control of the vital functionalities of the vehicle like engine, braking unit, etc. The purpose of this experiment (50) was to convey a message to the stakeholders that the implementation of security in the electric car is a vital issue. With the advent of autonomous vehicles (AVs), interest in the safety and security of vehicles is increasing. These autonomous vehicles are considered a breakthrough in modern technology; however, they come with cyber vulnerability risks.

Existing state-of-the-art countermeasures against spoofing and impersonation attacks on in-vehicle CAN networks can be divided into (i) message authentication code (MAC) based approaches (18; 89; 20; 21; 22; 23; 24; 90; 91; 92; 17), and (ii) intrusion detection based approaches (25; 26; 44; 27; 28; 73; 74; 31; 30; 29; 32; 33; 34; 35; 36; 37; 38; 39; 40; 41; 42; 93; 43; 45; 46; 94). Intrusion detection based approaches can be further subdivided into, (a) fingerprinting-based approach (26; 25; 44), (b) parameter monitoring-based approach (73; 74; 31; 30; 29), (c) information theory-based approach (32; 33; 34), and (d) machine learning-based approach (37; 36; 35; 40; 41; 38; 39). Current solutions (17; 49) for CAN communication are limited in their ability and

scope as they are unable to identify the transmitting ECU responsible for the received packet. This chapter presents a novel approach to identify the source ECU of a CAN signal to authenticate the message. The idea of fingerprinting the source is also used for other electronic devices such as microphones(77; 78). In CAN communication, when a transmitter sends a message, the expected signal at the receiver is a rectangular waveform. However, due to channel imperfections, it is not a perfect rectangular waveform. The channel has a transfer function, and the input to this channel is a rectangular wave, which can be modeled as a step function. Hence, the input to the receiver can be quantified as the step response of the channel, through which the signal propagates. Fig. 5.1 shows the step response of the channel to the CAN signal. It can be observed that the received signal is not the same as the ideal signal. This chapter proposes a new technique to address the limitations of the existing state of the art CAN security methods by exploiting the channel based step response; and then finding the transient, and steady-state parameters like overshoot, peak time, settling time, peak value, steady-state value, damping ratio and natural frequency. Furthermore, these parameters are used as inputs to a neural network classifier using supervised learning to authenticate a message by identifying its transmitter.

*Contributions:* The main contributions of this chapter are:

- Modeling of channel-specific step response for CAN signals
- Channel-specific step response uniqueness analysis
- Propose a reliable framework for identifying the source ECU of any given CAN message

The rest of this chapter is structured as follows: Section 5.2 provides mathematical modeling of channel-specific distortion and its uniqueness analysis; it also provides mathematical modeling of the system; section 5.3 outlines the experimental setup, data collection, results, and analysis.

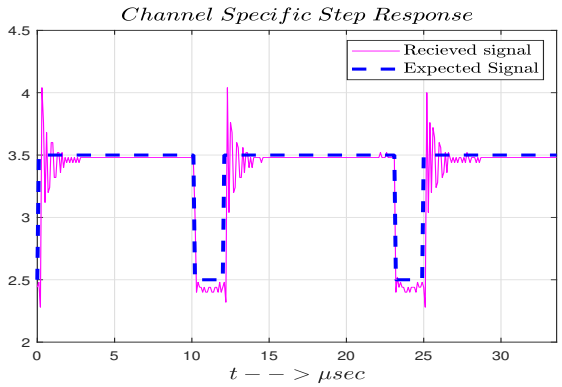


Figure 5.1: Channel specific step response

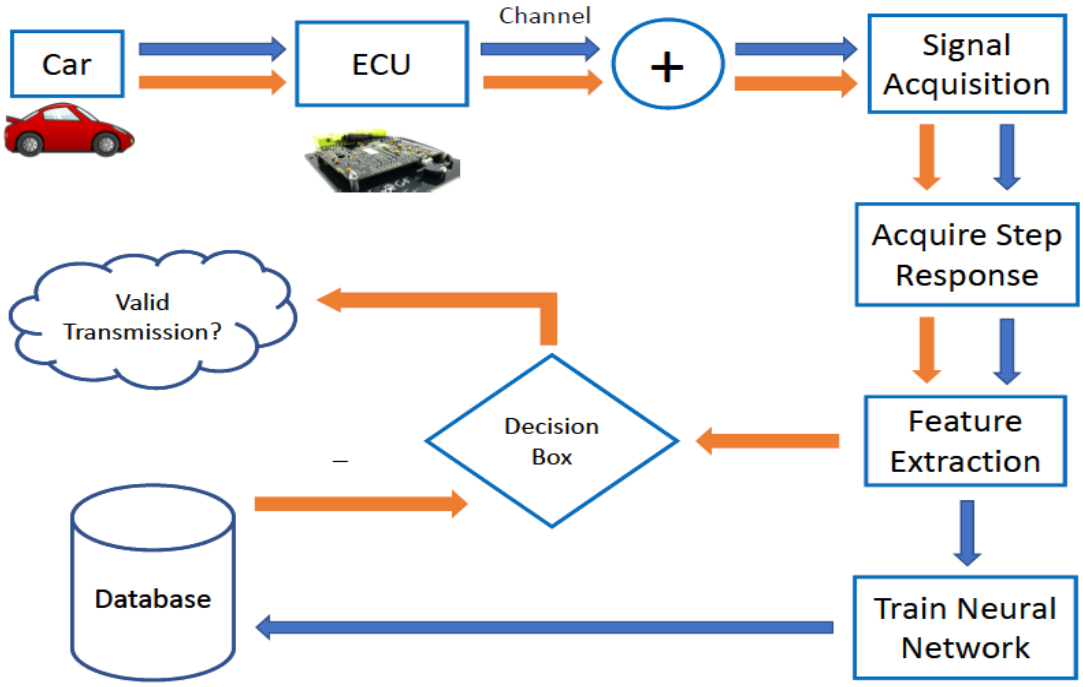


Figure 5.2: Block diagram of transmitter identification system

## 5.2 System and Mathematical Modelling

Fig. 5.2 shows the architecture of the method presented for transmitter identification for message authentication. This architecture is deployed at fingerprinting ECU $E_{(FP)}$ as shown in Fig. 5.3. In this architecture, the subnet has a total of $N$ ECUs connected to $N$ channels, where any ECU can be represented as $E_{(i)}$, $\lambda = \{1, 2, \ldots, N\}$, where $i \in \lambda$. The fingerprinting ECU $E_{(FP)}$ sniffs the signal transmitted by all trans-

mitters, and the response due to each channel at the $E_{(FP)}$ is then used for feature extraction to generate the feature vector $\mathbf{X_{(r)}} = \{x^{(1)}, x^{(2)}, \ldots, x^{(M)}\}$ for $E_{(i)}$, where $M$ represents the number of features, $r \in R$ and $R$ is the total number of records. The feature vector is then passed to a neural network architecture, which is pre-trained on $R$ records for message authentication.

### 5.2.1 Channel Modeling

The signal transmitted by $E_{(i)}$ represented by $x_{(i)}(t)$ propagates through the $i^{th}$ channel ($h_{(i)}(t)$), the channel behaves like a linear time-invariant system, the output of the channel is represented as $y_{(i)}^{(a)}(t)$. The relationship between $x_{(i)}(t)$ and $y_{(i)}^{(a)}(t)$ can be represented as eq. (5.1). Here, '$*$' operator represents convolution.

$$y_{(i)}^{(a)}(t) = x_{(i)}(t) * h_{(i)}(t). \tag{5.1}$$

Shown in the right column of Fig. 5.4 is the equivalent circuit of an infinitesimally small piece of a transmission line. According to the LEM, the transmission line is represented as a series resistance (R'), series Inductance (L'), a parallel Conductance (G') and a parallel capacitance (C'). Shown in the left column of Fig. 5.4 is the physical structure of the CAN channel. In this structure, $D$ is the distance between 2 wires, $d$ is the diameter of the wires. Let $\mu$ be the permeability, and $\sigma$ be the conductivity of the copper. The ideal line parameters R', C', L' and G' (95) can be

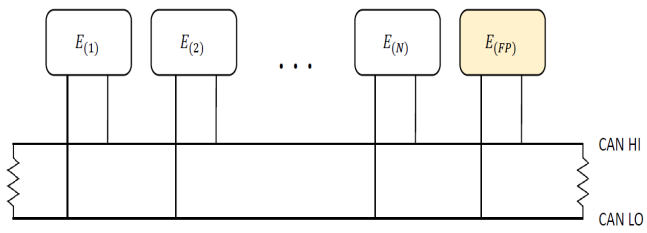


Figure 5.3: Message authentication by $E_{(FP)}$

expressed in the following equations:

$$R' = \frac{2R_s}{\pi d} \tag{5.2}$$

$$L' = \frac{\mu}{\pi} \ln \left\{ \frac{D}{d} + \sqrt{(\frac{D}{d})^2 - 1} \right\} \tag{5.3}$$

$$C' = \frac{\pi \epsilon}{\ln \left\{ \frac{D}{d} + \sqrt{(\frac{D}{d})^2 - 1} \right\}} \tag{5.4}$$

$$G' = \frac{\pi \sigma}{\ln \left\{ \frac{D}{d} + \sqrt{(\frac{D}{d})^2 - 1} \right\}} \tag{5.5}$$

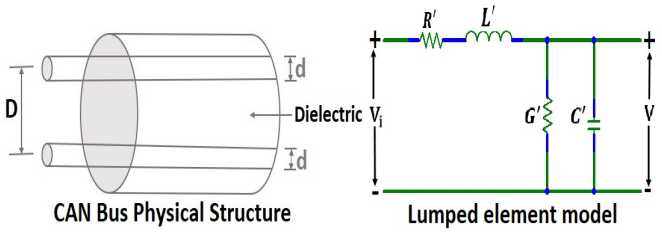


Figure 5.4: LEM for transmission line

I hypothesize that due to manufacturing imperfections, the length of the cable, $h_{(i)}(t)$ is unique for every channel, so the step response of each channel is unique. In this research, eight channels are used. Given the circuit is shown in Fig. 5.4, transfer function can be found relating $V_i$ to $V$ in the Laplace domain. Let $H_i(s)$ represent the Laplace transform of $h_{(i)}(t)$, which is represented as eq. (5.6).

$$H_i(s) = \frac{1}{L'C'} \left[ \frac{1}{s^2 + \frac{R'C'+L'G'}{L'C'} s + \frac{1+G'R'}{L'C'}} \right] \tag{5.6}$$

### 5.2.2 Signal Acquisition

The $E_{(FP)}$ acquires the analog signal $y_{(i)}^{(a)}(t)$ generated by $E_{(i)}$ and transmitted through $h_{(i)}(t)$ and converts this signal into digital signal $y_{(i)}(n)$ as represented in eq.

(5.7).

$$y_{(i)}(n) = y_{(i)}^{(a)}(t)|_{t=nT_s}, T_s = 0.5 \times 10^{-9} \tag{5.7}$$

Where $T_s$ represents sampling time of $0.5\,nsec$ and the sampling rate of $2\,GSa/sec$ for the signal. The reason to generate $y_{(i)}(n)$ is that the $y_{(i)}^{(a)}(t)$ occurs at infinite instants of time, thus demands large memory to get stored. However, the $E_{(FP)}$ has limited memory, therefore, the analog-to-digital conversion is performed.

### 5.2.3 Step Response Acquisition

After the signal acquisition, the next step is to acquire a step response of the system. The motivation behind finding the step response is to find control system parameters that are used as a feature set.

After digitization, an algorithm is used to isolate individual dominant bits from the $CANH$ bus. This section will focus purely on the detection of the rising and falling edges of each $CANH$ pulse, such that these pulses can be extracted. Individual pulses are needed to observe individual step response and model only that single step response in terms of its control parameters. For the following, assume that a single ECU's signal which passes through channel $i$ is being processed, $y_{(i)}(n)$, which contains $X$ samples in total and $K$ pulses. A small subset of samples from the digital pulse train $y_{(i)}(n)$ is shown in Fig. 5.5.

To isolate the individual pulses shown in Fig. 5.5 to extract their control parameters, an algorithm is designed to locate the edges of each dominant bit. For this edge detection, first, $y_{(i)}(n)$ is filtered using a P-sample ($P = 10$) digital moving average filter to smooth oscillations, reducing the likelihood of false edge detection. By suppressing high-amplitude oscillations with this filter, there is less chance to detect oscillations as edges. The new moving average signal is shown in Fig. 5.6 and is mathematically represented as a difference equation in eq. (5.8). Allow $y_{(i)}^{(MA)}(n)$ to represent the moving average filtered output of $y_{(i)}(n)$.

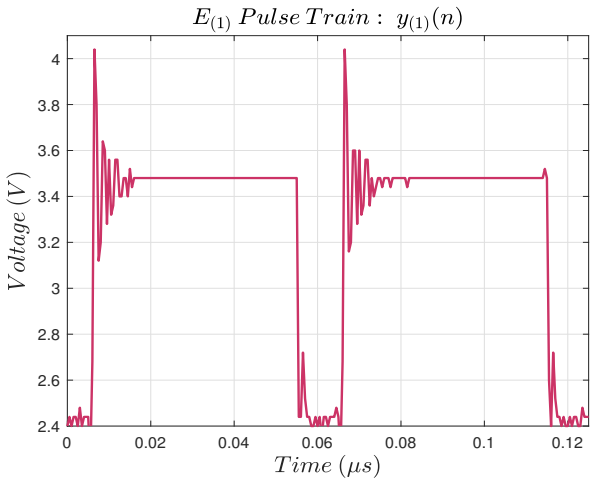


Figure 5.5: ECU's digitally sampled pulse train

$$y_{(i)}^{(MA)}(n) = \frac{1}{P}\sum_{i=1}^{P} y_{(i)}(n-i) \tag{5.8}$$

Next, the signal is converted to an ideal digital signal by using a threshold of $3V$ to classify individual samples as being part of either a dominant or a recessive bit as shown in eq. (5.9).

$$y_{(i)}^{(D)}(n) = \begin{cases} 0, & y_{(i)}^{(MA)}(n) < 3 \\ 1, & 3 \leq y_{(i)}^{(MA)}(n) \end{cases} \tag{5.9}$$

Fig. 5.7 shows this digitally-threshold signal. Allow $y_{(i)}^{(D)}(n)$ to be the digitally-threshold signal of $y_{(i)}^{(MA)}(n)$.

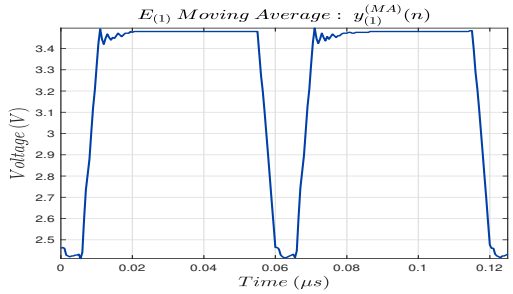


Figure 5.6: Moving average pulse train

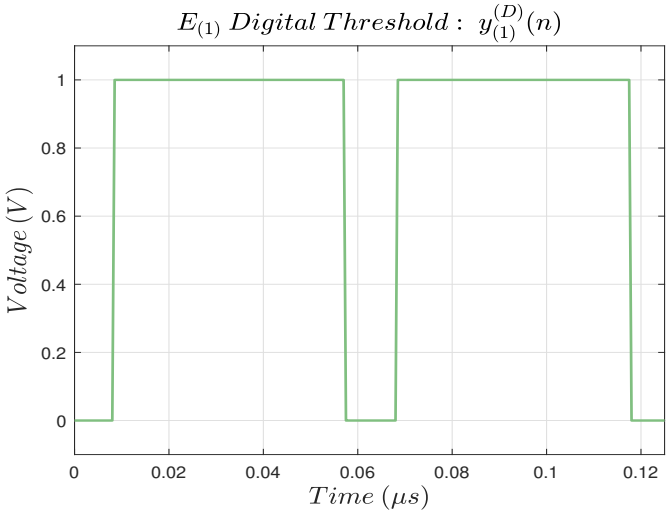

Figure 5.7: Digitally threshold pulse train

The ideal digital signal contains no oscillations, the digital derivative of the digital signal $y_{(i)}^{(D)}(n)$ is found using the first difference, as shown in eq. (5.10), to find the rising and falling edges.

$$\frac{dy}{dn}y_{(i)}^{(D)}(n) = y_{(i)}^{(D)}(n) - y_{(i)}^{(D)}(n-1), \quad n = 0, 1, ..., X-1 \tag{5.10}$$

All rising edges are now shown by a positive impulse with some delay, in samples, while all falling edges are shown by a negative impulse, as shown in Fig. 5.8. All non-edges assume a value of zero. The indices for rising edges and falling edges can be extracted as shown in eq. (5.11). Allow $n_{rise}$ and $n_{fall}$ to represent the sample indices of the detected rising and falling edges respectively.

$$n_{rise} = n|\frac{dy}{dn}y_{(i)}^{(D)}(n) > 0 \quad n_{fall} = n|\frac{dy}{dn}y_{(i)}^{(D)}(n) < 0 \tag{5.11}$$

In CAN, it is possible for an ECU to occasionally transmit erroneous pulses which do not exhibit similar characteristics to a typical dominant bit from this ECU. It is necessary to remove these anomalies because their signatures will be vastly different from the typical signature of their respective ECU and channel combinations. One possible anomaly scenario is when a single $CANH$ dominant bit is significantly shorter

than the typical dominant bit, which may produce anomalous steady-state parameter calculations, since the pulse may not have sufficient time to settle and reach its steady-state. Another anomaly is when a single dominant bit is sent, followed by a second dominant bit that comes too early. This often results in an abnormally long pulse and strange transient behaviors. These short and long pulse behaviors are demonstrated in Fig. 5.9 and Fig. 5.10 respectively.

