Detecting CAN Bus Intrusion by Applying Machine Learning Method to Graph Based Features

Rafi Ud Daula Refat(✉), Abdulrahman Abu Elkhail, Azeem Hafeez, and Hafiz Malik

University of Michigan -Dearborn, 4901 Evergreen Rd, Dearborn, MI, USA
{rerafi,abdkhail,azeemh,hafiz}@umich.edu

Abstract. Modern vehicle is considered as a system vulnerable to attacks because it is connected to the outside world via a wireless interface. Although, connectivity provides more convenience and features to the passengers, however, it also becomes a pathway for the attackers targeting in-vehicle networks. Research in vehicle security is getting attention as in-vehicle attacks can impact human life safety as modern vehicle is connected to the outside world. Controller area network (CAN) is used as a legacy protocol for in-vehicle communication, However, CAN suffers from vulnerabilities due to lack of authentication, as the information about sender is missing in CAN message. In this paper, a new CAN intrusion detection system (IDS) is proposed, the CAN messages are converted to temporal graphs and CAN intrusion is detected using machine learning algorithms. Seven graph-based properties are extracted and used as features for detecting intrusions utilizing two machine learning algorithms which are support vector machine (SVM) & k-nearest neighbors (KNN). The performance of the IDS was evaluated over three CAN bus attacks are denial of service (DoS), fuzzy & spoofing attacks on real vehicular CAN bus dataset. The experimental results showed that using graph-based features, an accuracy of 97.92% & 97.99% was achieved using SVM & KNN algorithms respectively, which is better than using traditional machine learning CAN bus features.

Keywords: In-vehicular network security · Intrusion detection system · Controller area network · Feature engineering · Machine learning

1 Introduction

A modern vehicle is a complex system that consists of electronic, mechanical & software components. A typical modern vehicle is equipped with 70 electronic control units (ECUs) on average [42]. These ECUs are called the brain of the vehicle and are responsible for the safety of the passengers and vehicle functionality. In order to communicate, the ECUs are connected using in-vehicle

K. Arai (Ed.): IntelliSys 2021, LNNS 296, pp. 730–748, 2022.
https://doi.org/10.1007/978-3-030-82199-9_49

communication protocol networks i.e. local interconnected network (LIN), controller area network (CAN), media oriented system transport (MOST) etc. [16]. Among them, the CAN bus was designed in 1983 to allow safety-critical ECUs to communicate inside a vehicle [12]. The introduction of CAN drastically resolved the wiring problem caused by point to point connection. It is a single bus system where the ECUs can be easily connected and was designed mainly for closed vehicular systems.

Vehicular systems are not considered closed systems anymore. Current vehicles are connected with external networks via Bluetooth, WIFI, installed apps, etc. They are even connected to each other while on road [40]. The connectivity of modern vehicles is becoming an access point for the attackers and becoming a security threat. In most cases, the motivation of the attackers is to access the safety-critical ECUs like brake, gas, powertrain, etc. The majority of the attackers take advantage of the CAN as it gives access to all the safety-critical ECU components and can take full control of the vehicle [40].

By design, CAN allows to broadcast messages and a single message does not have any information about the sender [40]. This vulnerability gives the attacker a chance to send a message to the CAN after gaining illegitimate access. Researchers have demonstrated these security vulnerabilities and were able to take control of the vehicle remotely [1]. Using the CAN arbitration ID, three types of attacks can be performed by the researchers [35]. They are DoS (denial of service) attack, spoofing attack and fuzzy attack. These attacks are concerning as it is directly related to passenger safety. To increase CAN bus security researchers have proposed several solutions by providing message authentication [39,43,44]. But, these solutions are not practical on CAN bus protocol since CAN bus has limited data byte (8 bytes). Additionally, the implementation of message authentication will add overhead and will limit the existing bandwidth (500 kbps). So, the techniques like designing IDS (intrusion detection system) is becoming more popular as they do not limit bandwidth [35] and do not modify existing CAN bus protocol.

In this paper, an IDS is proposed to detect DoS, fuzzy and spoofing CAN bus attacks by using machine learning. Popular machine learning algorithms SVM (support vector machines) and KNN (k-nearest neighbors) are used for intrusion detection. SVM and KNN have been used recently by [1] to increase CAN security. Unlike [1], the proposed IDS uses seven features (six novel features and a single feature from the state-of-the-art [21]) with high feature differences among benign and malicious CAN messages, which gives better classification accuracy. The experimental results show that the selected seven features represent the accurate behavior of CAN benign and malicious messages. The proposed work is inspired by the work done in [21]. The author converted the CAN messages into graphs and used a single graph property i.e. edges to detect CAN bus attack by using the statistical chi-square method. Unlike [21], new seven graph-based CAN bus features are explored that are used to detect CAN intrusions. Moreover, the proposed IDS can detect attacks using a single graph, whereas [21]

needs distribution of graphs for CAN bus attack detection. The followings are the contribution of this paper:

- To the best of our knowledge, this is the first machine learning-based CAN bus IDS that uses graph-based features.
- The IDS takes advantage of a total of seven graph-based properties that represent the actual behavior of the CAN bus.
- The experimental results show that the classification based on graph-based features are performing better than classification based on traditional CAN bus message features.
- The proposed IDS can detect three types of CAN attack and is applicable to in-vehicle networks.

2 Background and Related Work

2.1 CAN Bus

In modern electric vehicles, actuators and sensors are controlled through the electronic control units (ECUs). ECU is a device in modern vehicles which control electric subsystems. The ECUs are responsible for a variety of vehicle functions including engine control, braking, airbag deployment, door lock/unlock, antilock braking system (ABS), parking support system. Various network protocols have been proposed for in-vehicle communication between ECUs, such as controller area network (CAN), local interconnected network (LIN), media oriented system transport (MOST) [12]. CAN protocol is most commonly used for in-vehicle communication due to its robustness. Robert Bosch GmbH developed the CAN Protocol and published CAN 2.0 specification A and B in 1991 [20]. In 1993, the international organization for standardization (ISO) released standard ISO 11898 for CAN protocol [20]. Some of the advantages of CAN protocol are it decreased the cost of wiring in vehicles, had built-in error detection, increased robustness, higher speeds, and much more flexibility [2]. CAN protocol consists of multiple abstraction layers. The two important layers are the physical Layer and the transfer Layer.

