MACHINE LEARNING BASED ANOMALY DETECTION TECHNIQUE FOR IN-VEHICLE NETWORKS

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES OF

ÇANKAYA UNIVERSITY

BY

ARİF AKAR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF

MASTER OF SCIENCE

IN

THE DEPARTMENT OF

ELECTRONIC AND COMMUNICATION ENGINEERING

JUNE 2017

Title of the Thesis **: Machine Learning Based Anomaly Detection Technique For In-Vehicle Networks**

Submitted by **Arif Akar**

Approval of the Graduate School of Natural and Applied Sciences, Çankaya University

Prof. Dr. Can Çoğun
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Hüseyin Selçuk Geçim
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Doç. Dr. Klaus Werner Schmidt
Supervisor

Examination Date : 30.06.2017

Examining Committee Members

Assistant Prof. Dr. Selma ÖZAYDIN

Assoc. Prof. Dr. Klaus Werner SCHMIDT

Prof. Dr. Mehmet Kemal LEBLEBİCİOĞLU

STATEMENT OF NON-PLAGIARISM

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as require by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : **Arif AKAR**

Signature :

Date : 30.06.2017

ABSTRACT

AKAR, Arif
M.Sc., Department of Electronic and Communication Engineering
Supervisor: Assoc. Prof. Dr. Klaus Werner SCHMIDT

July 2017, 86 pages

The automotive industry faces a revolution by connecting vehicles to the communication infrastructure in the scope of intelligent transportation systems (ITS). The idea of internet of things (IoT) entering the automotive domain raises much skepticism about security and privacy issues. The information received from and sent to vehicles bears considerable risks for all components in the transportation system.

Commonly, the IT industry uses firewall devices to filter communication in both receiving and transmitting directions that require heavy maintenance personnel support and instant configuration changes. Considering the mobility of vehicles and the light-weight nature of in-vehicle networks, firewalls require too many resources and miss automated decision making. Intrusion detection systems (IDS) are widely used in traditional IT networks and try to close gaps resulting from stateful firewalls.

This thesis proposes the In-Vehicle Anomaly Detection Engine (IVADE) as an anomaly based intrusion detection algorithm for in-vehicle controller area network (CAN) applications using machine learning methods. The algorithm aims at detecting malicious manipulations of vehicle mobility data (such as position, speed, direction) which are exchanged in the form of Cooperative Awareness Messages on vehicle to vehicle (V2V) and vehicle to infrastructure (V2I) networks. The functionality of IVADE is validated by simulations of a Lane Keeping Assistance system that is implemented on a CAN bus together with the electronic control units (ECUs) for signal measurement and control computations. The relevant features for applying machine learning in IVADE are derived from received CAN message fields, supported with automotive domain-specific knowledge of the dynamic system behavior and trained with Decision Trees. The obtained simulation results indicate that IVADE successfully detects anomalies in in-vehicle applications and hence supports safety-critical functions.

Key words: Intelligent Transportation Systems, Anomaly Detection, Intrusion Detection System, Decision Tree, Intelligent Decision Making, Security.

ÖZ

AKAR, Arif
Yüksek Lisans, Elektronik ve Haberleşme Mühendisliği Anabilim Dalı
Tez Yöneticisi: Doç. Dr. Klaus Werner SCHMIDT

Temmuz 2017, 86 sayfa

Otomotiv dünyası, araçları birbirlerine ve Akıllı Ulaştırma Sistemleri'nin (ITS) haberleşme altyapısına bağlayacak bir devrimle karşı karşıya kalmıştır. Otomotiv dünyasına Şeyler'in İnternetinin (IoT) girmesi, güvenlik ve gizlilik konularında soru işaretleri oluşturmuştur. Araçlara iletilen ve araçtan çevreye iletilen bilgi, Ulaştırma sistemindeki tüm bileşenler için riskler taşımaktadır. IT endüstrisi hem alım hem de iletim yönündeki haberleşmeyi filtrelemek için yoğun bakım desteği ve anlık konfigürasyon değişiklikleri gerektiren "Firewall" ekipmanları kullanır. Araçların hareketliliğini ve araç içi ağların düşük yoğunluğu düşünüldüğünde, "firewall" ekipmanları çok fazla kaynak gerektirmektedir ve otomatize edilmiş karar verme yeteneğinden yoksundur. Saldırı Tespit Sistemleri (STS), bilişim teknolojileri ağlarında yaygın olarak kullanılmakta ve "Firewall" ekipmanlarının durağan doğasında kaynaklı boşlukları kapatmaya çalışmaktadır. Bu tez, araç içi kontrol ağları (CAN) uygulamaları için Makine Öğrenmesi metotlarını kullanan anomali tabanlı araç içi saldırı tespit motorunu (IVADE) önermektedir. Araçtan araca ağlarda (V2V) ve araçtan altyapıya ağlarda (V2I) *Kooperatif Farkındalık Mesajı* (CAM) içeriği olarak paylaşılan ve aracın konum, hız ve yön bilgisini içeren Hareket Verisine yönelik veri bozma saldırılarını tespit etmeyi amaçlamaktadır. Algoritmanın işlevselliği, Şerit Takip Asistanı (LKA) sistemine ait modelin sinyal ölçümleri ve kontrol işlemleri için Elektronik Kontrol Birimleri (ECU) ile bir CAN haberleşme hattı üzerine simülasyonu uygulanarak doğrulanmıştır. IVADE'de uygulanan makine öğrenmesi özellikleri, araç içi ağdaki CAN ağı üzerindeki mesajların veri alanlarından toplanmış, otomotiv sistemlerine özgü dinamik sistem davranışı bilgileriyle desteklenmiş ve Karar ağaçları ile öğrenilmiştir. Simülasyon sonuçları, önerilen algoritmanın araç içi uygulamalar için anomali tespitini başarıyla yaptığı ve emniyet-kritik fonksiyonları koruduğunu göstermiştir.

Anahtar Kelimeler: Akıllı Ulaştırma Sistemleri, Anomali Tespiti, Saldırı Tespit Sistemleri, Karar Ağaçları, Akıllı Karar Verme, Güvenlik

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CAN	Controller Area Network
CPS	Cyber-Physical System
DENM	Decentralized Environmental Notification Messages
DBIR	Data Breach Investigations Report
DSRC	Dedicated Short Range Communication
ECU	Electronic Control Unit
HIDS	Host-Based Intrusion Detection system
ICS	Industrial Control System
IDS	Intrusion Detection system
IOT	Internet of Things
ITS	Intelligent Transportation System
LKA	Lane Keeping Assist
M2M	Machine-to-Machine
NIDS	Network-Based Intrusion Detection system
OBD	On-Board Diagnostics
OBU	On-Board Unit
RSU	Road Side Unit
SRAM	Static Random Access Memory
TCAM	Ternary Content Addressable Memory
V2I	Vehicle-to-Infrastructure
V2V	Vehicle-to-Vehicle
V2X	Vehicle-to-X

1. INTRODUCTION

1.1 MOTIVATION

The enormous amount of information exchanged in the business sectors and the emerging use of information technologies at almost every layer of the IT industry poses a significant risk on the maintenance of reliability of the systems and perseverance of the internal databases [50]. Traditional sectors such as banking, telecommunications and IT in particular have a long history of battles for security and privacy, especially after the rapid increase of Internet services being used. Consequently, many methods and tools have emerged as a result of the increasing needs for security and privacy.

Anti-virus software and firewalls are seen as the main defense mechanisms, yet there is still a necessity for complementary approaches. Firewalls are powerful devices which are used to restrict unauthorized access to an internal network or host machines and to prevent undesirable outbound access. A firewall is integrated to networks in order to prevent suspicious traffic based on the selection of firewall rules. However, it also has to allow traffic into the network to support continuous use of internet services. Firewalls are pre-programmed and their rules are generally based on packet header information, typically *Port* numbers and *IP addresses* that are used by application layer protocols [1].

Pre-programmed and stateful defense mechanisms have revealed a necessity for detection of possible intrusions in an investigative manner starting from the end of the 90s. The fast emergence of the Internet lacks the necessary standardization for a more systematic approach to security. With this motivation and increase of hacking activities and network worms, Intrusion Detection was initiated as a complementary solution together with conventional security mechanisms [2].

Intelligent Transportation Systems (ITS) are developed with the objective of improving the traffic efficiency, comfort and safety by using information technologies, communication and control [57,58]. Similar to the Internet history, security and privacy concerns for the ITS infrastructure are growing due to the increasing connectivity between different entities and the significant rise in the amount of data generated through the whole ITS network. There are many encryption and trust mechanisms with certificate issuance proposed for ITS system protection, especially for the perseverance of system security and privacy of users [6]. At the vehicle level, research works for authentication [3] and fingerprinting [4] of ECUs are examples of such efforts.

Different from these works, this thesis focuses on detecting malicious manipulation of vehicle mobility data which are transmitted periodically to the ITS network via vehicle to vehicle (V2V) or vehicle to infrastructure (V2I) communication in the form of Cooperative Awareness Messages (CAM). The main motivation for our study is that most of the ITS applications depend on the correctness on the mobility data of vehicles through CAM transmission [5] and a plausibility check for V2V or V2I message content is seen as “the last line of defense” [6]. In particular, we consider that malicious manipulations of mobility data might endanger the safety of human and risk the resilience of the ITS system.

The automotive world has been introduced to security research works after 2010. In an experimental study [7], it is demonstrated that it is possible to inject messages into the controller area network (CAN) bus of a modern automobile so as to alter the physical state of the car. The idea was staggering in the sense that safety-critical functions of a vehicle have become exposed to external attacks. This experiment required a physical access to inner parts of the vehicle and the idea was re-used to extend the attack surface of the vehicle by realizing remote control [8]. In this work, it was possible to execute own code with a remote access. Both works clearly show that ECU firmware is vulnerable to physical and remote access and can be altered when there is not enough protection mechanism implemented. The hacking spree continued with a 2013 research from Miller and Valasek [9] in which they were able to effectively use onboard diagnostic surfaces (OBD) and could manipulate ECU firmware for two different car models. Lastly, a remote compromise of an SUV vehicle was demonstrated in a 2015 study [10]. The physical state of the vehicle was changed remotely from anywhere in USA. In the end, the car brand was forced to recall nearly 1.4 million vehicles for software update and security patching.

Automotive networks are not the sole control networks that are vulnerable to such cyber-attacks. Industrial Control Systems (ICS), including electric grid networks, Supervisory Control and Data Acquisition (SCADA) networks and nuclear stations control networks are not immune to possible cyber threats. STUXNET was the first malware that is ICS domain specific and was able to make alterations in programmable logics, download proprietary information and evade state-of-the-art security technologies [11]. Hacking incidents and emerging complex worms have changed the conventional understanding of confidentiality, integrity and authenticity as security essentials. In the December of 2016, the electric grid system was hit with a cyber bomb in the city of Kiev resulting in a blackout. The worm is later identified as CrashOverride which is essentially an insider attack that can run its own code and can map out hardware units installed in the ICS network. The incident report released in June 2017 states that the highest capability to detect this attack and similar threats would be the behavioral analytics to identify the communications on the network [47].

The Verizon DBIR 2016 report [12] states that 99% of malware hashes are seen for only 58 seconds or less and most of them were seen only once. The report implies that hackers quickly adapt their attacks and create variants in short terms making it difficult to form unique signatures for the detection of attacks. Known signatures in the databases will serve for only one percent of all incidents in the future according to the observation. In summary, the scene for future security problems makes it necessary to focus on behavioral protection for 0-day attacks instead of relying on signature-based approaches. In addition to the evolution of attack trends, the abundance of data generated in the world and dramatic developments in the computing power of hardware technology necessitate an increase in research of anomaly-based Intrusion Detection System (IDS) approaches and quick penetration into industry is foreseen as well. This thesis contributes to this effort in the scope of ITS.

1.2. CONTRIBUTION

In order to use ITS applications cooperatively, vehicles are expected to exchange mobility data among each other and with road side units (RSU) in the form of CAM messages. CAM messages are predicted to be vital heart beats of ITS systems [15]. This thesis work proposes a method for detecting possible manipulations on mobility data that are generated by compromised ECUs. The contributions are summarized in the following:

- The In-Vehicle Anomaly Detection Engine (IVADE) is proposed as an original machine learning-based anomaly detection technique. IVADE trains a decision tree based on the nominal behavior and generated anomalies of a vehicle application. IVADE is then able to detect manipulations of mobility data transmitted in CAM messages

- Anomalies for IVADE are generated based on physical laws as a truth mechanism. In the thesis, this concept is applied using the dynamic model of a Lane Keeping Assistance System. Labeling of in-vehicle data for the use of supervised learning technique for both normal and anomalous instances of CAN messages is realized.

- A Matlab/Simulink implementation including the in-vehicle CAN bus of the proposed anomaly detection technique is performed. To this end, a decision tree is trained with hours of driving data. The evaluation and analysis of the implementation is given in the form of anomaly detection performance criteria. The obtained results show a high level of success in detecting anomalies.

1.3 OUTLINE

The flow of the thesis work is given as follows. The first chapter emphasizes the context of security needs for future transportation system, the motivation behind the thesis work and summarizes the main contributions. Chapter 2 provides a detailed background on related topics including ITS, V2X communication, In-Vehicle Networks and their relations with security concepts. Anomaly Detection (AD) as a form of Intrusion Detection is explained since it is expected that AD-based techniques will be a necessity to meet future security objectives. Chapter 3 introduces the proposed In-Vehicle Anomaly Detection Engine (IVADE) by describing techniques, preprocessing of the data set, feature extraction and rules for generating anomalies synthetically. Chapter 4 gives implementation details of the IVADE algorithm for a Lane Keeping Assistance (LKA) system as an application example. The dynamic model of the LKA system is realized in Simulink and definitions of the related design blocks are explained. A decision tree is trained using in-vehicle CAN bus data of LKA model. Chapter 5 presents results of tests performed on the trained decision tree with different driving profiles. The results are evaluated against performance criteria with an emphasis on false positive and false negatives of the test profile. Finally, conclusions of the thesis work are given in Chapter 6.

2. BACKGROUND

2.1 INTELLIGENT TRANSPORTATION SYSTEM (ITS)

2.1.1 DESCRIPTION

Intelligent Transportation Systems (ITS) are an ecosystem for future transportation needs of the society in which communication, information and control technologies are cooperatively used for automated and connected services [13]. ITS can be thought of in the context of *smart cities*, where several cyber physical systems are built to perform intended operations in a dynamic and interactive way. Exchange and evaluation of information is considered as the basis for a successful implementation of any ITS application.

The technical aspects of ITS have observed a speed-up due to two main causes. First is the centralization of population in cities and its effects on emerging patterns of transportation between and within cities. According to World Bank statistics, the population of rural areas has decreased from 65% to 45% in the last 50 years [14]. As a consequence, urban centers have gained complexity in city planning in terms of transportation, which required a comprehensive and complementary approach for transportation solutions. The second factor is the recent technological revolution for the last 20 years mainly in Internet and in mobile technologies. Increasing computing capabilities of hardware and scalability of the emerging technologies have lead to data generation in every layer of the technology. Data is considered as a valuable source of information for intelligent decision making in automated environments [51].

The complexity of city planning requires several complications to be solved. Modern society needs environment-friendly, energy and time efficient solutions for transportation. In addition to these, safety and security should be taken as two goals that deserve to be highlighted. ITS are considered as a framework which has the potential to decrease the fatality rate in accidents significantly and to maintain an efficient transportation system with a reduced carbon footprint [15]. All applications and services of ITS have a strong potential for solving the mentioned issues with the help of emerging technologies in all domains.

Fatalities from vehicle accidents have been the primary driving force for the development of new transportation technologies. A report by the National Highway Traffic Safety Administration (NHTSA) [16] states that an average of 30.000 fatal incidents and an average of 2.500.000 individual injuries were caused by motor vehicle traffic crashes in the five-year period between 2009 and 2014. Numbers were in a decreasing trend from the previous years between 2003 and 2008 thanks to an increase in traffic safety legislation, education efforts for drivers and automotive safety systems deployment [5].

ITS can only be built with the contribution of several parties including governments, highway and transportation agencies, regulatory institutions and companies including automotive manufacturers and equipment suppliers [15]. Both in Europe and in the United States, there are several initiatives that have been defining and describing the ITS ecosystem, services and regulations. Consortium projects including the IntelliDrive Project from USA and the Car2Car Communications Consortium (C2C-CC) from Europe are supported by governments. Additionally, IEEE, ISO, European Committee for Standardization (CEN), Society of Automotive Engineers (SAE) are among Standards Development Organizations that invested and supported the standardization process of ITS [5]. However, it is critical to reach a worldwide regularized standardization since ITS have a large-scale and involve challenging technical issues such as communication technologies and computer vision technologies for vehicles. This requires a collaboration of all stakeholders to build, operate and maintain the ecosystem. The road map of ITS and road safety from Europe is given in Figure 1 [15].