Pulses similar to those shown in Fig. 5.9 and Fig. 5.10 can be treated as outliers, and will not be considered at all for training or testing the classification model, or for deployment. In effect, the fingerprinting ECU is blocked from making any decisions about the authenticity of these outlier pulses. To remove these outliers, the following algorithm is formulated. Allow $\mathbb{E}(T)$ to represent the expected pulse length of a typical CAN pulse, in seconds. Also, allow $\Delta t$ to represent some tolerance, in seconds. Pulses of length $T$ are selected such that the condition is shown in eq. (5.12) is obeyed.

$$\mathbb{E}(T) - \Delta t < T < \mathbb{E}(T) + \Delta t \tag{5.12}$$

Since, the current indices stored are $n_{rise}$ and $n_{fall}$ from eq. (5.11), where $n_{rise}^{(k)}$

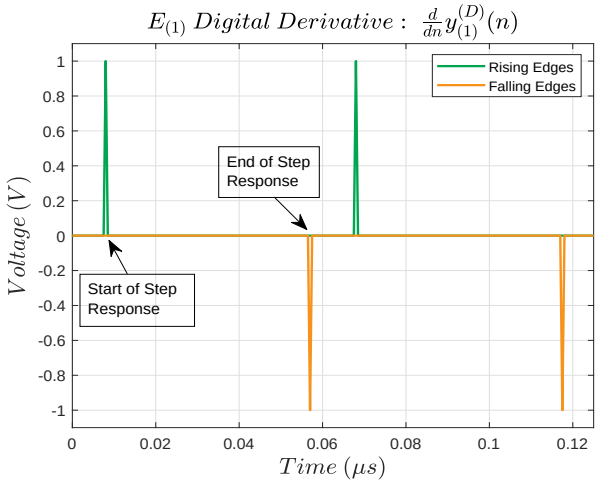


Figure 5.8: Rising and falling edges of pulse train

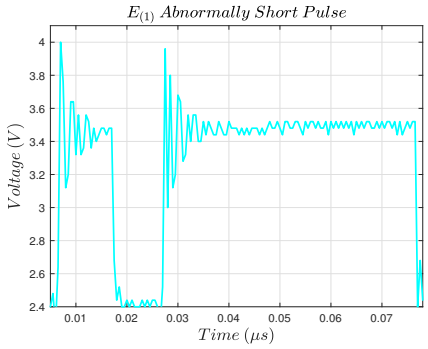


Figure 5.9: Abnormally short pulse

and $n_{fall}^{(k)}$ correspond to the rising and falling edge sample indices of pulse $k$, length in $T^{(k)}$ in seconds is computed, and then decide to keep or discard the pulse as an anomaly using the rule discussed in eq. (5.12):

**for** $1 \leq k \leq K$ **do**

$T^{(k)} = n_{fall}^{(k)} - n_{rise}^{(k)}$

**if** $\mathbb{E}(T) - \Delta t < T^{(k)} < \mathbb{E}(T) + \Delta t$ **then**

Keep $n_{fall}^{(k)}$ and $n_{rise}^{(k)}$

**else**

Discard $n_{fall}^{(k)}$ and $n_{rise}^{(k)}$

**end if**

**end for**

After removing indices corresponding to the outliers, the $k^{th}$ individual pulse of

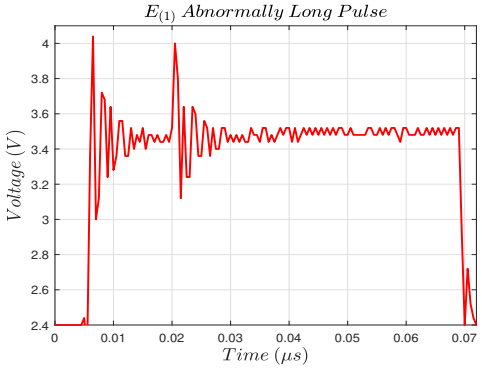


Figure 5.10: Abnormally long pulse

$E_{(i)}$ is extracted, denoted by $y_{(i,k)}(n)$, using the indices $n_{rise}$ and $n_{fall}$ which have not been discarded. An example of a typical single pulse is shown in Fig. 5.11.

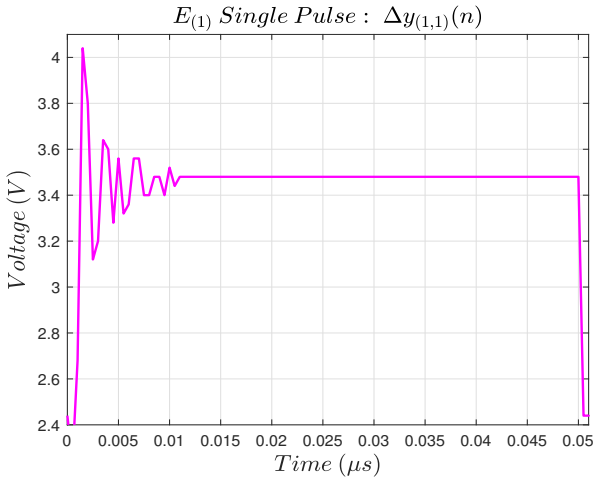


Figure 5.11: Single pulse

### 5.2.4 Feature Extraction

After outlier removal, feature extraction is performed to represent each pulse as a collection of control-theory-based parameters. This reduces a relatively high-dimensional input into a very low-dimensional representation, which allows for a lower-complexity model. This low model of complexity is computationally desirable. The parameters used for this study include peak time, $T_p$, percent overshoot, $\%OS$, settling time, $T_s$, and steady-state value, $V_{ss}$. Peak time and percent overshoot describe a second-order system's transient response while settling time and steady-state value describe the system's steady-state behavior. These parameters are described below:

- **Peak Time:** The time required for a step response to reach its peak value from the time that it first begins to rise.
- **Percent Overshoot:** The percentage by which the step response's peak value exceeds the response's value at steady-state.

- **Settling Time:** The amount of time it takes to form the time that the step response peaks until it has settled within some tolerance of its steady-state value.

- **Steady-State Value:** The value that a step response assumes after all transient oscillation has diminished.

Note that the second-order system is described in the Laplace domain by the standard equation shown in eq. (5.13).

$$H(s) = \frac{K\omega_n^2}{s^2 + 2\zeta\omega_n + \omega_n^2} \tag{5.13}$$

The three main parameters in this standard form are known as the natural frequency ($\omega_n$), the damping ratio ($\zeta$), and the D.C. gain of the system, $K$. In this application, $K$ will not be varied among ECU and receiver combinations, $K = 1$ for analysis. $\omega_n$ and $\zeta$ are computed using the following relationships, shown in eq. (5.14), (5.15).

$$\zeta = \frac{|\ln \frac{\%OS}{100}|}{\sqrt{\pi^2 + (\ln \frac{\%OS}{100})^2}} \tag{5.14}$$

$$\omega_n = \frac{\pi}{T_p\sqrt{1-\zeta^2}} \tag{5.15}$$

From these equations, extracting $T_p$ and $\%OS$ are theoretically enough to perfectly represent an ideal second-order system's transient response, however, settling time and steady-state value calculations are included to enhance the model's predictions since the pulse digitization can cause small errors in parameter calculation. In addition, the ECUs may not behave exactly as ideal second-order systems, so these additional parameters could provide the necessary information if the response does not exactly follow the second-order behavior.

The algorithm for calculating these parameters is non-trivial. an individual pulse $k$ from $E_{(i)}$ is analyzed, $y_{(i,k)}(n)$, with length $T$. First, the peak value is found and its corresponding sample index, as shown in eq. (5.16) and eq. (5.17).

$$V_{peak} = max(y_{(i,k)}(n)) \tag{5.16}$$

$$n_{peak} =_n y_{(i,k)}(n) \tag{5.17}$$

The peak time is computed by finding the number of samples from the sample before the rising edge occurs to the sample corresponding to the peak value and multiplying by the sampling period, as shown in eq. (5.18).

$$T_p = (n_{peak} - n_{rise})T_s \tag{5.18}$$

I can calculate the steady-state value, assuming that the pulse does eventually settle, by selecting some number of samples $Q$ of $y_{(i,k)}(n)$ to average just before the falling edge occurs, as shown in eq. (5.19). For analysis, I use $Q = 10$.

$$V_{ss} = \frac{1}{Q} \sum_{n=n_{fall}-Q}^{n_{fall}-1} x(n) \tag{5.19}$$

Afterwards, percentage overshoot is computed, represented as a decimal, as shown in eq. (5.20).

$$\%OS = \frac{V_{peak} - V_{ss}}{V_{ss}} \tag{5.20}$$

Lastly, settling time is computed by using a bit more complex of an algorithm. The first step is to define a kernel window of length $J$. then the beginning of the kernel window is aligned with the peak index, $n_{peak}$, of $y_{(i,k)}(n)$, and it is checked whether all values of $y_{(i,k)}(n)$ within the kernel window have magnitudes less than

a threshold, $\rho$. If not, the kernel is shifted by one sample ahead until either this condition is met, which means the settling index has been found, or until the end of the kernel reaches the falling edge of the pulse, indicating the pulse does not settle. To find the settling time of a single pulse, first, a parameter $\rho$ is selected, which is the percentage tolerance that a pulse voltage must remain from its steady-state value to be considered settled. Now, a tolerance value $\beta$ can be calculated as $\beta = \rho V_{ss}$. A pulse has settled at index $n_{settled}$ when the following condition shown in eq. (5.21) has been satisfied.

$$V_{ss} - \beta \leq y_{(i,k)}(n) \leq V_{ss} + \beta, \quad n \geq n_{settled} \tag{5.21}$$

I can re-write eq. (5.21) as shown in eq. (5.22):

$$|y_{(i,k)}(n) - V_{ss}| \leq \beta, \quad n \geq n_{settled} \tag{5.22}$$

The following shows a more mathematical formulation of this algorithm. Allow $n_0$ to denote the first sample index in which the kernel window overlaps. $y^{(0)}_{(i,k)}(n)$ is the same signal as $y_{(i,k)}(n)$ but with the D.C. steady-state offset removed.

$n_0 \leftarrow n_{peak}$
$y^{(0)}_{(i,k)}(n) \leftarrow y_{(i,k)}(n) - V_{ss}$
**while** $|y^{(0)}_{(i,k)}(n)| > \beta$ for all $n \in \{n_0, ..., n_0 + J - 1\}$ **do**
  $n_0 \leftarrow n_0 + 1$
  **if** $n_0 + J - 1 == n_{fall}$ **then**
    break
  **end if**
**end while**
$n_{settle} \leftarrow n_0$

Now that the settling index has been found, settling time is calculated by eq.

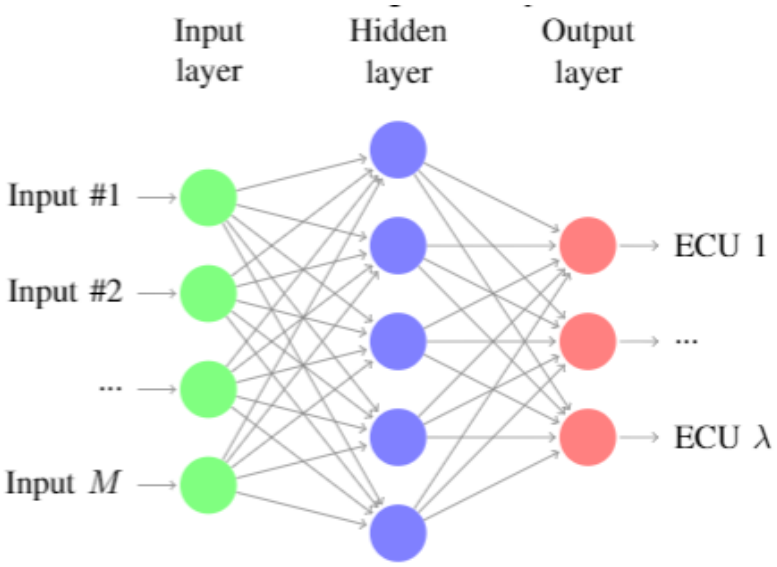


Figure 5.12: Artificial neural network architecture

(5.23).

$$T_s = (n_{settle} - n_{peak})T_s \tag{5.23}$$

The calculated parameters now comprise a lower-dimensional representation of a single-step response and can be used for pulse classification.

### 5.2.5 Training Neural Network

After finding the feature-set, any supervised machine learning method including support vector machines, artificial neural networks, deep learning, etc. can be used to achieve this goal. In this part of research, an artificial neural network is used, implementing the batch gradient descent optimization algorithm to train using feature vectors from the database. An artificial neural network (ANN)-based model (Fig. 5.12) is used to identify the source transmitter. For this the ANN gets $X_{(r)} = \{\mathbf{x_{(1)}}, \mathbf{x_{(2)}}, \ldots, \mathbf{x_{(R)}}\}$ as input-set and the output of neural network is the channel through which the signal propagated, which is further used to identify transmitter (ECU) for message authentication.

## 5.3 Experimental Setup, Dataset and Results

### 5.3.1 Experimental Setup

The proposed approach evaluates channel variability, to identify channel transmitter (ECU) for message authentication. Eight channels were used in this study and data was recorded through the $CANH$ pin. The technical specifications of channels are given in Table 5.1. The hardware is comprised of eight Arduino UNO-R2 micro-controller kits; eight CAN-Bus shield boards with $MCP\,2515$ CAN-bus controllers, $MPC\,2551$ CAN transceivers; and a $DSO1012A$ oscilloscope using a 2GSa/s sampling rate and $100\,MHz$ bandwidth to record the measured voltage samples. $Matlab\,R\,2018a$ software was used for the analysis of the recorded samples. A computer simulation was written that continuously transmitted the messages from different ECUs and pins. Afterward, these messages were then used as the dataset for model training and evaluation.

Table 5.1: Technical specifications of channels

| Label | Length(m) | Conductor | Insulation | Model |
|---|---|---|---|---|
| $Class$ –1 | 2 | Copper | XLPO | SAE J1939-15 |
| $Class$ –2 | 6 | Copper | XLPO | SAE J1939-15 |
| $Class$ –3 | 10 | Copper | XLPO | SAE J1939-15 |
| $Class$ –4 | 2 | Copper | XLPO | SAE J1939-19 |
| $Class$ –5 | 6 | Copper | XLPO | SAE J1939-19 |
| $Class$ –6 | 2 | Copper | XLPO | SAE J1128 |
| $Class$ –7 | 6 | Copper | XLPO | SAE J1128 |
| $Class$ –8 | 10 | Copper | XLPO | SAE J1128 |

### 5.3.2 Dataset Description

A dataset consisting of CAN packets at the output of 8 channels is recorded using an oscilloscope. There are 8 CAN channels, and 991 records in total comprising the

Table 5.2: Confusion matrix for transmitter classifier

| | | Target Class | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| - | - | *Class* –1 | *Class* –2 | *Class* –3 | *Class* –4 | *Class* –5 | *Class* –6 | *Class* –7 | *Class* –8 | Acc. % |
| *Predicted Class* | *Class* –1 | 115 | 0 | 0 | 8 | 0 | 0 | 1 | 0 | 92.7 |
| | *Class* –2 | 0 | 116 | 0 | 0 | 2 | 0 | 0 | 0 | 98.3 |
| | *Class* –3 | 0 | 0 | 118 | 0 | 0 | 0 | 0 | 0 | 100 |
| | *Class* –4 | 9 | 0 | 0 | 114 | 0 | 0 | 0 | 0 | 92.7 |
| | *Class* –5 | 0 | 3 | 0 | 0 | 125 | 0 | 0 | 0 | 97.7 |
| | *Class* –6 | 1 | 0 | 0 | 0 | 0 | 122 | 2 | 0 | 97.6 |
| | *Class* –7 | 0 | 0 | 0 | 0 | 0 | 0 | 129 | 0 | 100 |
| | *Class* –8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 126 | 100 |
| | Acc. % | 92 | 97.5 | 100 | 93.4 | 98.4 | 99.2 | 98.5 | 100 | 97.4 |

dataset. For model evaluation, 70% of the dataset is randomly partitioned into a training set and 30% into testing set respectively. All data used is collected under the same controlled conditions i.e. under the same temperature and using identical message patterns to observe the unique behaviors of the sampled signals.

### 5.3.3 Performance Evaluation Measures

I used precision, recall, $F_1Score$, accuracy and error rate as performance evaluation measures. The effectiveness of the method proposed is determined by the rate at which transmitters were correctly identified in the response to messages sniffed by $E_{(FP)}$. Let $TP$ represent the number of true positive predictions, $FP$ represents the number of false-positive predictions, $TN$ represents the number of true negative predictions, and $FN$ represents the number of false-negative predictions. Precision, recall, $F_1$ $–Score$, accuracy and error rate are computed which are as follows:

$$Precision = \left(\frac{TP}{TP + FP}\right) \tag{5.24}$$

$$Recall = \left(\frac{TP}{TP + FN}\right) \tag{5.25}$$

$$F_1–score = 2 \times \left(\frac{Precision \times Recall}{Precision + Recall}\right) \tag{5.26}$$

$$Accuracy = \left( \frac{TP + TN}{TP + TN + FP + FN} \right) \tag{5.27}$$

$$Error\text{–}rate = 1 - Accuracy \tag{5.28}$$

### 5.3.4 Experimental Results and Analysis

A series of experiments are performed for performance evaluation. For the performance evaluation presented, a neural network classifier is trained on the feature vectors of all different channels. It is important to highlight here that the selection of neural networks for the classifier is just a matter of choice and not a limitation of the proposed method. Classification accuracy is used as a performance measure here. Table 5.2 shows that the method proposed achieves a correct detection rate of 97.4%.

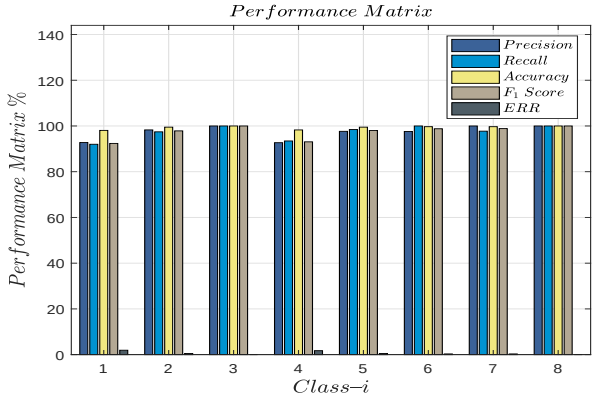


Figure 5.13: Bar graph of PM for channel classifier

Fig. 5.13 gives a visual representation of various useful metrics in the performance measurement of the classifier. The ideal classifier will have a precision, recall, accuracy, and $F_1$ $–Score$ of 100%, and an error of 0%. Table 5.3 shows the numerical performance metrics per ECU-channel combination. A relatively good performance is observed when looking at the accuracy and error scores, which only considers whether each prediction was correct or not. Thus, the accuracy and error consider $TN$ and

$TP$ both as correct predictions, and $FN$ and $FP$ predictions as incorrect. These are useful metrics to get an idea for how the model performs, but they do not give much insight into where the failures in the model exist. Precision and recall scores are useful for this purpose. The $F_1$ $-Score$ combines the recall and precision into one score, which gives another comprehensive summary of model performance.

Table 5.3: Performance matrix of ECU classifier

| - | $Precision$ | $Recall$ | $Accuracy$ | $F_1$ $-Score$ | $ERR$ |
|---|---|---|---|---|---|
| $Class$ –1 | 92.7% | 92.0% | 98.1% | 92.4% | 1.9% |
| $Class$ –2 | 98.3% | 97.4% | 99.5% | 97.9% | 0.5% |
| $Class$ –3 | 100% | 100% | 100% | 100% | 0.0% |
| $Class$ –4 | 92.7% | 93.4% | 98.3% | 93.1% | 1.7% |
| $Class$ –5 | 97.7% | 98.4% | 99.5% | 98.0% | 0.5% |
| $Class$ –6 | 97.6% | 100% | 99.7% | 98.8% | 0.3% |
| $Class$ –7 | 100% | 97.7% | 99.7% | 98.8% | 0.3% |
| $Class$ –8 | 100% | 100% | 100% | 100% | 0% |

The receiver operating characteristic curve, shown in Fig. 5.14, depicts the true-positive rate ($TPR$) and false-positive rate ($FPR$) plotted against each other. In the ideal case, the $TPR$ will be 100% and $FPR$ will be 0%. This means that the area under the ROC curve is 1 in the ideal case. With an imperfect classifier, this curve will begin to bend such that the area under the curve is reduced. Comparing the area under the ROC curve is one way to determine which classes were predicted best and worst by the model. Using Fig. 5.14, it is observed that $Class$ –1 is the least similar to the ideal case. Note that this bend in the curve for $Class$ –1 shows that initially if there are no false positives, there will not be high true positive rate. The other classes show that they can achieve very high $TPR$ without much sacrifice in $FPR$.

