IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 23, 2021, accepted December 13, 2021, date of publication December 16, 2021,
date of current version December 27, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3136147

An Efficient Intrusion Prevention System for CAN: Hindering Cyber-Attacks With a Low-Cost Platform

PAULO FREITAS DE ARAUJO-FILHO[1,2], (Graduate Student Member, IEEE), ANTÔNIO J. PINHEIRO[2,3], GEORGES KADDOUM[1], (Senior Member, IEEE), DIVANILSON R. CAMPELO[2], (Member, IEEE), AND FABIO L. SOARES[4]

[1]Electrical Engineering Department, École de Technologie Supérieure (ÉTS), Université du Québec, Montreal, QC H3C 1K3, Canada
[2]Centro de Informática, Universidade Federal de Pernambuco (UFPE), Recife 50740-560, Brazil
[3]Centro de Estudos e Sistemas Avançados do Recife (CESAR), Recife 50030-390, Brazil
[4]Intrepid Control Systems GmbH, 76131 Karlsruhe, Germany

Corresponding author: Paulo Freitas de Araujo-Filho (paulo.freitas-de-araujo-filho.1@ens.etsmtl.ca)

This work was supported in part by the Tier 2 Canada Research Chair on the Next Generations of Wireless IoT Networks, in part by the Fonds de Recherche du Québec–Nature et Technologies (FRQNT) B2X Scholarship, in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), in part by the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), and in part by the Fundação de Amparo à Ciência e Tecnologia de Pernambuco (FACEPE).

ABSTRACT The controller area network (CAN), which is still today the most used in-vehicle network, does not provide any security or authentication mechanism by design. Since current vehicles, which have numerous connectivity technologies, such as Bluetooth, Wi-Fi, and cellular radio, can be easily accessed from the exterior world, they can be easy targets of cyber-attacks. It is therefore urgently necessary to enhance vehicle security by detecting and stopping cyber-attacks. In this paper, we propose a novel unsupervised intrusion prevention system (IPS) for automotive CANs that detects and hinders attacks without modifying the architecture of the electronic control units (ECUs) or requiring information that is restricted to car manufacturers. We compare two machine learning algorithms' ability to detect fuzzing and spoofing attacks, and evaluate which of them is most accurate with the fewest number of data bytes. The fewer data bytes required, the sooner detection can start and the sooner attacking frames can be detected. Experiment results show that our proposed detection mechanism achieves accuracy higher than 99%, F1-scores higher than 97%, and detection times shorter than 80μs for the types of attacks considered. Moreover, when compared to four state-of-the-art intrusion detection systems, it is the only solution that is capable of discarding attacking frames before damage occurs while being deployed on inexpensive Raspberry Pi. Such an inexpensive deployment is particularly desirable, as cost is one of the automotive industry's primary concerns.

INDEX TERMS Intrusion detection system (IDS), intrusion prevention system (IPS), machine learning, controller area network (CAN).

I. INTRODUCTION

Today's vehicles are equipped with different connectivity technologies, such as Bluetooth, Wi-Fi, and cellular radio, that make a plethora of automotive applications feasible. However, such interfaces expand vehicles' attack surface so they are more exposed to security vulnerabilities that may compromise their operation and put drivers and passengers at risk [1], [2]. In-vehicle networks, which connect several electronic control units (ECUs), have recently been the target of cyber-attacks [1], [3]. The controller area network (CAN), the most commonly used in-vehicle network, carries signals from critical vehicle systems, such as the braking and steering systems. Thus, the need for timely detection of potential cyber-security incidents in cars and rapid response to them has been pushed by the industry and government [4]–[6].

Since the CAN does not have any authentication or security mechanism by design, the automotive industry employs several layers of security in vehicles. These include protection of individual ECU software and integrity, protection of critical signals and messages, isolation of automotive domains by a gateway, firewall and security solutions for external

The associate editor coordinating the review of this manuscript and approving it for publication was Mehdi Sookhak.

interfaces, and intrusion detection systems (IDSs) / intrusion prevention systems (IPSs) [2]. IDSs and IPSs detect cyber-attacks that other security mechanisms were not able to prevent. IPSs also hinder detected cyber-attacks. They are both particularly valuable defense mechanisms for CANs due to their low impact on system performance and the fact that they do not require any modification to the CAN protocol [7].

Unsupervised anomaly-based IDSs learn a normal behavior profile and detect cyber-attacks by measuring deviations from this profile [8], [9]. In contrast to other approaches that usually present fewer false alarms, such as signature-based techniques, unsupervised anomaly-based IDSs do not depend on a database of known attacks and are able to detect unknown attacks. Since new attacks are constantly being launched and obtaining labeled attack data can be very time-consuming and is sometimes not even possible, such IDSs are deemed best for securing networks and systems [9]–[11] and are increasingly being adopted by the industry [12]–[14].

However, existing IDS and IPS solutions present have limitations, and there are still crucial challenges to be addressed. Some rely on information that is restricted to car manufacturers or require modifications to ECUs. Most are ineffective against attacks that use legitimate frame identifiers, i.e., attacks that modify the content of a CAN message without affecting its identifier or the frequency with which the identifier is found in the network [15]. Others propose the use of machine learning techniques to overcome this limitation but require considerable computing power and expensive hardware to be deployed [16]. If less sophisticated hardware is used, the detection time requirement for hindering cyber-attacks is not met; intrusions are detected after their transmission is completed, and damage cannot be avoided. Therefore, even though CANs have been widely used for decades, existing cyber-attack intrusion detection and prevention systems cannot hinder attacking malicious frames using inexpensive hardware, such as Raspberry Pi. Furthermore, the current literature lacks a proper analysis of the time requirement to hinder intrusions and an IDS that can meet that requirement using inexpensive hardware.

A. CONTRIBUTIONS

In this paper, we propose a novel unsupervised IPS for detecting and discarding known and unknown attacks in CANs without relying on labeled attack data or on what each CAN message means, i.e., no restricted information from car manufacturers is required. In addition, our proposed system does not require modifications to ECUs or to the in-vehicle network architecture. Instead, it is deployed on additional low-cost hardware, such as Raspberry Pi, that is connected to the CAN bus. Although that additional device is inexpensive, it does not impact the performance of ECUs or in-vehicle networks and can detect cyber-attacks fast enough to hinder them and prevent damage. In order to achieve low detection times even when using less-sophisticated inexpensive hardware, our proposed detection mechanism uses only the first N bytes of the CAN frames' payload. If not all eight data bytes in a payload are used, detection can start sooner and be carried out before the transmission of the evaluated frames is completed, which makes it possible to hinder attacks.

Since using fewer data bytes may degrade detection performance, we evaluate two machine learning algorithms' ability to detect attacks when using different numbers of data bytes. We consider the same three types of attacks (fuzzing, drive gear spoofing, and RPM gauge spoofing) that are considered by the state-of-the-art IDSs we use as a baseline to compare our results. Our proposed IPS then uses the algorithm that is shown to need the fewest data bytes to detect cyber-attacks, so that detection finishes sooner and there is enough time to stop attacks before damage is caused. The first algorithm evaluated is the one-class support vector machine (OCSVM), which is firmly based on mathematical and optimization concepts. This algorithm finds a function that is positive for regions with a high density of points and negative for those with a low density of points. According to [17], the OCSVM has very fast inference, which makes it attractive for latency constrained problems like detecting cyber-attacks in CANs. The second algorithm considered is the isolation forest (iForest) [18], which is an ensemble of isolation trees that provides a measure of normality for observations [19]. It is a powerful classification algorithm that isolates anomalies from the rest of the observations by recursively and randomly partitioning data. It is specially designed for detecting anomalies and has low linear time complexity [18], such that it is also very attractive for detecting attacks in CANs. The OCSVM and iForest algorithms are usually used as benchmarks in anomaly detection research and have been suggested for detecting attacks in automotive CANs due to their performance and short detection times [18], [20], [21]. Finally, our proposed system is shown to be capable of detecting and discarding attacking frames before damage occurs and outperforming four state-of-the-art IDSs used as a baseline. In a nutshell, the main contributions of our work are:

1) A performance evaluation of the OCSVM and iForest algorithms' ability to detect cyber-attacks using different numbers of CAN frame data bytes.
2) A novel unsupervised IPS that can be deployed on inexpensive hardware, such as Raspberry Pi, and still satisfy the time requirement for discarding attacking frames before damage is caused.

B. ORGANIZATION

The remainder of this paper is organized as follows. Section II introduces related works. Section III introduces the threat model considered in this work. Section IV presents our proposed architecture by describing the system model, training procedure, and detection strategy. Section V explains the experiments conducted. In Section VI, we present and discuss the results. Finally, Section VII concludes the paper.