Physical Layer. CAN is a broadcast-based communication protocol that is utilized in many different applications that have complex structure topology and require reliable communication between devices e.g. automotive, aerospace and trains, etc. [4]. CAN has two types of physical layer standards, low speed and high speed, which determine how the CAN bus is structured and the speeds of the CAN bus [12]. The low speed standard has a baud rate up to 125 *Kbps* that requires a single wired bus and devices that self terminate by 120 *ohm* resistors on the CAN Bus [20]. A high speed CAN Bus consists of 2 wired half duplex serial network technology [20]. The wires are called CAN High (CANH) and CAN Low (CANL), which terminate at 120 *ohms* resistor. CAN is equipped to operate smoothly in different types of environments because of the electromagnetic shielding. CAN prevents electromagnetic interference (EMI) and protects

communications from electromagnetic radiations that an automobile under goes daily. To prevent magnetic field radiation, the pair of wires are twisted. Furthermore, to prevent electric field radiation, a coaxial cable is used for the two wires. Also, another issue that can occur is electrostatic discharge (ESD) on the CAN Bus and it is prevented by the CAN transceiver.

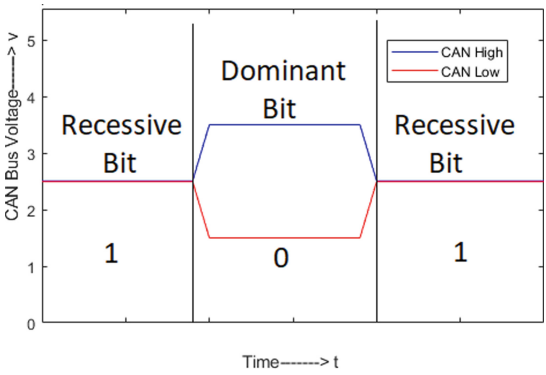

Fig. 1. CAN-bus differential voltage representation.

One main key feature of the CAN Protocol is that it supports centralized communication control over ECU [4]. ECUs can communicate with other ECUs on the network, and each ECU requires a micro-controller, CAN Controller, and CAN Transceiver as shown in Fig. 2. The micro-controller controls when the message should be transmitted and analyzes messages received from the bus. The micro-controller is connected to the CAN Controller which has two pins, transmitter (CAN-TX) and receiver (CAN-RX) [20]. These two pins are connected to the CAN Transceiver and have digital voltages of 0V for logical '0' and 5V for logical '1'. The actual CAN bus does not support these voltages. Therefore, the CAN Transceiver converts the digital logic voltages into a differential signal [4]. The CAN Transceiver drives and detects data communication to and from the bus. The differential voltages are outputs and consist of 2 states of voltages, dominant (or logical 0) and recessive (logical 1) [4]. The differential voltages for the dominant (0) are 3.5V on CANH and 1.5V on CANL [4,11,13–17,37]. In addition, the differential voltages for the recessive (1) are 2.5 V on both CANH and CANL. Fig. 1 shows the CAN bus differential voltage representation. The two pins on the CAN Transceiver are connected directly to the bus which allows the ECU to transmit and receive messages from the bus [20]. How the messages are transferred over through the bus is discussed in the following section.

Transfer Layer. The transfer layer abstraction receives messages from the physical layer and transmits those messages using the CAN bus. This layer is responsible for timing synchronization, message framing, arbitration, acknowledgment, fault confinement, error detection, and signaling [22]. These properties

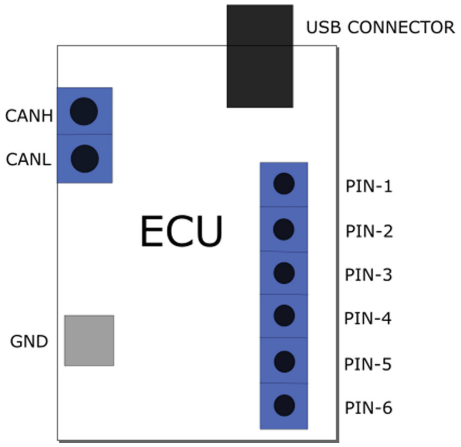

Fig. 2. Electronic control unit.

of the transfer layer are very important to the robustness of the CAN protocol which allows safe message communication between ECUs. These messages allow ECUs to communicate with any other ECU by the way of broadcasting the message to the shared CAN bus. Messages in CAN Protocol are usually event driven which means an event must occur before any communication is established [7,14]. All other ECUs receive the transmitted message and depending on the parameters in the message, the ECU will either accept or reject the message. Communication in CAN protocol consists of four types of frames which are sent to all ECUs. These four types of frames operate differently and consists of different number of parameters. The types of frames are as follows, the Data Frame, Remote Frame, Error Frame, and Overload Frame [20]. The four frames can be classified as error message frames or data message frames. The error message frames communicate errors that occur on the data message frames during the transmission on the CAN bus and they consist of the Error Frame and Overload frame. The data messages communicate actual data or request data to be communicated, which include the Data Frame and the Remote Frame. A data message frame, as shown in Fig. 3, can have a maximum of 126 bits and consists of the following parameters:

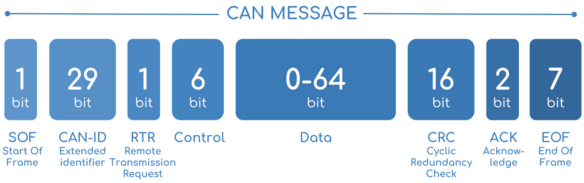

Fig. 3. CAN data message frame.