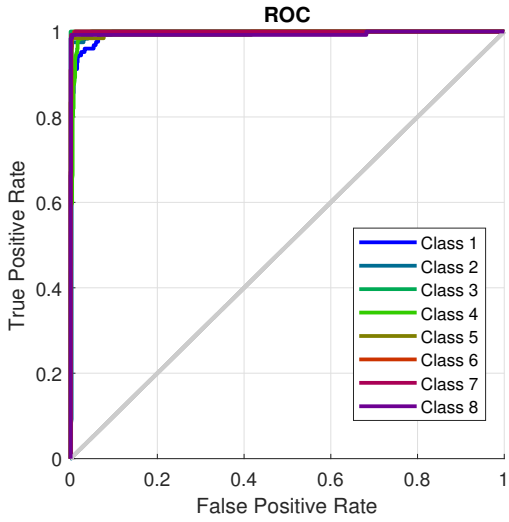


Figure 5.14: Receiver operating characteristic curve

# CHAPTER VI

# Channel Specific Distortion Based Transmitter Identification Using Neural Networks

## 6.1 Introduction

Cyberattacks on financial and government institutions, critical infrastructure, voting systems, businesses, modern vehicles, etc., are on the rise. Fully connected autonomous vehicles are more vulnerable than ever to hacking and data theft. This is due to the fact that the industry still relies on a controller area network (CAN) protocol for in-vehicle control networks. The CAN protocol lacks basic security features such as message authentication, which makes it vulnerable to a wide range of attacks including spoofing attacks. This chapter presents a novel method to protect CAN protocol against packet spoofing, replay and denial of service (DoS) attacks. The proposed method exploits physical unclonable attributes in the physical channel between transmitting and destination nodes and uses them for linking the received packet to the source. Impurities in the physical channel, imperfections in design, material, and length of the channel are contributing factors behind physically unclonable artifacts. The lumped element model is used to characterize channel-specific distortions. Non-parametric modeling is used to estimate distortion distribution, which is used for transmitting node identification. The performance of the proposed method

is evaluated on a dataset collected from a CAN network with channel lengths of 1 to 10 meters. Detection results show that the proposed method achieves the average accuracy of 99.8% with a false positive rate of 0.2%.

This research proposes a novel technique to address the aforementioned limitations in the existing state of the art CAN security by exploiting uniqueness in the channel-specific artifacts. Fig. 6.1 shows the comparison between the ideal CAN waveform and the actual waveform at the receiver. It can be observed that the channel-specific distortions do exist in the received signal, and they depend on the underlying physical channel.

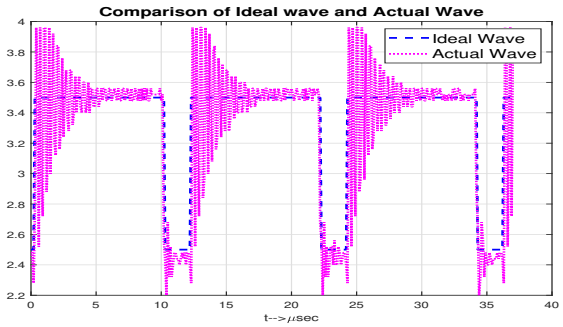


Figure 6.1: Channel-specific distortion in the CAN signal

This research hypothesizes that the distortion in the signal is physically inimitable, which can be used to link the signal to the channel. The proposed method exploits uniqueness in the channel-specific distortion for linking the received CAN packet to the transmitting source. The impurities in the physical channel, imperfections in design, material and length of the channel contribute to the uniqueness of received signal. The lumped element model (for the transmission lines) is used to characterize channel artifacts. Kernel density estimation, a non-parametric density estimation approach, is used to estimate the distribution of the artifacts. Estimated distribution is used for transmitting node identification. The performance of the proposed method is evaluated on a dataset collected from a CAN network with channel lengths of 1 to 10 meters.

*Contribution:* Main contributions of this chapter are:

- To model channel-specific distortion for CAN channel.
- Channel-specific distortion extraction and its uniqueness analysis
- To propose a reliable framework for linking received CAN packet to the true transmitting source.

The rest of this chapter is organized as follows: Section 6.2 provides the system model, mathematical modeling of the channel-specific distortion and its uniqueness analysis; Details of the proposed method are outlined in section 6.3; Experimental setup, data collection, and experimental results & analysis are discussed in section 6.4.

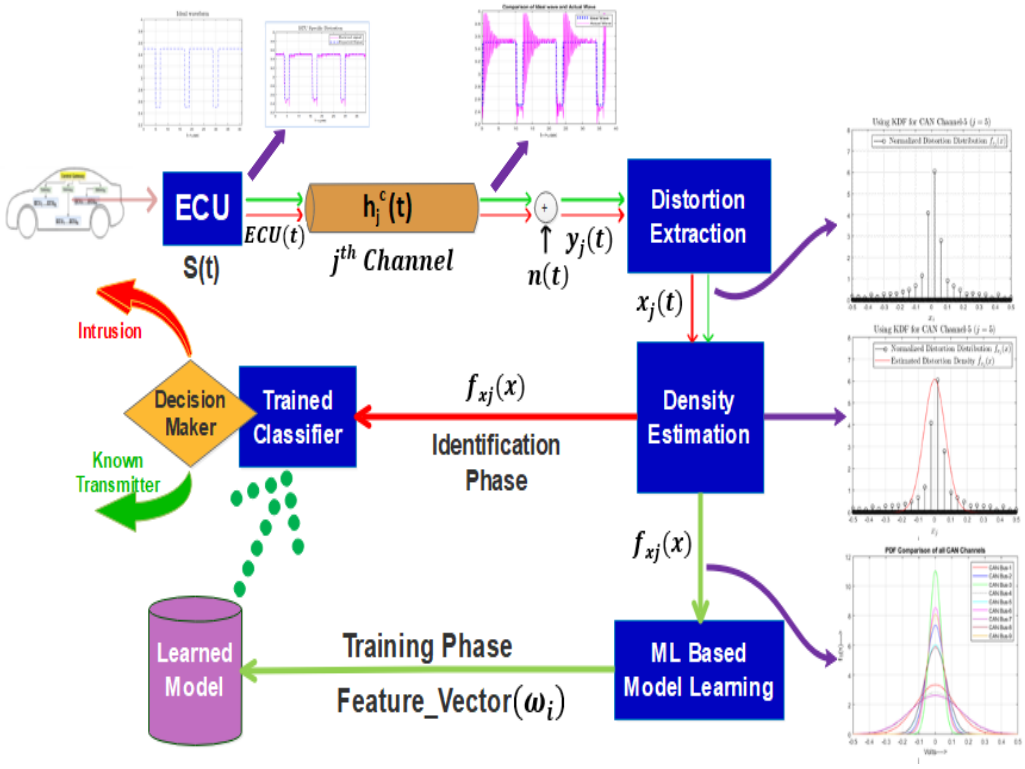


Figure 6.2: Block diagram of the proposed system

## 6.2 Proposed System Model and Channel Distortion Modeling

It can be observed from Fig. 6.1 that the received signal differs from its ideal level. Distortion in the received signal can be attributed to the physical channel imperfections and the channel response. The physical CAN channel behaves like a transmission line on which the signal propagates to carry information from the transmitting ECU to another ECU. The transmission line can be represented by the lumped element model (LEM) that breaks the transmission line into smaller segments (95).

This research hypothesizes that channel-specific distortions, $x_j(t)$, are inimitable, therefore it can be used to link a received network packet to the transmitter. Fig. 6.2 shows the block diagram of the proposed system. In a vehicle, different modules communicate with each other with the help of electronic control units (ECUs). The output of the ECU should be ideally a rectangular waveform, but practically it is not an ideal rectangular waveform, due to distortion added by the digital to analog converter (DAC). This signal then propagates through the channel which adds some distortion as well in the signal. The distortion extraction block extracts distortion from the signal, which is represented by $x_j(t)$. In the next stage, the density function of the distortion estimated is fed into a neural network. The neural network uses a gradient descent algorithm to find the feature vector which is stored in the database. The green lines in Fig. 6.2 indicates the training phase and the red lines represent the identification/testing phase. Once the model is trained, the transmitter cannot only be identified but also it can be located by feeding the estimated density function of the signal to the trained classifier. The trained classifier uses feature vectors and estimated density function to identify the transmitter using the channel distortion as a signature.

Shown in the right column of Fig. 6.3 is the equivalent circuit of an infinitesimally small piece of a transmission line. According to LEM, the transmission line is represented as a series resistance (R'), series Inductance (L'), a parallel Conductance (G') and a parallel capacitance (C'). Shown in the left column of Fig. 6.3 is the physical structure of the CAN channel. In this structure, $D$ is the distance between 2 wires, $d$ is the diameter of the wires.

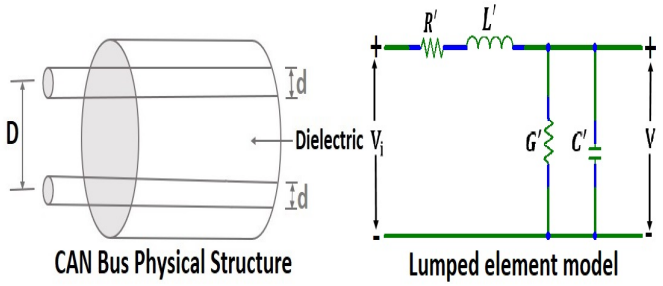


Figure 6.3: Lumped element model for CAN

Let $\mu$ be the permeability, and $\sigma$ be the conductivity of the copper. The ideal line parameters R',C',L' and G' (95) can be expressed as follows:

$$R' = \frac{2R_s}{\pi d} \tag{6.1}$$

$$L' = \frac{\mu}{\pi} \ln \left\{ \frac{D}{d} + \sqrt{(\frac{D}{d})^2 - 1} \right\} \tag{6.2}$$

$$C' = \frac{\pi \epsilon}{\ln \left\{ \frac{D}{d} + \sqrt{(\frac{D}{d})^2 - 1} \right\}} \tag{6.3}$$

$$G' = \frac{\pi \sigma}{\ln \left\{ \frac{D}{d} + \sqrt{(\frac{D}{d})^2 - 1} \right\}} \tag{6.4}$$

It is important to highlight that the material and the manufacturing imperfections are expected to contribute a small change (say $\delta$) in the expected line parameter values. Therefore, true values of line parameters can be expressed as: $R'_n = R' + \delta R$,

$L'_n = L' + \delta L$, $C'_n = C' + \delta C$, and $G'_n = G' + \delta G$. As imperfections are not uniform, therefore, resulting changes in the line parameters, e.g., $\delta$R, $\delta$L, $\delta$G and $\delta$C, can be treated as random variables which make resulting distortion in the signal traveling over the channel a random variable.

Similarly, channel material imperfections (impurities) also contribute to transmission distortion. For instance, propagation, $u_p = c/\sqrt{\varepsilon_r}$, depends on relativity permittivity $\varepsilon_r$. Material impurities can cause $\varepsilon_r$ to deviate, which further makes $u_{\mathrm{p}}$ to deviate from its expected value. Likewise, these imperfections also cause line parameters to deviate from their expected values. Channel dispersion is another source of channel-specific distortions. According to Fourier analysis, when a square wave travels over a physical channel, it experiences spreading effect. Finally, the channel length is another source of distortion. Specifically, channel length contributes to attenuation that signal experiences while traveling over the channel. The attenuation constant, $\alpha$, is directly dependent on the channel length, which can be expressed as eq. 6.5,

$$\alpha = Re\left\{\sqrt{\frac{R' + j\omega L'}{G' + j\omega C'}}\right\}. \tag{6.5}$$

Uncertainty in line parameters also contributes to uncertainty in $\alpha$, which can be expressed as, $\alpha_n = \alpha + \delta\alpha$.

## 6.3 Non-parametric Density Estimation

Fig. 6.4 represents the network model of a modern electric vehicle which is a heterogeneous distributed real-time system consisting of multiple ECUs interconnected with an in-vehicle network (73). In the proposed method, there are M sub-networks connected to each other by the central gateway, and in each sub-network, there are N electronic control units. The gateway is responsible for authentication of the sig-

nal. In this model, N = 9 and M can be any positive integer. Hence, the model can authenticate any number of transmitters based on channel characteristics.

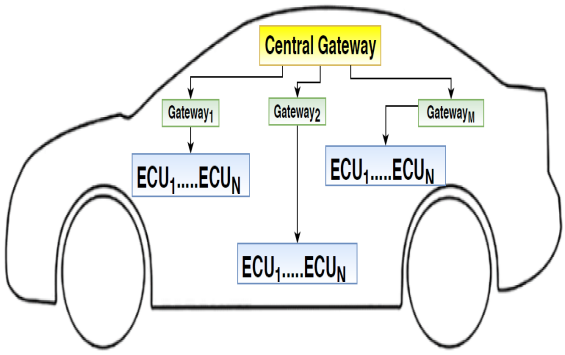


Figure 6.4: Architecture of sub-networks with gateways

Let $S(t)$ be the output of an ECU and $x_j(t)$ be the distortion induced in $j^{th}$ channel due to environmental factors such as temperature, electromagnetic, etc. The output of $j^{th}$ channel $y_j(t)$, can be expressed, $y_j(t) = S(t) + n_j(t)$. If $y_j(t)$ is the ideal channel output and $y_j^{(a)}(t)$ is the actual measurement then the channel-specific distortion, $x_j(t)$, can be computed as eq. 6.6,

$$x_j(t) = y_j^{(a)}(t) - y_j(t). \tag{6.6}$$

Here, $x_j(t)$ is a random variable obeying underlying density, $f_{x_j}(x)$. Shown in Fig. 6.5 is the distribution of $x_j(t)$. The density of $x_j(t)$ can be used to localize the source of the received CAN packet. The real challenge here is to estimate the underlying density of $x_j(t)$. Both parametric and non-parametric density estimation methods can be used to achieve this objective. Kernel density estimation (KDE) - a non-parametric density estimation method - is used here.

This is a data reduction stage, in this processing stage large dataset, i.e. estimated channel-specific distortion, $x_j(t)$, is processed to extract a feature vector that can be used for channel identification. The KDE package available with $Matlab^R$, which supports all known kernels such as Gaussian, logistic, Laplacian, Epanechnikov, etc.

is used for density estimation. The proposed scheme, however, uses the *Gaussian kernel*, i.e. $K_g(x) = \frac{1}{\sqrt{2\pi}} exp(-\frac{1}{2}u^2)$ for the results present in this research. The motivation behind the Gaussian kernel is that for a given bandwidth, $h$, the Gaussian kernel yields an optimally smooth density estimate (96). Shown in Fig. 6.5 is the plot of the distribution of channel-specific distortion and estimated density, $\hat{f}_{\mathrm{x}_j}(x)$, using Gaussian Kernel.

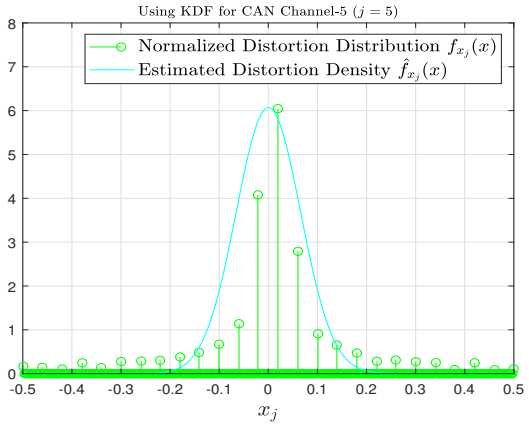


Figure 6.5: Distortion distribution and estimated density function

### 6.3.1 Channel Identification

Estimated channel distortion density is then used for channel identification. To achieve this goal, both the parametric binary hypothesis testing and the non-parametric hypothesis testing based on machine learning can be used. As estimated density is non-parametric, machine learning-based testing is used. To this end, estimated density, $\hat{f}_{\mathrm{x}_j}(x)$, is used to train a supervised classifier. Any supervised machine learning method including support vector machines, artificial neural networks, deep learning, etc. can be used to achieve this goal.

Shown in Fig. 6.6 is the block diagram of the training and test phases. During the training phase, the acquired signal from each channel is processed to extract channel-specific distortion, which is used to estimate the underlying density using

KDE, $\hat{f}_{\mathrm{x}_j}(x)$. The $\hat{f}_{\mathrm{x}_j}(x)$ is used for training a classifier. During the testing phase, channel-specific distortion is extracted from the test signal, $y_{test}(t)$, which is used for underlying density estimation, $\hat{f}_{\mathrm{x}_{test}}(x)$. The $\hat{f}_{\mathrm{x}_{test}}(x)$ is then applied at the input of a trained classifier for class label assignment.

A neural network with a scaled conjugate gradient back-propagation algorithm is used for channel classification. Let $p_1$,$p_2$.....$p_k$ be a set of non-zero weight vectors in $\Re^N$ (97). The set is said to be a conjugate system with respect to the non-singular matrix A if the following holds,

$$p_i^T A p_j = 0 \quad (i \neq j, i = 1, 2, .....k). \tag{6.7}$$

The set of points w in $\Re^N$ satisfying,

$$w = w_1 + \alpha_1 p_1 + \alpha_2 p_2 + ... + \alpha_k p_k, \alpha_i \in \Re. \tag{6.8}$$

where, $w_1$ is a point in weight space and $p_1$,$p_1$.....$p_k$ is a subset of conjugate system, is called a k-plane (97). This algorithm adjusts the weights $w_i$ such that the error is minimized. The iterations continue to find the set of $w_i$ till error is minimized and the local minimum is reached. The main purpose of the scaled conjugate gradient backpropagation algorithm is to find the weights $w_i$ in the training phase and then weights $w_i$ for classification in the testing phase.

## 6.4 Experimental Setup, Results and Analysis

### 6.4.1 Experimental Setup

**CAN channel specifications:**

To realize CAN channel, CAN cables commonly used by the automotive industry under the SAE J1939 standard from 2 manufacturers, are used. To this end, nine

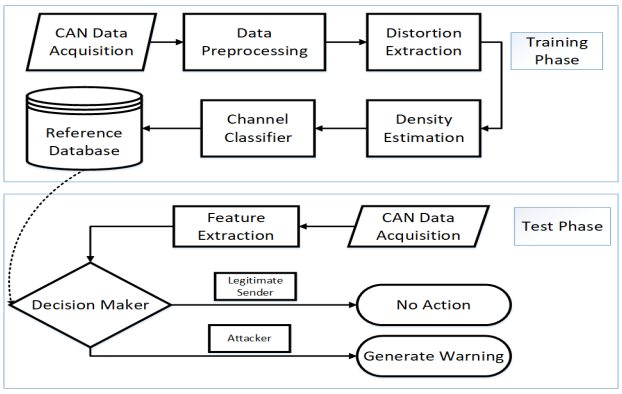


Figure 6.6: Training and test phases for intrusion detection system

(9) different CAN channels by either varying manufacturer or channel length are realized. Technical specifications of CAN channels used for performance evaluation of the proposed method are given in Table 6.1.