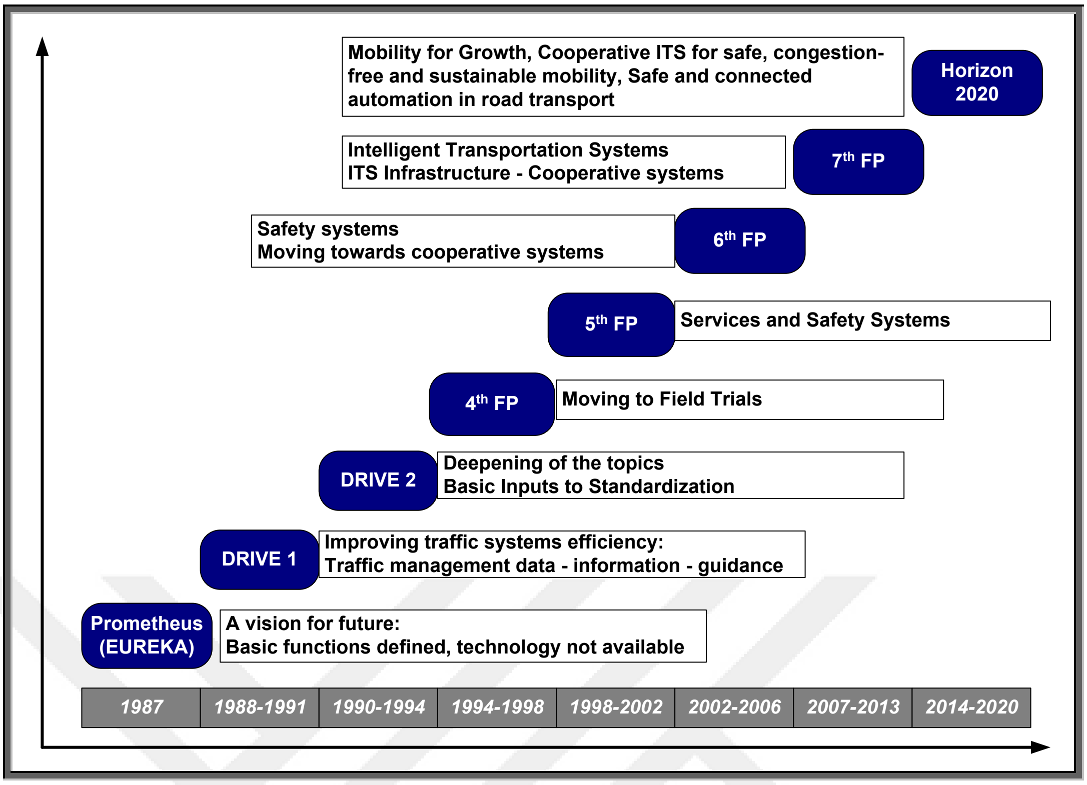
Mobility for Growth, Cooperative ITS for safe, congestion-free and sustainable mobility, Safe and connected automation in road transport
Horizon 2020
Intelligent Transportation Systems
ITS Infrastructure - Cooperative systems
7th FP
Safety systems
Moving towards cooperative systems
6th FP
5th FP
Services and Safety Systems
4th FP
Moving to Field Trials
DRIVE 2
Deepening of the topics
Basic Inputs to Standardization
DRIVE 1
Improving traffic systems efficiency:
Traffic management data - information - guidance
Prometheus (EUREKA)
A vision for future:
Basic functions defined, technology not available
1987
1988-1991
1990-1994
1994-1998
1998-2002
2002-2006
2007-2013
2014-2020

Figure 1. Europe ITS Roadmap.

The scope of this thesis in related to two vehicular networks of ITS in particular. Firstly, Vehicle-to-X (V2X) is known as the inter-vehicle network between vehicles (V2V) and between vehicles and infrastructure (V2I). There will be road side units, traffic lights and other smart units in the infrastructure that are deployed to communicate with vehicles in the system. They also act as a mediator between the decentralized part (V2V communication) and centralized (central management entities) part of ITS. Secondly, in-vehicle networks of vehicles that are typically control networks to perform mobility functions are in the scope of this thesis work. ITS is a data driven system and data is mainly generated in V2X and in-vehicle networks.

2.1.2 ARCHITECTURE

Efforts from a number of Standard Development Organizations resulted in a standardized ITS architecture. The concept of ITS Station is described in ISO 21217 based on the starting standard Communication Access for Land Mobiles (CALM) [49]. ITS Station (ITS-S) defines a set of functionalities in a bounded, secured and managed domain which is a communication basis for inhabitant applications [15].

ETSI EN 302 665 adopted the concept of ITS-S and defined four main types: Vehicle ITS-S, Roadside ITS-S, Central ITS-S and Personal ITS-S. Vehicle ITS-S are typically onboard embedded devices which are responsible for the communication establishment and functionalities required on vehicle. On the other hand, Roadside ITS-S are installed at the infrastructure side, i.e. roadside units, traffic lights and gateways. Central ITS-S are installed in the centralized part of the network to handle the data flow from and to the transportation channels. Personal ITS-S can be considered as user owned mobile devices such as tablet PCs and smart phones. ITS integrates all types of ITS stations in a connected framework.

A reference architecture of ITS Station is provided in both [17] and [49] and shown in Figure 2.

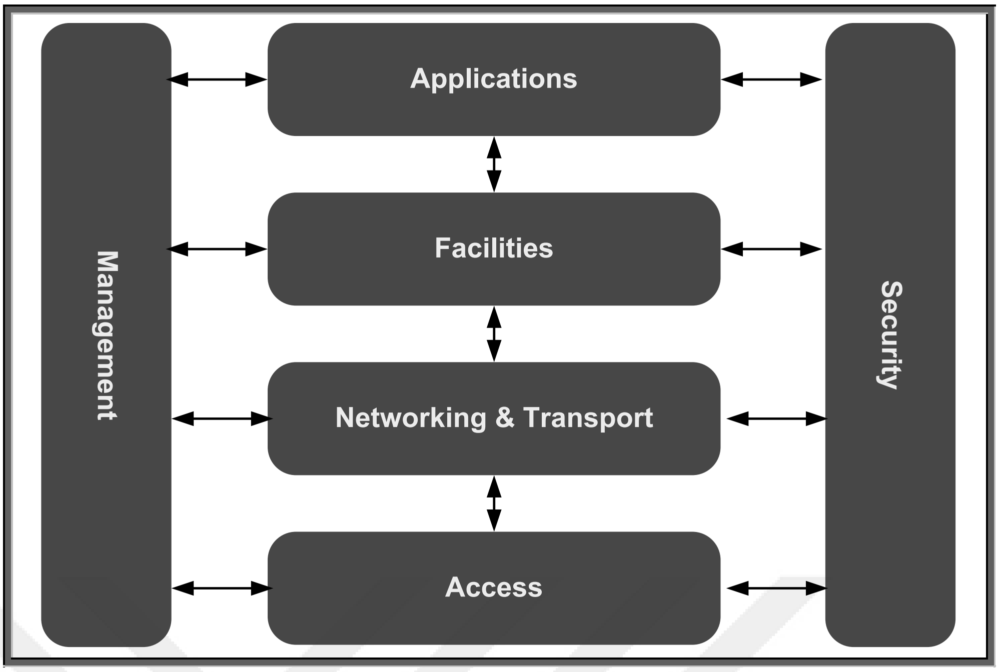

Figure 2. Architecture of ITS-S.

2.1.3 SERVICES AND APPLICATIONS

Applications to improve safety and efficiency have been primary goals for ITS development. Vehicles are increasingly getting smarter by the penetration of more ECUs and latest sensor technologies. The awareness of vehicles will increase with the help of emerging sensor and camera technologies penetrating into vehicles together with wireless and cellular communication connectivity.

ITS applications can be classified into four classes based on the primary goals of ITS infrastructure. These are hard safety applications, soft safety applications, mobility/efficiency applications and convenience applications [5]. This thesis work aims to contribute to security aspects of ITS. Security risks of ITS might endanger the safety of ITS applications. Therefore, hard-safety applications are at focus of interest considering the relation between security and safety.

Hard-safety applications

These are the most time-critical applications that require guaranteed minimal latency since there may be hazardous situations for a vehicle’s state. Latencies must be typically less than 100ms for this class [18]. Hard-safety applications explained in the study [5] are given below.

a. Emergency Electronic Brake Lights (EEBL)**:** This application sends “Hard Brake” message to surrounding vehicles.

b. Forward Collision Warning (FCW)**:** This application warns the driver of a collision risk with a vehicle in the forward direction.

c. Lane Change Warning (LCW) and Blind Spot Warning (BSW)**:** This application warns the driver about possible vehicles in the blind spot in which a lane change maneuver could be dangerous.

d. Do Not Pass Warning (DNPW)**:** This application warns the driver when a passing maneuver cannot be completed safely.

e. Intersection Movement Assist (IMA)**:** This application warns the driver when approaching to an intersection about a risk of collision with other vehicles.

f. Control Loss Warning (CLW)**:** This application sends a broadcast message to warn surrounding vehicles of a loss of control in vehicle maneuvering.

Soft-safety applications

Icy Bridge Warning, Disabled Vehicle and Construction Zone Warning applications can be counted as examples of this class of applications. This type of applications are not as demanding as hard safety ones in regard of latency and mostly for warning the driver of road and weather conditions.

Mobility Applications and Convenience Applications

While mobility applications mainly put emphasis on efficiency of time and energy consumed in transportation system, convenience applications focus on entertainment and personal activities to increase the passengers' life quality during transportation. Convenience applications are less dependent on time requirements, yet bandwidth requirements might be tight to enable audio and video streaming. These applications also involve synchronization to consumer electronics to personalize experience and allow for smart phone capabilities.

2.1.4 ITS COMMUNICATION NETWORKS

ITS stations mentioned in the previous section are communication end units in the network. Personal ITS-S (smart devices), Central ITS-S (server-side central units), Roadside ITS-S which is often denoted as Roadside Units (RSU) and Vehicle ITS-S which is often called as Onboard Unit (OBU) constitute ITS communication networks. The ITS network including all end units is shown in Figure 3.

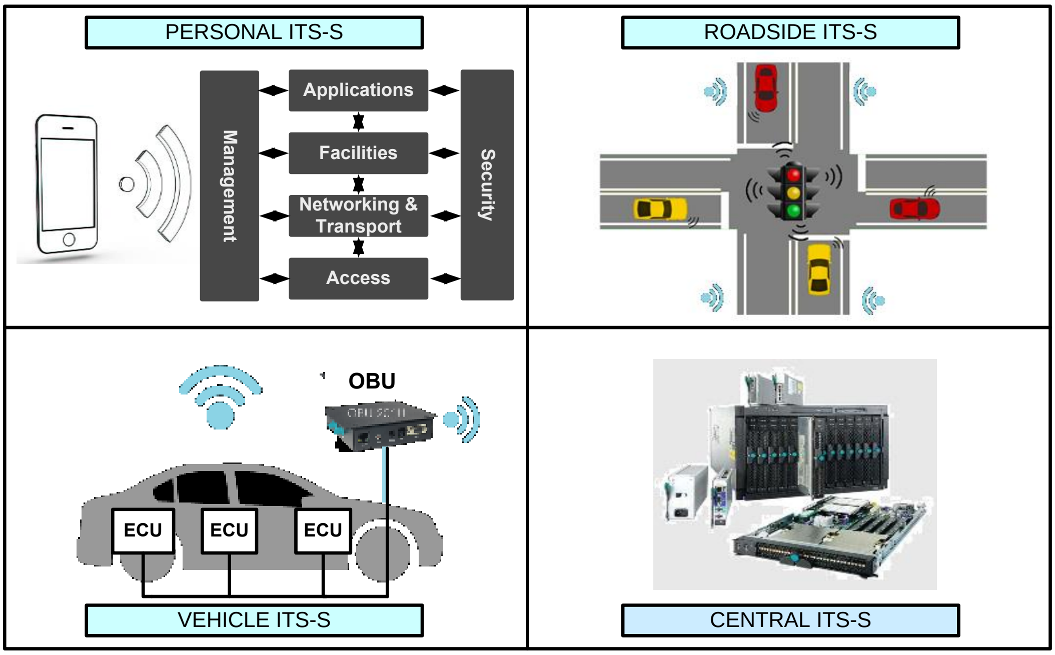

Figure 3. ITS Networks.

V2X Networks

V2X denotes a communication in vehicular networks as X refers to a single end unit according to the type of communication. X refers to an OBU when it is between two vehicles (V2V) and refers to an RSU when communication is between vehicle and infrastructure (V2I). The communication technology is usually a mix of protocols and technologies. It might be Dedicated Short-Range communications (DSRC), cellular networks or Wireless Access in Vehicular Environments (WAVE) which is built on IEEE 802.11p [15].

There are two safety messages standardized by ETSI which can be referred to as Decentralized Environmental Notification Messages (DENM) and Cooperative Awareness Messages (CAM) in V2X networks. CAMs are sent from vehicles to the surroundings in the region of the awareness range. They are transmitted periodically with high frequency in order to update neighborhood tables. Neighborhood tables are considered to be an important element for safety of future automotive applications. DENMs are event-triggered messages delivered to vehicles about a triggering event. Road safety applications, especially hard-safety applications rely on these two messages to perform their operations [17].

Onboard Unit

It is noteworthy to mention the connection interfaces of an onboard unit (host type ITS-S) that is used in the Vehicle Safety Communications-Applications Project is shown in the Figure 4 [5]. The VSC-A project is the product of a collaborative work between the US Department of Transportation and Vehicle Safety Communications 2 (VSC2) Consortium which includes automotive manufacturers Ford, GM, Honda, Mercedes-Benz and Toyota. The basic architecture is shown in Figure 4. As can be seen from the figure, there is a dedicated onboard unit that has connection surfaces with both internal CAN network and the outside world. Almost all V2X projects involve similar onboard units to establish communication with surroundings.

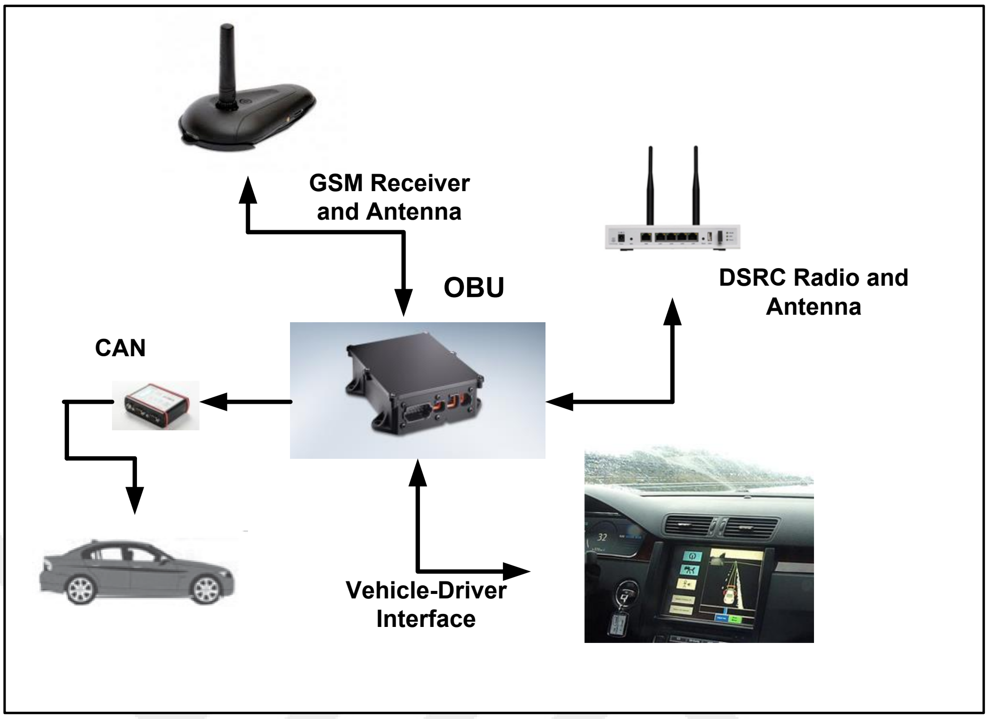

Figure 4. VSC-A Hardware Components [5].

In-Vehicle Networks

In-vehicle networks are onboard communication networks that interconnect subsystems of a vehicle to perform segregated or distributed functions. In-vehicle networks comply with the typical controller network model by having three different nodes: a sensor, an actuator and a controller. An ECU is an *electronic control unit* that usually controls actuators by observing inputs from sensors. In-vehicle networks have sub-networks in which ECUs are connected for a functional purpose. These functions can be stated as powertrain control, chassis control, infotainment control and body control [55].

In-vehicle networks are traditionally simpler than IT networks, yet it is crucial to ensure message transmission by eliminating conflicts and to be resilient to noise. As the number of electronic components penetrating into vehicle subsystems is increasing, the in-vehicle networks are given significance as a medium for exchange of information. The distributed functions among different subsystems triggered the complexity level of networks and transform in-vehicle networks into advanced controller area networks. Upcoming autonomous driving abilities and ITS applications that require collaboration of several subsystems force automotive manufacturers and equipment suppliers to add new hardware and software layers to their products, making in-vehicle networks a source of data. By increasing connectivity surfaces of vehicles to support ITS, vehicles are becoming vulnerable against malice and misbehaviors [8].

Controller Area Network (Can)

Controller Area Network (CAN) is the dominant in-vehicle network in the automotive industry because of its integrity, simplicity and resilience against noise [52]. CAN is message based and works on carrier sensing multiple access with collision detection (CSMA/CD) protocol with prioritization by arbitration. The priority of the message is designated by its header that has normal and extended forms as 11 bits (CAN2.0A) and 29 bits (CAN2.0B) length, respectively. Prioritization of messages makes CAN an attractive choice for real-time environment [53].

The usage of V2X is transforming in-vehicle networks into large-scale networks with complex functions and increasing connection surfaces. Vehicles can be regarded as safety critical systems for drivers, passengers and pedestrians in the sense that uncontrolled behaviors or actions might result in highly undesirable outcomes. Therefore, mobility-related functions, their corresponding ECUs and the control network in which these safety-critical functions are deployed are vital for the perseverance of ITS. CAN bus is historically the main choice for implementing mobility-related safety-critical functions.

2.2 SECURITY

2.2.1 OBJECTIVES

For internet and communication technologies, there are four conventional security principles for information security. Preservation of these objectives is intended to ensure secure and private communication in the ITS infrastructure. These security principles are defined in the following with the perspective of vehicle networks [6].

***Confidentiality*:**

Confidentiality requires the content of a message not to be disclosed to any parties other than the intended ones. Encryption mechanisms are proposed for ITS networks to satisfy this objective.

***Authenticity*:**

One of the primary objectives in vehicle communication is to ensure only trustworthy entities are included in networks. Authenticity is being able to verify the sender of any message in the system.

***Integrity*:**

To maintain the accuracy and completeness of message content is the objective of Integrity principle. Public Key Infrastructure (PKI) solutions proposed for ITS communication ensure integrity and freshness of messages.

***Availability*:**

Availability urges the nodes of networks and information or a service for necessary receivers shall be available whenever a request is ready. To preserve the functional state of ECUs in a vehicle network is to be ensured to complement the security requirements of a complex system.

2.2.2 THREAT MODEL

Threat model in a security mechanism refers to the main characteristics of possible adversaries, their capability level and an estimation of their tools [30]. A threat is an unauthorized attempt to access valuable assets, to manipulate information and to alter a system’s state into instability or unreliability [7]. Threats violate essential measures of security in a way to leverage their purpose.

It is important to describe a threat model in order to implement a layered and holistic security mechanism. Once a threat model is defined, attacker types and attack vectors can be derived according to the capabilities of adversaries, their possible intentions and the valuable assets in a host PC or network. The attack vector can be implemented as test procedures later to verify security implementation. The threat models are valuable simulations of possible attacks, yet, it is nearly impossible to cover the complete space of attack types in real life.