II. RELATED WORKS

The work in [22] proposes an IPS for CANs based on the assumption that it will work for systems that are able to detect attacks before their transmission is completed. It considers only two detection techniques that are based on the frame's identifiers and does not analyze the time requirement for hindering attacks. Moreover, it requires changes to the ECUs. The authors of [23] propose an IPS that assumes each message identifier can be sent by only one node. Then, each node monitors the bus and sends error frames to overwrite unauthorized transmissions. The work in [24] proposes a lightweight CAN authentication approach that overwrites malicious messages by sending error frames. However, the works in [23] and [24] also require that all ECUs be modified, and hence impose a significant burden on existing systems. The work in [25] detects when nodes have been disconnected from the bus by exploiting the CAN's fault confinement mechanism. However, it prevents only attacks that disconnect CAN nodes. The authors of [15] propose a defense method for CANs that coordinates all CAN nodes to shift a legitimate identifier once it has been used in an attack. However, it cannot stop ongoing attacks, only prevent future ones. Moreover, it requires that all ECUs be modified to comply with its mechanism. In contrast, the technique we propose does not require modifying ECUs and discards malicious frames during transmission, thereby stopping ongoing attacks.

Most of the other works on protecting CANs focus on detecting intrusions and also present several limitations. The work in [26] introduces a specification-based IDS that is based on the timing model of CAN messages. The work in [27] detects injected CAN messages using the cumulative sum change-point algorithm to monitor changes in the frequency of messages. The authors of [28] propose an IDS that detects behavior changes in CAN transmissions using the wavelet transform. The authors of [7] propose a lightweight IDS that analyzes the time intervals between CAN frames that have the same identifier. In [29], an IDS method that analyzes the sequence of CAN frame identifiers is presented. Similarly, the work in [30] proposes an IDS that uses generative pretrained transformer models to detect attacks based on the sequence of CAN frame identifiers. The works in [31] and [32] fingerprint ECUs by exploiting the intervals of periodic messages and the ECUs' clock offset, respectively. However, the works in [7] and [26]–[32] all assume that the adversary injects messages but does not modify regular messages. They are then ineffective against attacks that modify the frame's payload and preserve the messages' identifier and frequency.

The work in [33] presents the throughput method, which is based on the computation of statistical metrics and tests. In [34], fuzzy algorithms are proposed to detect injection attacks on the CAN protocol. However, these two works incur significant error rates. The work in [35] proposes a graph-based IDS that models the sequence of exchanged CAN messages. However, it cannot detect attacks by examining isolated frames and also has significant error rates for fuzzing attacks. In [16], a deep neural network is used to identify normal and attacking frames in a CAN bus with the frames' data bytes as features. The work in [36] presents a hybrid anomaly detection system based on the replicator neural network algorithm. In [20], a hybrid anomaly detection system for ECUs is introduced by combining an efficient specification-based system with machine learning techniques. However, none of these works propose mechanisms to hinder intrusions. Moreover, [16] and [36] are computationally costly and require expensive hardware to be deployed, while [20] relies on confidential specifications that are restricted to car manufacturers.

The authors of [37] propose a deep learning-based IDS to detect cyber-attacks in CANs. It relies on long short-term memory (LSTM) autoencoders and achieves 99% accuracy at detecting attacks. However, LSTM networks usually have long detection times, since they do not permit the parallelization of computations. The authors of [38] propose GIDS, an unsupervised IDS that uses generative adversarial networks (GANs). It models the system and detects attacks by measuring deviations from a learned normal behavior. Although it achieves accuracy higher than 96% at detecting cyber-attacks in CANs, its long detection time may thwart the hindering of attacks. The work in [39] proposes an IDS that is based on a deep convolutional neural network (DCNN) that learns the network traffic pattern. The DCNN achieves accuracy higher than 99% and outperforms several other machine learning techniques, such as LSTM networks, when it comes to detecting attacks in CANs. However, both DCNN and LSTM networks have long detection times even when using sophisticated GPU hardware, as it is shown in [39]. Their computational complexity per layer is $\mathcal{O}(knd^2)$ and $\mathcal{O}(nd^2)$, respectively, where n is the sequence length, d is the representation dimension, and k is the kernel size of convolutions. Finally, the work in [40] proposes a multi-tiered hybrid IDS (MTH-IDS) that combines signature and anomaly-based detection and also achieves accuracy higher than 99%. We compare the results of our work to those of GIDS [38], DCNN [39], an LSTM-based IDS [39], and MTH-IDS [40]. Table 1 summarizes our related works.

III. CAN VULNERABILITIES AND THREAT MODEL

In this section, we present the vulnerabilities of CANs and the threat model considered in this paper. Then, we discuss how to discard malicious frames and the time requirement for stopping cyber-attacks before damage occurs.

A. CONTROLLER AREA NETWORKS

Although the CAN protocol is considered a legacy technology and there are other network protocols that offer higher throughput, such as media oriented systems transport (MOST) and automotive Ethernet, it is still the most used protocol in today's cars [26], [37], [39]. Its robustness and low complexity allow car manufacturers to still rely on it for many modern applications, so CANs are expected to continue being widely used in vehicles. Besides, the CAN is

TABLE 1. Related works.

Reference	Method	Detection Attributes			Prevention Attributes	
		Requires Labeled Attack Data	Detects Modified Payloads	Requires Restricted Information	Modifies ECUs	Satisfies Time Requirement to Discard Malicious Frames using Inexpensive Hardware
[22]	Transmits error frames and reboots ECUs when an associated IDS detects attacks	N.A.	N.A.	N.A.	Yes	N.A.
[23]	Assumes that message identifiers are sent only by a single node and overwrites unauthorized messages	No	No	No	Yes	Yes
[24]	Introduces message authentication codes	No	Yes	No	Yes	Yes
[25]	Detects nodes disconnected from the bus by exploiting the CAN's fault confinement mechanism	No	No	No	No	Only after it detects disconnected ECUs
[15]	Changes CAN message identifiers after they have been detected in attacks by an associated IDS	N.A.	N.A.	N.A.	Yes	No
[26]	Models the timing of CAN messages to detect deviations	No	No	No	No	N.A.
[27]	Monitors changes in the frequency of messages using the cumulative sum change-point algorithm	No	No	No	No	No
[28]	Detects behavior changes in CAN transmissions using the wavelet transform	No	No	No	No	No
[7]	Analyzes the time intervals between CAN frames with the same identifier	No	No	No	No	No
[29]	Analyzes the sequence of frame identifiers	No	No	No	No	N.A.
[30]	Detects attacks based on the sequence of CAN frame identifiers using generative pretrained transformers	No	No	No	No	N.A.
[31]	Fingerprints ECUs by exploiting the intervals of periodic messages	No	No	No	No	N.A.
[32]	Fingerprints ECUs by exploiting their clock offset	No	No	No	No	No
[33]	Computes statistical metrics on message timing intervals over a sliding window	No	No	No	No	N.A.
[34]	Detects attacks using fuzzy algorithms	Yes	Yes	No	No	N.A.
[35]	Models the sequence of exchanged CAN messages using graph theory	No	No	No	No	No
[16]	Detects attacks using deep neural networks	Yes	Yes	No	No	No
[36]	Detects attacks using replicator neural networks	Yes	Yes	Yes	No	No
[20]	Combines a specification-based system with machine learning techniques	No	Yes	Yes	No	N.A.
[37]	Detects attacks using LSTM autoencoders	No	Yes	No	No	No
[38]	Detects attacks using GANs	No	Yes	No	No	No
[39]	Detects attacks using deep neural networks	Yes	Yes	No	No	No
[40]	Combines signature and anomaly-based detection mechanisms	Yes	Yes	No	No	No
Our Proposed IPS	**Proposes an unsupervised anomaly-based IDS using the isolation forest algorithm**	**No**	**Yes**	**No**	**No**	**Yes**

a well-known inexpensive technology that has a high enough data rate to transmit most vehicle signals [39]. It is compliant and perfectly interoperable with the CAN flexible data rate (CAN-FD), which enhances CANs and allows larger payloads to be transmitted with higher data rates [41]. Finally, the adoption of other in-vehicle networks for infotainment applications and autonomous driving, for example, does not mean CANs will be completely replaced. In fact, different in-vehicle networks already coexist and are connected through gateways [42].

The CAN protocol is a message-oriented broadcast protocol in which frames can transmit up to eight data bytes in their payload with bit rates of up to 1 Mbps [43]. Frames are transmitted and received depending on their message identifiers, which are used by a vehicle's ECUs to recognize whether they should process or ignore a message. These identifiers are also used in the CAN arbitration mechanism to determine which message should be prioritized when multiple ECUs want to transmit at the same time. Essentially, the lower a message's identifier, the higher its priority. Thus, when multiple ECUs try to transmit at the same time, the message with the lowest identifier is transmitted and the others are discarded.

The CAN was designed when vehicles were isolated environments without any external connectivity, such that there was no concern for security. It therefore does not have any authentication or encryption mechanism by design. Once access to the CAN bus is acquired, attackers can monitor all transmitted messages as well as corrupt frames and inject malicious ones. For instance, attackers can exploit the network's arbitration mechanism to continuously send messages with low identifiers, such that they always have priority and gain access to the bus. As a result, legitimate messages are not transmitted, as in a denial-of-service (DoS) attack. Fortunately, since this type of attack significantly affects the frequency and periodicity of CAN messages, it can be easily detected by exiting IDSs [7], [29], [33]. On the other hand,

other types of attacks, such as fuzzing and spoofing attacks, are much more challenging to detect.