**ECU and CAN communication**:

For CAN communication, Arduino Uno R2 microcontroller kit with CAN bus shield board with MCP2515 CAN bus controller and MPC2551 CAN transceiver is used. To avoid device-specific distortions, the same ECU is used for CAN packet transmission over each CAN channel. The same script is used for sending an identical message continuously from the same ECU over all channels. For CAN bus data

Table 6.1: Technical specifications of channels

| Label | Length(m) | Conductor | Insulation | Model |
|---|---|---|---|---|
| C-1 | 1 | Copper | XLPO | SAE J1939-15 |
| C-2 | 5 | Copper | XLPO | SAE J1939-15 |
| C-3 | 10 | Copper | XLPO | SAE J1939-15 |
| C-4 | 1 | Copper | XLPO | SAE J1939-19 |
| C-5 | 5 | Copper | XLPO | SAE J1939-19 |
| C-6 | 10 | Copper | XLPO | SAE J1939-19 |
| C-7 | 1 | Copper | XLPO | SAE J1939-19 |
| C-8 | 5 | Copper | XLPO | SAE J1939-19 |
| C-9 | 10 | Copper | XLPO | SAE J1939-19 |

Table 6.2: Confusion matrix for channel classifier

| Target Class | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| - | - | C-1 | C-2 | C-3 | C-4 | C-5 | C-6 | C-7 | C-8 | C-9 | Acc. % |
| Predicted Class | C-1 | 128 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 98.5 |
| | C-2 | 0 | 128 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 100 |
| | C-3 | 0 | 0 | 128 | 0 | 0 | 0 | 0 | 0 | 0 | 100 |
| | C-4 | 0 | 0 | 0 | 128 | 0 | 0 | 0 | 0 | 0 | 100 |
| | C-5 | 0 | 0 | 0 | 0 | 128 | 0 | 0 | 0 | 0 | 100 |
| | C-6 | 0 | 0 | 0 | 0 | 0 | 128 | 0 | 0 | 0 | 100 |
| | C-7 | 0 | 0 | 0 | 0 | 0 | 0 | 126 | 0 | 0 | 100 |
| | C-8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 128 | 0 | 100 |
| | C-9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 128 | 100 |
| | Acc. % | 100 | 100 | 100 | 100 | 100 | 100 | 98.4 | 100 | 100 | **99.8** |

acquisition, Oscilloscope DSO1012A for the voltage samples recording with Sampling Rate of 2GSa/s, 100MHz bandwidth, and 8-bit vertical resolution, is used.

### 6.4.2 Dataset Description

A dataset consisting of CAN packets at the output of 9 channels is recorded using an oscilloscope. For each CAN channel realization, 460,800 (3600*128) samples are collected. There are 9 CAN channel, so in total 4,147,200 (3600*128*9) samples are collected. For performance evaluation, random partitioning is performed to divide the dataset into the training and test sets. For Training set: 70% and for Test set: 30% of the data is used. The dataset used here is collected in the same environment i.e. under the same temperature and using an identical message to observe the minute and unique variation of the digital signals. It can be observed from Table 6.2 that the proposed method achieves very high transmitter classification accuracy.

### 6.4.3 Experimental Results and Analysis

A series of experiments are performed to evaluate the performance of the proposed method. For the performance evaluation presented, a neural network classifier is

trained on estimated densities for different channels. It is important to highlight here that the selection of neural networks for the classifier is just a matter of choice and not a limitation of the proposed method. Classification accuracy is used as a performance measure here. Table 6.2 shows that the proposed method achieves an overall correct detection rate of 99.8%.

**Experiment 1 - Channel-specific distortion discrimination:**

The goal of this experiment is to investigate the characteristics of a channel-specific distortion. To achieve this objective, the estimated density for each CAN-channel is compared. Shown in Fig. 6.7 are the plots of estimated densities for all 9 CAN channels. It can be observed from Fig. 6.7 that the estimated densities are different. It is important to highlight that the estimated densities from C-4 and C-7 are hard to differentiate. This is due to the fact that both channels are of the same length.

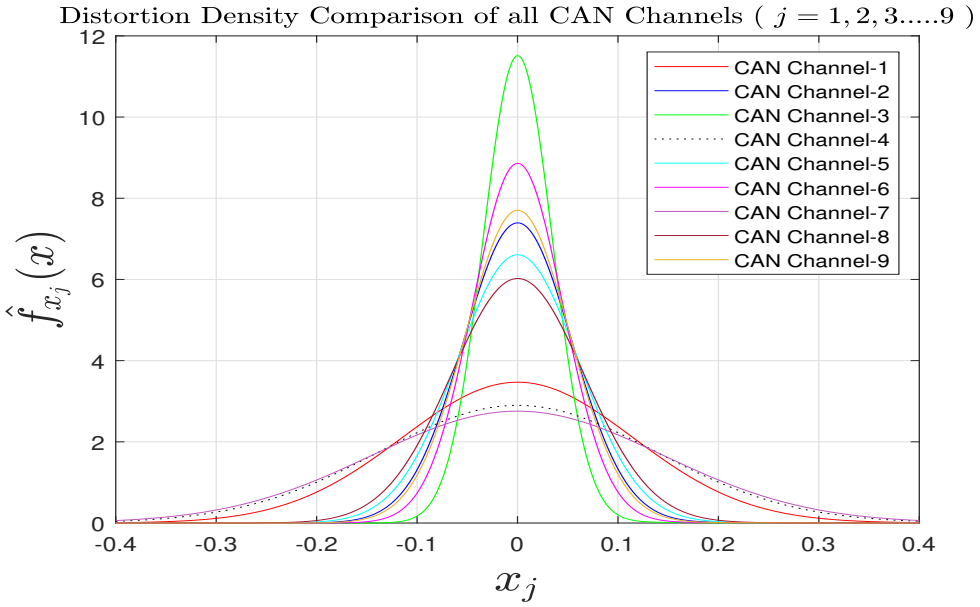


Figure 6.7: Estimated distortion density comparison

**Experiment 2 - Channel identification:**

Validating the uniqueness of channel-specific features is the main purpose of this experiment. This part of research hypothesize that channel-specific distortions are unique due to the unique material and the design imperfections in the channel. To

validate this claim, data is recorded for each cable family and each channel length with an identical channel input, transmitted using the same ECU. Specifically, for this experiment 'cable type' and 'length' are the only variables. For this experiment, a feature vector consisting of 2001 points (which is the dimension of an estimated density function) is considered. During the training phase, a neural network is trained on estimated densities (shown in Fig. 6.7) for all CAN channels. The neural network is trained with a "scaled conjugate gradient backpropagation" training algorithm, with 2001 input nodes, 10 hidden nodes and 9 output nodes, stopping criteria of Epochs = 2000 and gradient = $10^{-7}$.

Shown in Table 6.2 are the confusion matrices for the channel (CAN) classification. It can be observed from Table 6.2 that the proposed method achieves an overall correct detection rate of 99.8%. The proposed method achieves detection accuracy comparable with existing state-of-the-art. For instance existing voltage-based IDSs proposed in (44; 25) and (45) reported detection rates of 96.48%, 99.8% and 99.85%, respectively. It can also be noticed from Table 6.2 that CAN channel C-1 and C-7 are the only sources of error here, that is, these channels are of the same length and model. If these 2 channels are ignored, then the proposed system achieves a perfect detection rate of 100%. This method has advantage over other fingerprinting IDS based approaches. Since this approach uses the distortion of the channel as a signature, it is nearly impossible for any transmitter to inject the distortion based signature. The proposed method in this research also faces a limitation, if a legitimate ECU is removed from its location and an adversary ECU is connected to the same place in the network, in that scenario, the proposed framework is unable to detect the spoofing attack.

# CHAPTER VII

# Statistical Parameters Based Transmitter Identification

## 7.1 Introduction

Controller area network (CAN) is used as a legacy protocol for in-vehicle data communication between electronic control units (ECUs). Moreover, this protocol is also used to provide an interface between the vehicles and the external world via the internet; thus, makes it vulnerable to spoofing attacks. The primary reason behind vulnerability is that the sender's information in CAN messages is missing; hence, the receiving ECU is unable to authenticate an incoming CAN packet that may be injected by an adversary. In order to overcome this vulnerability, this research proposes a framework for message authentication through channel identification. The proposed distortion based intrusion detection system (DIDS) exploits channel distortion to create a unique fingerprint of the CAN signals and link the received packet to the transmitter. For message authentication, a novel parametric approach is developed to quantify the distortion in form of probability density function (PDF). The best fit PDF over histogram of distortion, is gamma distribution function. Afterwards, the $\alpha$ and $\beta$ parameters of gamma distribution function are computed for multiple records in each channel to obtain the joint density function. Finally, the likelihood

ratio test is applied on joint density function to identify the channel and transmitter. Experimental results shows that my method is robust and computationally efficient with less processing requirements for message authentication.

The modern electric vehicle is a cyber-physical system (CPS) equipped with many wireless and wired communication interfaces to connect ECUs through various in-vehicle networks (IVNs) (8; 1; 2; 3; 4; 5), such as the CAN (8), LIN (1), MOST (2), FlexRay (3), etc. These networks connect safety-critical components of the vehicle e.g. brakes, airbags, engine control, adaptive cruise control, and electronic stability program, etc. through ECUs (14; 98). Integration of wireless interfaces, e.g., Bluetooth, Wi-Fi, etc., with IVNs pose serious security threats to electric vehicles (9), because the CAN is prone to variety of internal as well as external attacks including packet spoofing attacks, data injection attacks, denial of service (DoS) attacks, etc. (16; 15; 14; 99; 17) as CAN does not take into account the sender's information for message authentication. Recently, researchers have successfully hijacked vehicles from a remote location, and killed the vehicle engine in middle of a highway (16; 10; 11; 12; 13; 100). In case of autonomous vehicles (AVs), it is highly likely that concerted attackers might take a vehicle or its occupants hostage for a ransom, or direct an AV to relocate it to an unintended destination, etc. Thus, automakers and other stakeholders are focused on developing safeguards against growing attack vectors before bringing AVs to the marketplace.

Researchers have proposed various solutions to detect and prevent attacks for in-vehicle control networks. These methods can be classified into 2 types: (i) Message authentication code (MAC) based methods (18; 19; 20; 21; 22; 23; 24), and (ii) Intrusion detection systems (IDS) (29; 30; 31; 32; 33; 34; 35; 36; 37; 38; 39; 40; 41; 44; 25; 45; 43; 26; 46; 27; 42). The MAC-based methods, achieve security and privacy by encrypting the payload of the CAN-packet before transmission. For instance, in (17), Wang et. al. demonstrated a MAC-based framework VeCure for CAN secu-

rity. In VeCure, 64-bit MAC for every 64-bit message was transmitted between the ECUs. Intuitively, the method exhibited high computational cost, 50% additional transmission overhead, and also required a higher data rate. In (21), Hiroshi et. al. presented a MAC-based message authentication mechanism for the CAN protocol against spoofing attacks. The monitoring ECU provided the authentication code for all ECUs and verified the code for all CAN messages. In (19), Hazem et. al. proposed a lightweight CAN authentication protocol (LCAP). The LCAP required to append a "magic number" that was generated by a one-way hash function employed on TESLA prototype (47). The protocol required 16-bits of the data field to append the authentication code, which still creates 25% overhead. The MAC-based approaches have the intrinsic overhead that lowers the transmission performance, hence makes them unreliable for the CAN security (17; 46; 23).

To address the limitations of MAC-based solutions, researchers have proposed intrusion detection-based approaches for CAN network traffic analysis (26; 27; 21; 19; 48). The intrusion detection-based approaches have lower data rate requirements because no additional bits are transmitted during the message transmission; hence, it does not add additional network overhead. In (26), Cho and Shin demonstrated a clock-based intrusion detection system (CIDS) that used ECU fingerprinting. Each ECU contained a crystal oscillator known as a clock; the ECU fingerprinting measured the clock skewness against the received packets and detect the attack. However, Sang et. al. (28) and Tayyab et. al. (49) demonstrated that CIDS can be bypassed by estimating the clock parameters. In (46), message authentication was performed through ECU detection by applying higher-order moments of the CAN signal in both time and frequency-domains. However, this approach was intolerant against the transmitter induction and the performance of the system seriously decays if the number of transmitters is increased. Therefore, we need an IDS-based approach that extracts unique fingerprints from the signal, works for a higher number of transmitters, and

also exhibits low computational complexity.

To address the aforementioned limitations of existing in-vehicle security techniques, a novel IDS-based message authentication approach is presented in this research. This approach exploits the uniqueness in channel-specific distortions e.g., channel imperfections, material impurities, length of the channel, etc., for channel fingerprint generation to identify transmitter for message authentication. This research hypothesizes that signal distortion is channel dependent; hence can be used to link the received signal to the source transmitter. Moreover, as distortion is channel-dependent, thus, it is not easy for the adversary to duplicate this information to attack the CAN. Therefore, I associate the received packet through a specific channel, by computing channel distortion using a parametric approach. The proposed parametric approach estimates multiple density functions of distortion, and adopt the best fit density function to identify the source transmitter. In my case, the gamma distribution function is the best fit density function that is used to compute the $\alpha$ and $\beta$ parameters of the gamma distribution. Through empirical analysis, I observed that the parameters $\alpha$ and $\beta$ have random nature thus can be represented through a joint normal distribution function. Afterward, for a test signal, $\alpha$ and $\beta$ are computed, that are then used to identify the channel for message authentication and attack prevention through a generalized likelihood ratio test approach.

Rest of the chapter is organized as follows: Section 7.2 provides detailed description of the proposed methodology. Section 7.3 covers the experimental results and analysis.

## 7.2 System Model

For in-vehicle communication between the ECUs, the expected signal at the receiving ECU should have perfect rectangular waveform. However, as shown in Fig. 7.1, the expected waveform and the actual waveform received are not the same due

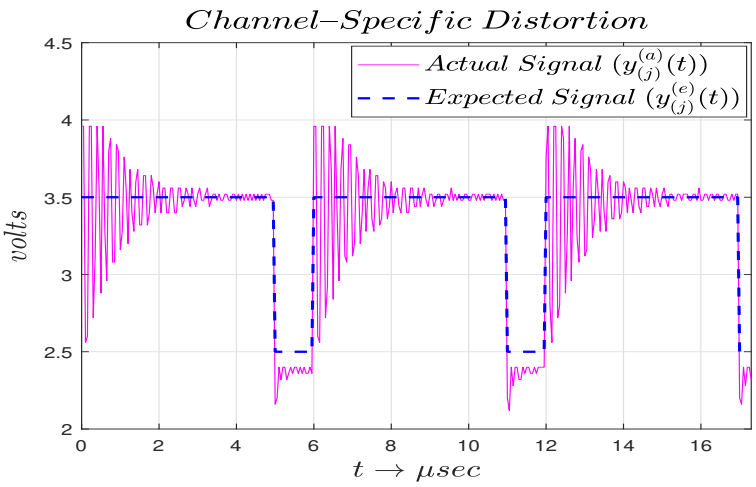


Figure 7.1: Comparison of expected and actual signal

to channel distortion, and I have exploited this fact for model generation to identify the CAN channel used to propagate the signal, and the source ECU. For this I have considered a *subnet* in CAN that contains $\psi = \{1, 2, \ldots, n\}$ channels represented as $C_{(j)}$, where $j \in \psi$, with a fingerprinting unit $E_{(s)}$ that sniffs the transmission of analog signal $y^{(a)}_{(j)}(t)$ which propagated through the channel $C_{(j)}$. The $E_{(s)}$ converts the signal $y^{(a)}_{(j)}(t)$ to a digital signal $y^{(a)}_{(j)}(n)$ with a sampling rate of $20\,MSa/sec$. Afterwards, the $E_{(s)}$ computes the expected signal $y^{(e)}_{(j)}(n)$ that is used for distortion computation $x_{(j)}(n)$ by representing the distortion in form of a histogram. The histogram is then used to estimate the probability density function (PDF) that is gamma distribution function $\hat{f}(x_j)$ with two parameters $\alpha_j$ and $\beta_j$. For each channel with $M$ records, the $\alpha_j$ and $\beta_j$ are collected in form of vectors i.e. $\hat{\alpha_j} = [\alpha_{j1}, \alpha_{j2} \ldots \alpha_{jM}]$ and $\hat{\beta_j} = [\beta_{j1}, \beta_{j2} \ldots \beta_{jM}]$, and due to their random nature, I further compute the PDF of $\hat{\alpha_j}$ and $\hat{\beta_j}$. Afterwards, the mean $\mu_j$ and the square root of variance (101) which is standard deviation $\sigma_j$ values of $\hat{\alpha_j}$ and $\hat{\beta_j}$ are utilized as the channel representation model. For any test signal after obtaining the $\alpha_j$ and $\beta_j$, the channel and transmitter identification is performed through likelihood test ratio using $\mu_j = \mu_{\alpha_j}, \mu_{\beta_j}$ and $\sigma_j = \sigma_{\alpha_j}, \sigma_{\beta_j}$. Fig. 7.2 shows the architecture of the proposed method that is described in detail in following subsections.