- Start of Frame (SOF): The Start of Frame is a single dominant(0) bit which marks the start of a message and is used to synchronize the ECUs on the bus [20].
- Identifier (ID): The Identifier determines the priority of the message. The lower the ID value, the higher the priority and vice-versa [20]. There are 2 types of ID's which are standard ID'S and extended ID'S. Standard ID's consist of 11 bits and extended ID's consists of 29 bits [20].
- Remote Transmission Request (RTR): The Remote Transmission Request is made up of one bit [20]. If the bit is set to dominant (0), the message is considered as a Remote Frame [20]. However, if the message is set to recessive (1), the message is considered as a Data Frame [20].
- Control: The Control consists of 6 bits which defines the type of data that will be transmitted [45]. The first bit is the Identifier Extension (IDE) bit which determines if the ID is a standard ID (11-bits) set to dominant(0) or extended ID (29-bits) set to recessive(1) [45]. The second bit is the Reserved Bit (R0) which is always dominant (0) and reserved for future needs [45]. The next 4 bits are the Data Length Code (DLC) which determine the size of the data (in bytes) being transmitted [45].
- Data: The data consists of a maximum of 8 bytes depending on the set value of the DLC in the control setup [20]. The data can send any type of information such as the temperature, speed and tire pressures.
- Cyclic Redundancy Check-(CRC): The Cyclic Redundancy Check consists of 16 bits, 15 message error correction bits and a recessive (1) delimiter bit [20]. The CRC checks if the message transmitted is the same without any corruption and corrects any data corruption [20].
- Acknowledge (ACK): The Acknowledgment bits consist of the ACK bit and a recessive (1) delimiter bit [20]. The ACK indicates an error free message has been sent [20]. Every ECU that has received an accurate message overwrites this recessive bit from the original message as a dominant bit indicating success [20]. If any of the ECUs detect an error, this bit is left as recessive indicating that there was an error and the message should be discarded and resent [20].
- End of Frame (EOF): The End of Frame consists of 7 recessive (1) bits [20].
- Inter-frame Space (IFS): The Inter-frame Space consists of 3 consecutive recessive (1) bits which separate a data frame and remote frame [20]. The proceeding bit will be regarded as the SOF bit of the next frame.

These message parameter fields allows CAN Protocol to be very flexible and have many different applications. The most important parameters from above are the ID, control, and the data fields.

One of the issues when using a single communication bus for transmitting and receiving messages is determining which ECU has control over the bus when two ECUs request bus access simultaneously. To resolve the issue, CAN Protocol implements bit-wise arbitration on the ID field of a frame to determine its priority [20]. As stated above under the Identifier parameter, the lower the ID the higher the priority, and the higher the ID the lower the priority. For an ECU to

win bit-wise arbitration, the ID's will be compared bit by bit and the dominant bit will always win the arbitration over the recessive bit [20]. The ECU with the recessive bit will forfeit the arbitration until another opportunity arises [20].

2.2 Related Work

Multiple bodies of work have adopted machine learning techniques for detecting intrusions on the CAN bus [5,26,38]. For example, Theissler [38] proposed an anomaly detection system based on multivariate time series. An ensemble anomaly detector was made comprising of two-class and one-class classifiers in order to detect both known and unknown fault types in various driving conditions. However, his approach has limitations, especially when the in-vehicle environment changes frequently; these limitations can be the continuous need for calibration and data update. Other work by Barletta et al. [5] proposed an IDS based on a combination of an unsupervised Kohonen Self-Organizing Map (SOM) network and k-means algorithm. The CAN IDs, time stamp, DLC and data field were used as features in order to identify attack messages sent on the CAN bus. Other work by Markovitz and Wool [26] proposed an anomaly detection system based on monitoring Constant fields, Multi-Value fields and Counter or Sensor fields of the CAN bus traffic. They used the Ternary Content Addressable Memory (TCAM) model to characterize those fields and build a model for the CAN bus messages based on those field types. Although they were able to achieve a low false-positive rate by evaluating their system on synthetic CAN bus traffic simulating 10 different message IDs. However, they didn't evaluate their system on actual attacks and against real CAN bus messages.

Additionally, Minawi et al. [28] proposed an IDS that utilizes machine learning and provides critical alerting features to protect vehicle operations. The CAN ID and Data field were the primary features used to determine if a message is benign or malicious. Furthermore, the Random Tree algorithm was used to achieve high accuracy in detecting DoS, impersonation, and Fuzzy injection attacks. Other work by Martinelli et al. [27] proposed an IDS by considering the eight data bytes of the CAN packet as a primary feature to determine if a message is benign or malicious. Four fuzzy algorithms of classification were used: FuzzyRoughNN, NN, DiscernibilityClassifier and FURIA. These algorithms were applied to the eight bytes features and they were able to achieve 0.85 to 1 precession. Other work by Avatefipour et al. [3] proposed an IDS for CAN bus based on the frequency of message IDs patterns that are transmitted in given normal traffic. A modified one class SVM was constructed and used based on a new meta-heuristic optimization algorithm called the Modified Bat Algorithm (MBA). Their IDS was evaluated on two datasets in the scope of CAN bus traffic anomaly detection. Although their IDS achieved a low false-positive rate. Nevertheless, it can't detect massages injection stacks. Additionally, Yang et al. [41] proposed an IDS based on tree-based machine learning algorithms. The CAN IDs and the data field were used as features to detect threats both on the CAN bus and external networks. Although their system was able to achieve high accu-

racy by testing their IDS on two data sets for both intra-vehicle and external networks. However, their IDS has a high computational cost.