B. THREAT MODEL

Due to their lack of security in vehicles and their widespread and ever-increasing connectivity capabilities, automotive CANs can be accessed through different types of communication interfaces [3], [37]. Attackers can physically access CANs by connecting to a vehicle's On-Board Diagnostics 2 (OBD-II) port, which allows dealerships and repair shops to extract information from a vehicle. They can also remotely access CANs by exploiting vulnerabilities of wireless interfaces, e.g., Bluetooth and cellular networks, which are used for telematics applications and over-the-air (OTA) ECU updates. For instance, in [44], control over a 2014 Jeep Cherokee CAN was gained by remotely exploiting its U-connect communication system. Similarly, in [45] and [46], Wi-Fi was exploited to reprogram a Tesla's ECUs.

In our work, we assume that attackers can gain access to vehicles' CANs through physical or remote attack surfaces. Moreover, once such access is obtained, we consider that they can sniff messages, learn the legitimate frame identifiers and transmission patterns, and acquire full knowledge about how CAN messages control ECUs' functionalities. Furthermore, we assume that attackers can inject and modify CAN messages. Injecting a message corresponds to transmitting a CAN frame. Since a CAN does not authenticate by design, an injected frame is treated as legitimate by the receiving nodes. On the other hand, modifying a message corresponds to corrupting it, i.e., overwriting bits of a legitimate CAN frame being transmitted. Since the CAN does not ensure integrity by design, bits of a legitimate frame may be overwritten as long as the frame's cyclic redundancy check field is modified accordingly.

Attackers can launch different types of cyber-attacks by injecting and manipulating CAN frames. We consider in our work fuzzing and spoofing attacks, since they have been shown to cause a lot of damage by taking control of and disabling ECUs [31], [38], [39], [47]. For instance, the authors of [47] were able to use fuzzing packets to unlock the doors, adjust interior and exterior lighting levels, and disable the key lock relay to lock the key in the ignition, among other things. Similarly, the authors of [31] manipulated and impaired vehicle functions by injecting arbitrary messages with a spoofed ID into the in-vehicle network. Fuzzing attacks inject CAN frames with random values for the data bytes and frame identifier to trigger hidden functionalities or generate errors in the devices connected to the CAN bus. Spoofing attacks, on the other hand, involve attackers injecting or corrupting frames with a known functionality to trick devices to act in a certain way, e.g., messages with the CAN identifier related to the drive gear are injected or modified in order to control it. Our proposed IPS aims to mitigate fuzzing and spoofing attacks by detecting and discarding malicious CAN frames that were injected or modified in automotive CANs.

IV. PROPOSED ARCHITECTURE

The most direct approach for discarding CAN frames is to interrupt their transmission by forcing an error condition that violates the CAN protocol. For instance, the CAN protocol has a bit stuffing mechanism to synchronize all nodes in the bus [43]. This mechanism inserts a bit 0 after the transmission of five consecutive bits 1 and inserts a bit 1 after five consecutive bits 0. Thus, if six consecutive bits are transmitted with the same logic level, the CAN protocol has been violated. When this occurs, all nodes in the bus discard the frame being transmitted and transmit an error frame, after which a new frame can be transmitted.

Rather than being a security mechanism, error frames exist in the CAN protocol to prevent errors. Several works have, however, used them to discard malicious frames, demonstrating it to be an effective approach [22]–[24]. Thus, malicious CAN frames can be discarded by forcing an error condition, such that all nodes in the bus transmit an error frame. Those error frames overlap and contain from 14 to 20 bits. Thus, at most 20 extra bits are transmitted upon discarding a malicious frame. It could be even less, since the malicious frame's transmission is interrupted before its last bits are sent. In our work, we discard malicious frames by forcing a logic level 0 during the transmission of six consecutive bits, which violates the bit stuffing mechanism. In contrast to [22]–[24], we evaluate whether an error frame can be sent soon enough to discard an attacking malicious frame and not require modifications to ECUs.

Due to the broadcast nature of the CAN, our proposed system is deployed on a single additional device that is connected to the CAN bus under surveillance. It probes the *CAN High* and *CAN Low* signals, to which all CAN nodes are connected, decodes the frames being transmitted, and detects cyber-attacks using the frames' data bytes. When an attack is detected, our system transmits six consecutive bits 0 to the bus to force a logic level 0 in the *CAN High* and *CAN Low* signals so that all ECUs discard that malicious frame. Since our system essentially monitors the bus and only transmits bits to discard malicious frames, it does not modify ECUs or impact communication between them.

Our architecture consists of a CAN transceiver, a CAN controller, and a detection agent. The CAN transceiver converts voltage to logic levels to receive and transmit bits to the bus through the *CAN High* and *CAN Low* signals. It is connected to the CAN controller through the *CAN RX* and *CAN TX* signals, which receive and forward bits. Most CAN transceivers disable themselves following many consecutive attempts to transmit dominant bits to prevent data corruption. However, most of them, such as the Microchip MCP2551 [48], allow up to 20 consecutive dominant bits to be transmitted. Therefore, since our IPS needs to send only 6 consecutive dominant bits, it can use a standard CAN transceiver. The CAN controller decodes frames with the bits it receives from the transceiver, such that the data bytes are acquired. Finally, the detection agent deploys the machine learning algorithm used to detect cyber-attacks. It receives

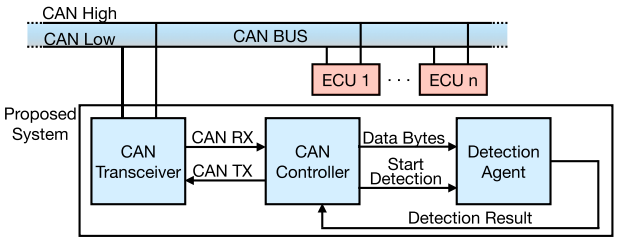


FIGURE 1. Proposed IPS architecture.

a *Start Detection* signal from the CAN controller that triggers the start of detection once all bytes used have been received. When an attack is detected, the detection agent instructs the CAN controller to transmit six consecutive bits 0 through the CAN transceiver so that the malicious frame is discarded. Figure 1 shows the architecture of our proposed IPS. Algorithm 1 shows the algorithm of our proposed IPS.

Algorithm 1: Proposed IPS

1: Train the detection agent's machine learning model
2: **while** The CAN transceiver reads the bus voltages and transmits bits to the CAN controller **do**
3: The CAN controller decodes the CAN frames
4: The CAN controller sends the incoming frames' data bytes to the detection agent
5: The detection agent applies its machine learning model to the data bytes used as features and decides whether the frame being transmitted is malicious
6: **if** The frame is considered malicious **then**
7: The detection agent commands the CAN controller and CAN transceiver to transmit six consecutive bits 0
8: **end if**
9: **end while**

A. TIME REQUIREMENT FOR PREVENTING ATTACKS

In order to stop cyber-attacks, malicious frames must be detected and discarded before their transmission is completed. Since malicious frames are discarded by transmitting six consecutive bits 0, they must be detected before their last six bits are transmitted, i.e., by the first bit of the end-of-frame (EOF) field, as depicted in Figure 2. Otherwise, the error condition would occur after the malicious frame transmission is completed and the ECUs would process attacking frames and follow their instructions that compromise the vehicle's operation.

We define the time available to detect and discard a malicious frame as

$$T_{\text{Available}} = (L - K)T_{\text{bit}}, \tag{1}$$

where T_{bit} is the transmission time of one bit, L is the number of bits in the frame, and K is the number of bits already

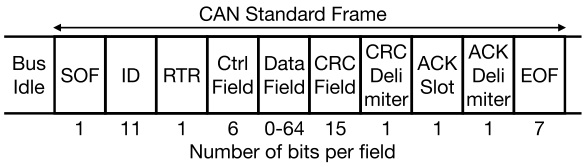


FIGURE 2. Fields and number of bits per field of standard CAN frames.

transmitted before detection starts. T_{bit} is defined by the CAN controller through the configuration of bit timing parameters that specify the CAN baud rate. Most car manufacturers operate with 500 Kbps CANs, which means that bits are transmitted every $2\mu s$, i.e., $T_{\text{bit}} = 2\mu s$. T_{bit} is a standard configuration that works with any CAN controller and does not depend on the hardware used to detect attacks.