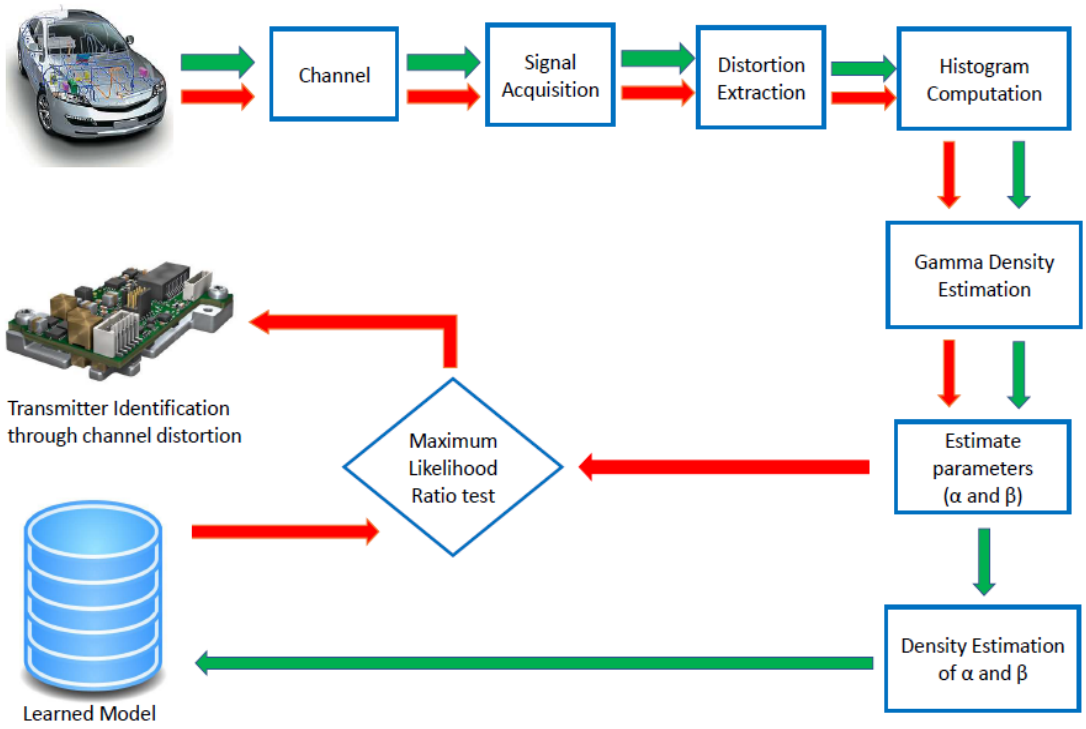


Figure 7.2: Block diagram of the system model

### 7.2.1 Signal Acquisition

The fingerprinting unit $E_{(s)}$ acquires the analog signal $y_{(j)}^{(a)}(t)$ generated by $C_{(j)}$ and converts this signal into digital signal $y_{(j)}^{(a)}(n)$ as represented in eq. (7.1).

$$y_{(j)}^{(a)}(n) = y_{(j)}^{(a)}(t)|_{t=nT_s}, T_s = 50 \times 10^{-9} \tag{7.1}$$

The $T_s = 50 \times 10^{-9}$ represents sampling time of $50\,nsec$ and sampling rate of $20\,MSa/sec$ for the signal. The reason to generate $y_{(j)}^{(a)}(n)$ is that the $y_{(j)}^{(a)}(t)$ occurs at infinite instants of time, thus demands large memory to get stored. However, as the $E_{(s)}$ has limited memory, therefore, the analog-to-digital conversion is performed. I compute expected signal $y_{(j)}^{(e)}(n)$ by first using a *p–sample* digital moving average filter to smooth high-amplitude oscillations. The new moving average signal, $y_{(j)}^{(MA)}(n)$, is mathematically represented as a difference equation as represented in eq. (7.2).

$$y_{(j)}^{(MA)}(n) = \frac{1}{p}\sum_{i=1}^{p} y_{(j)}^{(a)}(n-i) \tag{7.2}$$

Where, $p = 7$ represents the length of moving average filter. The $y_{(j)}^{(MA)}(n)$ is converted to an ideal (expected) digital signal by using a threshold of $3V$ to classify individual samples of signal as either dominant bit (logic 0) or recessive bit (logic 1) as described in eq. (7.3).

$$y_{(j)}^{(e)}(n) = \begin{cases} 2.5, & y_{(j)}^{(MA)}(n) < 3 \\ 3.5, & y_{(j)}^{(MA)}(n) \geq 3 \end{cases} \tag{7.3}$$

For the dominant bit, the voltage level is $3.5\,V$ and an ideal recessive bit has a voltage level of $2.5\,V$. In my case, the channel distortion is extracted in the dominant bits.

### 7.2.2 Distortion Extraction

After signal acquisition, I acquire the distortion for channel fingerprinting. The actual signal deviates from expected signal as shown in Fig. 7.3, and quantified as distortion $x_{(j)}(n)$, which can be captured using eq. (7.4).

$$x_{(j)}(n) = y_{(j)}^{(a)}(n) - y_{(j)}^{(e)}(n). \tag{7.4}$$

The distortion $x_{(j)}(n)$ as shown in Fig. 7.3 is a random variable, such that $\{x_{(j)}(n) \in n_{(j)} \mid n_{(j)} : [L \rightarrow U]\}$, where $L = -1.00$ and $U = 1.00$ represent the *lower* and *upper* distortion values respectively. The $x_{(j)}(n)$ is then used for histogram computation. The mathematical model of channel distortion is presented as follows.

The physical CAN bus behaves like a transmission line on which the signal propagates to carry information from one ECU to another ECU. Hence, the CAN channel

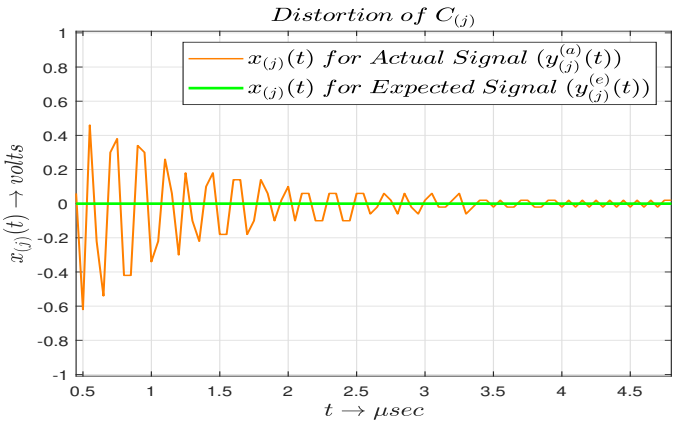


Figure 7.3: Distortion in the signal

can be represented by the lumped element model (LEM) (95) as shown in Fig. 7.4. According to LEM, the transmission line with the diameter $D$ of outer cable shield and, with the diameter $d$ of the inner copper wire can be modeled by circuit elements i.e. resistance ($R'$), inductance ($L'$), conductance ($G'$) and capacitance ($C'$).

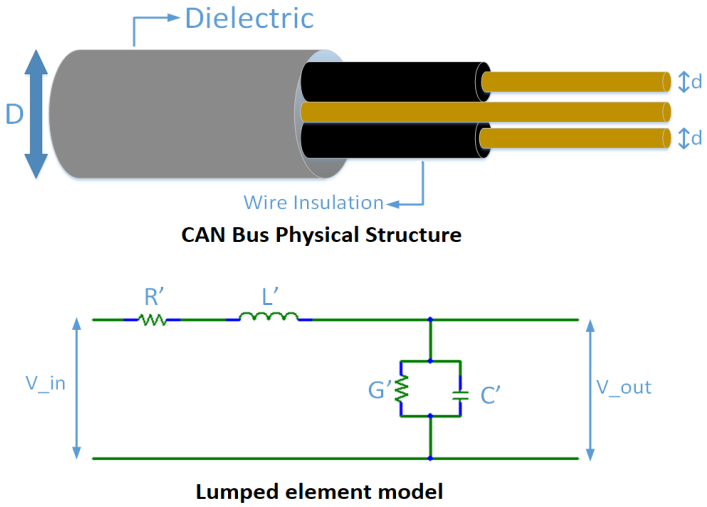


Figure 7.4: Lumped element model

The ideal line parameters $R'$, $C'$, $L'$ and $G'$ (95) can be expressed in the form of eq.(7.5)-(7.8).

$$R' = \frac{2R_s}{\pi d}. \tag{7.5}$$

$$L' = \frac{\mu}{\pi} \ln \left\{ \frac{D}{d} + \sqrt{\left(\frac{D}{d}\right)^2 - 1} \right\}. \tag{7.6}$$

$$C' = \frac{\pi\epsilon}{\ln \left\{ \frac{D}{d} + \sqrt{\left(\frac{D}{d}\right)^2 - 1} \right\}} \tag{7.7}$$

$$G' = \frac{\pi\sigma}{\ln \left\{ \frac{D}{d} + \sqrt{\left(\frac{D}{d}\right)^2 - 1} \right\}} \tag{7.8}$$

$$R_s = \sqrt{\frac{\pi f \mu}{\sigma}}. \tag{7.9}$$

In the equations above, $\mu$ is the permeability, $\epsilon$ is the permitivity of the dielectric separating the 2 copper wires; and $\sigma$ is the conductivity of the copper. It is important to highlight that material and manufacturing imperfections (small variation in $D$ and $d$) are expected to contribute a small change i.e. $\delta$ in the expected line parameter values. Therefore, true values of line parameters can be expressed as: $R'_n = R' + \delta_R$, $L'_n = L' + \delta_L$, $C'_n = C' + \delta_C$, and $G'_n = G' + \delta_G$. As imperfections are not uniform, therefore, resulting changes in the line parameters, e.g., $\delta_R$, $\delta_L$, $\delta_G$ and $\delta_C$, are random variables that makes resulting distortion in the signal random in nature. Let $x^{(R)}_{(j)}(n)$ represent the distortion due to $\delta_{R_{(j)}}$, $x^{(L)}_{(j)}(n)$ represent the distortion due to $\delta_{L_{(j)}}$, $x^{(C)}_{(j)}(n)$ represent the distortion due to $\delta_{C_{(j)}}$, and $x^{(G)}_{(j)}(n)$ represent the distortion due to $\delta_{G_{(j)}}$ for $j^{th}$ channel. Similarly, channel material imperfections (impurities) also contribute to transmission distortion. The velocity of signal propagation, $u_p = c/\sqrt{\varepsilon_r}$ depends on relative permittivity $\varepsilon_r$. Therefore, material imperfections can cause $u_{\mathrm{p}}$ to deviate from its expected value. Likewise, these imperfections also cause line parameters to deviate from their expected values. Channel dispersion is another source of channel-specific distortions. Let $x^{(D)}_{(j)}(n)$ be the distortion due to dispersion

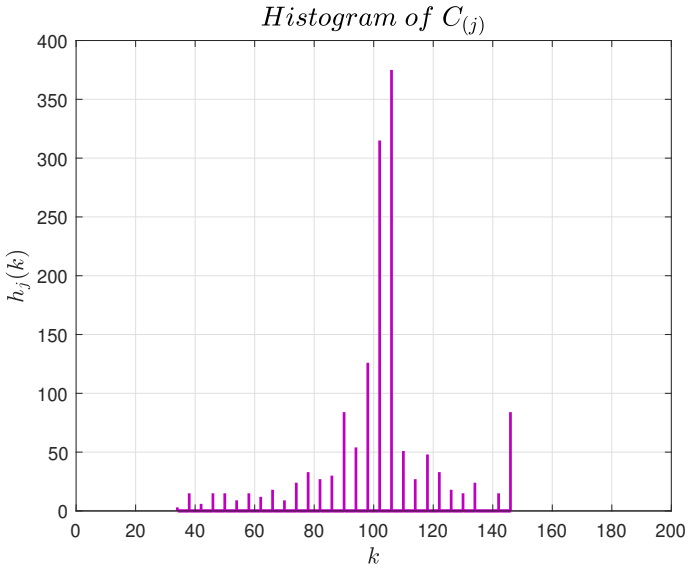


Figure 7.5: Histogram of $x_{(j)}(n)$ for $C_{(j)}$

which is attributed towards channel impurities. When a square wave travels over a physical channel it experiences spreading effect (95). The total distortion $x_j(n)$ (due to channel imperfections) for $j^{th}$ channel ($C_j$), can be expressed as eq. (7.10).

$$x_j(n) = x_{(j)}^{(R)}(n) + x_{(j)}^{(L)}(n) + x_{(j)}^{(C)}(n) + x_{(j)}^{(G)}(n) + x_{(j)}^{(D)}(n). \tag{7.10}$$

Similarly, using eq. (7.4), $y_{(j)}^{(a)}(n)$ can be expressed as follows:

$$y_{(j)}^{(a)}(n) = y_{(j)}^{(e)}(n) + x_{(j)}(n). \tag{7.11}$$

As $x_{(j)}(n)$ is unique for all the signals propagated over the same channel thus makes $y_{(j)}^{(a)}(n)$ dynamic, hence becomes an effective measure for channel fingerprinting for message authentication.

### 7.2.3 Histogram Computation

After distortion modeling, I use $x_{(j)}(n)$ for histogram generation, $n = \{0, 1, 2, \ldots, N-1\}$.. The histogram will then be used for density estimation of $C_{(j)}$. Using $m$ bins

with bin index $k$, and $N = 1500$ samples, the histogram is generated as follows:

$$h_{(j)}(k) = \sum_{n=1}^{N} \left[ \delta\left( \frac{x_{(j)}(n)}{\Delta} + 100 \right) + h_{(j)}(k) \right] \tag{7.12}$$

and

$$\Delta = \left( \frac{U - L}{m} \right) \tag{7.13}$$

Where $\Delta$ is the step size, $L$, and $U$ represents the lower and upper values for distortion. In my case $m = 200$, $L = -1$, $U = +1$. The $\delta(.)$ denotes Kronecker delta function (102), that can be computed through eq. (7.14).

$$\delta(n-k) = \begin{cases} 1 & : \quad n = k \\ 0 & : \quad Otherwise \end{cases} \tag{7.14}$$

Fig. 7.5 shows a sample histogram $h_{(j)}(k)$ for $j^{th}$ channel $C_{(j)}$.

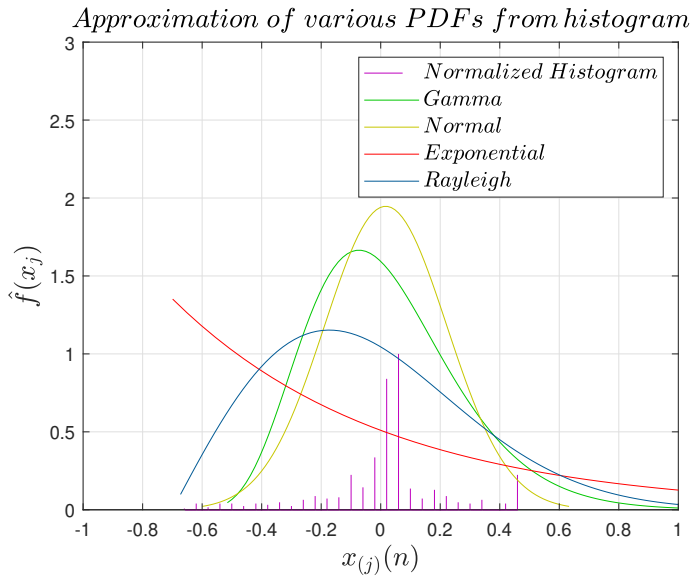


Figure 7.6: Approximation of various PDFs from histogram

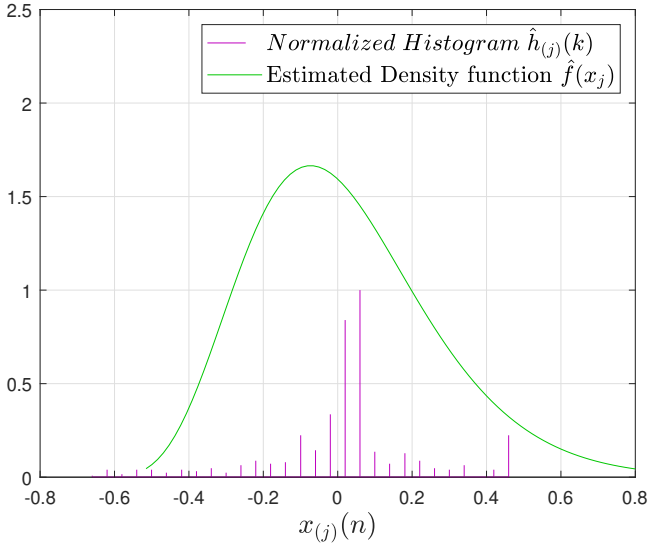


Figure 7.7: Estimating density function

### 7.2.4 Density Estimation

The purpose of this stage is to find best fit density function $\hat{f}(x_j)$ from $h_{(j)}(k)$. The real challenge here is how to estimate the underlying density of the distortion $x_j(n)$. Both parametric and non-parametric density estimation methods can be used to achieve this objective. For density estimation different options are available e.g.

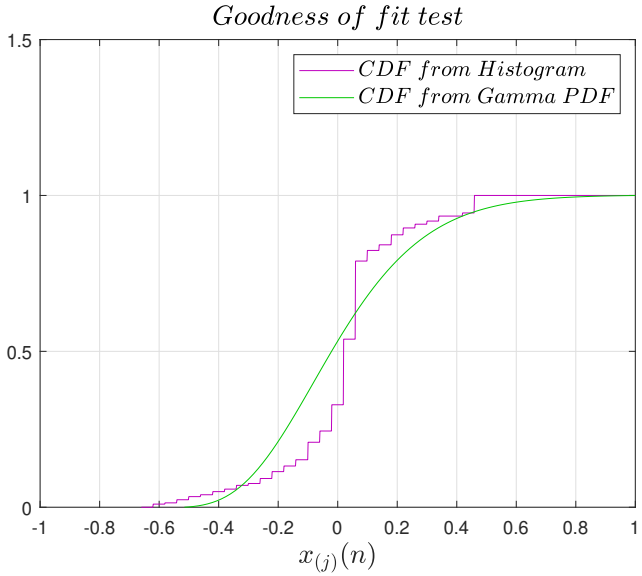


Figure 7.8: P-P plot

normal, exponential, rayleigh, gamma, etc. After extensive experimentation, the proposed scheme, however, uses the *gamma distribution*. The motivation behind gamma distribution is that for a given histograms, the gamma distribution yields optimally smooth density estimate (96) as confirmed in (Fig. 7.6). From Fig. 7.6, it can be observed that the normal distribution also fits the histogram, but as normal distribution is symmetric in nature, and histogram of $x_j(n)$ is non-symmetric in nature, hence, normal fit is not optimal choice. In order to compute the gamma distribution, I compute the mean $\mu_{x_j}$ and standard deviation $\sigma_{x_j}$ of the histogram using eq. 7.15, and eq. 7.16 and then compute the gamma distribution parameters $\alpha_j$ and $\beta_j$ using eq. 7.17, and eq. 7.18.

$$\mu_{x_j} = \frac{\left( \sum_{k=0}^{200} (k \times h_j(k)) \right)}{\left( \sum_{k=0}^{200} h_j(k) \right)} \tag{7.15}$$

$$\sigma_{x_j} = \sqrt{\left[ \frac{\left( \sum_{k=0}^{200} (k^2 \times h_j(k)) \right)}{\left( \sum_{k=0}^{200} h_j(k) \right)} - \mu_{x_j}^2 \right]} \tag{7.16}$$

and

$$\alpha_j = \left[ \frac{\mu_{x_j}}{\sigma_{x_j}} \right]^2. \tag{7.17}$$

$$\beta_j = \left[ \frac{\sigma_{x_j}^2}{\mu_{x_j}} \right]. \tag{7.18}$$

Afterwards, the gamma distribution function is computed as follows:

$$\hat{f}(x_j) = \frac{1}{\beta_j^{-\alpha_j} \tau(\alpha_j)} x^{\alpha_j - 1} e^{-\beta_j x_j}. \tag{7.19}$$

Here, $\tau(\alpha_j)$ is gamma function. Fig. 7.7 shows that the density function maps