2.2.3 ITS SECURITY

Traditional IT networks have a long history of security approaches which resulted in many solutions and measures aiming to maintain holistic security coverage. However, the characteristics of in-vehicle networks are different than traditional IT networks in many aspects. We assess these differences from the perspective of network behaviors, in order to explore distinguishing properties that can be captured by our anomaly detection algorithm. The study in [21 investigates differences between IT and CPS networks intrusion detection systems with a four step methodology. We restructure these steps to explore similarities and differences between traditional IT networks and in-vehicle networks as follows:

Physical Process Monitoring: IT networks can be monitored by measuring user and machine/server activities. In-vehicle networks can be monitored by measuring physical properties. This provides a ground for laws of physics that can explain the behavior of vehicle motion.

Closed Control Loops: IT networks are triggered by user inputs and the nature of the traffic being multivariate increases the unpredictability of behavior. On the other hand, although there is also user input triggering for in-vehicle networks, in-vehicle network messages are usually periodical or event-based. In-vehicle networks reflecting a time driven and semi-automated behavior provides predictability for network profiling.

Attack Sophistication: The infiltration into a complex social transportation system might be a valuable target for adversaries to gain leverage. Therefore, for ITS infiltration, attack sophistication is at least as high as for penetration into a closed intra network of a bank or state.

Legacy Technology: Hardware units of IT networks are usually physically inaccessible while ECUs of an in-vehicle network can be easily accessed and tampering of in-vehicle hardware might present trivial ways of compromise to adversaries.

The nature of in-vehicle network data according to the four mentioned criteria presents the basis for developing a behavior-based intrusion detection technique for in-vehicle networks.

2.3 INTRUSION DETECTION SYSTEMS

The main security mechanisms include firewall solutions, anti-virus products and even training of users in a system to raise awareness of threats. On the other hand, the approach of this thesis work is more comprehensive than proposing mainstream security measures. In other words, the proposed work in this thesis aims to investigate whether the implementation of an anomaly-based intrusion detection system (IDS) is feasible for in-vehicle networks. Therefore, a section of IDS is included with the intention of presenting basic IDS principles, common techniques and classification taxonomies in the literature.

2.3.1 DEFINITION

“IDS” is a system that detects possible intrusions or the effect of the intrusions in a system. A research in 1987 [22], was one of the earliest works that proposed a model for detecting intrusions. IDS are deployed to detect and identify malicious activities and prevent them to access to valuable assets such as information, control and process abilities of a network or a machine. An asset is a resource that might have a value or power in its context. All the approaches being software-based or hardware-based that are dedicated to detect or prevent a malicious activity can be included in the IDS concept.

Intrusion means an ill affected and anti-policy access to a valuable asset or an attempt to break the control hierarchy of a system. Every machine or network that has a valuable asset or a control power might be a gainful target for intrusions [23]. Especially, systems that have safety critical functions are sensitive targets and may not tolerate vulnerability in defense mechanisms. The higher the value of assets and control power of a system, the higher the possibility of being a target is an inherently expected principle. Therefore, it is reasonable that either the value of the asset must be decreased or more precautionary measures must be deployed to increase the safety and security of such systems.

Many IDS tools have been developed since the end of 90s. Bro, Snort, EMERALD and NETSTAT were early examples of software-based IDS [23]. Snort is a free open-source network intrusion detection software that has dominated the software-based IDS market. As the throughput requirement increased, IDS algorithms were also implemented in hardware [24].

IDS itself might also be a target of attack due to its critical functions for defense mechanism. It shall be developed in a way to survive attacks and to carry out its functions since it is one of the last defending mechanisms. In addition to traditional IDS, newly emerging machine learning based IDS also pose vulnerabilities for adversarial training. An attacker might explore ways to train the IDS with adversarial samples and try to leverage its false learning for the purpose of exploitation. For the scope of this thesis work, a threat to IDS is not considered and it is assumed to be in safe operating conditions.

2.3.2 CLASSIFICATION OF IDS

Intrusion Detection Systems have been classified in the literature using different taxonomies. Acquisition of data, detection techniques and time of detection are among the main dimensions for the classification of IDS [21]. Various selections of these three criteria also imply the purpose of the IDS, its scope of the protection and the effective time behavior of the used system.

Using the data acquisition dimension, IDS are classified into two categories; namely, host based IDS (HIDS) and network based IDS (NIDS) [25]. HIDS is designed to protect a single machine entity and operates by using the resources of the hardware and software properties of its host computer system. The data is collected among the operating system and all the necessary data analysis work is performed at the host level. On the other hand, NIDS requires a collection of network data from communication media and usually works through network sensor agents placed across the network topological structure. It monitors and collects the data for analysis and creates alarms for handling the intrusion incidents [23]. The IDS proposed in this thesis work are of type NIDS due to the usage of the CAN bus in-vehicle network data.

IDS are classified into three categories in a study according to detection techniques; namely, pattern-based detection, anomaly-based detection and specification based detection [21]. Pattern-based detection is also known as misuse detection in which a known attack is detected by searching a match of a special encoding of the attack which is called “signature”. Signature is a trace of a particular attack that is usually represented as strings. They are stored in databases and pattern matching algorithms use signatures to match in the ongoing or logged network stream. Anomaly-based IDS outline a normal behavior of a network profile and try to detect intrusions by discovering anomalies in the data stream. Pattern-based IDS and anomaly-based IDS systems are explained in the next sections in detail. Specification-based IDS systems can be defined under the anomaly-based IDS class since they also generate a profile for the normal state of the system based on system specifications. It creates rules depending on the specification used and labels every deviation from these rules and normal operation states as an anomaly.

Pattern-based IDS

The idea of pattern-based IDS lies in exploitation of knowledge for known attacks in the form of signatures. Signatures are pre-configured encoded patterns of known attacks and are constantly registered to the IDS database. Updating of this database is crucial to mitigate vulnerabilities until necessary patch is applied to the aforementioned software or the system [24].

The implementation of pattern-based IDS algorithms were done as both software-based and hardware-based [24]. The initial proposals for IDS were software-based implementations and evolved into very advanced tools that have been dominating the IDS market. Since the first examples of IDS solutions, pattern-based IDS systems are currently deployed in traditional IT systems extensively. They need to act within the speed of network connection in order to provide a timely and exhaustive monitoring. Therefore, a need for fast processing is required for keeping up with the incoming traffic stream, resulting in looking for hardware-based solutions [24]. For the pattern-matching and classification tasks, hardware implementation is performed to increase the speed of processing by dedicated hardware units.

Advantages and Disadvantages of Pattern Based IDS

For most networks, pattern-based intrusion detection systems are deployed since they generate low false positive rates against known attacks [23]. Signatures of attacks are constantly added to databases which in the end provide a comprehensive solution for known attacks. Another advantage of this type of IDS is that it is immediately effective after installation. Different from anomaly-based IDS, it doesn’t need a learning or configuration period before effectively performing detection process. On the other hand, pattern-based IDS are inevitably helpless against unknown and new (zero-day or 0-day) attacks since there hasn’t been a signature pattern formed yet for the particular attack type and registered to the signature database [21]. The effective time of reaction to intrusion is a rather significant criterion for the performance of IDS systems, and hence, detecting an intrusion unfortunately becomes only possible after analyzing the incident and creating a corresponding signature for it. The time passes during such process is a critical period for most of the industries that can’t tolerate vulnerability in their internal networks [24], [25].

Anomaly-based IDS

The definition of anomaly is originated from statistics and probability. Anomaly refers to a point in a data set that does not conform to other points in forming a pattern [25]. Being different from an outlier which can be the indicator of the data model dysfunction, anomaly is more likely to be the result of an unexpected event meaning an illegitimacy of appearing in the data set. For this reason, phenomena like an unexpected credit card transaction, a sudden fault in a system, a cyber-attack and an abrupt stock market change are all considered to be represented as anomalous events [26].

In [26], types of anomalies are stated as point anomalies, contextual anomalies and collective anomalies. As an example to this classification, a message of an in-vehicle CAN network can be classified under contextual anomaly category due to the reason that it can be normal or anomalous in a specific context, i.e. according to the state of the vehicle and according to the past and current inputs. Considering the automotive domain, a CAN message from the in-vehicle network might represent an anomaly although in other cases, the state of the vehicle can justify the same CAN message as a normal packet. Therefore, it is necessary to think in contexts for a better understanding of anomalies from in-vehicle networks.

Anomaly is similar to noise, both being undesirable deviation from the main signal or data set. The underlying difference is that noise can be explained with the deviations of the system dynamics and; most of the time it is expected and precautions are taken against it [26]. Hence, even intense noise levels do not result in dramatic consequences. On the other hand, anomaly indicates a transition into an unwanted state or reveals an unprecedented input given to the system and probably results in unexpected consequences which may not be tolerated by the system.

Anomaly detection is based on the idea of creating a profile of the *normal behavior* for a system. The deviations from the normal behavior are assumed to indicate possible intrusions. The challenges of anomaly detection rise exactly from this assumption. That is, it is a very difficult and exhaustive task to define the normal behavior of a system in a comprehensive way [27]. Objects of a system, attributes defining these objects, links connecting them and the data exchanged between them all contribute to the baseline profile of the normal state of a system. In [23], an illustration of an anomaly detection model is given.

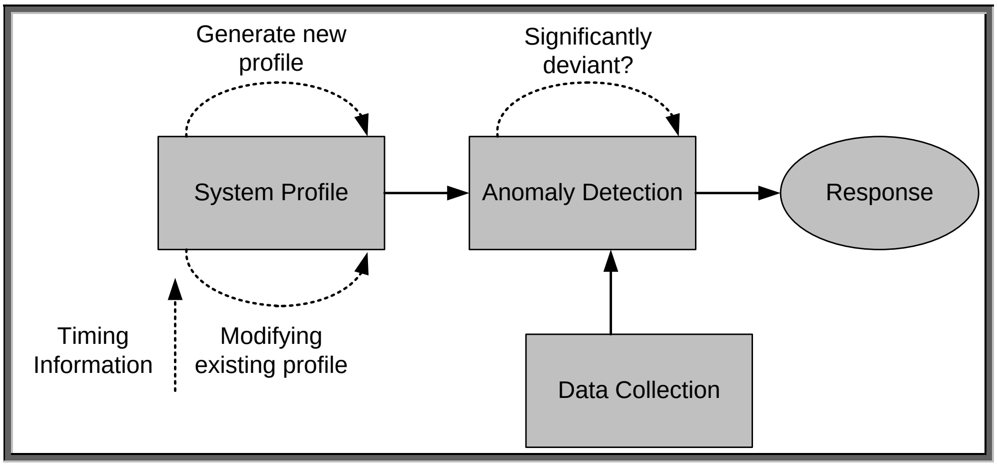

Figure 5. Anomal Detection Framework.

The assumption of building IDS by using anomaly detection techniques underlines that all intrusions create anomalies in the system behavior which are necessarily different than the normal (profiled) behavior [28]. The set of intrusions is assumed to be a subset of anomalies [25]. Therefore, by applying anomaly detection techniques, IDS is considered to be used as a security mechanism.

Classification of Anomaly Detection Methods

A good review work on the classification of anomaly detection is done by [25]. According to this study, the classification of network-based anomaly detection can be based on the used techniques and consists of the following techniques given in Figure 6.

In Chapter 2.1.3, we made a comparison of in-vehicle networks and conventional IT networks. It was noted that in-vehicle networks can be monitored by measuring physical signals or observing messages that are usually periodical or event-based. Knowledge based anomaly detection methods can be used expressing in-vehicle network behavior by defining rules of the system. Additionally, since the nature of network data is periodical or related with events, exploitation of statistical inference of in-vehicle network data is possible by using statistical anomaly detection methods.

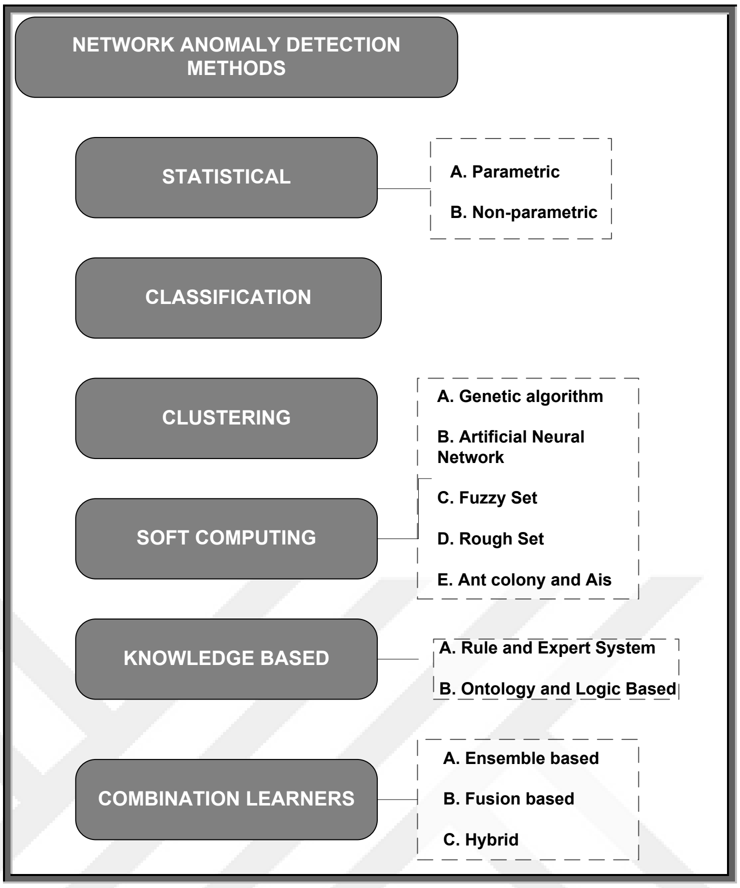

Figure 6. Classification of Anomaly Detection Methods [25].

Advantages of Anomaly-Based IDS

The principal benefit of anomaly-based IDS is that it can detect unknown (0-day) attacks, having a significant advantage over pattern-based IDS systems. 0-day attacks pose a great risk for the healthy operation of systems. For systems having safety-critical functions such as a vehicle’s driving operation and considering its relation with the future V2X infrastructure, anomaly-based techniques have a great potential for mitigating these risks. Security is not an easy task to be accomplished; it needs a holistic approach with layered defense and trust mechanisms. The most sophisticated attacks will be performed on the most critical systems [21]. For insider attacks in CPS architecture, anomaly detection-based IDS research proposes most promising ways by detecting 0-day attacks [21].

Attackers tend to explore ways to overcome security measures and try to extend their attack over time as in the case of CrashOverride that resulted in a blackout in Kiev. It is stated that attackers have been inside of the network for months [47]. Anomaly-based IDS might be effective for this type of stealthy attacks too. Additionally, since it is difficult for hackers to predict what knowledge is learnt inclusively in the model, it is more likely that they will cause alarms for their actions.

Challenges of Anomaly Based IDS

The challenges to develop AD-based IDS might be summarized as:

It is an extremely difficult task to profile the normal behavior of the system. An exhaustive process including defining rules and specifications for the system might be required.

It is also difficult to locate genuine anomalies for the system. Therefore, there might be a need to generate anomalies synthetically which also might require expensive labeling work and making undesirable assumptions.

It can be hard to generalize to scale the AD-based IDS due to proprietary knowledge of every system model. The inherited knowledge in the model might not work for other networks due to difference between attributes, communication media, object features…etc.

There might be a need of a huge amount of data to model the system in the case of classification-based anomaly detection.

AD-based IDS might require IT maintenance support to evaluate false alarms in case of high False Positive rates.

AD-based IDS might not scale up to limits of speed of the communication. Hardware implementation of algorithms might be required as in the case of pattern-based IDS.

The practical contribution of this thesis work is demonstrated with the implementation of the IVADE algorithm to detect any manipulation in CAM messages. The collaborated ITS applications such as collision avoidance, autonomous driving and platooning heavily based on CAM messages for receiving and validating the mobility information of surrounding vehicles. CAM messages contain position, speed and global data of the host vehicle and is shared between vehicles in short range. Mobility data is updated as a result of vehicle’s motion and the change in mobility data is originated from inputs given into control network of the vehicle. In a study [19], eight types of detection sensors are presented for automotive bus systems. With our implementation, (S-3) Range Sensor, (S-7) Plausibility Sensor and (S-8) Consistency Sensor check are applied as a form of anomaly detection between in-vehicle network parameters and mobility information in CAM messages.

3. IN-VEHICLE ANOMALY DETECTION ENGINE (IVADE)

3.1 DESCRIPTION AND MOTIVATION

IVADE is an anomaly-based network IDS which is proposed to detect insider attacks in the CAN bus of an automotive system. Anomaly-based algorithms essentially require definition of a “normal” profile for the system in order to detect anomalies by observing deviations from the “normal” profile. In [25], it is stated that when the degree of deviation is high enough with respect to the profile of the system then the instance of the system is classified as anomaly. The behavioral profiling of IVADE algorithm is performed using machine learning techniques. In this sense, IVADE algorithm is suitable to generalize the profiling of any vehicle subsystem’s network behavior by monitoring in-vehicle network data.

The goal of the IVADE system is to detect insider or internal attacks in the form of compromised ECU and CAN injection attacks performed onto in-vehicle communication of vehicles by monitoring CAN channels. An internal attacker definition is given in [30] as adversaries which are equipped with cryptographic keys and credentials to overcome security protocols. The threat model is assumed as a sophisticated entity with physical and remote access to the in-vehicle network which is able to run its own code to compromise ECUs of the vehicle.

Mobility functions of a vehicle are carried out by multiple ECUs on a basis of CAN message exchanges in a networked model. The engine and transmission networks are dedicated to generate outputs for creating motion according to inputs coming from steering wheel, gas and brake pedals in classical vehicles. In modern vehicles, most of these inputs are expected to be autonomous. Computer vision systems have found significant use within the recent development of automotive technologies. Cameras with high resolution capability have penetrated in the car supply chains and consequently, there has been an increase in the vision capabilities of vehicles. In addition to cameras, emerging sensor technologies such as LIDARs are being used in autonomous vehicle designs both to support current automotive technologies and upcoming technology rush for autonomous driving capabilities [31]. Additionally, vehicles are driven by auto-pilots that are capable of arranging traction force and brake force inputs automatically in semi-autonomous and fully-autonomous cases.

Whether coming from a human or from dedicated sensors, inputs given into the motion-related ECUs and the outputs of these ECUs are transmitted to and received from in-vehicle networks. Therefore, the observation of the vehicle’s state can be performed at the network level. The dynamics of the vehicle define the relationship between inputs and outputs as the driving experience continues. In other words, for the inputs and outputs of a vehicle’s subsystem; there is always a dependency dictated by the dynamic model of the vehicle [44]. Although proprietary design by the vehicle manufacturer is the decisive factor for this dependency, we aim to build a model of network in order to predict the authenticity of outputs depending on the inputs and the state of the vehicle at the time of inputs.