On the other hand, we define the processing time of our IPS as

$$T_{\text{IPS}} = T_{\text{Detection}} + T_{\text{Reaction}} + T_{\text{Delay}}, \tag{2}$$

where $T_{\text{Detection}}$ is the time it takes to detect a malicious frame, T_{Reaction} is the time it takes to transmit six consecutive bits 0, i.e., $T_{\text{Reaction}} = 6T_{\text{bit}}$, and T_{Delay} includes the time it takes to instruct the transceiver and the delay introduced by the hardware. Our IPS processing time must be shorter than the time available to detect and stop an attack, hence

$$\begin{aligned} T_{\text{IPS}} &< T_{\text{Available}} \\ T_{\text{Detection}} + T_{\text{Reaction}} + T_{\text{Delay}} &< (L - K)T_{\text{bit}} \\ T_{\text{Detection}} + 6T_{\text{bit}} + T_{\text{Delay}} &< (L - K)T_{\text{bit}} \\ T_{\text{Detection}} &< (L{-}K - 6)T_{\text{bit}} - T_{\text{Delay}}. \end{aligned} \tag{3}$$

Therefore, the time requirement for preventing attacks, i.e., detecting and stopping attacks before they can cause damage, is that the time it takes to detect malicious frames must be shorter than

$$T_{\text{Detection}_{\text{Max}}} = (L{-}K - 6)T_{\text{bit}} - T_{\text{Delay}}, \tag{4}$$

so that $T_{\text{IPS}} < T_{\text{Available}}$ is satisfied.

Furthermore, since the detection of attacks can start only after all bytes used by the detection algorithm are received, the fewer bytes used, the sooner detection can start. In reference to Figure 2, if all eight data bytes in a CAN frame's data field are used, the detection algorithm can start only after the eighth data byte has been transmitted, after which there are only 25 bits left to be transmitted in the frame. Thus, $(L - K) = 25$ and $T_{Detection_{Max}} = 19T_{bit} - T_{Delay}$. Similarly, if only the first seven data bytes in the data field are used, detection can start after the transmission of the seventh data byte, after which there are 33 bits left to be transmitted. Thus, $(L - K) = 33$ and $T_{Detection_{Max}} = 27T_{bit} - T_{Delay}$. The same formula applies when fewer of the data field's data bytes are used. To generalize, if the first N data bytes of a CAN frame's data field are used in detection, the maximum time to detect

attacks is given by

$$T_{\text{Detection}_{\text{Max}}} = 19T_{\text{bit}} + 8(8-N)T_{\text{bit}} - T_{\text{Delay}}. \quad (5)$$

Since using fewer data bytes tends to degrade detection rates, our proposed system needs to use a detection algorithm that minimizes the impact of that reduction and still performs well with a minimum number of data field data bytes. On the other hand, although using fewer data bytes grants more time for detecting attacks, that time is still very short and challenging to meet. For instance, for 500 Kbps networks ($T_{\text{bit}} = 2\mu s$), even when using only the first six data bytes in the data field, our proposed IPS has only $70\mu s$ to detect a cyber-attack in the best-case scenario ($T_{Delay} = 0$). For this reason, it is essential to use a detection algorithm that not only performs well with minimal data, but is also fast. This constraint significantly inhibits most sophisticated anomaly detection algorithms that are based on deep learning techniques, such as GANs [38], [49], as they usually have longer detection times. In this paper, we evaluate whether the OCSVM and iForest algorithms, which have been largely advocated in the literature for anomaly detection tasks [18], [20], [21], can satisfy the requirements discussed.

B. ONE-CLASS SUPPORT VECTOR MACHINE

The OCSVM algorithm constructs an optimal hyperplane for separating data patterns that are similar to training data from abnormalities, i.e., data patterns that do not conform with the training data [50], [51]. It is widely adopted in anomaly detection problems. Consider a training set D with N samples having dimension d, i.e., $D = \{\mathbf{x}_i\}_{i=1}^{N} | \mathbf{x}_i \in \mathbb{R}^d$, and a feature mapping $\Phi : \mathbf{x} \mapsto \Phi(\mathbf{x})$ that maps training samples from the input space into a higher dimensional feature space. The OCSVM tries to find a decision boundary function f that delimits the smallest region in the high-dimensional feature space such that f returns $+1$ if a data sample lies within the defined region and -1 otherwise. In the context of our proposed detection mechanism, we train the OCSVM with only normal CAN frames and find a decision function f such that $f(\mathbf{x}) = +1$ when $\mathbf{x}$ is a normal frame and $f(\mathbf{x}) = -1$ when $\mathbf{x}$ is a malicious frame.

The OCSVM solves the following quadratic optimization problem to find the decision boundary:

$$\min_{w,\rho} \frac{1}{2}||w||^2 + \frac{1}{\nu^N}\sum_{i=1}^{N}\xi_i - \rho, \quad (6)$$

such that $w\Phi(x_i) \geq \rho - \xi_i$ and $\xi_i \geq 0$, where ρ is a margin parameter, ξ_i is a slack variable that allows a data point to be outside the decision boundary, and $\nu \in (0, 1]$ is a trade-off parameter that represents an upper bound on the fraction of outliers and a lower bound on the fraction of support vectors.

Considering that the inner product of two training samples, i.e., $\Phi(x_i)^T\Phi(x_j)$, can be replaced by a kernel function K such that $K(x_i, x_j) = \Phi(x_i)^T\Phi(x_j)$, the dual optimization problem of (6) is obtained as follows:

$$\min_{\alpha} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j K(\mathbf{x}_i, \mathbf{x}_j), \quad (7)$$

such that

$$\sum_{i=1}^{N}\alpha_i = 1,$$

$$0 \leq \alpha_i \leq \frac{1}{\nu^N} \quad \forall i = 1, \ldots, N,$$

where α_i is the Lagrange multiplier. Solving that dual optimization problem yields the following decision function f:

$$f(x) = \sum_{i=1}^{N}\alpha_i K(\mathbf{x}_i, \mathbf{x}) - \rho. \quad (8)$$

C. ISOLATION FOREST

The iForest algorithm consists of an ensemble of isolation trees, which are tree structures constructed to isolate every single instance of a set from the others. Each tree recursively partitions and isolates data instances by randomly selecting a data attribute and a split value that is between the maximum and minimum values of the selected attribute [18], [52]. Since anomalies are considered to make up a small percentage of the normal data and have very different attributes than normal data, they tend to be isolated with fewer partitions than normal data. Therefore, the iForest algorithm detects anomalies by computing the path length $h(x)$, which is the average number of partitions required by isolation trees to separate a data pattern x.

The iForest training stage builds isolation trees using sub-samples of a training set. To obtain a normalized anomaly score, it estimates the average number of partitions of a data pattern given a sub-sample size m. That estimate, $c(m)$, corresponds to an unsuccessful search in a binary tree and is given by:

$$c(m) = \begin{cases} 2H(m-1) - \frac{2(m-1)}{m}, & \text{for } m > 2 \\ 1, & \text{for } m = 2 \\ 0, & \text{for } m = 1, \end{cases}$$

where H is the harmonic number, which can be estimated by $H(i) = ln(i) + 0.5772156649$ (Euler's constant). Then, it computes the anomaly score s of the data pattern x as

$$s(x, m) = 2^{(-\frac{E(h(x))}{c(m)})},$$

where $E(h(x))$ is the average path length of x for a collection of isolation trees, and $c(m)$ is the average of $h(x)$ given m.

Finally, since isolation trees have a limited number of nodes, the time required to build them is also limited, such that, in the worst-case scenario, it is a $\mathcal{O}(m^2)$ function. Similarly, in the worst-case scenario, the time complexity to detect anomalies during data pattern evaluation is $\mathcal{O}(ntm)$, where n is the testing data size and t is the number of isolation trees.

TABLE 2. Number of messages in the three datasets.

Dataset	# of Messages	# of Normal Messages	# of Attacking Messages
Fuzzing	3,838,860	3,347,013	491,847
Drive Gear Spoofing	4,443,142	3,845,890	597,252
RPM Gauge Spoofing	4,621,702	3,966,805	654,897

V. METHODOLOGY AND EXPERIMENTAL SETUP

In this section, we briefly present the datasets used in our experiments, which contain both normal and attacking CAN frames. Then, we explain the experiments conducted to compare the OCSVM and iForest algorithms, the workflow of our methodology, and the experimental setup for discarding malicious frames.

A. DATASET PRESENTATION

We evaluate our proposed IPS using three datasets from the Hacking and Countermeasure Research Lab [53] that are publicly available for academic purposes. Each of them contains from 30 to 40 minutes of regular traffic data logged from a CAN via the OBD-II port of a real vehicle. In addition, each dataset contains attacking CAN messages from a fuzzing, a drive gear spoofing, or an RPM gauge spoofing attack, which were conducted during data acquisition. In the fuzzing attacks, CAN messages have random identifiers and data bytes; in the drive gear spoofing attacks, CAN messages use the identifier related to gear information; and in the RPM gauge spoofing attacks, CAN messages use the identifier related to RPM information. Table 2 shows the number of normal and attacking CAN messages in each of the three datasets considered.