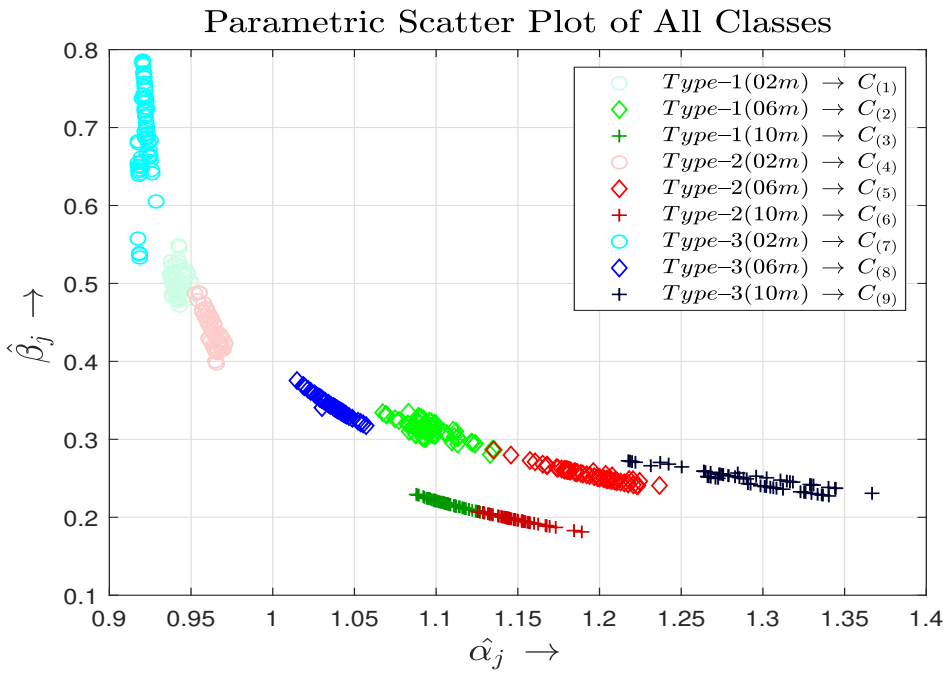


Figure 7.9: Scatter plot of $\alpha$ vs $\beta$ for $C_{(j)}$

over histogram of $x_j(n)$ which signifies that the distortion can be represented through $\alpha_j$ and $\beta_j$. Fig. 7.8 show the goodness of fit test test. It can be observed that the theoretical cumulative distribution function which is obtained from histogram of distortion is very close to the CDF of estimated gamma PDF represented by parameters $\alpha_j$ and $\beta_j$. By using $\alpha_j$ and $\beta_j$ my feature space significantly reduces as well, which shows that even the simple classification approaches can utilize this information effectively. I repeated this distribution approximation for all channels, and I observed that the distribution function of all the channels follow gamma density function. Therefore, if I plot $\alpha_j$ against $\beta_j$ for all channels $C_j$, where each channel has $M$ records as $\hat{\alpha_j} = [\alpha_{j1}, \alpha_{j2} \ldots \alpha_{jM}]$ and $\hat{\beta_j} = [\beta_{j1}, \beta_{j2} \ldots \beta_{jM}]$, as shown in Fig. 7.9, we can observe that the data points corresponding to each channel observe lower intra-class distance, however, there inter-class distance is high; thus, the benefit of my scheme is that the data points for each channel $C_j$ becomes linearly separable and can be captured through data clustering approaches. For capturing data points, I can apply any data clustering approach i.e. k-means (103), fuzzy c-mean clustering (104) etc.

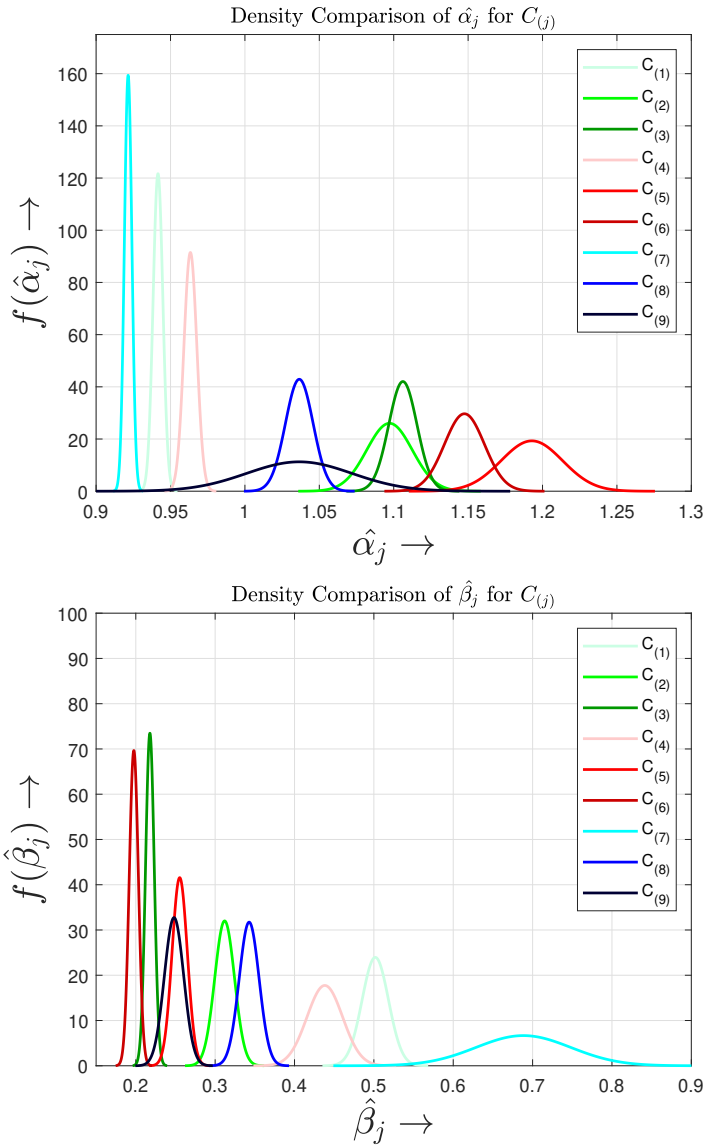


Figure 7.10: Distribution functions of $\alpha$ and $\beta$

to identify the channel. However, as these methods explicitly required the number of channels to compute the cluster centeroids, therefore, cannot be used in the situation where this information is not predetermined. Therefore, in this research I have used likelihood ratio test (96) as data clustering approach.

### 7.2.5 Adaptive Channel Specific Clustering Approach

As it can be observed from Fig. 7.9 that although data points corresponding to $\hat{\alpha}_j$ and $\hat{\beta}_j$ for $C_j$ have lower intra-class distance, but still there is small variation amongst the data points in $\alpha$ and $\beta$ planes that highlights their symmetric random nature. Therefore, in order to capture this symmetric randomness in both planes, I approximate the PDF of $\hat{\alpha}_j$ and $\hat{\beta}_j$ by normal distribution function as described in eq. (7.20) and eq. (7.21) respectively. The density functions, $f(\hat{\alpha}_j)$ and $f(\hat{\beta}_j)$ for $C_j$ are shown in Fig. 7.10.

$$f(\hat{\alpha}_j) = \frac{1}{\sqrt{2\pi\sigma_{\alpha_j}^2}} exp(-\frac{1}{2}(\alpha_j - \mu_{\alpha_j})^2). \tag{7.20}$$

$$f(\hat{\beta}_j) = \frac{1}{\sqrt{2\pi\sigma_{\beta_j}^2}} exp(-\frac{1}{2}(\beta_j - \mu_{\beta_j})^2). \tag{7.21}$$

I assume that $f(\hat{\alpha}_j, \hat{\beta}_j)$ are independent of each other. By using eq. (7.20) and eq. (7.21), I compute the joint density function, $f(\hat{\alpha}_j, \hat{\beta}_j)$, as described in eq. (7.22). The joint density function, $f(\hat{\alpha}_j, \hat{\beta}_j)$ for $C_j$ is shown in Fig. 7.11.

$$f(\hat{\alpha}_j, \hat{\beta}_j) = f(\hat{\alpha}_j).f(\hat{\beta}_j) \tag{7.22}$$

or,

$$f(\hat{\alpha}_j, \hat{\beta}_j) = \frac{\exp\left\{-\frac{1}{2}\left[\left(\frac{\hat{\alpha}_j - \mu_{\alpha_j}}{\sigma_{\alpha_j}}\right)^2 + \left(\frac{\hat{\beta}_j - \mu_{\beta_j}}{\sigma_{\beta_j}}\right)^2\right]\right\}}{2\pi\sigma_{\alpha_j}\sigma_{\beta_j}} \tag{7.23}$$

where, $\mu_{\alpha_j}$, $\mu_{\beta_j}$, $\sigma_{\alpha_j}$ and $\sigma_{\beta_j}$ can be expressed by eq. (7.24)-(7.27) respectively.

$$\mu_{\alpha_j} = \frac{\sum_{i=1}^{M}(\alpha_{ji})}{M} \tag{7.24}$$

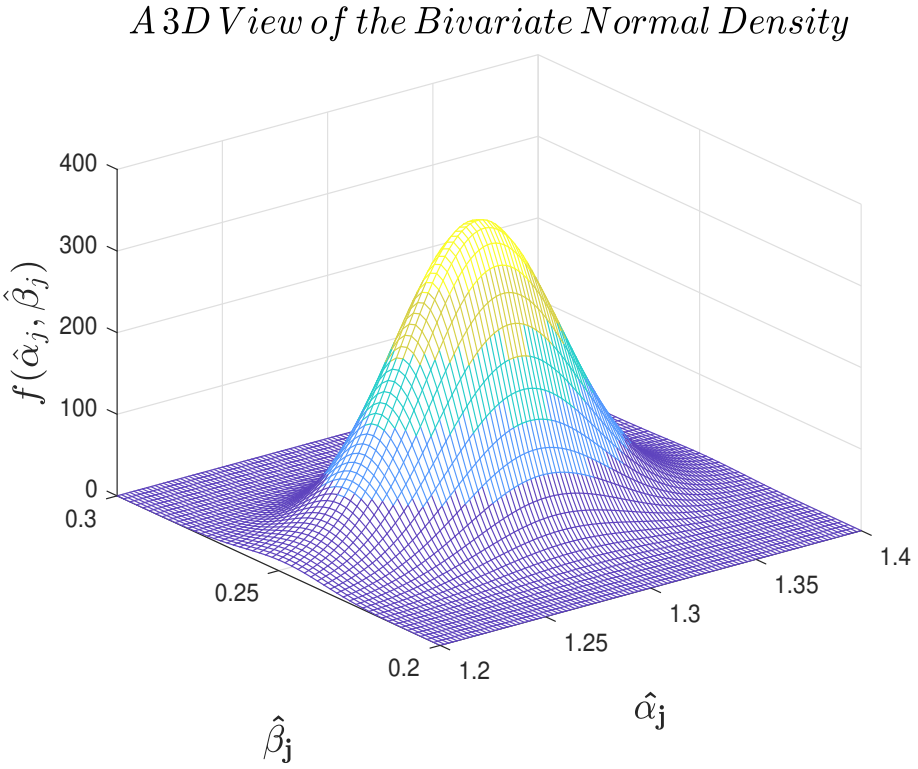


Figure 7.11: 3-D plot of joint density function for single channel

$$\mu_{\beta_j} = \frac{\sum_{i=1}^{M}(\beta_{ji})}{M} \tag{7.25}$$

$$\sigma^2_{\alpha_j} = \frac{\sum_{i=1}^{M}(\alpha_{ji} - \mu_{\alpha_j})^2}{M} \tag{7.26}$$

$$\sigma^2_{\beta_j} = \frac{\sum_{i=1}^{M}(\beta_{ji} - \mu_{\beta_j})^2}{M} \tag{7.27}$$

$f(\hat{\alpha}_j, \hat{\beta}_j)$ is used to apply likelihood ratio test to identify the transmitting channel using the following hypothesis:

$$H_r : x_{test} = x_r \tag{7.28}$$

$$H_q : x_{test} = x_q$$

Here, $r \neq q$, $r \in \psi$, $q \in \psi$, $test \in \psi$, and $\psi = \{1, 2, \ldots, n\}$ and n represents the unique channel index.

$$\Lambda(x) = \frac{f(x_{test}; \theta_r, H_r)}{f(x_{test}; \theta_r, H_q)} = \begin{matrix} H_q \\ \lessgtr \\ H_r \end{matrix} 1 \tag{7.29}$$

In eq. (7.29), $\theta_r$ and $\theta_q$ can be expressed as eq. (7.30) and eq. (7.31) respectively.

$$\theta_r = \{\hat{\mu}_{\alpha_r}, \hat{\mu}_{\beta_r}, \hat{\sigma}_{\alpha_r}, \hat{\sigma}_{\beta_r}\} \tag{7.30}$$

and

$$\theta_q = \{\hat{\mu}_{\alpha_q}, \hat{\mu}_{\beta_q}, \hat{\sigma}_{\alpha_q}, \hat{\sigma}_{\beta_q}\} \tag{7.31}$$

Where, $\hat{\mu}_{\alpha_r}$, $\hat{\mu}_{\beta_r}$, $\hat{\mu}_{\alpha_q}$ and $\hat{\mu}_{\beta_q}$ are show in eq. (7.32) - (7.35).

$$\hat{\mu}_{\alpha_r} = \frac{\sum_{i=1}^{M}(\alpha_{ri})}{M} \tag{7.32}$$

$$\hat{\mu}_{\beta_r} = \frac{\sum_{i=1}^{M}(\beta_{ri})}{M} \tag{7.33}$$

$$\hat{\mu}_{\alpha_q} = \frac{\sum_{i=1}^{M}(\alpha_{qi})}{M} \tag{7.34}$$

$$\hat{\mu}_{\beta_q} = \frac{\sum_{i=1}^{M}(\beta_{qi})}{M} \tag{7.35}$$

and, $\hat{\sigma}_{\alpha_r}$, $\hat{\sigma}_{\beta_r}$, $\hat{\sigma}_{\alpha_q}$ and $\hat{\sigma}_{\beta_q}$ are show in eq. (7.36) - (7.39).

$$\hat{\sigma}_{\alpha_r} = \sqrt{\left[\frac{\sum_{i=1}^{M}(\alpha_{ri} - \mu_{\alpha_r})^2}{M}\right]} \tag{7.36}$$

$$\hat{\sigma}_{\beta_r} = \sqrt{\left[\frac{\sum_{i=1}^{M}(\beta_{ri} - \mu_{\beta_r})^2}{M}\right]} \tag{7.37}$$

$$\hat{\sigma}_{\alpha_q} = \sqrt{\left[\frac{\sum_{i=1}^{M}(\alpha_{qi} - \mu_{\alpha_q})^2}{M}\right]} \tag{7.38}$$

$$\hat{\sigma}_{\beta_q} = \sqrt{\left[\frac{\sum_{i=1}^{M}(\beta_{qi} - \mu_{\beta_q})^2}{M}\right]} \tag{7.39}$$

The decision function considers $x_{test}(n)$ belongs to $r^{th}$ transmitter if the eq. (7.29) is true for all values of $q$.

Table 7.1: Technical specifications of channels

| Label | Length(m) | Conductor | Insulation | Model |
|---|---|---|---|---|
| $C_{(1)}$ | 2 | Copper | XLPO | SAE J1939-15 |
| $C_{(2)}$ | 6 | Copper | XLPO | SAE J1939-15 |
| $C_{(3)}$ | 10 | Copper | XLPO | SAE J1939-15 |
| $C_{(4)}$ | 2 | Copper | XLPO | SAE J1939-19 |
| $C_{(5)}$ | 6 | Copper | XLPO | SAE J1939-19 |
| $C_{(6)}$ | 10 | Copper | XLPO | SAE J1939-19 |
| $C_{(7)}$ | 2 | Copper | XLPO | SAE J1128 |
| $C_{(8)}$ | 6 | Copper | XLPO | SAE J1128 |
| $C_{(9)}$ | 10 | Copper | XLPO | SAE J1128 |

## 7.3 Experimental Results and Analysis

### 7.3.1 Experimental Setup

**CAN channel specifications:**

To realize CAN channel, CAN cables commonly used by the automotive industry under the SAE J1939 standard from two manufacturers are used. To this end, nine

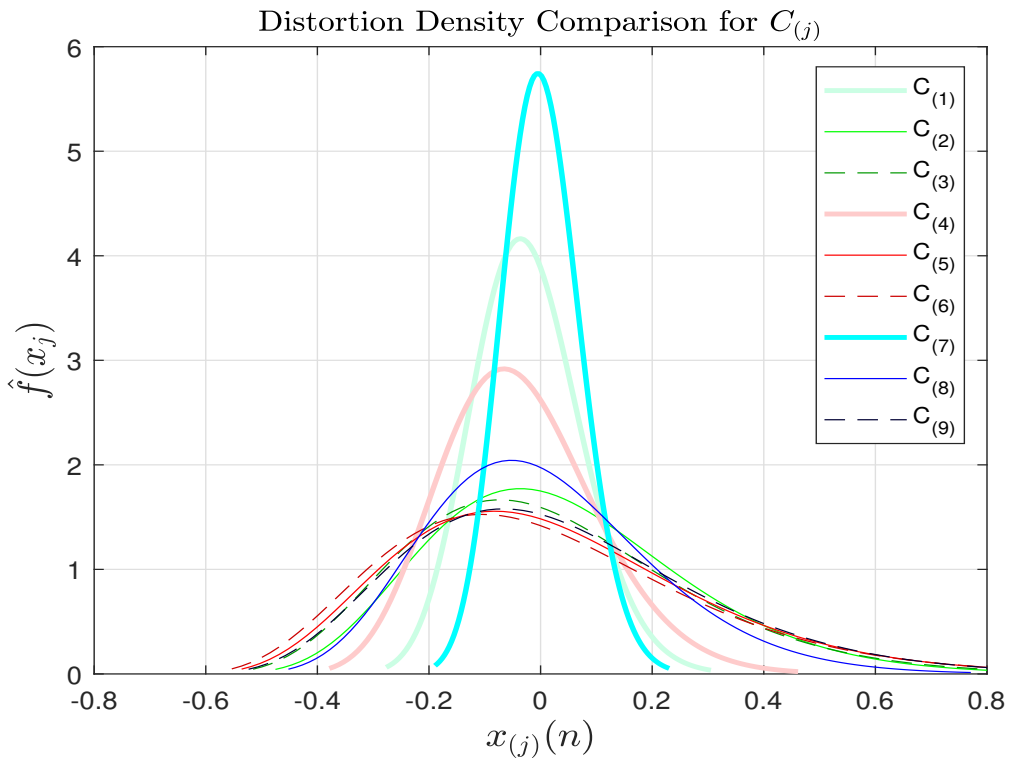


Figure 7.12: Comparison of estimated distortion density for $C_{(j)}$

different CAN channels by either varying manufacturer or channel length are used. Technical specifications of the CAN channel used for performance evaluation of the proposed method are given in Table 7.1.

**ECU and CAN communication**:

For CAN communication, Arduino Uno R2 micro-controller kit; CAN-Bus shield board with MCP2515 CAN-bus controller and the MPC2551 CAN transceiver are used. The same script is used for sending an identical message continuously over 9 channels. For CAN bus data acquisition, oscilloscope DSO1012A for the voltage samples recording with sampling sate of 20 $MSa/sec$ is used. The technical specifications of channels are provided in Table 7.1.