3.2 RELATED WORK

An IDS proposal for critical infrastructure system is presented in [29]. In this study, real network data is recorded from a Programmable Logic Controls (PLC) hardware of a critical infrastructure control system. The intrusion instances are generated artificially and randomly based on the intrusion attempts. A window-based feature extraction technique is applied since the network packet stream is described as time series data. Based on the extracted features, a combination of Error Back-Propagation and the Levenberg-Marquardt algorithm is applied with neural networks to train boundaries of the clusters of the recorded normal behavior. The results are promising that perfect detection rates and zero false positives are reached against an unknown test set.

An IDS using deep learning neural networks for in-vehicle networks is proposed in [43]. This research uses the probability of each bit in a CAN message to extract features that represent the statistical behavior of the in-vehicle network. Three techniques including the proposed technique, SVM and artificial neural network are compared, showing that the deep neural network based proposed algorithm performed the best. The provided results are promising but it is required to define attack scenarios as mode information so that weights of the neural network should be trained fitted to each scenario. Additionally, CAN bus data is simulated synthetically without any prior knowledge or precise model. Another research [44] uses in-vehicle data that is acquired from the CAN bus of a real vehicle. The CAN sub-network from which data is acquired consists of an engine control module, parking control, motor control and transmission control modules. A number of in-vehicle signals including RPM, Speed, MAP, MAF, AccPedal and Throttle sensor data are extracted as features of the algorithm for the representation of behavior. The features are trained with deep learning technique and it is stated that a model representing the normal operation of the vehicle can be learnt and any significant deviation from the model can be received as alarms for intrusions. The method of what the scale of deviation should be and how thresholds will be decided is not stated.

3.3 ALGORITHM FOUNDATION

IVADE is proposed to detect anomalies for in-vehicle networks by modeling in-vehicle network behavior. The flow of the algorithm showing phases and data transition is given in Figure 7.

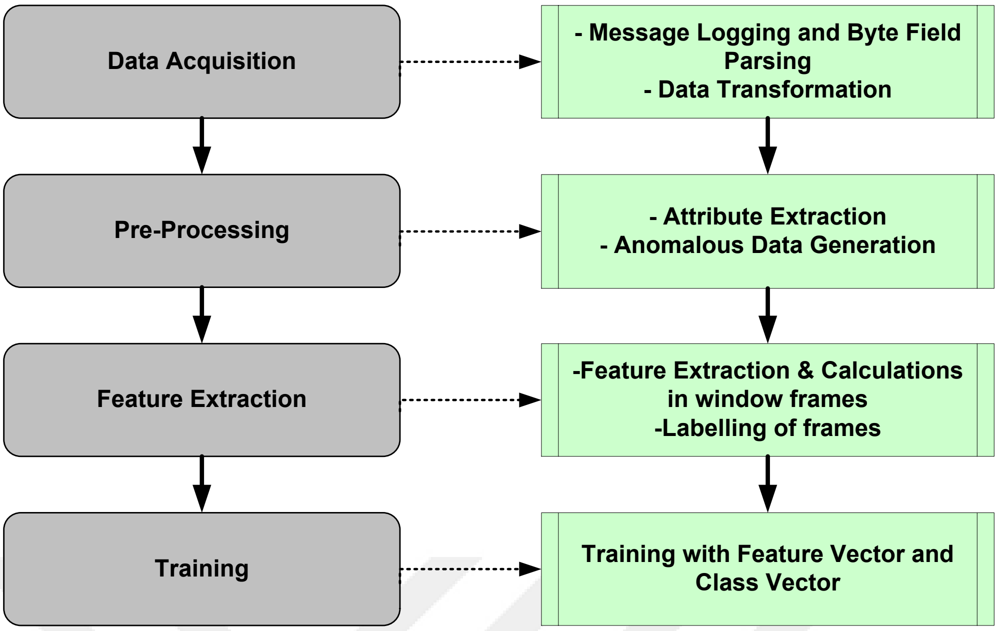
Data Acquisition
- Message Logging and Byte Field Parsing
- Data Transformation
Pre-Processing
- Attribute Extraction
- Anomalous Data Generation
Feature Extraction
-Feature Extraction & Calculations in window frames
-Labelling of frames
Training
Training with Feature Vector and Class Vector

Figure 7. Flow Of Alogirthm.

3.3.1 DATA STRUCTURES

Data to be used to train and test the proposed IVADE algorithm shall be acquired from the in-vehicle network of vehicle. CAN bus is chosen to demonstrate the algorithm since it is the most dominant communication standard for vehicles. According to the CAN specification, the payload of a CAN message can be as many as 8 bytes, but payload length is not a condition for our algorithm. The range of values that can be represented in a byte is between (0-255). Many different CAN messages each having a unique task are transmitted to the network.

In a study [34], 20 recordings of vehicle CAN data each having 100,000 messages are logged from a vehicle. From their observation, there were four different CAN field types; constant fields, multi-value fields, counter fields and sensor fields. Our algorithm uses all these types of fields as source of attribute selection.

3.3.2 ATTRIBUTE SELECTION

Idea 1: The analytics of motion essentially depends on the relation between input force, geometry of the road and the vehicle dynamics. Speed, Distance and their rate of change over time in a coordinated universe are correlated with inputs, i.e. input force and geometry of the road.

In a given time, i.e. in a window of time, expected values of mobility data might be calculated based on the current speed, distance taken and the global orientation of a vehicle. In this regard, in-vehicle data attributes extracted from CAN messages can be regarded as time-series multivariate data and might be used as attributes of a machine learning technique.

Input-output relations of ECUs can be exploited for causality of vehicle's physical state. Therefore, the input-output causality approach can be used for structuring data for the formation of an attribute vector of IVADE in order to correctly profile the in-vehicle network. For the case of in-vehicle networks, any motion-related signal can be an appropriate attribute for in-vehicle network profiling. As a matter of fact, the idea can be generalized to any vehicle subsystem by extracting ECU relations and related in-vehicle messages. In Figure 8, the mentioned approach is visualized.

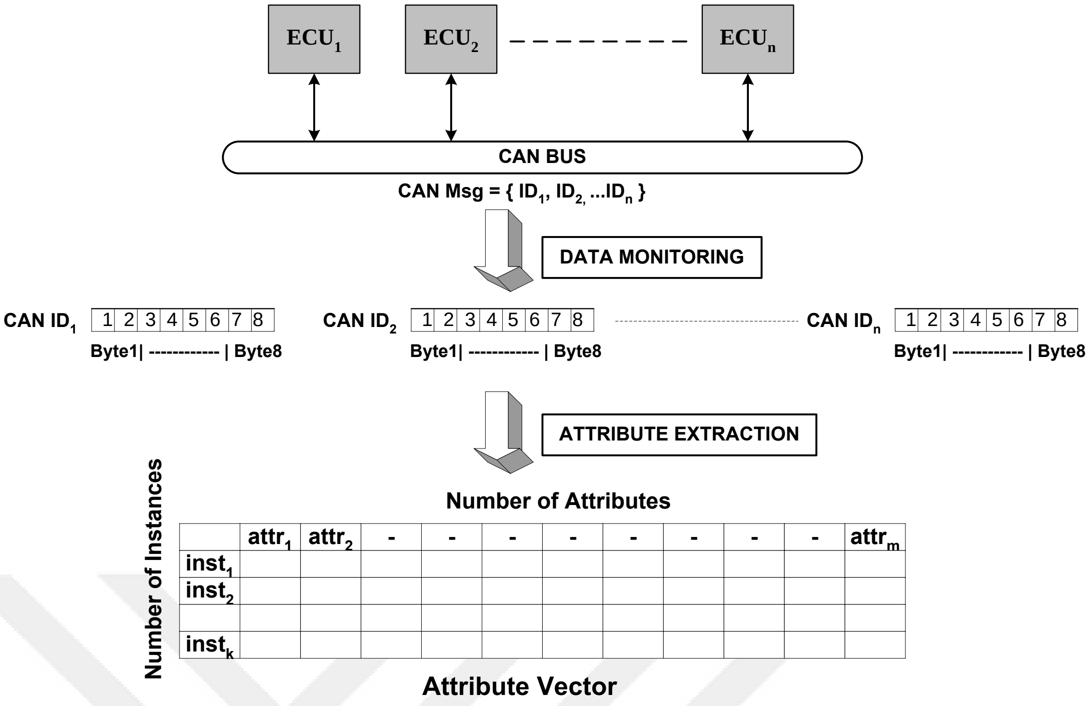

Figure 8. IVADE attribute extraction.

For any vehicle system that consists of a number of ECUs connected within a network, attributes can be extracted from all motion-related signals in order to profile the behavior of the network. While forming the attribute vector, related byte fields of all CAN messages that arrive in a time interval t_i are sampled.

Definition: The number of attributes is constant and equals the sum of all different CAN data fields from all unique CAN messages. *m*, being the number of attributes (byte fields chosen as attributes) per each CAN message and *n*, being the number of different CAN messages; the number of attributes sampled in one sampling period is:

$$num_{attributes} = \sum_{i=1}^{n} m_i$$

With a sampling period of t seconds, there will be T/t instances of every CAN message in a each time interval T; resulting in an attribute vector of dimension in T second:

$$Dim[attribute\ vector] = \frac{T}{t}\ X\ num_{attributes}$$

3.3.3 FEATURE EXTRACTION

A compromised ECU may alter the correct outputs with falsified values. By observing CAN messages independently, detecting a possible manipulation in one CAN instance is considered as a difficult task considering the missing contextual information. Detection of manipulation gains coherence when it is conceived in a time basis since the content of CAN messages shows a time-series behavior during runtime [32]. ECUs, by design, work with recently received CAN messages and perform according to their current state. Therefore, it is vital to define the normal profile of the CAN bus by considering conditional probabilities of interrelated events. In the studies [48] and [29], previous values of parameters are utilized to detect targeting interrelated events.

Idea 2: A sliding-window approach is more suitable for feature formation.

The instantaneous CAN messages deliver timely information, yet this transient form of information on the system state might be too volatile to take into consideration. Therefore, a sliding window approach to define a *feature vector* is used in the form of window-frames consisting of consecutive instances of attributes.

Idea 3: The length of a time window shall be long enough to store adequate amount of information on the system, while it is short enough to provide an ample resolution of measurement [33].

The length of the window depends on the physical properties and the objectives of the application. Experiments and observations on the dynamic model might help to determine the length of the window in the case of vehicles. Additionally, the response requirement of the application might dictate limitation on the length of the window when quick detection and response is needed.

In order to predict output signals of any ECU at a particular time, it is necessary to consider previous values of the input and output signals, i.e. derived attributes from CAN fields; by doing so it is possible to have a contextual understanding of the network behavior. *l* can be defined as the length of the sliding window frame. Let A be an instance of the attribute vector, and W be the corresponding instance of the feature vector and *l* being the length of the window:

$$W_w \stackrel{\text{def}}{=} \{A_{i-l+1}, A_{i-l+2}, A_{i-l+3}, \dots A_i\}$$

Feature extraction from the attribute vector with the sliding window approach is shown in Figure 9.

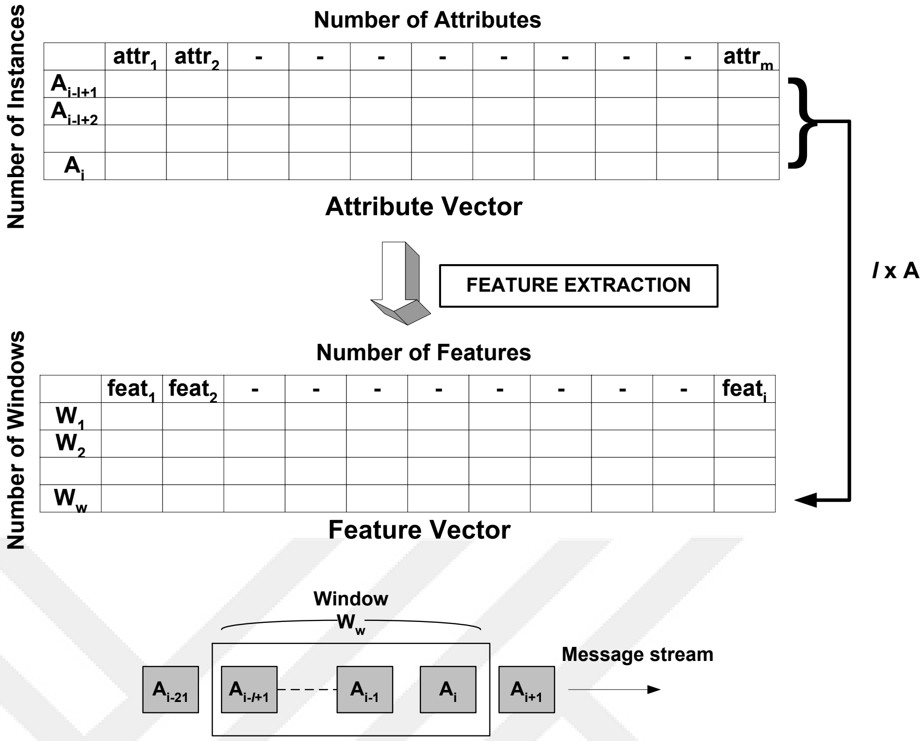

Figure 9. Feature Extraction.

A window of consecutive attribute instances is transformed into a feature vector by calculating some metrics. These metrics are generally chosen to be indicators of the deviation between an anomalous instance and a normal instance. In anomaly detection, distance metrics are a quantitative degree of how two objects are far or close from each other. The Chebyshev distance is used as a similarity measure in IVADE to indicate the deviation [25]. The Chebyshev distance represents the maximum distance between two vectors or two points along any coordinate dimension. Other metrics are mean values, expected values and tangent of consecutive attribute instances in a window.

$$\max_{i} \mid x_i - y_i \mid$$

Definition: num_{feat} being the number of features, a feature vector with window size *l* derived from attribute vector has dimension of:

$$Dim[feature\ vector] = (\frac{1}{l} * \frac{T}{t})\ X\ num_{feat}$$

3.3.4 ANOMALY GENERATION AND ATTACK SIMULATION

The rate of anomalies in the training dataset affects the learning bias of any machine learning algorithm intrinsically. Anomaly detection can be also considered as a rare class classification problem in the sense that anomalies in the data sets are almost all the time in minority. The unbalance between classes in a training set might result in a skewed and inherently biased learning which is not regarded as proper generalization [35]. Studies like [36] propose ways both to generate non-existing rare class instances and to over-sample rare class instances that have significance. IVADE also needs a particular amount of anomalous data for a proper learning.

In this thesis work, we take a different approach for anomaly generation by building rules of motion. Once these rules are formally stated, it becomes clear in what way anomaly class instances would be generated. First, we create physical rules as a ground check mechanism, then build up violation scenarios for each rule and lastly, generate synthetic anomalies for each scenario by injecting anomalous information in the data set. It has to be noted that observation on behavior of each parameter and experiments on the boundaries before introducing any violation scenario are required to embed system knowledge into rules and conditions. These rules, conditions and violation scenarios are stated in the following part.

Rules for Anomaly Generation

a) Violation of Parameter Range

Knowledge of parameters’ range in a system can be used as a ground truth.

b) Violation of Similarities

Anomaly class instances are at a distance from normal class instances. A distance manipulation can be used to generate anomaly samples from normal samples.

c) ***Violation of Physical Rules***

Vehicle dynamics is a valuable source of knowledge to found a ground truth.

d) ***Violation of Input-Output Relation***

The input output relations of a system are proprietary information and predefined at design phase. Distortion of input-output relationship can be used to generate anomalies.

e) ***Violation of Message Frequency***

In-vehicle networks transmit periodic and event-based messages. The frequency knowledge of message transmissions can be exploited for anomaly generation.

3.4 TRAINING WITH DECISION TREES

Supervised machine learning techniques have a training phase to represent the inductive inference among a data set as a classifier. The classifier of IVADE is formed by training the *Feature Vector* of CAN messages with decision trees. Decision trees are easy to build by quickly fitting the data set and use low memory resources with a fast response of prediction [56].

Feature Vector including features for windows of CAN messages and its corresponding *Output Vector* representing classes of every instance are used to train a decision tree in MATLAB. The *Fitctree* function is used to build a classification decision tree that uses the CART algorithm.

3.4.1 DECISION TREES

A decision tree is a representation of a tree from root to leaf nodes, applying classification of instances. The core of the decision tree algorithm is the Iterative Dichotomiser 3 (ID3) algorithm [37] that performs a greedy search in the space of all possible decision trees. A successor of the ID3 algorithm, C4.5 is introduced in [38] with a number of extensions to the core algorithm. The search begins to find the best candidate attribute for the root node of the tree by applying a statistical test to decide which attribute is able to classify the instances best by itself. After the selection of the root node, the search continues repetitively to find the descendant nodes using instances related with each branch.

The statistical test used as classification measure is *information gain*. The ID3 algorithm processes the information gain test repetitively to find the best attributes that can classify the instances from root to down. The information gain is stated formally as the decrease in the entropy as a result of partitioning by an attribute. In other words, the best attribute is the one that causes the minimum decrease in entropy by partitioning the data set. Entropy of system S is given as

$$Entropy\ (S) \equiv -\rho_+ \ \log_2 \rho_+ - \rho_- \log_2 \rho_-$$

Then, the information gain using the concept of entropy is given as;

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Training decision trees is essentially a search for the best tree fitting the data set among a hypotheses universe. There will be many decision trees that can be justified by different hypothesis. On the other hand, the bias learnt through heuristic information gain test can be stated as shorter (simpler) trees are preferred over larger trees [39].

Apart from ID3 and C4.5, another known algorithm is CART [41] which is also the default algorithm used in MATLAB for building decision trees. Instead of information gain, CART uses the Gini index. It is stated [40] that when there are 2 classes as in the binary classification problems, the regarding function for building decision trees *fitctree* always uses an exact search by using a computational shortcut described in [41]. With this method, categories can be ordered by probability for one of the classes in a classification problem and by mean response in regression problems. Consequently, computational challenges are avoided in classification cases with two classes (*normal* and *anomaly*).