We constructed training, validation, and testing sets for each dataset. The training and validation sets were used to train the proposed detection algorithms and find their optimal hyper-parameters, respectively. The testing sets were used to evaluate the performance of each model, and hence that of our proposed IPS. To avoid overfitting, we followed the 10-fold cross-validation technique and constructed the training and validation sets by randomly selecting 250,000 normal frames and splitting them into 10 folds of 25,000 frames, of which nine were used for training, and the remaining one was used for validation along with 25,000 randomly selected malicious frames. Thus, the training sets contained only normal frames, and the validation sets contained both normal and malicious frames. All remaining normal and attacking frames formed the testing set. The reason behind using folds of 25,000 frames is that initial experiments we conducted showed that using less than 225,000 normal frames for training resulted in lower detection rates. On the other hand, using more than 225,000 normal frames increased the training time without improving the detection results. Tables 3, 4, and 5 show the number of normal and attacking messages in the training, validation, and testing sets of the three datasets considered.

B. METHODOLOGY AND EXPERIMENTAL SETUP

To compare the OCSVM and iForest algorithms' performance when using different numbers of data field data bytes, we defined eight models for each algorithm, which are described in Table 6. The hyper-parameters of each model were optimized using a cross-validated grid search over a parameter grid. For instance, in the case of the OCSVM models, different kernel functions were considered for tuning, and the radial basis function (RBF) was chosen as it produced the best results on the validation set. Then, the detection performance of each model was evaluated on the testing sets. Finally, since our goal is to deploy our system on an inexpensive hardware platform, all experiments were conducted on Raspberry Pi 4 Model B with 4GB of RAM. Figure 3 shows a flow diagram of our proposed approach.

Figure 4 shows our experimental setup for verifying that our IPS can discard malicious frames before their transmission is completed. Our setup includes two neoVI FIRE 2 [54] from Intrepid Control Systems that are configured to work as nodes exchanging CAN messages. As shown in Figure 1, our IPS has a detection agent, a CAN controller, and a CAN transceiver. The detection agent implements on Raspberry Pi 4 Model B the model that has the highest detection rates from those in Table 6. The Raspberry Pi is connected to the CAN controller MCP2515 from Microchip [55] through the serial peripheral interface (SPI). The chip select (CS), interrupt (INT), master in slave out (MISO), master out slave in (MOSI), and clock (CLK) signals are standard signals

TABLE 3. Fuzzing training, validation, and testing sets.

Fuzzing Dataset	# of Normal Messages	# of Attacking Messages
Training	225,000	0
Validation	25,000	25,000
Testing	3,097,013	466,847

TABLE 4. Drive gear spoofing training, validation, and testing sets.

Drive Gear Spoofing Dataset	# of Normal Messages	# of Attacking Messages
Training	225,000	0
Validation	25,000	25,000
Testing	3,595,890	572,252

TABLE 5. RPM gauge spoofing training, validation, and testing sets.

RPM Gauge Spoofing Dataset	# of Normal Messages	# of Attacking Messages
Training	225,000	0
Validation	25,000	25,000
Testing	3,716,805	629,897

TABLE 6. Detection models.

Model	Description
OCSVM_1F	OCSVM algorithm is used with the first CAN data byte
OCSVM_2F	OCSVM algorithm is used with the first 2 CAN data bytes
OCSVM_3F	OCSVM algorithm is used with the first 3 CAN data bytes
OCSVM_4F	OCSVM algorithm is used with the first 4 CAN data bytes
OCSVM_5F	OCSVM algorithm is used with the first 5 CAN data bytes
OCSVM_6F	OCSVM algorithm is used with the first 6 CAN data bytes
OCSVM_7F	OCSVM algorithm is used with the first 7 CAN data bytes
OCSVM_8F	OCSVM algorithm is used with the first 8 CAN data bytes
iForest_1F	iForest algorithm is used with the first CAN data byte
iForest_2F	iForest algorithm is used with the first 2 CAN data bytes
iForest_3F	iForest algorithm is used with the first 3 CAN data bytes
iForest_4F	iForest algorithm is used with the first 4 CAN data bytes
iForest_5F	iForest algorithm is used with the first 5 CAN data bytes
iForest_6F	iForest algorithm is used with the first 6 CAN data bytes
iForest_7F	iForest algorithm is used with the first 7 CAN data bytes
iForest_8F	iForest algorithm is used with the first 8 CAN data bytes

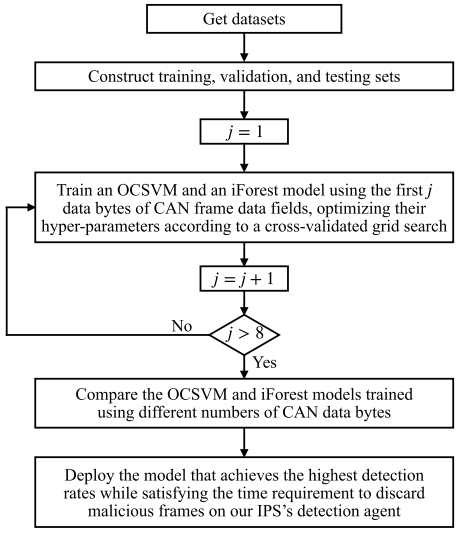


FIGURE 3. Flow diagram of our proposed approach.

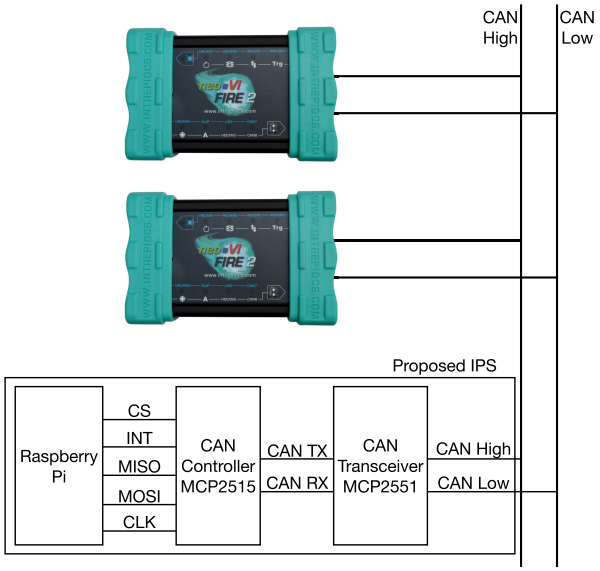


FIGURE 4. Experimental setup.

used in the SPI. The CAN controller MCP2515 is connected to the CAN transceiver MCP2551 from Microchip [48], which allows up to 20 consecutive dominant bits to be transmitted through the CAN TX and CAN RX signals. Finally, the transceiver MCP2551 connects to the CAN bus by means of the CAN High and CAN Low signals. Note that the CAN controller and transceiver used are standard and inexpensive. Equivalent hardware can also be used.

VI. RESULTS AND DISCUSSION

In this section, we present and discuss the results of our experiments. First, we present the performance and comparison results of the models defined in Table 6. Then, we present the detection times of the models that achieve the best detection results, and evaluate whether they can be used in our proposed architecture to detect and discard malicious frames before their transmission is completed. Afterwards, we show the results obtained from our experimental setup for discarding malicious frames. Finally, we compare our proposed solution to four state-of-the-art IDSs that also aim to secure automotive CANs: GIDS [38], DCNN [39], an LSTM-based IDS [39], and MTH-IDS [40].

TABLE 7. Fuzzing attack detection performance.

Model	Accuracy (%)	Precision (%)	Recall (%)
OCSVM_8F	97.0633 ± 1.8694	83.0497 ± 9.2978	99.9177 ± 0.002
OCSVM_7F	96.5949 ± 2.2679	81.1490 ± 10.5017	99.7820 ± 0.0040
OCSVM_6F	95.6728 ± 3.3571	79.0108 ± 13.9457	97.6339 ± 0.1175
OCSVM_5F	86.7426 ± 4.9369	52.3206 ± 11.0698	91.6551 ± 0.2771
iForest_8F	99.7215 ± 0.0404	97.9964 ± 0.2959	99.9711 ± 0.0084
iForest_7F	99.5916 ± 0.0558	97.0744 ± 0.3946	99.9738 ± 0.0075
iForest_6F	99.3140 ± 0.0933	95.1984 ± 0.6400	99.9375 ± 0.0159
iForest_5F	99.3128 ± 0.0517	95.2214 ± 0.3403	99.8977 ± 0.0215
iForest_4F	99.1429 ± 0.1384	94.2371 ± 0.9410	99.7269 ± 0.0635
iForest_3F	98.5657 ± 0.2796	90.6627 ± 1.7729	99.6195 ± 0.0951
iForest_2F	95.7681 ± 1.5233	77.0781 ± 6.0160	98.6587 ± 0.2897

A. PERFORMANCE EVALUATION

After training and tuning the defined models, we applied them to the testing sets to classify frames as normal or malicious. The classification results were then used to compute three metrics: accuracy, precision, and recall. Accuracy gives the percentage of correct predictions, i.e., the percentage of frames correctly classified as either normal or malicious. Precision measures the percentage of correctly predicted malicious frames out of all frames classified as malicious. Recall measures the percentage of correctly predicted malicious frames out of all malicious frames.