Several methods were recently proposed to detect intrusions on the CAN bus based on deep learning techniques [18,19,23,25,34,35]. Such a method includes an IDS based on a deep convolutional neural network (DCNN) to protect the CAN bus of the vehicle. The DCNN learns the network traffic patterns and detects malicious traffic without hand-designed features [35]. Furthermore, work by Loukas et al. [25] proposed a cloud-based cyber-physical IDS for vehicles by using the deep learning technique. Eight features were used to detect an intrusion which are network incoming and outgoing rates, CPU utilization, the rate of the written data to the disk, the time between two consecutive encoders, accelerometer readings, power consumption and the overall current drawn by the vehicle. However, by using RNN, they were able to achieve only 79% accuracy. Other work by Hossain et al. [19] proposed a long short-term memory (LSTM) deep learning model-based on the intrusions on the CAN bus. The CAN ID, DLC and data field were used as features for in-vehicle CAN bus network attack. Other work by Seo et al. [34] proposed an IDS model for the in-vehicle network based on GAN deep learning model. A large number of CAN IDs have been encoded and random fake data in the training process have been used instead of the real attack data. Although they were able to achieve an average of 98% accuracy. Nevertheless, their model was not able to distinguish anomalous traffic caused by normal malfunctioning of electronic components from anomalous traffic caused by intentional attacks by hackers. Additionally, Hanselmann et al. [18] proposed an IDS based on a neural network architecture that is trained in an unsupervised manner. The CAN IDs and timestamp were used as features in order to detect intrusions and anomalies on the CAN bus. Although they were able to achieve high accuracy by evaluating their system on synthetic CAN bus traffic. However, they didn't evaluate their system on actual attacks and against real CAN bus messages.

Although the above solutions provide some degree of security as shown in Table 1. However, in addition to the additional resources required and complex computation costs needed, the deep learning approach is not sufficient for a vehicular network due to the limited computing power of the ECUs to procedure the complex process. Unlike prior work, we propose a lightweight IDS based on a new eight features. We find that the newly explored eight features are significant features to detect attacks on CAN bus messages with a high detection rate with minimal time without any modification in the standard procedure of the CAN protocol.

2.3 Attack Model

In this section, we first present the adversary model. Afterward, we discuss three attack scenarios that can seriously ruin in-vehicle functions: spoofing attack, fuzzy attack, and Denial-of-Service (DoS) attack.

Table 1. A comparison of the state of the art IDSs using machine learning techniques

Ref. No	Machine learning algorithms	Features
[38]	Ensemble classifier	Timestamp
[5]	SOM and k-means	CAN IDs, timestamp, DLC and data field
[26]	One-class classifier	Constant, Multi-Value and Sensor fields
[28]	Random Tree	CAN IDs and data field
[27]	FuzzyRoughNN, NN, Discernibility Classifier and FURI	Data field
[3]	A modified one class SVM	The frequency of message IDs patterns
[41]	Tree-based	CAN IDs and data field
[35]	DCNN	The network traffic patterns
[25]	RNN	Network in and out rates, CPU utilization, written data rates, time between two consecutive encoders, accelerometer readings, power consumption, the overall current drawn by vehicle
[19]	Deep Learning	CAN IDs, DLC and data field
[34]	GAN Deep Learning	CAN IDs
[18]	DNN	CAN IDs and timestamp
[1]	SVM, KNN	CAN IDs and data field
Our IDS	SVM, KNN	Graph property-based

Adversary Model. In this paper, we assume an adversary can physically or remotely compromise in-vehicle ECUs via several attack surfaces such as OBD-II, CD players, USB, Bluetooth, and cellular [8]. We consider an attacker who wants to control or disable or paralyze in-vehicle ECUs' functionality. An attacker can accomplish this by either injecting arbitrary messages repeatedly into the CAN bus or by injecting unauthenticated messages with a spoofed ID into the in-vehicle network. In this paper, we discuss three kinds of attackers who can inject malicious messages in the in-vehicle network through the CAN bus. We assume an attacker can inject malicious messages in order to control and breakdowns vehicle functionality. Accordingly, we consider three types of message injection attacks which are spoofing, fuzzy and DoS attacks.

Attack Scenarios. Based on the above described adversary model, we consider the normal CAN-bus data (attack-free) in addition to three kinds of attack scenarios which are spoofing, fuzzy and DoS as shown in Fig. 4.

Spoofing attack: This attack happens when an attacker injects a single message of randomly spoofed CAN identifier with arbitrary data. Subsequently, it

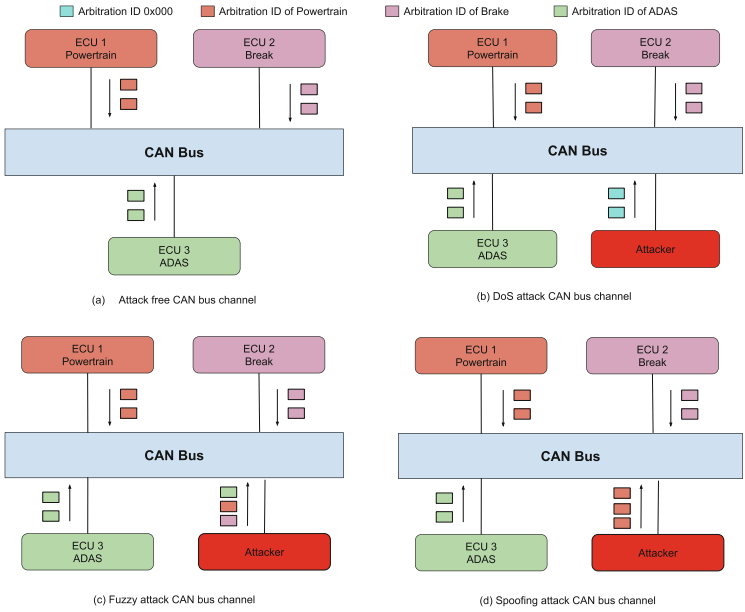

Fig. 4. Different CAN bus channel scenarios

causes unintended vehicle behaviors since all ECUs will receive that message. To exploit the spoofing attack, an attacker can inject arbitrary data into one message of the in-vehicle messages and chose the target identifier of that message to create unexpected behaviors for the vehicle. Such behaviors include turning the signal lamps light irregularly, flickering the instrument board in incalculable ways, disabling the braking system and shaking the steering wheel colossally.

Fuzzy attack: This attack occurs when an attacker injects multiple messages with arbitrary data of randomly multiple spoofed CAN identifiers, unlike the spoofing attack which occurs by injecting only a single message of randomly a single spoofed CAN identifier. As a result, all ECUs will receive various messages which cause unintended vehicle behaviors like gearshift changes automatically, disabling the braking system, instrument panel blinks in incalculable manners and the steering wheel shakes gigantically.