4. IMPLEMENTATION OF IVADE

The proposed in-vehicle anomaly detection engine (IVADE) is implemented and tested in Matlab/Simulink. The performance of the algorithm is demonstrated by securing CAM messages against manipulation of mobility data which is generated by a Lane Keeping Assistance (LKA) application. LKA has strict safety-critical functions and can be regarded as a cooperative application since it has also a V2X connection interface with ITS infrastructure. The rest of Chapter 4 starts with a description of LKA and its parameters, a signal generator ECU block that produces input signals for LKA and a V2X transmitter ECU block that transmits in-vehicle mobility data to outside world. The in-vehicle network is implemented as a CAN bus and by applying varying driving scenarios the CAN data is acquired from network for the implementation of IVADE.

4.1 LANE KEEPING ASSISTANCE

4.1.1 MODEL DESCRIPTION

Lane Keeping Assistance (LKA) model is used as a test bed to simulate vehicle mobility and to implement the in-vehicle CAN network for the thesis work. LKA is an Advanced Driver-Assistance System (ADAS) that is different from Lane Departure Warning Systems. LKA provides a semi-autonomous driving ability while the latter is essentially a warning system to inform the driver when the vehicle moves out of the intended lane. LKA is developed in the simulation and analysis environment Matlab/Simulink due to its renown abilities and wide spread use for control systems.

Automotive industry has already developed many LKA systems that actively apply steering torque in case of a deviation from the followed lane [42]. Existing models help to keep the vehicle in the center of the road lane by constantly observing lane markings through automotive vision systems.

In our implementation, the LKA system receives radius information of the road as an input from a dedicated ECU that is assumed to calculate the road radius constantly using automotive vision sensors. In practice, such automotive vision ECUs are being developed using front, back and top cameras, LIDARs and several other sensor technology.

4.1.2 TECHNICAL FOUNDATION

The LKA model is used to simulate vehicle mobility and to generate realistic mobility data for an in-vehicle CAN network in which different ECUs are connected. The goal of designing an in-vehicle network simulation is to generate in-vehicle network data which is required to train IVADE; therefore, acquisition of CAN bus data for several road profiles is the expected outcome of the simulation. Theoretical foundation and technical details of vehicle dynamics model of LKA will not be explained in detail as the emphasis is to demonstrate securing of a next-generation ADAS system and an ITS service message, Cooperative Awareness Message (CAM), by applying the IVADE algorithm. On the other hand, it is worth mentioning that the LKA model was able to deliver a close approximation to similar real-life LKA systems by having reasonable and compatible mobility data outputs and satisfying the deviation limits from intended road profiles. Previous studies as in [43] used purely hypothetical simulation data without considering any dynamic model or are contended with CAN bus data taken from a car’s control network with lack of proprietary knowledge of car manufacturers [44]. To the author’s best knowledge, our research is the only work in the literature that intends to develop and test an IDS system for in-vehicle automotive networks using a vehicle dynamic model and hence, is able to validate the results of the anomaly detection algorithm.

The LKA model considers lateral and longitudinal dynamics of the vehicle and applies active steering to keep up with the intended track of the lane. The LKA is assumed as a planar model in which the vertical dynamics are not considered for a simplified discussion. There are two controllers designed in the LKA control system as can be seen in Figure 10. The first controller C1 is computing a desired yaw rate r_d that is then regulated by the second controller C2 that computes the steering angle δ for the vehicle.

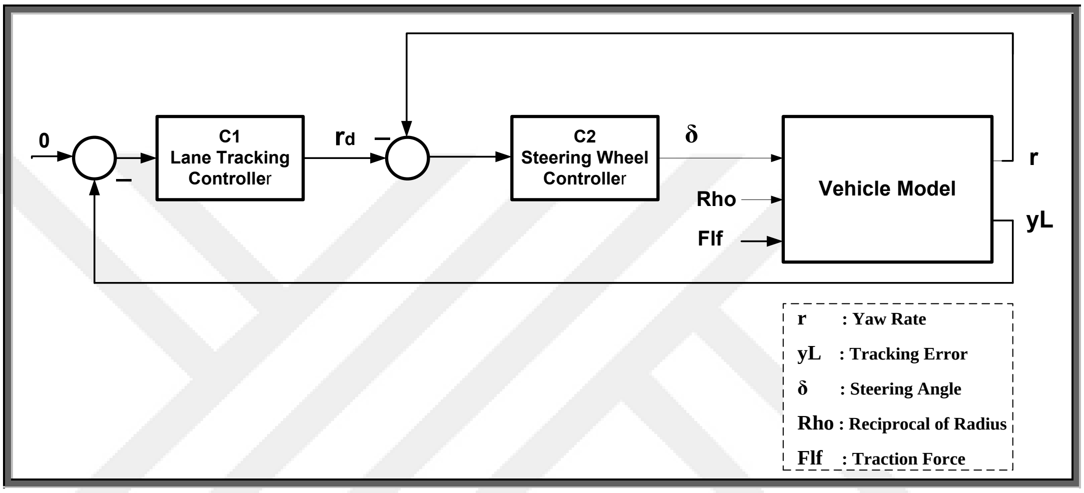

Figure 10. Block Diagram of LKA model.

As the vehicle model, a bicycle model as shown in Figure 11 is used. The steering angle δ is used to perform steering of the vehicle.

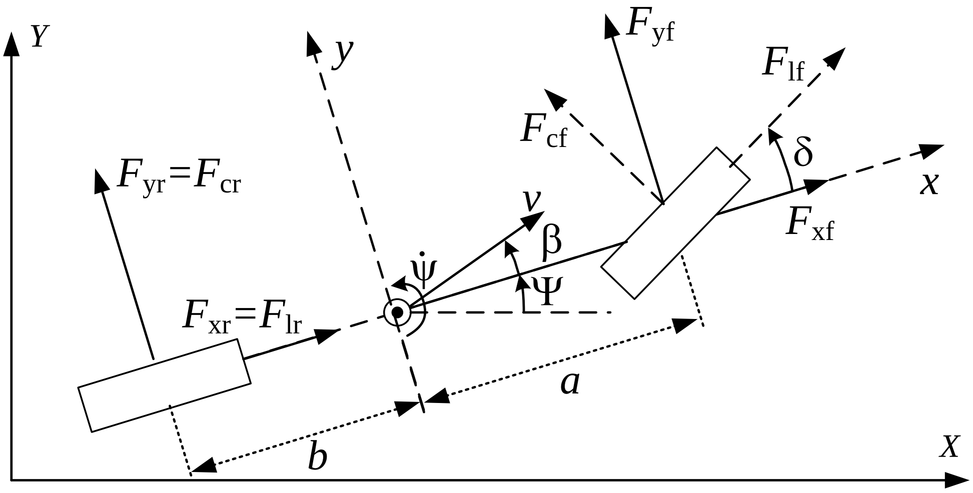

Figure 11. Schematic of Bicycle model

The vehicle is assumed as a front-wheel drive car and the following forces are considered to act on the wheels in the longitudinal and lateral wheel directions:

F_{lf}: Longitudinal force at the front wheel in the wheel direction,

F_{cf}: Lateral (cornering) force at the front wheel perpendicular to the wheel direction,

F_{lr} = 0: Longitudinal force at the real wheel in wheel direction. This force is zero since front-wheel actuation and rolling is assumed,

F_{cr}: Lateral force at the rear wheel perpendicular to the wheel direction.

The forces in the body frame can be computed from the forces in the wheel directions as follows:

$$F_{xf} = F_{lf}\cos\delta - F_{cf}\sin\delta \quad (front\ wheel, x - direction)$$

$$F_{yf} = F_{lf}\sin\delta + F_{cf}\cos\delta \quad (front\ wheel, y - direction)$$

$$F_{xr} = F_{lr} = 0\ (rear\ wheel, x - direction)$$

$$F_{yr} = F_{cr} = 0 \ (rear\ wheel, y - direction)$$

Dynamic Equations

The longitudinal velocity can be adjusted either by directly changing the longitudinal tire force or by controlling the longitudinal tire slip. The following dynamic equations are used:

$$\ddot{x} = \dot{y}\dot{\psi} + \frac{F_{xf} + F_{xr}}{m}$$

$$\ddot{y} = -\dot{x}\dot{\psi} + \frac{F_{yf} + F_{yr}}{m}$$

$$\ddot{\psi} = \frac{aF_{yf} + bF_{yr}}{I_{zz}}$$

Force Model

It is assumed that the tire force depends linearly on slip angle and longitudinal force at the front wheel Flf is used as an input signal to control the longitudinal acceleration

$$F_{cf} = C_f \alpha_f$$

$$F_{cr} = C_r \alpha_r$$

Where $C_f < 0$ and $C_r < 0$ are appropriate constants.

4.1.3 MODEL PARAMETERS

The LKA model itself can be described as a group of ECUs which compute the requested power for the forward motion and that calculate the necessary steering angle to keep the vehicle in the intended route by applying steering torque. For this model, LKA has an input interface from visual sensors and pedal actuators and it outputs mobility related parameters as the result of motion. The communication to and from LKA is assumed to be transmitted on the CAN bus. There are two signals LKA receives as input from other ECUs in the vehicle. These signals are the curvature (reciprocal of road radius) ρ and the traction force of the vehicle *Flf*. Additionally, there are 8 output signals that are transmitted to the outside of the model; global X-position *X*, global Y-position *Y*, global orientation ψ, velocity of the vehicle *v*, acceleration in the x-direction of body coordinate frame $\ddot{x}$, acceleration in the y-direction of body coordinate frame $\ddot{y}$, rotational acceleration $\ddot{\psi}$and deviation of the motion from the intended path *yl*.

For the scope of this thesis work, the curvature ρ *(= 1/ radius)* is assumed to be calculated by an automotive vision subsystems and transmitted to LKA model as an input. By applying different ρ profiles, it is possible to generate all types of road profiles, from roads with very tight turns to straight roads. In other words, a large value of ρ leads to a small radius and dictate a very tight turn while small values of ρmean a higher valued radius, even converging to a pseudo-straight road profile for marginally small values of ρ. Since there is a learning phase for IVADE algorithm, it is favorable to have different road profiles with a large variety of road data instances. Hence, the applicability of ρ input alteration finds a very functional use in generating various vehicle driving scenarios.

The second parameter to control the LKA model is the traction force of the vehicle, *Flf*. It is assumed in the dynamic model that the traction force simulates throttle and brake pedal actuation. Positive values of *Flf* demonstrate a stepping on the throttle pedal possibly resulting in acceleration while negative values correspond to applying of brakes that would result in deceleration. A zero value of *Flf* would imply no use of any of the pedals and results in coasting of the vehicle at a constant velocity (neglecting air drag and friction). Large valued (positive) traction forces result in a steeper acceleration than moderate values and low valued (negative) traction forces result in immediate slow down. Similar to ρ, alterations of *Flf* make it possible to have a large number of driving scenarios and hence, high numbers of different road data instances can be generated by applying varying traction forces both at positive and negative magnitudes.

Description of model parameters

Input Parameters

a) Reciprocal of radius ρ

The road curvature parameter is regulated by Highway and Traffic Agencies in many countries. Range of ρ to be given to LKA model as an input doesn’t have a theoretical limit. In order to have a realistic approach, curvature radius values are chosen according to the regulations and real life practices. Minimum road curvature radius is one of the road geometry design parameters that affects safety and also is regarded as a fundamental parameter for road regulation [45]. Table 1 from [46] and Table 2 from [45] show desirable minimum values that are stated as comfortable values dictated by design speeds. By observing values from these three countries, it would be a roughly correct estimate that a tight radius would be around 100m. Therefore, for the tight-turn scenarios, ρ_{ight} is chosen as 100m.

Table 1. UK highway standards.

	DESIGN SPEED (km/h)	120	100	85	70	60	50
HORIZONTAL CURVATURE (m)	**Minimum R without elimination of Adverse Camber and Transitions**	2880	2040	1440	1020	720	520
	Minimum R with Superelevation of 2.5%	2040	1440	1020	720	510	360
	Minimum R with Superelevation of 3.5%	1440	1020	720	510	360	255

	Desirable Minimum R with Superelevation of 5%	1020	720	510	360	255	180
	One Step below Desirable Minimum R with superelevation of 7%	720	510	360	255	180	127
	Two Steps below Desirable Minimum R with superelevation of 7%	510	360	255	180	127	90

Table 2. Germany and France highway standards for Road Radius.

Design Speed (km/h)	**120**	**100**	**90**	**80**	**70**	**60**
Germany	800m	500m	380m	280m	200m	135m
France	665m	425m	-	240m	-	120m

In addition to a tight-turn radius, a moderate-turn radius is chosen as 500m and a comfort-turn radius is chosen as 1000m respectively. Straight road radius is also simulated as a road curvature radius of 5000m. Four discrete radius values and corresponding eight ρ values including negatives used in the simulations are shown in Table 3. Positive ρ values represent motion in the counter-clockwise rotation while negative values represent motion in clockwise rotation.

Table 3. Discrete Values for ρ and Radius.

Tight-turn Curve Radius	**Moderate-Turn Curve Radius**	**Comfort-Turn Curve Radius**	**Straight Curve Radius**
100m	500m	1000m	5000m
ρ = 0.01, -0.01	ρ = 0.002, -0.002	ρ = 0.001, -0.001	ρ = 0.0002,-0.0002

b) Traction Force *Flf*

The traction force input, *Flf* depends on the requested motor torque and the road conditions such as the friction coefficient of the road. The higher the positive traction force input represents a greater gas pedal actuation involvement. Similarly, as the negative traction force given into LKA model decreases, it implies harder brakes that would cause deceleration and a probable stop of the motion in the end. It is also possible to simulate a driving profile of *coasting* by applying a traction force close to “0” representing neither gas nor brake pedal is involved.

The intention of the data generation in the thesis work is to create in-vehicle network data for all types of movement; acceleration, deceleration, coasting (constant-speed drive) and variations of these three main movement types with different rates. In other words, a hard acceleration produces significantly different data from a soft acceleration does and similarly, a hard brake generates quite different in-vehicle network data than a smooth slow down drive. In order to create profiling of many different movement scenarios, a set of discrete values are chosen for the traction force input. Table 4 summarizes all the values used in the simulation that represent different possible drive scenarios.

Table 4. Discrete Values for Traction Force, Flf

Hard Acceleration	2000
Moderate Acceleration	1000, 1250
Soft Acceleration	750, 500
Coasting	10
Soft Brake (deceleration)	-250, -500
Moderate Brake (deceleration)	-750, -1250
Hard Brake (deceleration)	-1500

Output Parameters

LKA model is a Lane Keeping Assistance system that applies steering torque to keep the vehicle in the intended route in accordance with the observed road radius and traction force input. As the drive goes on with given inputs, vehicle experiences a change in the motion parameters. There are 8 output signals from LKA model that are related with the change in the motion. These output signals contain information on global positions, orientation, velocity, acceleration and deviation from the intended path of the vehicle. All the output parameters are explained in this section to have a better understanding of the model.

Global positioning of the vehicle is given in a hypothetical planar universe with only X and Y coordinates. The vertical movement of the vehicle is omitted for the simplification of the discussion since the essential goal of the thesis is to simulate the in-vehicle network data for different road profiles. The starting point of the vehicle in each drive simulation is considered as the origin of the universe with the coordinates (X, Y) = (0, 0). As the movement goes on, global positions of the vehicle is updated according to the direction of the motion and the distance taken in both X and Y direction. Since the starting point of the drive is at point (0, 0), global position-X and global position-Y can take both positive and negative values.

The velocity of the vehicle is a result of the inputs and the model outputs the instantaneous velocity of the vehicle through the simulation. Velocity parameter can take only non-negative scalar values without implying any vectored direction. By adjusting inputs accordingly, velocity of the vehicle in the simulation is tried to be limited between 0 – 180 km/h in order to have a realistic driving simulation.

Output parameters X, Y and ψ are considered in global coordinates. Conversely, acceleration parameters$\ddot{x}$, $\ddot{y}$ and $\ddot{\psi}$ are given in the body frame coordinates. The velocity parameter designated as v is the velocity of the vehicle in the direction of the motion by not being represented in any coordinate system.

The lateral deviation of the vehicle from the intended route is designated with *yl* parameter. Lateral deviation information is informative on the success of the LKA model to keep the vehicle in the safe distance limits. In other words, higher values of *yl* would mean that the intended route is not achieved and the safety of operation is failed. There is not an official upper limit for this deviation, yet a 50cm deviation is considered as a suitable value to ensure a safe drive.

Global Position *X*

LKA model outputs the instantaneous X-coordinate of the global position of the vehicle and the signal is designated with *X*. LKA model outputs instantaneous X-coordinate of the current position, therefore distance taken in the coordinate system is represented in meters unit.

Global Position *Y*

LKA model outputs the instantaneous Y-coordinate of the global position of the vehicle and the signal is designated with *Y*. LKA model outputs instantaneous Y-coordinate of the current position, therefore distance taken in the coordinate system is represented in meters unit.

Global Orientation ψ

LKA model outputs the instantaneous global orientation of the vehicle and the signal is designated with ψ. The orientation of the unit is considered as 0 radian in the direction of +X coordinate and increases in the counter clockwise direction to a maximum of 2π. In other words, $\psi$is periodical in 2π radian scale, making a full circle turn of 360° would have a global orientation equals to 0 (= 2π).

Velocity *v*

LKA model outputs the instantaneous velocity of the vehicle in the direction of motion and the signal is designated with *v*. The velocity of the vehicle is in meter/second unit.

Lateral Deviation *yl*

LKA model outputs the instantaneous lateral deviation of the vehicle from the intended route and the signal is designated with *yl*. The lateral deviation of the vehicle is in meter unit.

4.2 SIMULATION OF IN-VEHICLE NETWORK

The scope of this thesis work is not to propose the technical realization of the LKA model and control system but is to develop a simulation and test platform for the implementation and evaluation of the proposed Intrusion Detection System IVADE. The LKA model is developed in Matlab/Simulink environment due to its wide spread use for control systems. In addition to control systems, Matlab/Simulink environment can be used as an integration platform for models from different specialization areas as it is done through this thesis work. A pure dynamic model of LKA system implemented in Simulink is transformed into an in-vehicle network using Vehicle Network Toolbox design blocks by introducing CAN bus channels as communication layer.

4.2.1 DESCRIPTION OF THE SIMULATION MODEL

The in-vehicle network is formed by four main components in the final phase; LKA dynamic model, a custom signal generation block to provide two inputs ρ and *Flf* to the LKA, a V2X transmitter ECU to send the mobility data of the vehicle to V2X infrastructure and lastly, the Intrusion Detection System which monitors CAN bus channels and detects anomalies. The main components of the network will be discussed in detail in the following section.