Table 7 shows the accuracy, precision, and recall results of the fuzzing attack dataset for the models that achieved more than 85% accuracy. The remaining models were disregarded. The results in Table 7 clearly show that the models that used the iForest algorithm outperform the models that used the OCSVM algorithm, since they achieved better accuracy, precision and recall results even when using fewer data field data bytes for detection. Moreover, the great disparity in the precision results indicates that the iForest models achieve much lower false positive rates than the OCSVM models. The false positive and false negative percentages for eight of the defined models are shown in the confusion matrices in Figure 5. Similar results were obtained for the drive gear spoofing and RPM gauge spoofing attacks.

B. MODEL COMPARISON

We also conducted D'Agostino and Pearson's hypothesis test to verify that the detection rates obtained for all the models can be approximated by a normal distribution. This verification allowed us to conduct the one-way ANOVA hypothesis test to verify whether there was a significant difference between at least two of the models defined for each attack type. For the three attack types considered, the ANOVA test confirmed that at least one of the models differed from the others such that there was a statistically significant

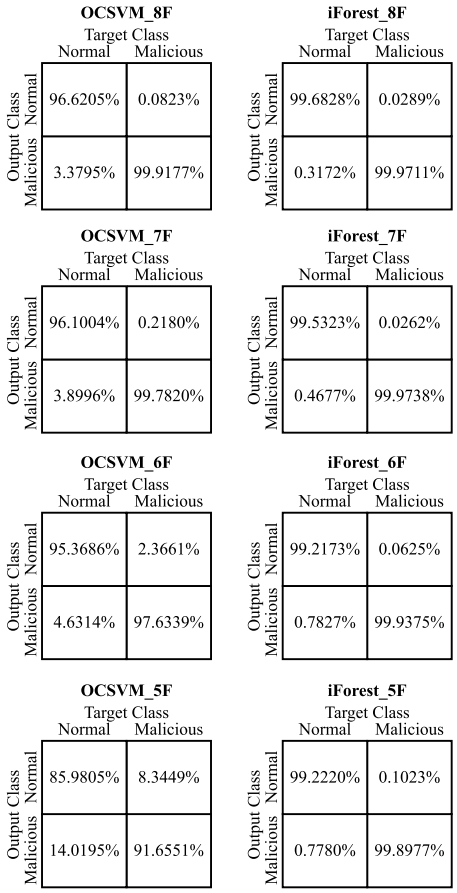


FIGURE 5. Models' confusion matrices for the fuzzing attack.

difference among the models. Since ANOVA is not able to determine which model differed, Tukey's honestly significant difference (HSD) post hoc test was also conducted. In contrast to the ANOVA test, it evaluates the models two-by-two to identify which model differed from the others.

For fuzzing attacks, the post hoc test indicated that there is no statistically significant difference between the iForest models with four or more data bytes as input and the OCSVM model with eight data bytes as input. Since the *iForest_8F* model achieves the highest accuracy, precision and recall values, as presented in Table 7, and the *iForest_4F* model needs the least data to detect malicious frames with statistically equivalent performance metrics, the *iForest_4F* model can then be considered the best one for fuzzing attacks. Similarly, for drive gear spoofing attacks, there is no statistically significant difference between the iForest models with five or more data bytes as input and the OCSVM models with six or more data bytes as input. Therefore, the *iForest_5F* model can be considered the most suitable for this type of attack. Finally, for RPM gauge spoofing attacks, there is no statistically significant difference between the iForest models with three or more data bytes as input and the OCSVM models with seven or eight data bytes as input. Thus, the *iForest_3F* model is the best for this type of attack. In order to have a general model to detect attacking frames of the three types considered, we choose the iForest model with five CAN frame data bytes as input. As shown in Table 7, this model achieves accuracy higher than 99% with a standard deviation of less than 0.06% for fuzzing attacks.

TABLE 8. Average detection times on Raspberry Pi 4 model B.

Attack Type	Average Detection Time (μs)
Fuzzing	79.4645
Drive Gear Spoofing	78.0415
RPM Gauge Spoofing	77.9576

C. DETECTION TIME EVALUATION

Since the iForest model that uses five data bytes was shown to be the best at detecting attacking frames, we use it as the detection model of our proposed IPS. Therefore, we can refer to (5) and determine that the time available to detect a malicious frame and discard it before its transmission is completed is $T_{\text{Detection}_{\text{Max}}} = 43T_{\text{bit}} - T_{\text{Delay}}$. Thus, for standard 500 Kbps CANs, for which $T_{\text{bit}} = 2\mu s$, our proposed IPS must detect cyber-attacks in less than $86\mu s$ to prevent damage.

To verify that our proposed IPS meets the $T_{\text{Detection}_{\text{Max}}} = 86\mu s$ constraint, we measured the detection time when testing and classifying frames as normal or malicious with the Raspberry Pi 4 Model B on which we deployed our IPS. The experiment was repeated 50 times for each attack type analyzed to compute their average detection times, which are depicted in Table 8. Since the average detection time computed for each of the three attack types considered is below the $T_{\text{Detection}_{\text{Max}}} = 86\mu s$ threshold, our proposed system is capable of detecting and discarding attacking frames before their transmission is completed, which then prevents damage and secures the network. Note that if more than five data field data bytes are used, i.e., $T_{\text{Detection}_{\text{Max}}} = 35T_{\text{bit}} = 70\mu s$, the time requirement for discarding malicious frames before their transmission is completed would not be met. Therefore, using fewer data bytes to detect cyber-attacks in CANs is not only possible, but necessary.

D. DISCARDING MALICIOUS FRAMES

The experimental setup shown in Figure 4 was used to verify that our proposed IPS can discard malicious frames before their transmission is completed. Once the two neoVI FIRE 2 were configured to exchange the CAN messages in the testing sets constructed, we used a PicoScope oscilloscope to analyze the transmitted CAN frames. Figure 6 shows one of the malicious CAN frames that were transmitted. Next, we connected our proposed IPS to the bus to use the trained detection mechanism as the Raspberry Pi's detection agent. When malicious frames were detected, the Raspberry Pi sent a command for the CAN controller to transmit six consecutive bits 0 through the CAN transceiver, which forces an error condition so that an error frame is transmitted by the nodes in the bus. Figure 7 shows a malicious frame that was discarded

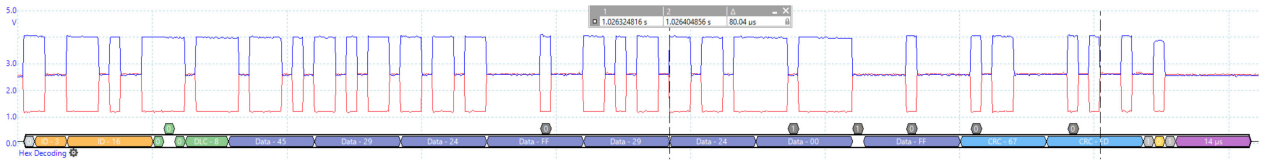


FIGURE 6. Malicious frame.

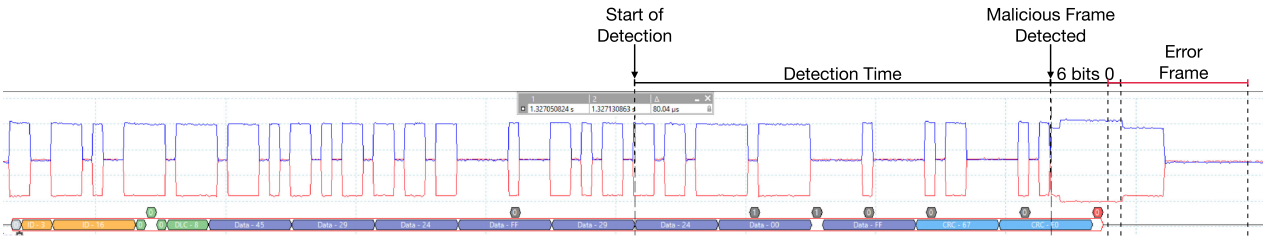


FIGURE 7. Discarded malicious frame.

TABLE 9. Detection results on the fuzzing dataset.

Detection Mechanism	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
GIDS	98.00	97.30	99.50	98.39
DCNN	99.82	99.95	99.65	99.80
LSTM-based IDS	99.35	99.36	99.16	99.26
MTH-IDS	99.99	99.99	99.99	99.99
Proposed IPS	99.29	95.07	99.93	97.44

TABLE 10. Detection results on the drive gear spoofing dataset.

Detection Mechanism	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
GIDS	96.20	98.10	96.50	97.29
DCNN	99.95	99.99	99.89	99.94
LSTM-based IDS	99.76	99.75	99.68	99.72
MTH-IDS	99.99	99.99	99.99	99.99
Proposed IPS	99.24	94.79	100.00	97.33

TABLE 11. Detection results on the RPM gauge spoofing dataset.