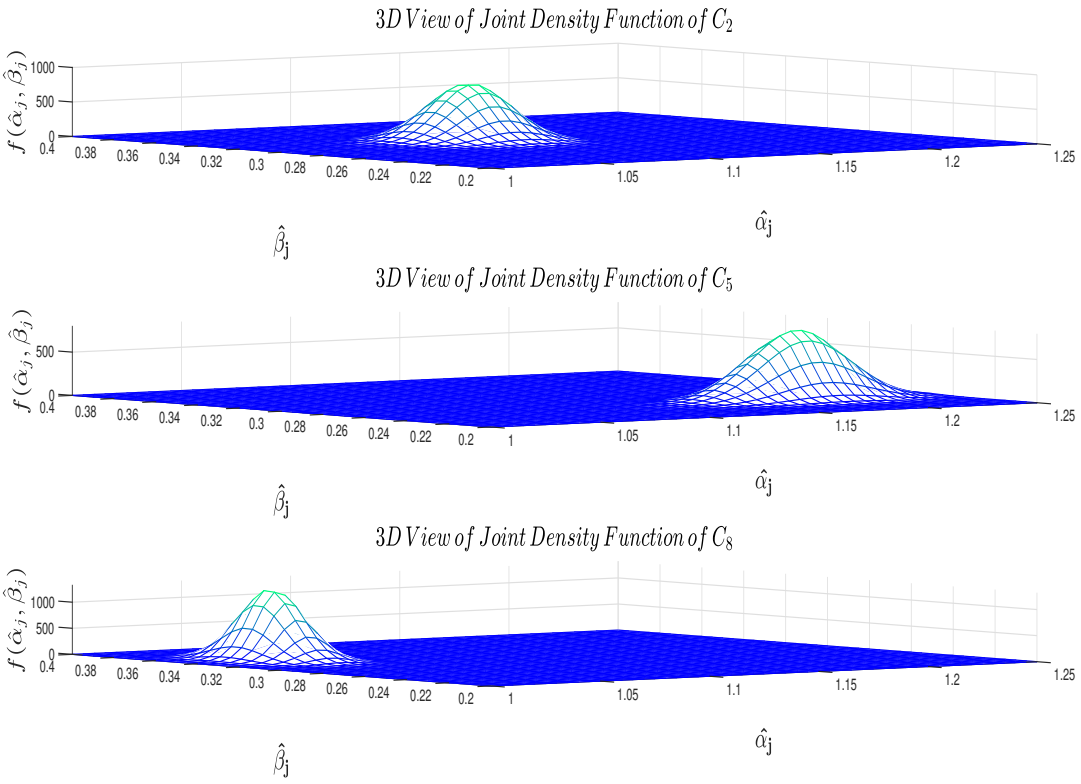


Figure 7.13: Comparison of joint density functions

### 7.3.2 Dataset Description

The channel identification dataset comprised of 1080 $(120 \times 9)$ records with 1500 samples in each record. For performance evaluation, partitioning is performed to divide the dataset into the training and test sets. For this 50% of the set is used for training and 50% of the set is used for testing. The data is collected in the same environment (i.e. under the same temperature and using an identical message) to observe the minute and unique variation of the digital signals.

### 7.3.3 Performance Evaluation Measures

For performance evaluation, I used precision, recall, $F_1$ $-score$, accuracy, and error rate as performance evaluation measures. To evaluate the effectiveness of the proposed

method, I determined how many channels were correctly identified in the response to messages sniffed by $E_{(s)}$. Let $TP$ represents true positive rate, $FP$ represents false positive rate, $TN$ represents true negative rate, and $FN$ represents false-negative rate, then precision can be defined as follows:

$$Precision = \left( \frac{TP}{TP + FP} \right) \tag{7.40}$$

Precision was used to measure the ratio of the true instances against the retrieved instances for a particular class. To measure the sensitivity I used the recall rates that can be computed as follows:

$$Recall = \left( \frac{TP}{TP + FN} \right) \tag{7.41}$$

The recall was computed to measure the total number of relevant instances that were actually retrieved. In order to combine both measures i.e. precision, and recall I used $F_1\, Score$ that was computed as:

$$F_1\, score = 2 \times \left( \frac{Precision \times Recall}{Precision + Recall} \right) \tag{7.42}$$

The higher $F_1\, Score$ signifies the robustness of the classification approach. In order to evaluate the overall performance by considering all the classes together, I computed the accuracy of the method as follows:

$$Accuracy = \left( \frac{TP + TN}{TP + TN + FP + FN} \right) \tag{7.43}$$

Accuracy was computed to measure all instances that were correctly classified, despite the fact, whatever class they belong to. Moreover, by using accuracy value I also computed the overall error rate of the method as follows:

Table 7.2: Confusion matrix for channel classifier

| | | *Target Class* | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| - | - | $C_{(1)}$ | $C_{(2)}$ | $C_{(3)}$ | $C_{(4)}$ | $C_{(5)}$ | $C_{(6)}$ | $C_{(7)}$ | $C_{(8)}$ | $C_{(9)}$ | Acc. % |
| *Predicted Class* | $C_{(1)}$ | 60 | 0 | 0 | 4 | 0 | 0 | 4 | 0 | 0 | 88.2 |
| | $C_{(2)}$ | 0 | 57 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 96.6 |
| | $C_{(3)}$ | 0 | 0 | 59 | 0 | 0 | 2 | 0 | 0 | 0 | 96.7 |
| | $C_{(4)}$ | 0 | 0 | 0 | 56 | 0 | 0 | 0 | 0 | 0 | 100 |
| | $C_{(5)}$ | 0 | 0 | 0 | 0 | 59 | 0 | 0 | 0 | 7 | 89.4 |
| | $C_{(6)}$ | 0 | 0 | 1 | 0 | 0 | 58 | 0 | 0 | 0 | 98.3 |
| | $C_{(7)}$ | 0 | 0 | 0 | 0 | 0 | 0 | 56 | 0 | 0 | 100 |
| | $C_{(8)}$ | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 59 | 0 | 95.2 |
| | $C_{(9)}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 53 | 100 |
| | Acc. % | 100 | 95 | 98.3 | 93.3 | 98.3 | 96.7 | 93.3 | 98.3 | 88.3 | 95.74 |

$$Error–rate = 1 - Accuracy \tag{7.44}$$

### 7.3.4 Performance Analysis

This section provides channel detection performance of the proposed method in terms of evaluation measures. A series of experiments were performed to evaluate the performance of the proposed method.

**Experiment 1 - Channel-specific parametric analysis of distortion:**

The goal of this experiment is to validate that the distortion is a channel specific entity thus has the ability to link the signal to the transmitting channel. The transmitted signal between ECUs using the transmitting channel does not by default contains the sender ID, thus is prone to the spoofing attack. However, my approach extracts the sender information through the signal analysis in terms of channel specific distortion profile. To achieve this objective, it is necessary that the distortion profile should be unique for each channel in the CAN network. Therefore, in this experiment I find the distortion profile for each channel in terms of gamma density function; and from Fig. 7.12 it can be observed that the estimated distortion densities for all the

9 channels are unique, which consequently generates unique $\alpha_j$ and $\beta_j$ parameters of gamma distribution for all channels that can be verified from Fig. 7.9. Further, it can be observed that channels with larger lengths (as listed in Table 7.1) have more spread in the density functions. This spread on $x$–axis is due to an increase in the variance of the density function. Gamma densities of some channels are correlated as they overlap with each other. If we find probability density functions of $\hat{\alpha}_j$ and $\hat{\beta}_j$ for each channel $C_j$ and their joint density function $f(\hat{\alpha}, \hat{\beta})$, it can be observed that they are uncorrelated as they do not overlap each other as shown in Fig. 7.13, the joint density function of $\hat{\alpha}_j$ and $\hat{\beta}_j$ is an ideal candidate for channel identification. Hence, as the density function of each channel is unique, so it can be used to link the signal to the source transmitter for message authentication.

**Experiment 2 - Transmitter identification based upon channel distortion:**

The main purpose of this experiment is to test the accuracy of the channel identification method proposed in this research. Estimated channel distortion density is then used for transmitter identification. To achieve this goal, both parametric binary hypothesis testing or non-parametric hypothesis testing based on machine learning can be used. As estimated density is a parametric approach, therefore, likelihood is used. During training phase, the acquired signal from each channel is processed to extract channel-specific distortion, which is used to estimate the underlying density, $\hat{f}(x_j)$ and its parameters . Further, $\hat{f}(x_j)$ is computed M times,to generate $\alpha_j$ and $\beta_j$ and find joint density function $f(\hat{\alpha}, \hat{\beta})$. The $f(\hat{\alpha}, \hat{\beta})$ is then used to classify channel. During testing phase, channel-specific distortion ($x_{test}(t)$) is extracted from the test signal ($y_{test}(t)$), $x_{test}(t)$, which is used for underlying density estimation, $\hat{f}(x_{test})$. Parameters $\alpha$ and $\beta$ are acquired from $\hat{f}_{\mathrm{x}_{test}}(x)$. Finally, maximum likelihood is applied for channel label assignment. The experimental results show that the proposed algorithm achieves a satisfactory performance with a false positive rate of 4.26% and

identification accuracy of 95.74% as shown in Table 7.2. It can also be noticed from Table 7.2 that the false positive which appear for CAN channel $C_{(1)}$ are the signals which belong to $C_{(4)}$ and $C_{(7)}$, these channels are of the same length. The same pattern can be observed for all the other classes as well, i.e. sources of errors/false positives are those classes that have the same cable length.

Table 7.3: Performance matrix of channel classifier

| - | *Precision* | *Recall* | *Accuracy* | $F_1$ *–score* | *ERR* |
|---|---|---|---|---|---|
| $C_{(1)}$ | 88.2% | 100% | 98.5% | 93.8% | 1.5% |
| $C_{(2)}$ | 96.6% | 95% | 99.1% | 95.8% | 0.9% |
| $C_{(3)}$ | 96.7% | 98.3% | 99.4% | 97.5% | 0.6% |
| $C_{(4)}$ | 100% | 93.3% | 99.2% | 96.6% | 0.8% |
| $C_{(5)}$ | 89.4% | 98.3% | 98.5% | 93.7% | 1.5% |
| $C_{(6)}$ | 98.3% | 96.7% | 99.4% | 97.5% | 0.6% |
| $C_{(7)}$ | 100% | 93.3% | 99.2% | 96.6% | 0.8% |
| $C_{(8)}$ | 95.2% | 98.3% | 99.2% | 96.7% | 0.8% |
| $C_{(9)}$ | 100% | 88.3% | 98.7% | 93.8% | 1.3% |

Table 7.3 shows the performance in terms of different performance evaluation measures. The performance of channel-classifier is quantified in terms of precision, recall, $F_1$ *–score*, accuracy, and error rate. In 3 out of 9 cases, my method achieved 100% precision, as observed from Table 7.3, $C_{(4)}$, $C_{(7)}$ and $C_{(9)}$ have 100% precision. It can also be observed that $C_{(1)}$ has a recall of 100%. Further, $C_{(3)}$ and $C_{(6)}$ stand out in terms of $F_1$ *–score*, they both have an $F_1$ *–score* of 97.5%. The overall detection rate of channel identification is 95.74%. The same results are graphically presented in Fig. 7.14.

### 7.3.5 Comparison Against State-of-the-art

In this section proposed method is compared against state-of-the-art methods that are also doing the transmitter identification. The performance is compared against transmitter detection using Viden (25), Inimitable characteristics of CAN signal (43) and VoltageIDS (44).

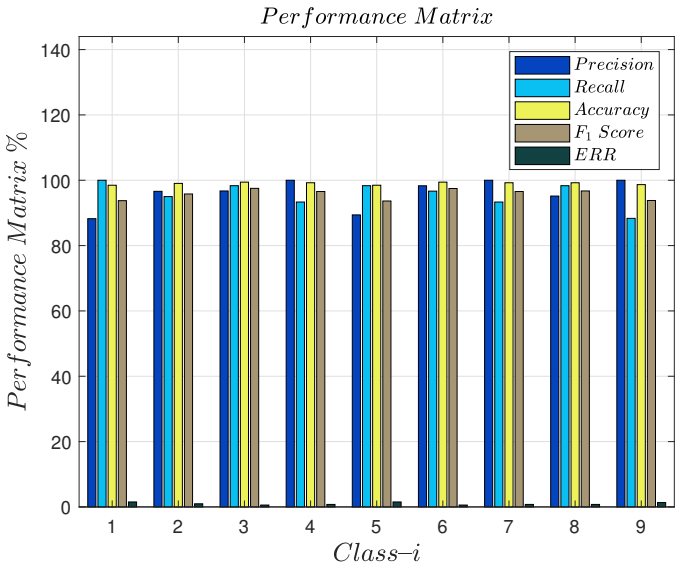


Figure 7.14: Bar graph of PM for classifier

In (25), Cho et. al. proposed a method named *Viden* that used voltage profile of acknowledgment (*Ack*) bits for transmitter identification. In the first phase, ACK bit was used to measure the message was originated from the genuine transmitter or not. Afterward, voltage measurements were used to generate transmitter fingerprints. Based on these fingerprints, the attacker transmitter was identified. In (43), a monitoring unit was installed in the vehicle that analyzes the electrical CAN signals and computes the statistical features. These features were then classified to identify the transmitter. In (44), transmitter detection based on inimitable voltage characteristics technique was proposed. The feature vectors proposed in (43) were extended both in time- and frequency domains, and were classified for transmitter identification in (44).

Table 7.4 shows the performance comparison of my method against (44; 43; 25). From the results, it can be observed that my method is giving accuracy as very close to (44; 43). However, the main advantage of my method is that feature extraction and message authentication can be done in any part of the signal without the latency. Whereas in case of Viden, voltage profile is estimated for message authentication

Table 7.4: Comparison with other methods

| *ResearchWork* | *Method* | *Accuracy* |
|---|---|---|
| Cho et. al. (25) | *Viden* | 99.57% |
| Choi et. al. (43) | *Inimitable Char. of CAN Signal* | 96.48% |
| Choi et. al. (44) | *VoltageIDS* | 95.54% |
| This method | *Channel Distortion based IDS* | 95.74% |

during the reception of the ACK bit but it also introduces the latency. Hence, from the aspect of latency, my method is more robust than the Viden. Although, the channel detection rates are slightly lower than Viden, as the channel-detection using parametric estimation is a novel concept, so the research efforts can be done in this area to further generate more interesting findings. In (43), message authentication is done based on using inimitable characteristics of signal. The feature vector consists of 40 features both in time domain and frequency domain, further a multilayered neural network is trained based on this feature-set. This approach is computationally expensive, hence, it adds a lot of overhead to the system, which makes this approach impractical to be implemented. Whereas, in my case, there are 2 advantages of my approach as compared to approach inimitable characteristics of signal which was proposed in (43). The first approach is that my model is trained only on 2 features $\alpha$ and $\beta$, whereas 40 features were used for training in (43). The second advantage is that likelihood ratio test is used for channel identification which is computationally less expensive as compared to neural networks.

# CHAPTER VIII

# Conclusion and Future Work

In this research, a novel approach for ECU identification is presented for message authentication. The main motivation behind ECU identification is that in CAN messages, as the sender (ECU) information is missing, therefore, they are prone to spoofing attacks; however, with the ECU identification, spoofing attacks can easily be prevented. In this research, the ECU specific distortion is utilized for ECU-fingerprinting, and ECUs are identified through a 4-layered double Neural network architecture. The uniqueness of my approach is that it is utilizing $DAC$ imperfections and semi-conductor impurities that are hard to replicate. The ECU imperfections are unique for all ECUs even from the same make, model, and manufacturer. Thus, this information has the potential to prevent spoofing attacks. Another novel contribution in this research is that the same fingerprinting technique can be used for ECU-pin identification, which can then be used to avoid pin-level attacks e.g. voltage-based attacks.

Another contribution of this research is that, it has been demonstrated that the channel-specific step response in the received signal is unique; hence, it can be utilized to associate CAN packets with their source transmitter. The proposed system uses the transient response parameters of the channel to capture uniqueness in the received signal, which is used as inputs to a multi-layer neural network classifier. The clas-

sification method's performance is evaluated on eight different CAN channels. The experimental results indicate that each channels' set of feature vectors is significantly different from all other channels. The proposed method achieves channel detection with an accuracy of 97.4%.

Another contribution of this research is that it uses a distribution profile of the channel-specific distortion to capture uniqueness in the received signal, which is used to train a neural network classifier. The performance of the classification method is evaluated on nine different CAN channels. The experimental results indicate that the estimated densities are significantly different even for the same length and manufacturer. This method achieves an accuracy of 99.4% using a single layer neural network.

The main contribution of this research is using parametric analysis in the received signal and then identify the transmitter. This method is computationally inexpensive as compared to other methods. This method achieves transmitter detection with an accuracy of 95.74%. This method is superior to encryption-based techniques since it does not adds additional overhead in the system. Hence, it makes the system robust. It is also superior to machine learning IDS techniques implemented at the physical layer since it is computationally less expensive as compared to other machine learning-based techniques at the physical layer. For future work, I will localize the transmitter based on channel distortion. In case when an attack is detected, the attacker transmitter will be isolated from the CAN, and will not be allowed to transmit any message. I also plan to test my method at different environmental conditions i.e. temperature, humidity, and electromagnetic interference, etc. in real-time scenarios. In this regard, fuzzy logic-based decision modeling will be employed to adapt channel-identification corresponding to different environmental conditions.

# BIBLIOGRAPHY

# BIBLIOGRAPHY


[1] "AUTOSAR, Specification of Lin Interface," http://bit.ly/33EJDyk, 2017.

[2] I. A. Grzemba, *MOST: the automotive multimedia network*. Franzis Verlag, 2012.

[3] S. Lorenz, "The flexray electrical physical layer evolution," *SPECIAL EDITION HANSER automotive FLEXRAY*, pp. 14–16, 2010.

[4] S. Kempainen, "Low-voltage differential signaling (lvds)," *Altera Co-operation*, 2002.

[5] "Audio Video Bridging," http://bit.ly/34EK7Vj, 2019.

[6] A. Hafeez, K. Topolovec, C. Zolo, and W. Sarwar, "State of the art survey on comparison of can, flexray, lin protocol and simulation of lin protocol," SAE Technical Paper, Tech. Rep., 2020.

[7] A. Hafeez, K. Rehman, and H. Malik, "State of the art survey on comparison of physical fingerprinting-based intrusion detection techniques for in-vehicle security," SAE Technical Paper, Tech. Rep., 2020.

[8] K. Tindell, H. Hanssmon, and A. J. Wellings, "Analysing real-time communications: Controller area network (can)." in *RTSS*. Citeseer, 1994, pp. 259–263.

[9] "Upstream Security Global Automotive Cybersecurity Report," http://bit.ly/2oTQtB0, 2019.

[10] M. Marchetti and D. Stabili, "Read: Reverse engineering of automotive data frames," *IEEE Trans. on Information Forensics and Security*, vol. 14, no. 4, pp. 1083–1097, 2018.