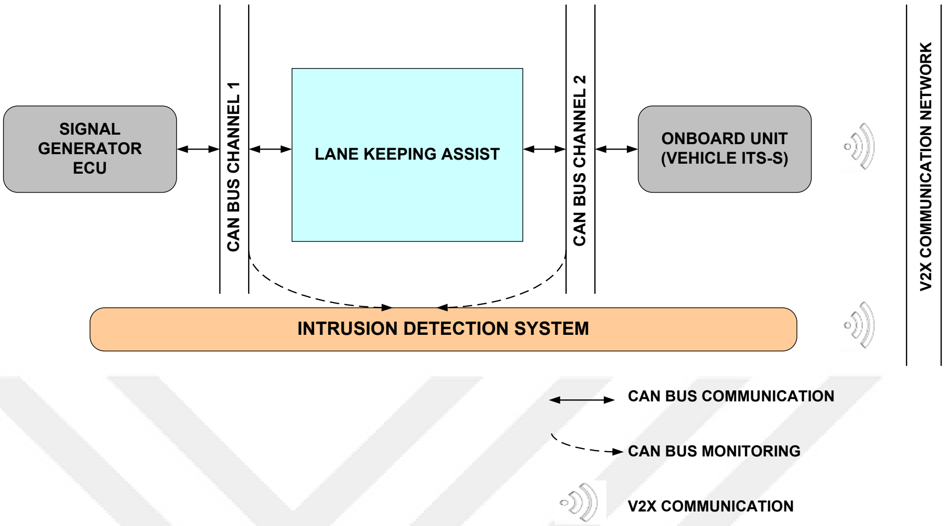

Figure 12. Schematic representation of In-Vehicle Network.

4.2.2 DIRECT AND NETWORKED MODEL

The first phase of simulation environment is called Direct Model in which three components in the network are directly connected in Simulink. The direct Model is basically the native form of the model without a CAN BUS implementation. Three main components of the network are; LKA dynamic model, input signal generation block and V2X transmitter ECU are linked to each other and the architecture is shown in FIGURE X. The direct Model provides an environment to test and observe the behavior of the LKA model, i.e. the response of output parameters that are to be sent to V2X infrastructure for various combinations of input parameters. The nominal behavior of the model is observed and noted before modifications for Networked Model are applied.

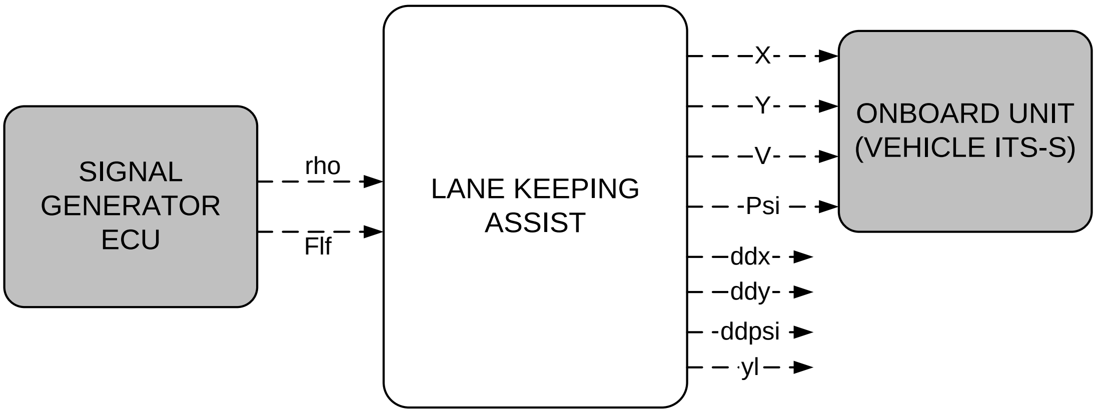

Figure 13. Schematic representation of Direct Model

The second phase of the simulation environment is called Networked Model in which modules of the Direct Model are assumed as ECUs of a vehicle control network which are connected through CAN Bus channels with each other. In other words, instead of using direct links to connect design blocks and LKA model, signals are transmitted through CAN bus networks as CAN messages from and to the LKA. This is the final form of the simulation set-up in which different driving scenarios are run by generating appropriate inputs. The CAN Bus network data including inputs to the model and outputs from it are logged.

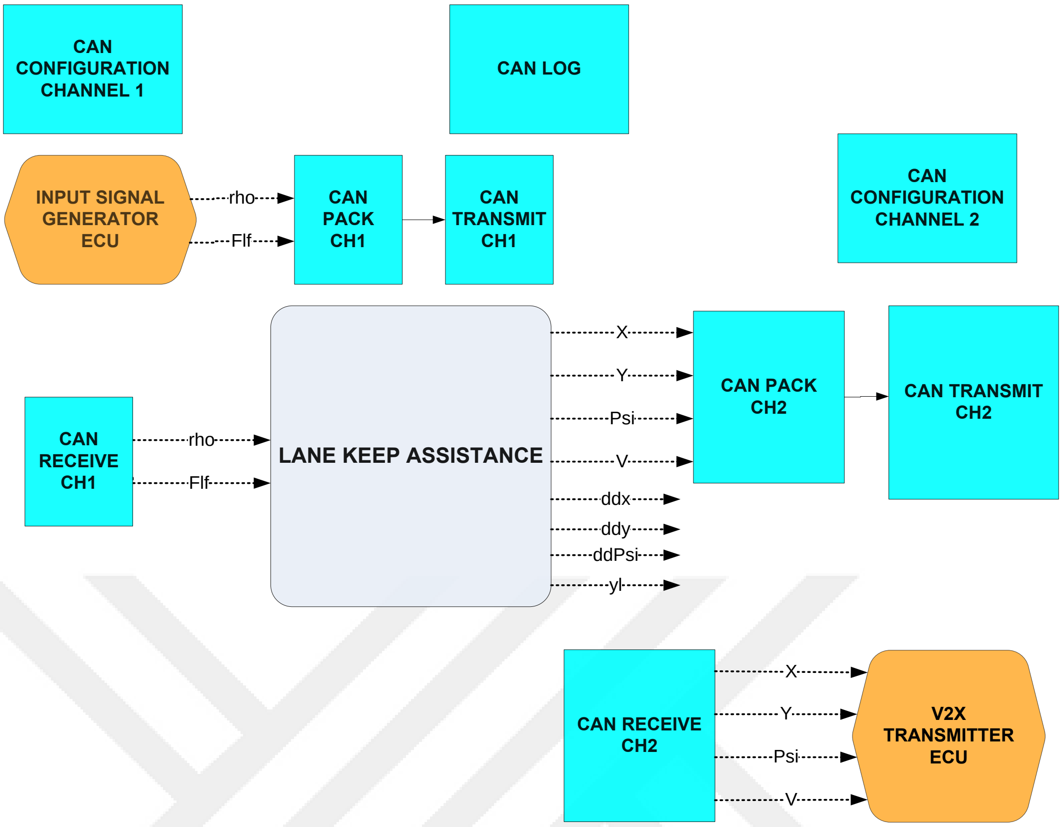

Figure 14. Schematic representation of Networked Model.

The real output of the simulation is the motion in the coordinate universe. Therefore, in order to compare the Direct Model and Networked Model, same input configuration is fed into both models and the route in the coordinate universe is observed. It would be naïve to expect the route of the Networked Model to be identical to the Direct Model considering that the signals are not transmitted instantly through a direct link but in message-wise communication through a CAN bus channel. On the other hand, Networked Model is a more realistic model in the sense that in real in-vehicle networks, information is also transmitted in the same fashion; through control networks with time multiplexing or event-based message transmission as in the Flexray, CAN or Ethernet case. DM and NM are run with same input signals and *speed (v)* output is observed. The difference between models can be seen in Figure 15. The lagging behavior of NM model can be explained with delays resulting from input bus CAN Channel 1 and output bus CAN Channel 2.

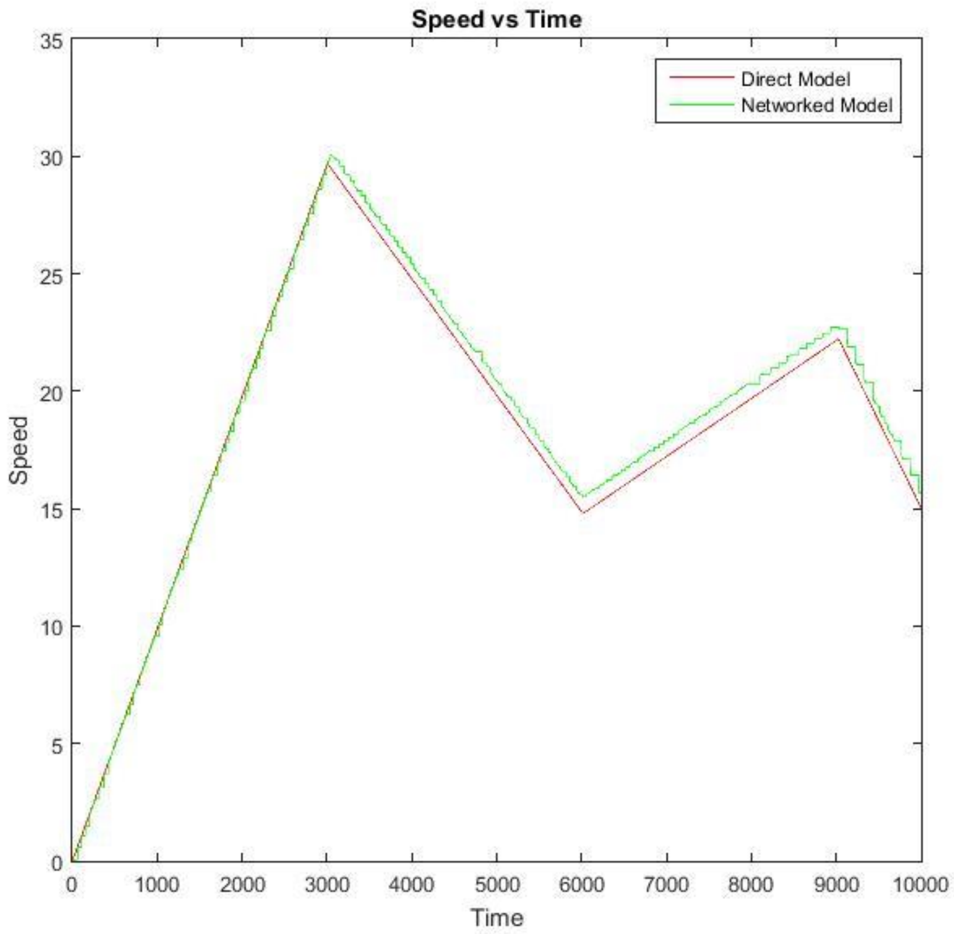

Figure 15. Speed vs Time in DM and NM.

4.2.3 IN-VEHICLE CAN BUS IMPLEMENTATION

The Direct Model is transformed into Networked Model by implementing CAN Bus Channel communication interface for transmission of signals. CAN Bus is a widely used communication protocol that has been in use for three decades now. Controller Area Network (CAN) is a serial De Facto Standard for in-vehicle control area networks especially for subsystems that is related to mobility functions, i.e. engine, transmission, steering and all type of ADAS. Although Ethernet is perceived as a strong alternative for future vehicles' internal networks due to its wide bandwidth and scalability, CAN is going to be dominating sub-control networks which carry out mostly mobility-related safety-critical functions. Therefore, because of the wide-spread use in ADAS systems and current dominance in the industry, CAN bus is chosen as the communication layer for the in-vehicle network.

Vehicle Network Toolbox (VNT) of Matlab/Simulink provides a connection interface with CAN devices, both through virtual channels for simulation purposes and through vehicle networks. It is possible to carry out operations such as sending, receiving, coding and decoding of messages. There are also functional blocks maintained in VNT for monitoring of CAN channels, analyzing messages and logging them for future use. Throughout the work in this study, many of these functions have been used and implemented.

4.2.4. MAIN COMPONENTS OF THE SIMULATION

LKA Model

LKA dynamic model is developed in Simulink and perceived as a black box in the simulation environment. The input and output interfaces of LKA model were transformed into CAN channels by using generic Vehicle Network Toolbox blocks of Simulink. The dynamic model is essentially used to simulate a Lane Keeping Assistance system driving the vehicle on a given route. The input and output parameters of the model is explained in detail in the previous section.

Input Signal Generator

The input parameters, reciprocal of the road radius ρ and traction force *Flf*, to be given to the LKA model is generated by a custom function block developed in Simulink. Input signal generator block simulates two possible ECUs which are dedicated to generate ρ information from automotive vision and sensor system by observing road radius and to generate gas and brake pedal actuation information *Flf*.

This function block produces all different input combinations of ρ and *Flf* for several driving scenarios.

Distinct values of inputs are chosen from the following sets:

$$\rho \in \{-0.01, -0.002, -0.001, -0.0002, 0.0002, 0.001, 0.002, 0.01\}$$
$$Flf \in \{-1500, -1250, -750, -500, -250, 10, 250, 500, 750, 1000, 2000\}$$

The number of different ρ values is Nρ = 8 and the number of different *Flf* values is N_{Flf} = 11. The number of all different input combinations is Nρ x N_{Flf} = 88. All possible input combinations are given in Table 5.

Table 5. Different Input Combinations.

ρ /Flf	-1500	-1250	-750	-500	-250	10	250	500	750	1000	2000
-0.01	S1	S9	S17	S25	S33	S41	S49	S57	S65	S73	S81
-0.002	S2	S10	S18	S26	S34	S42	S50	S58	S66	S74	S82
-0.001	S3	S11	S19	S27	S35	S43	S51	S59	S67	S75	S83
-0.0002	S4	S12	S20	S28	S36	S44	S52	S60	S68	S76	S84
0.0002	S5	S13	S21	S29	S37	S45	S53	S61	S69	S77	S85
0.001	S6	S14	S22	S30	S38	S46	S54	S62	S70	S78	S86
0.002	S7	S15	S23	S31	S39	S47	S55	S63	S71	S79	S87
0.01	S8	S16	S24	S32	S40	S48	S56	S64	S72	S80	S88

Signal generator function block outputs one of the 88 different input combinations for 5 seconds per one output state randomly. Since, the order of the state is not pre-decided, there is a risk of the vehicle speed exceeding or falling below the practical limits. In order to keep the vehicle in motion during the simulation, the velocity must be positive. Therefore, when velocity of the vehicle decreases below a threshold, a positive Flf is applied for 5 seconds. Similarly, when speed goes up to a higher threshold, a negative Flf is applied for 5 seconds.

The essential motion types were defined as acceleration, deceleration and coasting previously. These 88 different input combinations are all examples of the three basic motion types, acceleration, deceleration and coasting, with different rates of road curvature and traction force. The input sequence pattern is one of the significant factors that is critical to have a rich data set including all of the possible driving scenarios. The numerical information on simulation data is given in Chapter 5.

Can Bus Channel Blocks

In order to implement CAN Bus communication using Vehicle Network Toolbox, there are necessary blocks to be used. Two CAN channels are implemented for inputs and outputs of LKA model, i.e. CAN Channel 1 is created for inputs to LKA model, ρ and *Flf*; CAN Channel 2 is created for mobility parameters of CAM messages X, Y, ψ and v. IVADE is assumed to be connected to both channels to observe inputs and outputs of LKA model. The following blocks are used and configured with appropriate parameter settings to achieve a healthy and functional CAN communication.

Vehicle Network Toolbox Simulink Blocks:

CAN Configuration: This block is used to configure the setting of a CAN Channel. This block must be used for each channel created.

CAN Pack: This block is used to pack the signals into a CAN message.

CAN Transmit: This block is used to transmit packed CAN messages to a selected channel.

CAN Receive: This block is used to receive CAN messages from a selected channel

CAN Log: This block is used to log CAN messages into a file for future use.

4.3 ANOMALY GENERATION AND ATTACK SIMULATION

Supervised learning methods require both positive (normal) and negative (anomaly) class samples in the training set. We derived rules according to IVADE Anomaly Generation principles explained in 3.3.4 Anomaly Generation and Attack Simulation. We simulate manipulation of mobility data (Global Position, Speed, Global Orientation) that is vital for V2X CAM messages.

Rules for Anomaly Generation

a) ***Manipulation of Velocity Data***

Rule 1: Speed cannot decrease at certain conditions. Condition can be given as (radius is MODERATE to STRAIGHT) AND (Flf is POSITIVE).

This rule is violated by injecting point anomalies in which speed is decreased by a percent.

Rule 2: Speed cannot increase at certain conditions. Condition can be given as either Flf is negative OR (radius is TIGHT and Flf is close to ZERO).

This rule is violated by injecting point anomalies in which speed is increased by a percent.

Rule 3: Speed cannot increase instantly. There is no condition for this rule.

This rule is violated by injecting point anomalies in which speed is increased dramatically.

Rule 4: Speed cannot decrease instantly. There is no condition for this rule.

This rule is violated by injecting point anomalies in which speed is decreased dramatically.

Rule 5: The change in velocity in a window-frame should be bounded by a small constant.

This rule is violated by injecting point anomalies in which speed is manipulated by a constant deviation from the dynamic model behavior.

b) Manipulation of Position Data

Rule 6: The change in displacement in a window-frame should be bounded by a small constant.

This rule is violated by injecting point anomalies in which GPS-X and GPS-Y are manipulated by a constant deviation from the dynamic model behavior.

Rule 7: If Global Orientation is in the neighborhood of n*pi/2, i.e. if it is closer to x = y OR x = -y lines; X component of displacement should be comparably similar to Y-component of the displacement.

This rule is violated by injecting point anomalies in which GPS-X and GPS-Y are manipulated so that similarity of displacements is distorted.

Rule 8: If Global Orientation is NOT in the neighborhood of n*pi/2, i.e. if it is NOT close to x = y OR x = -y lines; the ratio between X component and Y component of displacement should be comparably similar to global orientation.

This rule is violated by injecting point anomalies in which GPS-X and GPS-Y are manipulated according to a percent of instantaneous global orientation.

c) Manipulation of Global Orientation Data

Rule 9: The change in global* orientation *in a window-frame should be bounded by a small constant.

This rule is violated by injecting point anomalies in which global orientation is manipulated by a constant deviation from the dynamic model behavior.

Rule 10: Change of ψ in a window-frame should be non-negative when radius of the road is counter-clockwise.

This rule is violated by injecting point anomalies that makes the change of Psi negative.