Detection Mechanism	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
GIDS	98.00	98.30	99.00	98.65
DCNN	99.97	99.99	99.94	99.96
LSTM-based IDS	99.87	100.00	99.71	99.85
MTH-IDS	99.99	99.99	99.99	99.99
Proposed IPS	99.85	98.97	100.00	99.48

by our IPS. Note that detection starts just after the fifth data byte and finishes after 80μs, which corresponds to the frame's detection time plus T_{Delay}. The error frame interrupts the malicious frame's transmission and protects the nodes. After transmission, the bus waits for the interframe space (IFS), which corresponds to the time it takes to transmit three bits, and then becomes idle so another transmission can start.

E. COMPARISON TO OTHER WORKS

We compare our proposed IPS, which uses the *iForest_5F* model, to four state-of-the-art CAN IDSs: GIDS [38], DCNN [39], an LSTM-based IDS [39], and MTH-IDS [40]. While GIDS leverages GANs to propose an unsupervised IDS, DCNN follows a supervised approach and is limited to detecting known types of attacks. The LSTM-based IDS, which the authors of [39] use as their baseline, is also supervised. Finally, MTH-IDS combines supervised and unsupervised techniques. Tables 9, 10, and 11 show the detection results of our proposed IPS and those state-of-the-art IDSs on the fuzzing, drive gear spoofing, and RPM gauge spoofing datasets, respectively.

Our experiment results show that our proposed IPS achieves better accuracy than GIDS on the three datasets studied and has a higher F1-score than GIDS on the two spoofing datasets, which balances the precision and recall metrics. On the other hand, DCNN and the LSTM-based IDS achieve better accuracy and have higher F1-scores than our proposed system and GIDS on the three datasets studied. However, contrary to our IPS and GIDS, DCNN and the LSTM-based IDS follow a supervised approach and rely on labeled training data and therefore cannot detect unknown attacks. Finally, MTH-IDS achieves the best accuracy and has the highest F1-scores as it combines both unsupervised and supervised techniques. Since our datasets are imbalanced, we rely on the F1-score to evaluate and compare our results. Moreover, although the fuzzing, drive gear spoofing, and RPM gauge spoofing datasets used include between 10% and 15% malicious samples, malicious messages usually account for less than 1% in real world cyber-security applications. However, this does not affect our solution and results as our technique is trained with only normal data and its performance does not depend on the number of malicious samples.

Although our proposed IPS achieves slightly lower detection rates than DCNN, the LSTM-based IDS, and MTH-IDS, it has the shortest detection time of all the solutions compared.

TABLE 12. Comparative analysis of detection times.

Detection Mechanism	Average Detection Time (μs)
GIDS	92.1187
DCNN	98.4375
LSTM-based IDS	160.9375
MTH-IDS	140.1094
Proposed IPS	78.4879

Moreover, it is the only one that meets the time requirement for discarding attacking frames before their transmission is completed, such that damage is prevented. Furthermore, it does so while being deployed on inexpensive Raspberry Pi, whereas GIDS, DCNN, and the LSTM-based IDS are deployed on expensive and sophisticated GPUs. For instance, GIDS is deployed on an NVIDIA GeForce GTX 1080 with 32GB of RAM, and DCNN and the LSTM-based IDS are deployed on an NVIDIA Tesla K80 GPU with 12GB of RAM. MTH-IDS is deployed on the same Raspberry Pi used by our IPS. Therefore, our proposed system is considered the best solution among the five as it is the only one that is capable of discarding malicious frames and preventing damage using inexpensive hardware, such as Raspberry Pi. Table 12 shows the detection times of our proposed IPS, GIDS, DCNN, the LSTM-based IDS, and MTH-IDS.

F. TRADE-OFFS, LIMITATIONS, AND FINAL CONSIDERATIONS

Our system was shown to achieve accuracy higher than 99% and F1-scores higher than 97% when it comes to detecting fuzzing, drive gear spoofing, and RPM gauge spoofing attacks in the datasets used, which contain data collected from an actual vehicle under attack. In addition, our IPS was shown to be able to discard malicious frames before their transmission is completed so that the bus nodes are protected. On the other hand, it brings two trade-offs or limitations that we discuss in this subsection along with considerations about the feature interpretability.

1) OVERHEAD VERSUS HARDWARE COST

Since our detection mechanism uses only the first five data bytes of a CAN frame's data field, it cannot detect attacks that affect only the last three data bytes. To avoid this issue, we suggest two countermeasures. The first is to treat the last three data bytes as overhead so that no data is sent on them. Although this reduces the amount of information transmitted per message, the automotive industry is used to having similar and even larger overheads, as in the case of authentication solutions [56]. The second suggestion is to use a more sophisticated hardware platform so that our detection mechanism can meet the time constraint for stopping ongoing attacks while using all eight data bytes of the CAN frame's data field. Therefore, manufacturers may choose between reducing the data transmission rate or spending more financial resources on better hardware. Since cost is an essential concern of vehicle manufacturers, such a decision depends on the market and car model, as different vehicles offer more or less technology at higher or lower prices.

2) FALSE POSITIVES VERSUS DETECTION OF UNKNOWN ATTACKS

False positives can potentially block messages and cause problems. Although they can be minimized, they result from unsupervised anomaly-based detection techniques, such as the one used in our proposed IPS. However, detection mechanisms that are based on signatures or labeled attack data instead cannot detect unknown attacks, whereas our solution can. Thus, there is a trade-off between mistakenly blocking some legitimate frames and being subject to new types of attacks that are constantly being developed and cannot be detected. If legitimate frames are mistakenly blocked (false positives), the ECUs that should have received them will not send acknowledgment bits. As a consequence, those blocked frames are automatically resent. If that situation persists, ECUs that cannot successfully send or receive messages raise software warnings called diagnostic trouble codes (DTC). Those warnings create a log of the system's malfunctioning and may even notify drivers through indicative lights or make the car enter fail-safe mode, which modifies the car's operation to prevent damage and safety risks [57]. On the other hand, if IDSs/IPSs fail to detect malicious messages (false negatives), vehicles must rely on other security layers, such as authentication and data integrity mechanisms. Although vehicles usually employ multiple security layers, not identifying attacks in the early stages may result in increased network overhead, the car being put into fail-safe mode, or even loss of control of ECUs to attackers if the other security layers also fail [58], [59]. As future work, we propose to study an ensemble of different detection mechanisms so that it is possible to detect unknown attacks while also keeping false positive rates near zero.

Furthermore, our solution can be configured to use different thresholds depending on whether it is more harmful to have more false positives or fewer true positives, which balances the trade-off between false positives and true positives. Figures 8, 9, and 10 show the receiver operating characteristic (ROC) curves and their area under the curve (AUC) for the testing sets of the datasets used. Each point in the curves represents the true positive and false positive rates achieved for a threshold. The accuracy, precision, recall, and F1-score values of Tables 9, 10, and 11 correspond to the results obtained with the threshold that maximizes the difference between the true positive and false positive rates in the validation sets.

3) FEATURE INTERPRETABILITY

Vehicle manufacturers usually use proprietary message mappings, i.e., their own confidential way of encoding information requests, sensor measurements, and commands in a CAN frame. In such cases, interpreting what each CAN message and data byte means is possible only if one has access to the restricted information from manufacturers or through reverse

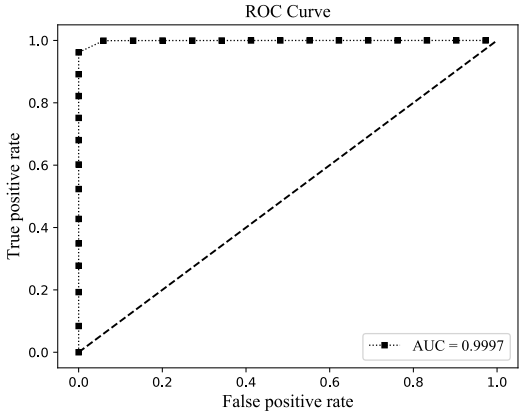


FIGURE 8. ROC curve on the fuzzing dataset.

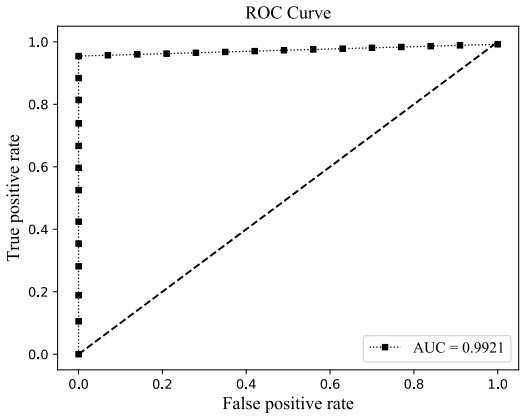


FIGURE 9. ROC curve on the drive gear spoofing dataset.