[11] J. Liu, S. Zhang, W. Sun, and Y. Shi, "In-vehicle network attacks and countermeasures: Challenges and future directions," *IEEE Network*, vol. 31, no. 5, pp. 50–58, 2017.

[12] S. Fröschle and A. Stühring, "Analyzing the capabilities of the can attacker," in *European Symposium on Research in Computer Security*. Springer, 2017, pp. 464–482.

[13] C. Miller and C. Valasek, "A survey of remote automotive attack surfaces," *black hat USA*, vol. 2014, p. 94, 2014.

[14] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, T. Kohno *et al.*, "Comprehensive experimental analyses of automotive attack surfaces." in *USENIX Security Symposium*, vol. 4. San Francisco, 2011, pp. 447–462.

[15] I. Studnia, V. Nicomette, E. Alata, Y. Deswarte, M. Kaaniche, and Y. Laarouchi, "Survey on security threats and protection mechanisms in embedded automotive networks," in *2013 43rd Annual IEEE/IFIP Conference on Dependable Systems and Networks Workshop (DSN-W)*. IEEE, 2013, pp. 1–12.

[16] A. Greenberg, "Hackers remotely kill a jeep on the highway—with me in it," *Wired*, vol. 7, p. 21, 2015.

[17] Q. Wang and S. Sawhney, "Vecure: A practical security framework to protect the can bus of vehicles," in *Internet of Things (IOT), 2014 International Conference on the*. IEEE, 2014, pp. 13–18.

[18] B. Gierlichs and A. Y. Poschmann, *Cryptographic Hardware and Embedded Systems–CHES 2016*. Springer, 2016.

[19] A. Hazem and H. Fahmy, "Lcap-a lightweight can authentication protocol for securing in-vehicle networks," in *10th escar Conf. Embedded Security in Cars, Berlin, Germany*, vol. 6, 2012.

[20] T. P. Doan and S. Ganesan, "Can crypto fpga chip to secure data transmitted through can fd bus using aes-128 and sha-1 algorithms with a symmetric key," SAE Technical Paper, Tech. Rep., 2017.

[21] H. Ueda, R. Kurachi, H. Takada, T. Mizutani, M. Inoue, and S. Horihata, "Security authentication system for in-vehicle network," *SEI Technical Review*, vol. 81, pp. 5–9, 2015.

[22] T. Sugashima, D. K. Oka, and C. Vuillaume, "Approaches for secure and efficient in-vehicle key management," *SAE International Journal of Passenger Cars-Electronic and Electrical Systems*, vol. 9, no. 2016-01-0070, pp. 100–106, 2016.

[23] A. Hafeez, H. Malik, O. Avatefipour, P. R. Rongali, and S. Zehra, "Comparative study of can-bus and flexray protocols for in-vehicle communication," SAE Technical Paper, Tech. Rep., 2017.

[24] M. Wolf, A. Weimerskirch, and C. Paar, "Security in automotive bus systems," in *Workshop on Embedded Security in Cars*, 2004.

[25] K.-T. Cho and K. G. Shin, "Viden: Attacker identification on in-vehicle networks," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2017, pp. 1109–1123.

[26] ——, "Fingerprinting electronic control units for vehicle intrusion detection." in *USENIX Security Symposium*, 2016, pp. 911–927.

[27] B. Groza and P.-S. Murvay, "Efficient intrusion detection with bloom filtering in controller area networks (can)," *IEEE Transactions on Information Forensics and Security*, 2018.

[28] S. U. Sagong, X. Ying, A. Clark, L. Bushnell, and R. Poovendran, "Cloaking the clock: emulating clock skew in controller area networks," in *Proceedings of the 9th ACM/IEEE International Conference on Cyber-Physical Systems*. IEEE Press, 2018, pp. 32–42.

[29] H. Lee, S. Jeong, and H. Kim, "Otids: A novel intrusion detection system for in-vehicle network by using remote frame," in *15th Annual Conf. on Privacy, Security and Trust (PST)*. IEEE, 2017, pp. 57–5709.

[30] A. Taylor, N. Japkowicz, and S. Leblanc, "Frequency-based anomaly detection for the automotive can bus," in *World Congress on Industrial Control Systems Security (WCICSS)*. IEEE, 2015, pp. 45–49.

[31] H. Song, H. Kim, and H. Kim, "Intrusion detection system based on the analysis of time intervals of can messages for in-vehicle network," in *Int. Conf. on information networking (ICOIN)*. IEEE, 2016, pp. 63–68.

[32] M. Marchetti, D. Stabili, A. Guido, and M. Colajanni, "Evaluation of anomaly detection for in-vehicle networks through information-theoretic algorithms," in *IEEE 2nd Int. Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI)*. IEEE, 2016, pp. 1–6.

[33] W. Wu, Y. Huang, R. Kurachi, G. Zeng, G. Xie, R. Li, and K. Li, "Sliding window optimized information entropy analysis method for intrusion detection on in-vehicle networks," *IEEE Access*, vol. 6, pp. 45 233–45 245, 2018.

[34] D. Stabili, M. Marchetti, and M. Colajanni, "Detecting attacks to internal vehicle networks through hamming distance," in *Int. Annual Conf. AEIT*. IEEE, 2017, pp. 1–6.

[35] A. Taylor, S. Leblanc, and N. Japkowicz, "Anomaly detection in automobile control network data with long short-term memory networks," in *IEEE Int. Conf. on Data Science and Advanced Analytics (DSAA)*. IEEE, 2016, pp. 130–139.

[36] M. Kang and J. Kang, "Intrusion detection system using deep neural network for in-vehicle network security," *PloS one*, vol. 11, no. 6, p. e0155781, 2016.

[37] M. Markovitz and A. Wool, "Field classification, modeling and anomaly detection in unknown can bus networks," *Vehicular Communications*, vol. 9, pp. 43–52, 2017.

[38] N. Jain and S. Sharma, "The role of decision tree technique for automating intrusion detection system," *Int. Jour. of Computational Engineering Research*, vol. 2, no. 4, 2012.

[39] R. Rieke, M. Seidemann, E. Talla, D. Zelle, and B. Seeger, "Behavior analysis for safety and security in automotive systems," in *25th Euromicro Int. Conf. on Parallel, Distributed and Network-based Processing (PDP)*. IEEE, 2017, pp. 381–385.

[40] S. Narayanan, S. Mittal, and A. Joshi, "Using data analytics to detect anomalous states in vehicles," *arXiv preprint arXiv:1512.08048*, 2015.

[41] M. Marchetti and D. Stabili, "Anomaly detection of can bus messages through analysis of id sequences," in *IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 1577–1583.

[42] A. Hafeez, M. Tayyab, C. Zolo, and S. Awad, "Fingerprinting of engine control units by using frequency response for secure in-vehicle communication," in *2018 14th International Computer Engineering Conference (ICENCO)*. IEEE, 2018, pp. 79–83.

[43] W. Choi, H. J. Jo, S. Woo, J. Y. Chun, J. Park, and D. H. Lee, "Identifying ecus using inimitable characteristics of signals in controller area networks," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 6, pp. 4757–4770, 2018.

[44] W. Choi, K. Joo, H. J. Jo, M. C. Park, and D. H. Lee, "Voltageids: Low-level communication characteristics for automotive intrusion detection system," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 8, pp. 2114–2129, 2018.

[45] M. Kneib and C. Huth, "Scission: Signal characteristic-based sender identification and intrusion detection in automotive networks," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2018, pp. 787–800.

[46] O. Avatefipour, A. Hafeez, M. Tayyab, and H. Malik, "Linking received packet to the transmitter through physical-fingerprinting of controller area network," in *2017 IEEE Workshop on Information Forensics and Security (WIFS)*. IEEE, 2017, pp. 1–6.

[47] A. Perrig, R. Canetti, J. D. Tygar, and D. Song, "Efficient authentication and signing of multicast streams over lossy channels," in *Proceeding 2000 IEEE Symposium on Security and Privacy. S&P 2000*. IEEE, 2000, pp. 56–73.

[48] S. Nürnberger and C. Rossow, "–vatican–vetted, authenticated can bus," in *Int. Conf. on Cryptographic Hardware and Embedded Systems*. Springer, 2016, pp. 106–124.

[49] M. Tayyab, A. Hafeez, and H. Malik, "Spoofing attack on clock based intrusion detection system in controller area networks."

[50] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," *Black Hat USA*, p. 91, 2015.

[51] A. Sawant, S. Lenina, and M. D. Joshi, "Can, flexray, most versus ethernet for vehicular networks." 2018.

[52] N. Navet, Y. Song, F. Simonot-Lion, and C. Wilwert, "Trends in automotive communication systems," *Proceedings of the IEEE*, vol. 93, no. 6, pp. 1204–1223, 2005.

[53] E. A. Bretz, "By-wire cars turn the corner," *IEEE Spectrum*, vol. 38, no. 4, pp. 68–73, 2001.

[54] R. M. Daoud, H. H. Amer, H. M. Elsayed, and Y. Sallez, "Fault-tolerant ethernet-based vehicle on-board networks," in *IECON 2006-32nd Annual Conference on IEEE Industrial Electronics*. IEEE, 2006, pp. 4662–4665.

[55] ——, "Ethernet-based car control network," in *2006 Canadian Conference on Electrical and Computer Engineering*. IEEE, 2006, pp. 1031–1034.

[56] W. Zeng, M. Khalid, and S. Chowdhury, "A qualitative comparison of flexray and ethernet in vehicle networks," in *2015 IEEE 28th Canadian Conference on Electrical and Computer Engineering (CCECE)*. IEEE, 2015, pp. 571–576.

[57] H. Kimm and H.-s. Ham, "Integrated fault tolerant system for automotive bus networks," in *2010 Second International Conference on Computer Engineering and Applications*, vol. 1. IEEE, 2010, pp. 486–490.

[58] T. Hoppe, S. Kiltz, and J. Dittmann, "Security threats to automotive can networks—practical examples and selected short-term countermeasures," *Reliability Engineering & System Safety*, vol. 96, no. 1, pp. 11–25, 2011.

[59] A. Hafeez, K. Topolovec, and S. Awad, "Ecu fingerprinting through parametric signal modeling and artificial neural networks for in-vehicle security against spoofing attacks," in *2019 15th International Computer Engineering Conference (ICENCO)*. IEEE, 2019, pp. 29–38.

[60] S. C. HPL, "Introduction to the controller area network (can)," *Application Report SLOA101*, pp. 1–17, 2002.

[61] "Lin," https://www.newark.com/pdfs/techarticles/introToLIN.pdf, (Accessed on 03/18/2019).

[62] "Can-bus specifications rep. robert bosch gmbh. postfach 50, d-7000. stuttgart 1print."

[63] A. Forsberg and J. Hedberg, "Comparison of flexray and can-bus for real-time communication," *IEEE transactions on industrial electronics*, vol. 58, no. 3, 2012.

[64] R. Makowitz and C. Temple, "Flexray-a communication network for automotive control systems," in *2006 IEEE International Workshop on Factory Communication Systems*. IEEE, 2006, pp. 207–212.

[65] E. Hackett, "Lin protocol and physical layer requirements," http://www.ti.com/lit/an/slla383/slla383.pdf, Feb 2018, (Accessed on 03/18/2019).

[66] D. Paret and R. Riesco, *Multiplexed Networks for Embedded Systems: CAN, LIN, FlexRay, Safe-by-Wire...* Wiley, 2007.

[67] "Introduction to the local interconnect network (lin) bus," http://www.ni.com/en-us/innovations/white-papers/09/introduction-to-the-local-interconnect-network–lin–bus.html, Feb 2019, (Accessed on 03/18/2019).

[68] R. Sharma, "In-vehicular communication networking protocol," *Indiana University, Purdue University, Indianapolis, IN*, (Accessed on 03/18/2019).

[69] K. T. Cho, "From attack to defense: Toward secure in-vehicle networks," 2018.

[70] "Flexray automotive communication bus overview - national instruments," http://www.ni.com/en-us/innovations/white-papers/06/flexray-automotive-communication-bus-overview.html, Feb 2019, (Accessed on 03/18/2019).

[71] "Introduction to flexray," https://elearning.vector.com/mod/page/view.php?id=381, Apr 2018, (Accessed on 03/18/2019).

[72] A. Fuchsberger, "Intrusion detection systems and intrusion prevention systems," *Information Security Technical Report*, vol. 10, no. 3, pp. 134–139, 2005.

[73] W. Wu, R. Li, G. Xie, J. An, Y. Bai, J. Zhou, and K. Li, "A survey of intrusion detection for in-vehicle networks," *IEEE Trans. on Intelligent Transportation Systems*, 2019.

[74] M. Han, J. Lee, A. Kang, S. Kang, J. Park, and H. Kim, "A statistical-based anomaly detection method for connected cars in internet of things environment," in *Int. Conf. on Internet of Vehicles*. Springer, 2015, pp. 89–97.

[75] E. Wang, W. Xu, S. Sastry, S. Liu, and K. Zeng, "Hardware module-based message authentication in intra-vehicle networks," in *ACM/IEEE 8th Int. Conf. on Cyber-Physical Systems (ICCPS)*. IEEE, 2017, pp. 207–216.

[76] M. Müter and N. Asaj, "Entropy-based anomaly detection for in-vehicle networks," in *IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2011, pp. 1110–1115.

[77] A. Hafeez, H. Malik, and K. Mahmood, "Performance of blind microphone recognition algorithms in the presence of anti-forensic attacks," in *Audio Engineering Society Conference: 2017 AES International Conference on Audio Forensics*. Audio Engineering Society, 2017.

[78] A. Hafeez, K. M. Malik, and H. Malik, "Exploiting frequency response for the identification of microphone using artificial neural networks," in *Audio Engineering Society Conference: 2019 AES International Conference on Audio Forensics*. Audio Engineering Society, 2019.

[79] S. Jana and S. K. Kasera, "On fast and accurate detection of unauthorized wireless access points using clock skews," *IEEE Transactions on Mobile Computing*, vol. 9, no. 3, pp. 449–462, 2010.

[80] S. Zander and S. J. Murdoch, "An improved clock-skew measurement technique for revealing hidden services." in *USENIX Security Symposium*, 2008, pp. 211–226.

[81] T. Kohno, A. Broido, and K. C. Claffy, "Remote physical device fingerprinting," *IEEE Transactions on Dependable and Secure Computing*, vol. 2, no. 2, pp. 93–108, 2005.

[82] S. Dey, N. Roy, W. Xu, R. R. Choudhury, and S. Nelakuditi, "Accelprint: Imperfections of accelerometers make smartphones trackable." in *NDSS*, 2014.

[83] G. Brown, A. Pocock, M.-J. Zhao, and M. Luján, "Conditional likelihood maximisation: a unifying framework for information theoretic feature selection," *Journal of machine learning research*, vol. 13, no. Jan, pp. 27–66, 2012.

[84] T. D. Godfrey, A. A. Eielsen, and A. J. Fleming, "Digital-to-analog converter considerations for achieving a dynamic range of 1 ppm in precision mechatronics systems," in *Control Applications (CCA), 2015 IEEE Conference on*. IEEE, 2015, pp. 786–791.

[85] S. Pavan, R. Schreier, and G. C. Temes, *Understanding delta-sigma data converters*. John Wiley & Sons, 2017.

[86] S. U. Sagong, X. Ying, R. Poovendran, and L. Bushnell, "Exploring attack surfaces of voltage-based intrusion detection systems in controller area networks," in *Proc. ESCAR Eur.*, 2018, pp. 1–13.

[87] S. M. Adnan, A. Irtaza, S. Aziz, M. O. Ullah, A. Javed, and M. T. Mahmood, "Fall detection through acoustic local ternary patterns," *Applied Acoustics*, vol. 140, pp. 296–300, 2018.

[88] D. Hirst and R. Espesser, "Automatic modelling of fundamental frequency using a quadratic spline function." 1993.

[89] A. Hazem and H. Fahmy, "Lcap-a lightweight can authentication protocol for securing in-vehicle networks," in *10th escar Embedded Security in Cars Conference, Berlin, Germany*, vol. 6, 2012.

[90] P.-S. Murvay and B. Groza, "Source identification using signal characteristics in controller area networks," *IEEE Signal Processing Letters*, vol. 21, no. 4, pp. 395–399, 2014.

[91] R. M. Gerdes, M. Mina, S. F. Russell, and T. E. Daniels, "Physical-layer identification of wired ethernet devices," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 4, pp. 1339–1353, 2012.

[92] O. Avatefipour, "Physical-fingerprinting of electronic control unit (ecu) based on machine learning algorithm for in-vehicle network communication protocol "can-bus"," 2017.

[93] R. M. Gerdes, M. Mina, S. F. Russell, and T. E. Daniels, "Physical-layer identification of wired ethernet devices," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 4, pp. 1339–1353, 2012.

[94] J. Hall, M. Barbeau, and E. Kranakis, "Radio frequency fingerprinting for intrusion detection in wireless networks," *IEEE Transactions on Defendable and Secure Computing*, vol. 12, pp. 1–35, 2005.

[95] F. T. Ulaby, E. Michielssen, and U. Ravaioli, *Fundamentals of Applied Electromagnetics 6e*. Prentice Hall, 2001.

[96] H. Malik, "Steganalysis of qim steganography using irregularity measure," in *Proceedings of the 10th ACM workshop on Multimedia and security*, 2008, pp. 149 – 158.

[97] M. Møller, "A scaled conjugate gradient algorithm for fast supervised learning," *Neural networks*, vol. 6, no. 4, pp. 525–533, 1993.

[98] K. Shin and H. Kyusuk, "Securing information exchanged between internal and external entities of connected vehicles," May 10 2018, uS Patent App. 15/805,806.

[99] J. Petit and S. Shladover, "Potential cyberattacks on automated vehicles," *IEEE Trans. on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 546–556, 2014.

[100] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham *et al.*, "Experimental security analysis of a modern automobile," in *Security and Privacy (SP), 2010 IEEE Symposium on*. IEEE, 2010, pp. 447–462.

[101] A. Hafeez, H. Arshad, A. Kamran, R. Malhi, M. A. Shah, M. Ali, and S. Malik, "Object recognition through kinect using harris transform," in *1st Mediterranean Interdisciplinary Forum on Social Sciences and Humanities, MIFS 2014, Vol. 2*, 2014, p. 413.

[102] J. G. Proakis and D. G. Manolakis, *Introduction to digital signal processing.* Prentice Hall Professional Technical Reference, 1988.

[103] S. Wang, A. Gittens, and M. W. Mahoney, "Scalable kernel k-means clustering with nyström approximation: relative-error bounds," *The Journal of Machine Learning Research*, vol. 20, no. 1, pp. 431–479, 2019.

[104] S.-B. Roh, S.-K. Oh, W. Pedrycz, K. Seo, and Z. Fu, "Design methodology for radial basis function neural networks classifier based on locally linear reconstruction and conditional fuzzy c-means clustering," *International Journal of Approximate Reasoning*, vol. 106, pp. 228–243, 2019.