DoS attack: This attack happens when an attacker injects high priority of CAN messages such as the 0×000 CAN ID packet in a short cycle on the CAN bus. To exploit the spoofing attack, an attacker can easily occupy the bus by injecting the highest priority identifier of CAN messages such as 0×000 in a short cycle on the CAN bus. Subsequently, it yields latencies of other messages and causes threats in regards to availability with no reaction to the driver's commands since all ECUs share a single bus. Unlike the spoofing and fuzzy attacks, the DoS attack delays the normal messages through the occupancy of the CAN bus rather than cripple the functions of a vehicle.

3 Methodology

This section presents the proposed IDS in detail. Subsection 3.1 starts with the overview of the IDS. It is followed by a CAN bus message to graph conversion. Subsection 3.3 represents the extraction of graph properties from CAN message based graphs. And finally we conclude the section with classification of CAN bus graphs section.

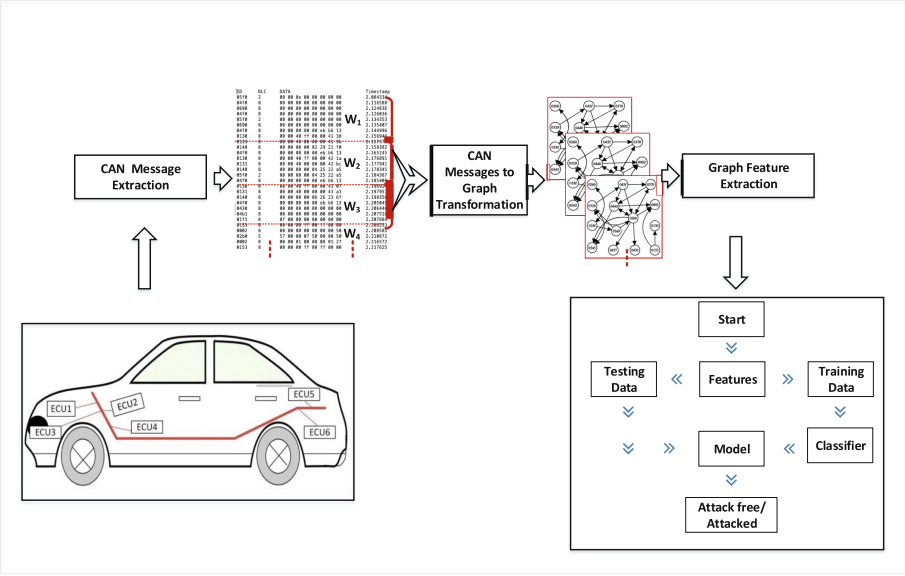

Fig. 5. Overview of the proposed IDS

3.1 Overview of the IDS

As shown in Fig. 5, the proposed IDS has three main sub-components of conversion of CAN bus messages to graph structure, extraction of graph properties from CAN message based converted graphs and classification of CAN bus graphs based on the graph features. First the CAN bus messages are converted into graph structures. In the feature extraction phase 8 features have been extracted based on the properties of the graph. Based on the feature differences between benign CAN bus graphs and malicious CAN bus graphs, 7 features were selected for classification phase. In the classification, support vector machine (SVM) [30] and K-nearest neighbor (KNN) [32] is used on the selected features based on attack free CAN bus message and attacked CAN bus message. Experimental result shows classification with SVM and KNN based on the selected features exhibits good accuracy in attack in CAN bus.

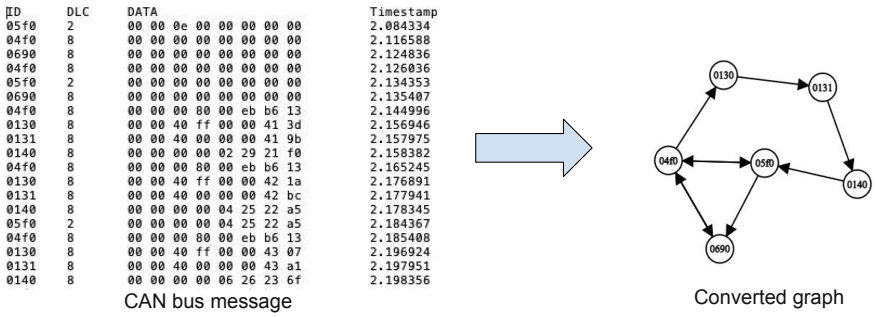

Fig. 6. CAN bus message to graph conversion

3.2 Conversion of CAN Bus Message to Graph

To extract the normal behavior of CAN bus system, author in [21], considers a window of CAN bus messages and converts it to a graph. Our proposed IDS takes these concepts and converts a chunk of CAN messages into a meaningful graph structure. To build a graph, the CAN arbitration ID of every CAN bus message is considered as a node and an edge is put between two graph nodes (arbitration IDs) if two CAN messages come sequentially on after another. Figure 6 shows how a chunk of CAN messages CAN be converted to a graph. As graphs can find meaningful hidden structure of data, so graphs converted from CAN bus data, represent the meaningful behavior of CAN bus.

3.3 Graph Properties as Features

In graph theory, the properties like number of nodes, number of edges etc. are totally dependent on graph structure. To distinguish between benign CAN bus data with malicious (DoS, fuzzy and spooning attack) data the initial plan is to choose the properties that show the characteristics of the graph. Hence, the proposed IDS extracts graph properties like number of nodes, number of edges [21], radius [9], diameter [10], density [24], reciprocity [6], average clustering coefficient [29] and assortativity coefficient [31]. Figure 7 shows the box plot for each of these graph properties mentioned above in different CAN bus attack free and attack scenarios. The plots clearly show that number of nodes is not distinguishable between benign and malicious CAN bus scenarios. Hence the proposed IDS excludes the number of nodes from classification feature list and selects the other seven differential features.