Rule 11: Change of ψ in a window-frame should be non-positive when radius of the road is clockwise.

This rule is violated by injecting point anomalies that makes the change of $\boldsymbol{\psi}$positive.

By applying these rules, manipulations of the CAN message instances are created in attribute vector and these anomalous instances are transferred to feature vector by modifying the 20th instance of each frame. In other words, manipulations are always applied to the newest element in a window frame. We limited the structure of algorithm by manipulating only the last element in the window for a simplified discussion. In the future, we aim to improve IVADE so that manipulations can be detected at any location of the window. Together with the attribute vector, feature vector representing windows and its corresponding output vector also updated with anomalous class frames.

5. EVALUATION AND RESULTS

5.1 SIMULATION PARAMETERS

This section includes all numerical data of the simulation experiments. The necessary details and explanations are provided accordingly.

5.1.1 CAN MESSAGES AND DATA FIELDS

CAN data fields are up-scaled with proper constants in order to have a better resolution. Up-scaled values of X, Y and V are too high to be transmitted with 1 byte of CAN message field. Therefore, these three signals are represented with 3 bytes. Therefore, up-scaled values are divided into multiple bytes before transmission and converted back to the original value at reception where the values are down-scaled with same constants. The scale constants and byte orders are given in the tables below.

As an example when X = 2425,1, it is up-scaled to X' = 24251 with constant = 10. Then, three bytes are calculated with the following formulas:

$$X' = 10000 * X_{up} + 100 * \ X_{mid} + \ X_{low}$$

$$X_{up} = rem(X', 10000) = 2$$
$$X_{mid} \ = rem\big((X' - X_{up} * 10000\big), 100) = 42$$
$$X_{low} = \big((X' - X_{up} * 10000\big) - X_{mid} \ * 100\big) = 51$$

The scaling constants and the used CAN messages are shown in Table 6 and 7.

Table 6. Scaling Constants.

Parameter	Scaling Constant
X	10
Y	10
V	100

Table 7. CAN Messages.

	CAN Channel 1	CAN Channel 2	
	CAN ID: 100	CAN ID: 200	CAN ID: 300
1. Byte	ρ	X_up	V_up
2. Byte	Flf_up	X_mid	V_mid
3. Byte	Flf_low	X_low	V_low
4. Byte	-	Y_up	Psi_up
5. Byte	-	Y_mid	Psi_mid
6. Byte	-	Y_low	Psi_low
7-8. Byte	-	-	-

5.1.2 LKA CONFIGURATION SETTINGS

The LKA model parameters are initialized with the values in Table 8. Before every drive simulation, a parameter file is executed for initialization.

Table 8. LKA Configuration Parameters.

Vehicle Mass	2023/2223/2423 kg
Distance from center of gravity to front wheel	1.26 m
Front wheel tire force constant	2.864e5
Inertia	6286
Distance from center of gravity to rear wheels	1.9 m
Rear wheel tire force constant	1984e5
Nominal Velocity	0.01 m/sec
Look-ahead distance for road detection	12 m
KP1	20
KI1	10
KP2	30
KI2	0.01
KI3	0.01
Kd	0.05
Tau	0.01 sec
Initial velocity in x-direction	V (=0.01)
Initial velocity in y-direction	0
Initial angular velocity	0

5.1.3 ATTRIBUTE AND FEATURE VECTORS

Attribute Vector is derived from sampled CAN messages. The attributes chosen is given in Table 9.

Table 9. Attribute Vector Parameters.

Attr1	Attr2	Attr3	Attr4	Attr5	Attr6
ρ	Flf	X	Y	V	Psi
Attr7	Attr8	Attr9	Attr10	Attr11	Attr12
$V_i - V_{i-20}$	$X_i - X_{i-20}$	$Y_i - Y_{i-20}$	$Psi_i - Psi_{i-20}$	tan(Psi)	$\frac{X_i - X_{i-20}}{Y_i - Y_{i-20}}$

Feature Vector is derived from attribute vector, by calculating consecutive instances of attribute vector in the same window. The length of the window is chosen as *l* = 20. The chosen metric as feature is calculated for 20 consecutive instances of attribute vector. The features selected are given in Table 10.

Table 10. Feature Vector Parameters.

Feat1	Feat2	Feat3	Feat4
$mean(\rho)$	$mean(Flf)$	$mean(V)$	$mean(Psi)$
Feat5	Feat6	Feat7	Feat8
$\tan(\boldsymbol{\psi})$	$\frac{X_i - X_{i-20}}{Y_i - Y_{i-20}}$	$dist(I(dV') - dV)$	$dist(I(dX') - dX)$
Feat9	Feat10	Feat11	
$dist(I(dY') - dY)$	$dist(I(dPsi') - dPsi)$	mass	

* $mean(p)$ is defined as average value of an attribute in a window.

* $dist(I(dp') - dp)$ can be defined as the Chebyshev distance between $p_i - p_{i-20}$ and interpolation of $p_i - p_{i-20}$ for the last 20 instances.

* mass is the current mass of vehicle.

* tan(p) is the tangent of p.

Principles for Attribute and Feature Selection

We have derived three main ideas for determining attributes and extracting features from these attributes. As a summary, the motion of a vehicle constitutes a physical behavior that can be observed by monitoring physical signals. Firstly, the relationship between input signals and output signals of ECUs comprise knowledge of the system behavior. This behavior can be learnt for profiling the in-vehicle network. As an example, the global orientation of a vehicle is strictly related with the input signal ρ, reciprocal of road radius. Similarly, the change of vehicle speed can be derived from the change of the traction force *Flf*. Secondly, network data instances have a transient state which is not suitable for observing system behavior. Such a sampling would introduce noise that would impair the success of the algorithm. Therefore, a time window is considered as a steady point for observation of state changes. Lastly, the length of the window has to be short enough for a quick response to detection while being long enough to contain necessary knowledge. Consequently, all signals of the in-vehicle network that are related to motion and indicators of the changes in a system are considered as good attributes. We also provide reasons of the selection and contribution of each attribute and feature in this section.

Evaluation of Attributes

1. Reciprocal of Radius: ρ, is an input signal that carries the information of road radius. It has a dramatic effect on the motion of vehicle, X and Y component of distances taken.

2. Traction Force: *Flf* is the signal coming from throotle and braking pedal actuators. Flf is decisive for the state of motion, whether it is acceleration, deceleration or coasting.

3. GPS Position X: *X* is the x coordinate of the vehicle’s instantaneous position. The signal is informative on the distance taken. Additionally, it also gives a clue about vehicle’s orientation when previous values of the signal are considered.

4. GPS Position Y: *Y* is the y coordinate of the vehicle’s instantaneous position. The signal is informative on the distance taken. Additionally, it also gives a clue about vehicle’s orientation when previous values of the signal are considered.

5. Speed: V is a vital property of vehicle that is informative of the motion.

6. Global Orientation: Psi is the heading direction of vehicle. It is particularly related with ρ and can be indicator of the shape of the road taken, in other words X and Y component of distance taken.

7. Change of Speed: This attribute is derived from Speed in a time window. The change in speed is decisive on the distance taken and it is related with *Flf*.

8. Change of Position-X: This attribute is derived from GPS Position X in a time window. The distance taken in the coordinate of X is informative on the trend of distance taken. It is related with speed, traction force and road radius.

9. Change of Position-Y: This attribute is derived from GPS Position Y in a time window. The distance taken in the coordinate of Y is informative on the trend of distance taken. It is related with speed, traction force and road radius.

10. Change of Orientation: This attribute is derived from $\boldsymbol{\psi}$ in a time window. The change in orientation is decisive on the shape of the road and the X and Y component of roads taken.

11. Tangent of Current Orientation: Tangent of instantaneous orientation is informative on the recent changes of other parameters. Theoretically, it should be related with delta(X)/delta(Y) in a long time window, but the relation is not linear for instantaneous response. This signal is representative for short term changes in orientation.

12. Delta(X)/Delta(Y): Unlike tan($\boldsymbol{\psi}$), the signal is chosen for long term representation of change in orientation.

Evaluation of Features

Features are derived from attributes with a window approach. The main principles for the selection of features are to exploit the behavior of input-output relationships, to leverage distance-based similarity comparison and to represent behavioral knowledge of the in-vehicle network. All selected features contribute to the representation of all three knowledge types, however it is worth to underline particular reasons on the selection of each feature.

1. Behavioral Knowledge: Average values of ρ, *Flf*, *V* and *Psi*: Average values of these signals are informative for behavioral learning of in-vehicle network. Additionally, the causality between inputs and outputs are useful on the prediction of signals.

2. Temporal Knowledge: Tangent of Current Orientation and Delta(X)/Delta(Y): These attributes are preserved as features in order to represent knowledge for the prediction of orientation and change of the distance taken in both short-term and long-term.

3. Similarity Knowledge: Distances between instances and interpolated instances are derived for *X*, *Y*, *V* and $\boldsymbol{\psi}$ in order to represent the similarity of instances in terms of distance metrics. This knowledge is especially critical in the existence of data manipulation.

5.2 EVALUATION CRITERIA

Anomaly detection is a classification problem and as it is performed, there are four possibilities for the outcome:

Missed Intrusions: Malicious CAN messages which are flagged as normal. These messages will be represented as False Negatives (FN).

False Alarms: Benign CAN messages which are flagged as anomalous. These messages will be represented as False Positives (FP).

Detected intrusions: Malicious CAN messages which are flagged as anomalous. These messages will be represented as True Positives (TP).

Undetected normal packets: Benign CAN messages which are flagged as normal. These messages will be represented as True Negatives (TN).

Performance Criteria

There are many accuracy measures to evaluate how an anomaly detection algorithm performs. Recall and Precision are among the most used metrics for classification problems. Recall is the ratio of anomalies that are correctly identified as anomalous compared to the whole number of anomalies. It is especially important for anomaly detection since the anomaly class has usually fewer numbers of instances in any data set. Precision is the measure of how an algorithm was sharp on the point to correctly detect anomalies in all alarms.

$$\boldsymbol{Recall(=TPR)} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FN}}$$

$$\boldsymbol{Precision} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FP}}$$

In addition, there are four more metrics that are used to analyze performance of classification problems: TPR is true positive rate, FNR, false negative rate, TNR, true negative rate and FPR being false positive rate.

$$\boldsymbol{TPR} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FN}}$$
$$\boldsymbol{FNR} = \frac{\boldsymbol{FN}}{\boldsymbol{FN+TP}}$$
$$\boldsymbol{TNR} = \frac{\boldsymbol{TN}}{\boldsymbol{TN+FP}}$$
$$\boldsymbol{FPR} = \frac{\boldsymbol{FP}}{\boldsymbol{FP+TN}}$$

F-measure is one effective metric for classification problems. By representing both precision and recall, which are usually inversely proportional, F-measure is used to find out a balance point for thresholds that generates the least amount of FP at the FN rate that system can tolerate.

$$\boldsymbol{F - measure} = \frac{\mathbf{2}}{\frac{\mathbf{1}}{\boldsymbol{Precision}} + \frac{\mathbf{1}}{\boldsymbol{Recall}}}$$

5.3 EXPERIMENTS

Several experiments are performed in order to improve feature selection process. In the beginning, we didn’t use any distance metrics, then we performed more experiments by adding distance features to feature vector.

After deciding final forms of feature vector, we train a decision tree as the final experiment and present results of all tests in this section.

5.3.1 Experiments for Feature Selection

A number of experiments are presented in this part to demonstrate the significance of distance features and the influence of the amount of data.

Experiment 1: Without a distance feature, we observed high numbers of FP and FN (Table 11&12).

Table 11. Feature vector.

Feat1	Feat2	Feat3	Feat4
$mean(\rho)$	$mean(Flf)$	$mean(V)$	$mean(Psi)$
Feat5	Feat6	Feat7	Feat8
$\tan(\boldsymbol{\psi})$	$\frac{X_i - X_{i-20}}{Y_i - Y_{i-20}}$	$dV = V_i - V_{i-20}$	$dX = X_i - X_{i-20}$
Feat9	Feat10	Feat11	
$dY = Y_i - Y_{i-20}$	$dPsi = Psi_i - Psi_{i-20}$	mass	

Table 12. Results of experiment 1.

Test ID	# anomaly	# normal	FP	FN	TPR	FPR	F
Train: 7200sec Test: 7200sec, Mass = 2023kg							
Test	10303	25696	9522	1247	0.8788	0.37	0.6267

Experiment 2: Distance metric for V is added as Feature 7 = $dist(I(dV') - dV)$.

Table 13. Results of experiment 2.

Test ID	# anomaly	# normal	FP	FN	TPR	FPR	F
Train: 7200sec Test: 7200sec, Mass = 2023kg							
Test	10513	25486	8091	1256	0.8805	0.31	0.6645

Experiment 3: Distance metric for Psi is added as Feat 10 = $dist(I(dPsi') - dPsi)$

Table 14. Results of experiment 3.

Test ID	# anomaly	# normal	FP	FN	TPR	FPR	F
Train: 7200sec Test: 7200sec, Mass = 2023kg							
Test	10416	25883	7777	603	0.9421	0.3040	0.7008

Experiment 4: Distance metric for X is added as Feat8 = $dist(I(dX') - dX)$

Table 15. Results of experiment 4.

Test ID	# anomaly	# normal	FP	FN	TPR	FPR	F
Train: 7200sec Test: 7200sec, Mass = 2023kg							
Test	10478	25521	2790	556	0.9469	0.1093	0.8557

Experiment 5: Distance metric for Y is added as Feat9 = $dist(I(dY') - dY)$. The results are improved significantly; therefore, we decided to use distance metrics in the final feature vector.

Table 16. Results of experiment 5.

Test ID	# anomaly	# normal	FP	FN	TPR	FPR	F
Train: 7200sec Test: 7200sec, Mass = 2023kg							
Test	10378	25621	4	4	0.9996	1.56e-4	0.9996

Experiment 6: We used one hour of drive data for training and one hour of drive for testing. Two data sets are randomly generated and test data is not seen by trained decision tree before. We observed more false negatives and positives. Then we used two hours of data for testing to observe the same trend. Lastly, we decided to use more data for training and we observed that more data worked better than the former case in terms of performance. Results of three test cases are given in Table 17-19.

Table 17. Results of experiment 6 – step 1.

Test ID	# anomaly	# normal	FP	FN	TPR	FPR	F
Train: 3600sec Test: 3600sec, Mass = 2023kg							
Test 1	5190	12809	8	6	0.9985	4.68e-4	0.9987
Test 2	5166	12833	6	5	0.9990	4.67e-4	0.9989
Test 3	5114	12885	5	6	0.9988	3.88e-4	0.9989

Table 18. Results of experiment 6 – step 2.

Test ID	# anomaly	# normal	FP	FN	TPR	FPR	F
Train: 3600sec Test: 7200sec, Mass = 2023kg							
Test 1	10375	25624	8	10	0.9990	3.12e-4	0.9991
Test 2	10341	25658	10	11	0.9989	3.89e-4	0.9990
Test 3	10484	25515	12	9	0.9991	4.70e-4	0.9990

Table 19. Results of experiment 6 – step 3.

Test ID	# anomaly	# normal	FP	FN	TPR	FPR	F
Train: 7200sec Test: 7200sec, Mass = 2023kg							
Test 1	10303	25696	5	1	0.9999	1.94e-4	0.9997
Test 2	10421	25578	9	0	1	3.52e-4	0.9996
Test 3	10395	25604	7	0	1	2.73e-4	0.9997

5.3.2 Main Experiment

Out main experiment includes training a decision tree using in-vehicle network data and testing this decision tree with randomly generated test drives.

a. Training Data

Different driving profiles are generated by applying inputs in a random sequence. The CAN log block records all in-vehicle network data in both CAN Channel 1 and CAN Channel 2. The recorded data is then processed for further analysis. The drive profiles are given in Table 20 for training drives.

Table 20. Training Data.

TrainingData	Vehicle mass	Duration	# of instances / CAN ID
Drive 1	2023	2 hours (7200s)	719,979
Drive 2	2023	2 hours (7200s)	719,979
Drive 3	2223	2 hours (7200s)	719,979
Drive 4	2223	2 hours (7200s)	719,979
Drive 5	2423	2 hours (7200s)	719,979
Drive 6	2423	2 hours (7200s)	719,979

Each of the 6 driving profiles is different in terms of input state sequence since a wide scanning of the training space is intended. Additionally, since the vehicle mass has a physical effect on the vehicle dynamic response, the driving profiles are generated for three different weights. Firstly, Drive 1-6 are combined to form a CAN bus data vector by applying necessary data transformation, then an *attribute vector* is derived from the CAN bus vector by preprocessing attribute values and is shown in Table 21. Lastly, a *feature vector* is extracted from the attribute vector with the window approach and features of each window are calculated. The feature vector is shown in Table 22.

Table 21. Content of attribute vector.

Duration	# of instances	# of attributes	Dimension
12 hours	4,319,880	12	4,319,880 X 12

Table 22. Content of feature vector.

Duration	# of instances	# of features	# of anomalies	# of normal
12 hours	215,994	11	61,688	154306

b. Test Data

The trained DT is tested with test drive data. Test drives are generated by random input sequences in order not to have similar profiles with training data. Three Drive profiles are given in Table 23. Each Drive data is tested with DT five times and results are provided in the next section.

Table 23. Content of test data.

Test Data	Vehicle mass	Duration	# of instances / CAN ID
Drive 7	2023	2 hours (7200s)	719,979
Drive 8	2223	2 hours (7200s)	719,979
Drive 9	2423	2 hours (7200s)	719,979

c. Decision Tree

A decision tree based on the *feature vector* and *output vector* designating the class of each window frame is trained. Classification rules and graphical description of the decision tree is given in Figure 16 and Table 24 respectively.