3.4 Classification Step

The classification step is dependent on the feature extraction and selection process. It takes the selected eight features as an input and classifies benign and malignant CAN bus data. Two popular machine learning algorithms i.e. support

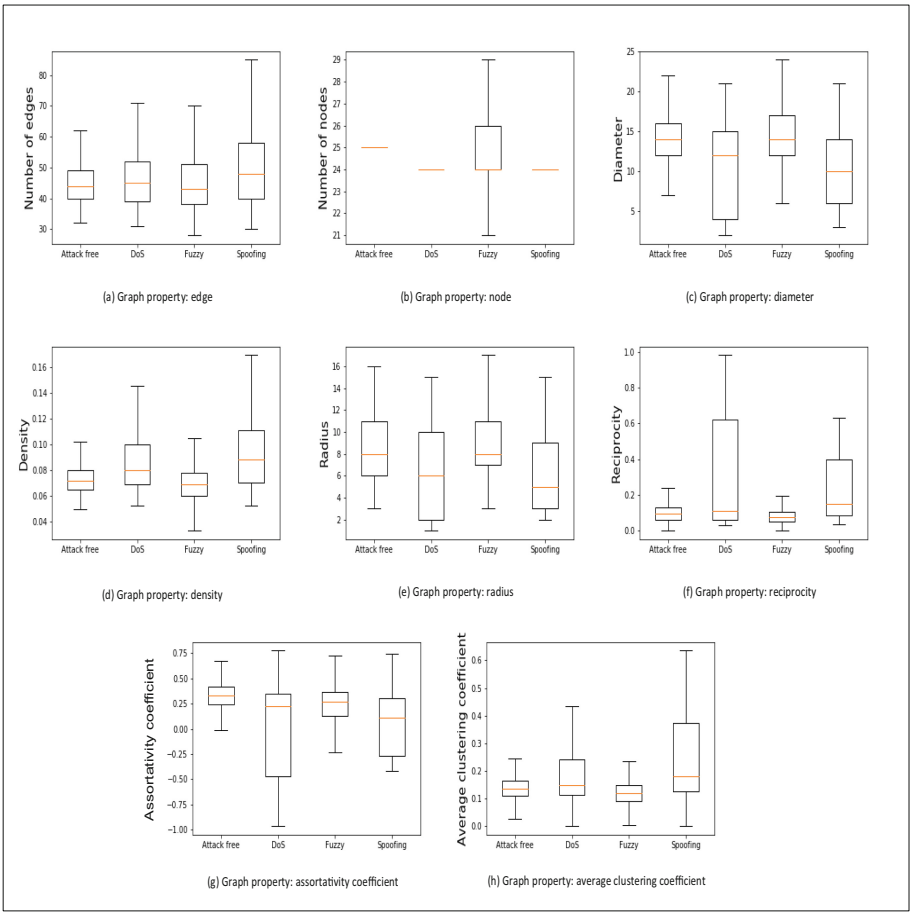

Fig. 7. Graph based CAN features

vector machines (SVM) and k-nearest neighbor (KNN) are applied to classify the performance.

- Support Vector Machine (SVM) [30]: SVM is a supervised learning algorithm used for classification and regression problems. It actually tries to find a suitable hyperplane in a finite dimension of data that clearly classifies the training data points. In the testing phase, the test instances are compared with the hyperplane to predict the appropriate category of the instance. The algorithm is used in many applications like classification of images, satellite data, categorization of text & hypertext data, handwritten recognition etc. successfully over the years [36].
- K-Nearest Neighbor (KNN) [32]: KNN is a non-parametric machine learning algorithm that can be used for both classification and regression. It does not make any assumptions on the underlying data distribution. In the training

phase it keeps the similar data near to each other and it is utilized during the testing phase. First, the distances between training data and a test instance are measured; And finally using majority voting among the k-nearest training instances, the class of the test instance is predicted.

Our assumptions and selected features exhibit excellent results in classifying attack free and attacked CAN bus messages.

4 Performance Evaluation

4.1 Description of the Dataset

To verify the effectiveness of the proposed IDS, an experiment is designed and performed on real vehicular dataset [35]. The dataset contains three kinds of CAN bus attack data along with benign CAN data. That is DoS attack, fuzzy attack and spoofing attack. Table 2 shows the summary of the dataset. To prepare the dataset for our proposed IDS, the CAN messages are converted to temporal graphs. Along the lines of the authors in [21], each graph is built using 200 CAN messages. Using the dataset, we were able to extract 5,558 DoS attack graphs, 2,802 fuzzy attack graphs and 11,263 spoofing attack graphs.

Table 2. Can traffic dataset details

Dataset	No of CAN messages	No of graphs	No of attack-free graphs	No of attack graphs
DoS attack	3,631,600	18,158	5,558	12,600
Fuzzy attack	3,053,400	15,267	2,802	12,465
Spoofing attack (RMP)	4,566,200	22,831	11,263	11,568

4.2 Validation Metrics

Each datafile of the selected dataset was fed into the classifier and was treated as a binary classification problem. The performance of the proposed IDS was validated based on precision, recall, F-1 score, accuracy and area under the receiver operating characteristic curve. Out of them precision and recall is considered to measure the quality and quantity respectively. The F-1 score is used to find how precise the classifier is. In the experiment accuracy means the percentage of correctly classified graphs. The area under the receiver operating characteristic curve is considered to measure the ability of the classifier to classify benign and malignant graphs. The equations for this performance metrics can be found here [33].

4.3 Simulation Result

The proposed IDS was designed and implemented using python programming language on a 1.8 GHz windows 10 computer system with 16 GB RAM. The selected dataset was fed into two types of machine learning classifier i.e. (i) SVM & (ii) KNN and the performance was measured based on the validation metrics discussed in Subsect. 4.2. First, 60% of the overall data is considered as the training dataset. The remaining 40% is divided into two equal halves as a validation dataset and test dataset. The reason for using a validation set is to fine tune the classifier hyper-parameters on unknown data while the classifier is fit using training data. The total separate test dataset helps to eliminate the overfitting tendency of a model. While feeding the dataset files into SVM classifier, we achieved accuracy of 99.90%, 99.93% & 96.43% for DoS, fuzzy and spoofing attack respectively. On the other hand, KNN provided accuracy of 99.86%, 99.79% & 96.55%. In order to check the robustness of the IDS, a mix attack by combining DoS, fuzzy & spoofing attack dataset is also performed. While defending the mix attack the SVM classifier achieved 97.92% accuracy while the KNN has achieved 97.99% accuracy. Table 3 shows the details about all the validation metrics discussed in Subsect. 4.2 while defending DoS, fuzzy, spoofing & combined attack.

Table 3. The results of the fuzzy and the dos attacks

Attack	Classifier	Accuracy	Precision	Recall	F-1	AUC-ROC score
DoS attack	SVM	99.90	0.9994	0.9969	0.9981	0.9969
	KNN	99.86	0.9993	0.9955	0.9977	0.9955
Fuzzy attack	SVM	99.43	0.9996	0.9952	0.9973	0.9952
	KNN	99.79	0.9989	0.9865	0.9926	0.9865
Spoofing attack	SVM	96.43	0.9761	0.9363	0.9537	0.9361
	KNN	96.55	0.9767	0.9384	0.9554	0.9385
Mix attack	SVM	97.92	0.9861	0.9609	0.9726	0.9608
	KNN	97.99	0.9895	0.9623	0.9737	0.9623

4.4 Comparison with the State of the Art

Finally, the proposed IDS is compared with one of the state of the art works [1]. The particular reason for choosing [1] for comparison is because it uses CAN bus data message frames as features to detect CAN intrusion using SVM and KNN classifiers. Therefore the effectiveness of the proposed IDS is compared with the state of the art by considering accuracy on real vehicular CAN bus data. While tested on 20% data, [1] achieved accuracy of 97.40% & 96.50% while using SVM & KNN. On the other hand the SVM & KNN based on our proposed graph based features achieved accuracy of 97.92% & 97.99% respectively. Figure 8 shows the

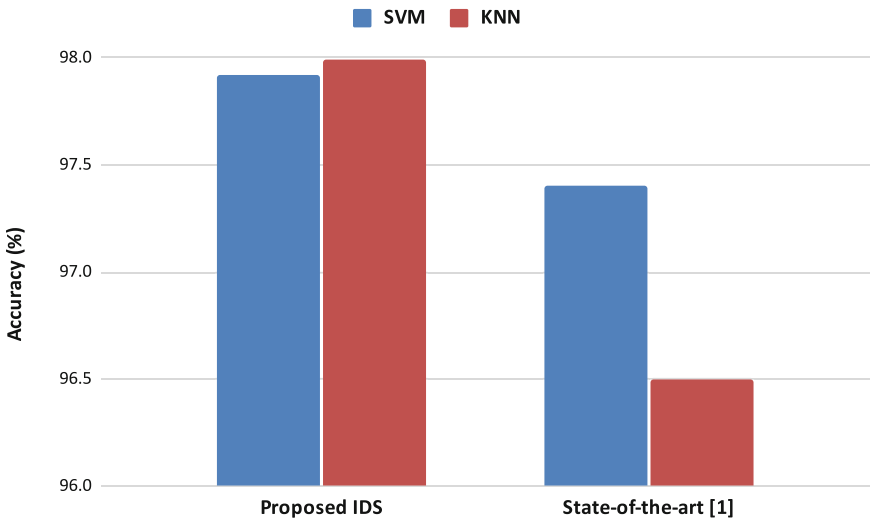

Fig. 8. Comparison with the state-of-the-art [1]

comparison between the state of the art [1] and proposed IDS. The figure clearly demonstrates that the proposed IDS has achieved better accuracy than the state of the art for both SVM & KNN.

5 Conclusion and Future Work

In the modern transportation system, more and more connectivity is added in vehicles with the outside world. More connectivity adds more threat surfaces in vehicles which poses serious threat to the safety of passengers and security of vehicles. In order to make the in-vehicle network secure a strong intrusion detection system is required. In this paper, a CAN intrusion detection system is proposed that uses graph based features to detect CAN attack. This novel and pragmatic approach uses graph based features to classify authentic and malicious CAN messages for in-vehicle communication. The experimental results showed that using graph-based features, an accuracy of 97.92% & 97.99% was achieved using SVM & KNN algorithms respectively. In future, we would like to consider other graph properties to detect intrusion for in-vehicle communication. In addition, we will apply different machine learning algorithms in place of the SVM & KNN to see the robustness of the selected features.

Acknowledgment. The authors extend their appreciation to the Deputyship for Research & Innovation, Ministry of Education in Saudi Arabia for supporting this work through the project # DRI-KSU-934. This research is also partly supported by National Science Foundation (NSF) under the award # 2035770.