Table 24. Classification rules for Decision Tree.

```
1  if x8<0.00541667 then node 2 elseif x8>=0.00541667 then node 3 else 0
 2  if x5<0.113278 then node 4 elseif x5>=0.113278 then node 5 else 0
 3  class = 1
 4  if x10<0.218567 then node 6 elseif x10>=0.218567 then node 7 else 0
 5  class = 1
 6  if x9<0.22013 then node 8 elseif x9>=0.22013 then node 9 else 0
 7  class = 1
 8  if x10<0.134464 then node 10 elseif x10>=0.134464 then node 11 else 0
 9  class = 1
10  if x9<0.186399 then node 12 elseif x9>=0.186399 then node 13 else 0
11  if x3<12.0808 then node 14 elseif x3>=12.0808 then node 15 else 0
12  if x3<1.391 then node 16 elseif x3>=1.391 then node 17 else 0
13  class = 0
14  class = 1
15  class = 0
16  if x3<1.379 then node 18 elseif x3>=1.379 then node 19 else 0
17  if x3<2.79375 then node 20 elseif x3>=2.79375 then node 21 else 0
18  if x10<0.112365 then node 22 elseif x10>=0.112365 then node 23 else 0
19  class = 1
20  if x3<2.79 then node 24 elseif x3>=2.79 then node 25 else 0
21  class = 0
22  class = 0
23  class = 1
24  class = 0
25  class = 1
```

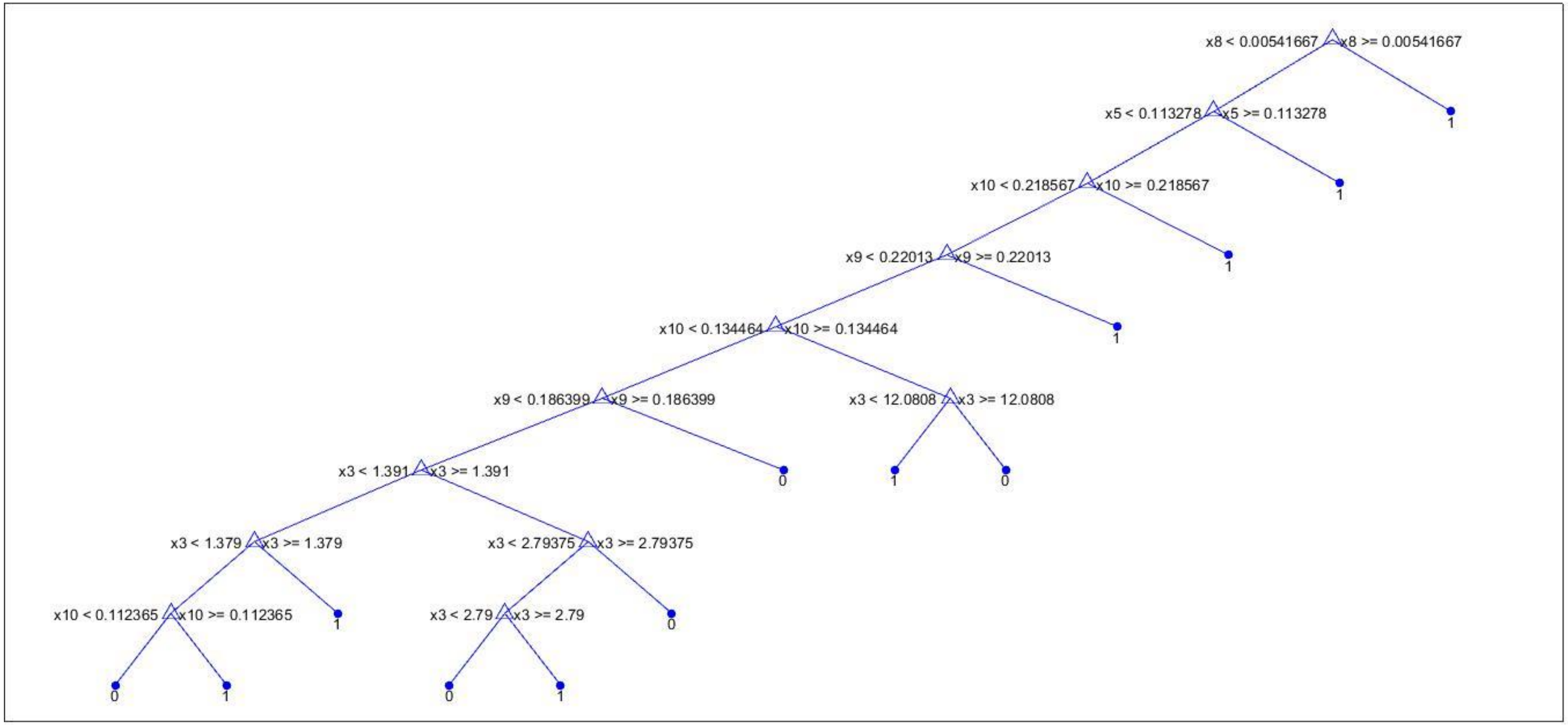

Figure 16. Decision Tree Graphical View.

5.4 RESULTS

Each of the three test drive profiles, Drive7-9, are used to form *feature vectors* as test vector. Since each processing involves random data manipulation for anomaly generation, each drive profile is run five times. Test cases and results are evaluated by using performance metrics that are common techniques for classification and anomaly detection.

The results of our simulation test are summarized in Table 25.

Table 25. Test Results.

Test ID	# anomaly	# normal	FP	FN	TPR	FPR	F
DRIVE 7, Mass = 2023kg							
Drive7.1	10463	25536	0	0	1	0	1
Drive7.2	10230	25769	1	0	1	3.88e-5	1
Drive7.3	10378	25621	0	1	0.9999	0	1
Drive7.4	10280	25719	1	0	1	3.88e-5	1
Drive7.5	10381	25618	1	0	1	3.90e-5	1
DRIVE 8, Mass = 2223kg							
Drive8.1	10166	25833	0	1	0.9999	0	1
Drive8.2	10219	25780	0	0	1	0	1
Drive8.3	10307	25692	0	0	1	0	1
Drive8.4	10281	25718	0	0	1	0	1
Drive8.5	10204	25795	0	0	1	0	1
DRIVE 9, Mass = 2423kg							
Drive9.1	10417	25581	1	1	0.9999	3.90e-5	0.9999
Drive9.2	10388	25610	0	1	0.9999	0	1
Drive9.3	10330	25668	0	0	1	0	1
Drive9.4	10417	25581	1	1	0.9999	0	1
Drive9.5	10348	25650	1	1	0.9999	3.898e-5	0.9999

We next comment on the main observations of our tests.

Key Points

At each run of Drive 7, 8 and 9, anomalies are generated according to the rules previously mentioned. The rate of anomalies to all instances is around 30% for all data sets. The result of the attack detection tests is promising that IVADE can detect a point anomaly at the first encounter with high accuracy. The response time of a detection is crucial for safety critical systems, therefore detection at first sight is favorable.

It has to be noted that the positive results are obtained where both the training data and test data are randomly generated. Test data is never seen by trained decision tree.

Results of Drive 1 and Drive 3 are similar whereas Drive 2 has better scores. This is partly due to the mass feature. Decision tree is trained with an equal number of instances from each mass, hence the average converges to 2223 kg. Therefore, DRIVE 8 can be expected to have better results.

The degree of manipulation of anomalies were the same for both training and test instances because the same constants and manipulation metrics were used. The values of manipulations are distributed but being very low close to 0. IVADE was successful at detecting manipulations with values lower than 0.5 m for position (X and Y); 0.30 m/s for speed (V) and 2° for orientation (Psi).

The abundance of data contributed to the accuracy of algorithm. Experiments with less data yielded more FP and FN.

6. CONCLUSION

The automotive world is about to face a revolution regarding the amount of data processed and exchanged via communication. The increasing trend in establishing connection surfaces between a vehicle’s internal control network and the outside world jeopardize one of the main objectives of transportation: safety. The conventional security measures may be ineffective for such large and distributed systems and complementary approaches for securing a vehicle’s internal network are needed. Behavioral profiling of in-vehicle networks is seen as a promising technique for security purposes due to the increasing computing capabilities and abundance of data.

In the described context, this thesis focuses on the detection of anomalies in in-vehicle networks. Hereby, we make use of the fact that in-vehicle networks are not as random as traditional IT networks in which user-based deviations can be difficult to model. Specifically, safety-critical messages in in-vehicle networks are transmitted periodically and the message contents depend on the dynamical vehicle behavior. As the main contribution of the thesis, we develop a general approach by exploiting the behavioral, temporal and distance characteristics of in-vehicle network data, for detecting anomalous traffic. The proposed In-Vehicle Anomaly Detection Engine (IVADE) collects and preprocesses data and derives attributes and features that are considered powerful indicators for the anomaly detection. Using machine learning, we train IVADE with Decision Trees because it quickly captures inductive inference while consuming modest resources. We use distance-based similarity concept as features and a window frame approach to capture contextual information.

For the evaluation, we implemented IVADE by using the dynamic model of a next generation Lane Keeping Assistance system that has safety-critical functions by controlling the mobility of vehicle. The model is extended by an internal CAN bus network in which signals are transmitted through CAN messages and mobility data including position, speed and direction of a vehicle are transmitted to V2X network with an onboard V2X transmitter unit. We collect and process the CAN bus data as a data source for IVADE. We use violations of the physical rules of motion for the vehicle in order to generate coherent anomalies, instead of generating random noise or outliers. The algorithm is trained with hours of drive data and tested against manipulations of the mobility data. The results of the implementation are promising and encouraging for behavioral protection of not only vehicles but all cyber-physical systems. It is observed that anomalies are detected with a very high success rate which is paramount for driving safety.

REFERENCES

[1] **Höfler T., Burkert C., Telzer M., (2004),** *“Comparative Firewall Study”.*

[2] **Axelsson S., (2010),** *“Intrusion detection systems: A survey and taxonomy”*, Vol. 99, Technical report, 2000.

[3] **Ueda H., Kurachi R., Takada H., (2015),** *“Security Authentication System for In-Vehicle Network”*, SEI TECHNICAL REVIEW, (81), 5.

[4] **Cho K. T., , Shin K. G., (2016),** *“Fingerprinting electronic control units for vehicle intrusion detection”,* 25th USENIX Security Symposium (USENIX Security 16). USENIX Association, 2016.

[5] **Zhang T., Delgrossi L., (2012),** *“Vehicle safety communications: protocols, security, and privacy”*, Vol. 103. John Wiley & Sons, 2012.

[6] **Stübing H., (2013),** *“Multilayered security and privacy protection in Car-to-X networks: solutions from application down to physical layer”*, Springer Science & Business Media, 2013.

[7] **Koscher K., Czeskis A., Roesner F., (2010),** *“Experimental security analysis of a modern automobile”*, Security and Privacy (SP), 2010 IEEE Symposium on. IEEE, pp. 447-462.

[8] **Checkoway, S. et. al., (2011),** *“Comprehensive Experimental Analyses of Automotive Attack Surfaces”*, USENIX Security Symposium, 2011.

[9] **Miller C., Valasek C., (2013),** *“Adventures in automotive networks and control units”*, DEF CON 21 (2013), pp. 260-264.

[10] **Miller C., Valasek C., , (2015),** *“Remote exploitation of an unaltered passenger vehicle”*, Black Hat USA 2015.

[11] **Milinkovic S. A., Lazic L. R. , (2012),** *“Some Facts about Industrial Software Security”*, XI International SAUM Conference on Systems, Automatic Control and Measurements, 2012.

[12] **Verizon (2016),** *“2016 Data Breach Investigations Report”*, Verizon.

[13] **Sladkowski A., Pamula W., (2015),** *“Intelligent Transportation Systems–Problems and Perspectives”*, Vol. 32. Springer, 2015

[14] **World Bank , Rural population (% of total population), (2017),** [Online], Available: http://data.worldbank.org/indicator/SP.RUR.TOTL.ZS

[15] **Picone, M. et al, (2015),** *"Advanced technologies for intelligent transportation systems"*, Vol. 139, pp. 1-199, Springer.

[16] **National Highway Traffic Safety Administration, (2016),** *"2015 motor vehicle crashes: overview"*, Traffic safety facts research note, 2016, 1-9.

[17] **ETSI EN 302 665 v1.1.1: Intelligent transport systems (ITS); Communications architecture (2010).**

[18] **Karagiannis, Georgios et al., (2011),** *"Vehicular networking: A survey and tutorial on requirements, architectures, challenges, standards and solutions."*, IEEE communications surveys & tutorials 13.4 (2011): pp. 584-616.

[19] **Müter M., Groll A., Freiling F. C., (2010),** *"A structured approach to anomaly detection for in-vehicle networks"*, Information Assurance and Security (IAS), 2010 Sixth International Conference on, pp. 92-98, IEEE.

[20] **Stübing, Hagen et al., (2010),** *"Sim TD: A Car-to-X system architecture for field operational tests [Topics in Automotive Networking]"*, IEEE Communications Magazine 48.5, 2010.

[21] **Mitchell R., Chen I. R., (2014),** *"A survey of intrusion detection techniques for cyber-physical systems"*, ACM Computing Surveys (CSUR) 46.4, (2014): 55.

[22] **Denning D. E., (1987),** *"An intrusion-detection model"*, IEEE Transactions on software engineering 2 (1987), pp. 222-232.

[23] **Ghorbani A. A., Lu W., Tavallaee M., (2009),** *"Network intrusion detection and prevention: concepts and techniques"*, Vol. 47. Springer Science & Business Media, 2009.

[24] **Tzur-David S., (2011),** *"Network Intrusion Prevention Systems: Signature-based and Anomaly Detection"*.

[25] **Bhuyan M. H., Bhattacharyya D. K., Kalita J. K., (2014),** *"Network anomaly detection: methods, systems and tools"*, IEEE communications surveys & tutorials 16.1 (2014), pp. 303-336.

[26] **Chandola V., Banerjee A., Kumar V., (2009),** *"Anomaly detection: A survey"*, ACM computing surveys (CSUR) 41.3 (2009): 15.

[27] **Patcha A., Park J. M., (2007),** *"An overview of anomaly detection techniques: Existing solutions and latest technological trends"*, Computer networks 51.12 (2007): pp. 3448-3470.

[28] **Kumar S., Spafford E. H., (1994),** *"An application of pattern matching in intrusion detection".*

[29] **Linda O., Vollmer T., Manic M., (2009**), *"Neural network based intrusion detection system for critical infrastructures",* Neural Networks, 2009, IJCNN 2009, International Joint Conference on. IEEE, 2009.

[30] **Papadimitratos, Panagiotis et al., (2008),** *"Secure vehicular communication systems: design and architecture"*, IEEE Communications Magazine 46.11 (2008).

[31] **RADAR, camera and LiDAR for autonomous cars, (2017),** [Online], Available: "https://blog.nxp.com/automotive/radar-camera-and-lidar-for-autonomous-cars"

[32] **Garcia-Teodoro, Pedro, et al., (2009),** "*Anomaly-based network intrusion detection: Techniques, systems and challenges"*, Computers & Security 28.1 (2009): pp. 18-28.

[33] **Wagner A., Plattner B., (2005),** *"Entropy based worm and anomaly detection in fast IP networks"*, Enabling Technologies: Infrastructure for Collaborative Enterprise, 14th IEEE International Workshops on. IEEE, 2005.

[34] **Markovitz M., Wool A., (2017),** *"Field classification, modeling and anomaly detection in unknown CAN bus networks"*, Vehicular Communications 9 (2017): pp. 43-52.

[35] **Joshi M. V., Agarwal R. C., Kumar V., (2001),** *"Mining needle in a haystack: classifying rare classes via two-phase rule induction"*, ACM SIGMOD Record 30.2 (2001): pp. 91-102.

[36] **Alhammady H., Ramamohanarao K., (2004),** *"Using emerging patterns and decision trees in rare-class classification",* Data Mining, 2004. ICDM"04. Fourth IEEE International Conference on. IEEE, 2004.

[37] **Quinlan J. R., (1986),** *"Induction of decision trees"*, Machine learning 1.1 (1986): pp. 81-106.

[38] **Quinlan J. R., (2014),** *"C4.5: programs for machine learning"*, Elsevier, 2014.

[39] **Mitchell T. M., (1997),** *"Machine learning. 1997."*, Burr Ridge, IL: McGraw Hill, 45(37), 870-877 Challenges in Splitting Multilevel Predictors, Matlab, (2017),

[40] **MATLAB Help for CART,** [Online], Available: http://www.mathworks.com/help/stats/splitting-categorical-predictors-for-multiclass-classification.html"

[41] **Breiman, Leo et al., (1984),** *"Classification and regression trees", Wadsworth & Brooks"*, Monterey, CA (1984).

[42] **"Active Lane Keeping Assist",** Mercedes-Benz USA, [Online], Available: "https://www.mbusa.com/mercedes/technology/videos/detail/title-safety/videoId-e84b9423c67a7410VgnVCM100000ccec1e35RCRD"

[43] **Kang M. J., Kang J. W., (2016),** *"Intrusion Detection System Using Deep Neural Network for In-Vehicle Network Security"*, PloS one 11.6 (2016): e0155781.

[44] **Jun Li, (2016),** *"CANSsee - An Automobile Intrusion Detection System*", HITB(2016).

[45] **Ruyters H. G. J. C. M., Slop M., Wegman F. C. M., (1994),** *"Safety effects of road design standards"*, Vol. 94. No. 7. SWOV Institute for Road Safety, 1994.

[46] **Highway Link Design, Volume 6 Road Geometry Section 1 Links, , (2002),** *"DESIGN MANUAL FOR ROADS AND BRIDGES".*

[47] **CRASHOVERRIDE: Analysis of the Threat to Electric Grid Operations,** Online Report, [Online], Available: https://dragos.com/blog/crashoverride/CrashOverride-01.pdf

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

[49] **ISO 21217, Intelligent transport systems —**Communications access for land mobiles (CALM) — Architecture.

[50] Qu, Tianmin. "Database security in assets of companies." *T-110.501 Seminar on Network Security. Retrieved April.*, Vol. 11., 2001**.**

[51] **Jennings, N. R., & Wooldridge, M. J. (1998).** "*Applications of intelligent agents".*

[52] **Watterson, C. (2012), "***Controller Area Network (CAN) Implementation Guide"***,** Application Note AN-1123, Analog Devices, Inc.

[53] **Corrigan, S. (2008), "***Introduction to the controller area network (CAN)***",** *Texas Instrument, Application Report*.

[54] **Bouard, A., (2014)**, "*Middleware-based Security for Future In-Car Networks"*, Doctoral dissertation, München, Technische Universität München, Diss., 2014.

[55] **Stumpf F., Pohl C. (2014)**, *"An Analysis and Comparison of Hardware Security Modules for the Automotive Domain"*, ESCAR USA 2014.

[56] **MATLAB, Choose a Classifier Type**, [Online], Available: https://www.mathworks.com/help/stats/choose-a-classifier.html

[57] **Varaiya P. (1993),** "*Smart cars on smart roads: problems of control*", Automatic Control, IEEE Transactions on 38(2), pp. 195–207.

[58] **Li L., Wen D. and Yao D. (2014),** "*A survey of traffic control with vehicular communications*", Intelligent Transportation Systems, IEEE Transactions on 15(1), 425–432.