References

1. Alshammari, A., Zohdy, M., Debnath, D., Corser, G.: Classification approach for intrusion detection in vehicle systems. Wirel. Eng. Technol. **9**, 79–94 (2018)
2. Alves, M., Pereira, M., Ramos, H.: CAN protocol: a laboratory prototype for field bus applications (2009)
3. Avatefipour, O., et al.: An intelligent secured framework for cyberattack detection in electric vehicles' CAN bus using machine learning. IEEE Access **7**, 127580–127592 (2019)
4. Avatefipour, O., Hafeez, A., Tayyab, M., Malik, H.: Linking received packet to the transmitter through physical-fingerprinting of controller area network (2017)
5. Barletta, V., Caivano, D., Nannavecchia, A., Scalera, M.: Intrusion detection for in-vehicle communication networks: an unsupervised Kohonen SOM approach. Future Internet **12**, 119 (2020)
6. Berg, J., Dickhaut, J., Mccabe, K.: Trust, reciprocity, and social history. Games Econ. Behav. **10**, 122–142 (1995)
7. Broster, I., Burns, A.: An analysable bus-guardian for event-triggered communication (2003)
8. Checkoway, S., et al.: Comprehensive experimental analyses of automotive attack surfaces (2011)
9. Ducoffe, G, Dragan, F.: A story of diameter, radius, and (almost) helly property. Networks (2020)
10. Eppstein, D.: Diameter and treewidth in minor-closed graph families. Algorithmica **27**, 275–291 (2000)
11. Hafeez, A.: A robust, reliable and deployable framework for In-vehicle security (2020)
12. Hafeez, A., Malik, H., Avatefipour, O., Rongali, P., Zehra, S.: Comparative study of can-bus and flexray protocols for in-vehicle communication (2017)
13. Hafeez, A., Ponnapali, S., Malik, H.: Exploiting channel distortion for transmitter identification for in-vehicle network security. Sae Int. J. Transp. Cybersecurity Priv. **3** (2019)
14. Hafeez, A., Tayyab, M., Zolo, C., Awad, S.: Finger printing of engine control units by using frequency response for secure in-vehicle communication (2018)
15. Hafeez, A., Topolovec, K., Awad, S.: ECU fingerprinting through parametric signal modeling and artificial neural networks for in-vehicle security against spoofing attacks (2019)
16. Hafeez, A., Topolovec, K., Zolo, C., Sarwar, W.: State of the Art Survey on Comparison of CAN. FlexRay, LIN Protocol and Simulation of LIN Protocol (2020)
17. Hafeez, A., Rehman, K., Malik, H.: State of the Art Survey on Comparison of Physical Fingerprinting-Based Intrusion Detection Techniques for In-Vehicle Security (2020)
18. Hanselmann, M., Strauss, T., Dormann, K., Ulmer, H.: CANet: an unsupervised intrusion detection system for high dimensional CAN bus data. Ieee Access **8**, 58194–58205 (2020)
19. Hossain, M., Inoue, H., Ochiai, H., Fall, D., Kadobayashi, Y.: LSTM-based intrusion detection system for in-vehicle can bus communications. IEEE Access **8**, 185489–185502 (2020)
20. Hpl, S.: Introduction to the controller area network (CAN). Appl. Rep. Sloa1011-17 (2002)

21. Islam, R., Refat, R., Yerram, S., Malik, H.: Graph-Based Intrusion Detection System for Controller Area Networks. IEEE Trans. Intell. Transp. Syst. (2020)
22. Jung, J., Park, K., Cha, J.-S.: Implementation of a network-based distributed system using the CAN protocol. In: Khosla, R., Howlett, R.J., Jain, L.C. (eds.) KES 2005. LNCS (LNAI), vol. 3681, pp. 1104–1110. Springer, Heidelberg (2005). https://doi.org/10.1007/11552413_157
23. Kang, M., Kang, J.: Intrusion detection system using deep neural network for in-vehicle network security. Plos One **11**, e0155781 (2016)
24. Kowalik, Ł: Approximation scheme for lowest outdegree orientation and graph density measures. In: Asano, T. (ed.) ISAAC 2006. LNCS, vol. 4288, pp. 557–566. Springer, Heidelberg (2006). https://doi.org/10.1007/11940128_56
25. Loukas, G., Vuong, T., Heartfield, R., Sakellari, G., Yoon, Y., Gan, D.: Cloud-based cyber-physical intrusion detection for vehicles using deep learning. IEEE Access **6**, 3491–3508 (2017)
26. Markovitz, M., Wool, A.: Field classification, modeling and anomaly detection in unknown CAN bus networks. Veh. Commun. **9**, 43–52 (2017)
27. Martinelli, F., Mercaldo, F., Nardone, V., Santone, A.: Car hacking identification through fuzzy logic algorithms (2017)
28. Minawi, O., Whelan, J., Almehmadi, A., El-khatib, K.: Machine learning-based intrusion detection system for controller area networks (2020)
29. Newman, M.: Random graphs with clustering. Phys. Rev. Lett. **103**, 058701 (2009)
30. Noble, W.: What is a support vector machine? Nature Biotechnol. **24**, 1565–1567 (2006)
31. Noldus, R., Vanmieghem, P.: Assortativity in complex networks. J. Complex Netw. **3**, 507–542 (2015)
32. Peterson, L.: K-nearest neighbor. Scholarpedia **4**, 1883 (2009)
33. Salo, F., Injadat, M., Nassif, A., Shami, A., Essex, A.: Data mining techniques in intrusion detection systems: a systematic literature review. IEEE Access **6**, 56046–56058 (2018)
34. Seo, E., Song, H., Kim, H.: Gids: Gan based intrusion detection system for in-vehicle network (2018)
35. Song, H., Woo, J., Kim, H.: In-vehicle network intrusion detection using deep convolutional neural network. Veh. Commun. **21**, 100198 (2020)
36. Suthaharan, S.: Machine Learning Models and Algorithms for Big Data Classification. ISIS, vol. 36. Springer, Boston (2016). https://doi.org/10.1007/978-1-4899-7641-3
37. Tayyab, M., Hafeez, A., Malik, H.: Spoofing attack on clock based intrusion detection system in controller area networks (2018)
38. Theissler, A.: Detecting known and unknown faults in automotive systems using ensemble-based anomaly detection. Knowl.-Based Syst. **123**, 163–173 (2017)
39. Ueda, H., Kurachi, R., Takada, H., Mizutani, T., Inoue, M., Horihata, S.: Security authentication system for in-vehicle network. SEI Tech. Rev. **81**, 5–9 (2015)
40. Wu, W., et al.: A survey of intrusion detection for in-vehicle networks. IEEE Trans. Intell. Transp. Syst. **21**, 919–933 (2019)
41. Yang, L., Moubayed, A., Hamieh, I., Shami, A.: Tree-based intelligent intrusion detection system in internet of vehicles (2019)
42. Charette, R.: This Car Runs on Code-IEEE Spectrum. IEEE Spectr. Technol. Engineering, And Science News https://spectrumieee.org/green-tech/advanced-cars/this-car-runs-on-code (2009)
43. Lin, C.: Sangiovanni-vincentelli, A.: Cyber-security for the controller area network (CAN) communication protocol (2012)

44. Groza, B., Murvay, S.: Efficient protocols for secure broadcast in controller area networks. IEEE Trans. Ind. Inf. **9**, 2034–2042 (2013)
45. Dinatale, M., Zeng, H., Giusto, P., Ghosal, A.: Understanding and using the Controller Area Network Communication Protocol: Theory and Practice. Springer, New York (2012) https://doi.org/10.1007/978-1-4614-0314-2