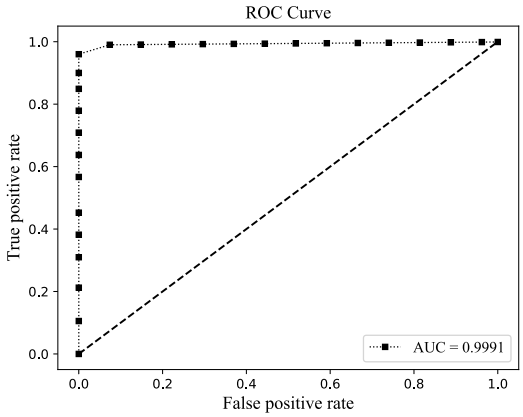


FIGURE 10. ROC curve on the RPM gauge spoofing dataset.

engineering. On the other hand, vehicle manufacturers are required to adopt standard encodings for specific purposes, such as standard diagnostic and emission control protocols. In these cases, it is possible to interpret the features used by our IPS. Nevertheless, in contrast to other techniques that depend on knowing message mappings, our proposed IPS does not rely on any previous knowledge about what each CAN message means.

VII. CONCLUSION

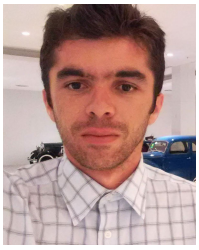
In this work, we propose a novel IPS that effectively and efficiently detects cyber-attacks in CANs while being deployed on inexpensive hardware, such as Raspberry Pi. We evaluated the cyber-attack detection performance of the one-class support vector machine and isolation forest algorithms using different numbers of CAN data field data bytes to ensure the time requirement was met for discarding attacking frames before their transmission is completed and preventing damage. We experimented on three datasets with three types of cyber-attacks: fuzzing, drive gear spoofing, and RPM gauge spoofing. For the three types of attacks considered, the isolation forest algorithm with five data bytes is shown to achieve the best trade-off between the detection rate and the number of data bytes used. Our proposed IPS achieved accuracy higher than 99% when using only the first five data bytes to detect attacks in CANs. Moreover, it had a shorter detection time than four state-of-the-art IDSs, and it is the only one capable of discarding malicious frames before damage occurs.

REFERENCES

[1] S. Woo, H. J. Jo, and D. H. Lee, "A practical wireless attack on the connected car and security protocol for in-vehicle CAN," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 993–1006, Sep. 2015.
[2] J. Liu, S. Zhang, W. Sun, and Y. Shi, "In-vehicle network attacks and countermeasures: Challenges and future directions," *IEEE Netw.*, vol. 31, no. 5, pp. 50–58, May 2017.
[3] S. Parkinson, P. Ward, K. Wilson, and J. Miller, "Cyber threats facing autonomous and connected vehicles: Future challenges," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 11, pp. 2898–2915, Nov. 2017.
[4] United States Department of Transportation. *Vehicle Cybersecurity*. Accessed: Nov. 15, 2021. [Online]. Available: https://www.nhtsa.gov/technology-innovation/vehicle-cybersecurity
[5] United Nations. *Uniform Provisions Concerning the Approval of Vehicles With Regards to Cyber Security and Cyber Security Management System*. Accessed: Nov. 15, 2021. [Online]. Available: https://unece.org/sites/default/files/2021-03/R155e.pdf
[6] *Road Vehicles—Cybersecurity Engineering*, Standard ISO/SAE 21434:2021, ISO, 2021. [Online]. Available: https://www.iso.org/standard/70918.html
[7] H. M. Song, H. R. Kim, and H. K. Kim, "Intrusion detection system based on the analysis of time intervals of CAN messages for in-vehicle network," in *Proc. Int. Conf. IEEE Inf. Netw. (ICOIN)*, Jan. 2016, pp. 63–68.
[8] R. Mitchell and I.-R. Chen, "A survey of intrusion detection techniques for cyber-physical systems," *ACM Comput. Surv.*, vol. 46, no. 4, pp. 55:1–55:29, Mar. 2014, doi: 10.1145/2542049.
[9] A. Nisioti, A. Mylonas, P. D. Yoo, and V. Katos, "From intrusion detection to attacker attribution: A comprehensive survey of unsupervised methods," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 3369–3388, 4th Quart., 2018.
[10] H. Choi, M. Kim, G. Lee, and W. Kim, "Unsupervised learning approach for network intrusion detection system using autoencoders," *J. Supercomput.*, vol. 75, no. 9, pp. 5597–5621, Sep. 2019.
[11] B. B. Zarpelo, R. S. Miani, C. T. Kawakani, and S. C. de Alvarenga, "A survey of intrusion detection in Internet of Things," *J. Netw. Comput. Appl.*, vol. 84, pp. 25–37, Apr. 2017, doi: 10.1016/j.jnca.2017.02.009.
[12] Splunk. *What is User Behavior Analytics (UBA)/User Entity Behavior Analytics (UEBA)?* Accessed: Oct. 10, 2021. [Online]. Available: https://www.splunk.com/en_us/data-insider/user-behavior-analytics-ueba.html
[13] Microsoft. *Tutorial: Detect Suspicious User Activity With UEBA*. Accessed: Oct. 10, 2021. [Online]. Available: https://docs.microsoft.com/en-us/cloud-app-security/tutorial-suspicious-activity#phase-2-tune-anomaly-detection-policies
[14] FireEye. *User and Entity Behavior Analytics (UEBA) Through FireEye Helix*. Accessed: Oct. 10, 2021. [Online]. Available: https://www.fireeye.com/content/dam/fireeye-www/products/pdfs/pf/helix/ds-fe-helix-ueba.pdf

[15] K. Cheng, Y. Bai, Y. Zhou, Y. Tang, D. Sanan, and Y. Liu, "CANeleon: Protecting CAN bus with frame ID chameleon," *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 7116–7130, Jul. 2020.
[16] M.-J. Kang and J.-W. Kang, "A novel intrusion detection method using deep neural network for in-vehicle network security," in *Proc. IEEE 83rd Veh. Technol. Conf. (VTC Spring)*, May 2016, pp. 1–5.
[17] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, p. 15, 2009.
[18] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation-based anomaly detection," *ACM Trans. Knowl. Discovery Data*, vol. 6, no. 1, pp. 3:1–3:39, Mar. 2012.
[19] O. Avatefipour, A. S. Al-Sumaiti, A. M. El-Sherbeeny, E. M. Awwad, M. A. Elmeligy, M. A. Mohamed, and H. Malik, "An intelligent secured framework for cyberattack detection in electric vehicles' CAN bus using machine learning," *IEEE Access*, vol. 7, pp. 127580–127592, 2019.
[20] M. Weber, S. Klug, E. Sax, and B. Zimmer, "Embedded hybrid anomaly detection for automotive CAN communication," in *Proc. 9th Eur. Congr. Embedded Real Time Softw. Syst. (ERTS)*, 2018, pp. 1–11.
[21] S.-F. Lokman, A. T. Othman, and M.-H. Abu-Bakar, "Intrusion detection system for automotive controller area network (CAN) bus system: A review," *EURASIP J. Wireless Commun. Netw.*, vol. 2019, no. 1, p. 184, Jul. 2019.
[22] H. Olufowobi, S. Hounsinou, and G. Bloom, "Controller area network intrusion prevention system leveraging fault recovery," in *Proc. ACM Workshop Cyber-Phys. Syst. Secur. Privacy (CPS-SPC)*, 2019, pp. 63–73.
[23] T. Matsumoto, M. Hata, M. Tanabe, K. Yoshioka, and K. Oishi, "A method of preventing unauthorized data transmission in controller area network," in *Proc. IEEE 75th Veh. Technol. Conf. (VTC Spring)*, May 2012, pp. 1–5.
[24] R. Kurachi, Y. Matsubara, H. Takada, N. Adachi, Y. Miyashita, and S. Horihata, "CaCAN-centralized authentication system in CAN (controller area network)," in *Proc. 14th Int. Conf. Embedded Secur. Cars (ESCAR)*, 2014, pp. 1–9.
[25] S. Longari, M. Penco, M. Carminati, and S. Zanero, "CopyCAN: An error-handling protocol based intrusion detection system for controller area network," in *Proc. ACM Workshop Cyber-Physical Syst. Secur. Privacy (CPS-SPC)*, 2019, pp. 39–50.
[26] H. Olufowobi, C. Young, J. Zambreno, and G. Bloom, "SAIDuCANT: Specification-based automotive intrusion detection using controller area network (CAN) timing," *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 1484–1494, Feb. 2020.
[27] H. Olufowobi, U. Ezeobi, E. Muhati, G. Robinson, C. Young, J. Zambreno, and G. Bloom, "Anomaly detection approach using adaptive cumulative sum algorithm for controller area network," in *Proc. ACM Workshop Automot. Cybersecur.*, Mar. 2019, pp. 25–30.
[28] M. Bozdal, M. Samie, and I. K. Jennions, "WINDS: A wavelet-based intrusion detection system for controller area network (CAN)," *IEEE Access*, vol. 9, pp. 58621–58633, 2021.
[29] M. Marchetti and D. Stabili, "Anomaly detection of CAN bus messages through analysis of ID sequences," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2017, pp. 1577–1583.
[30] M. Nam, S. Park, and D. S. Kim, "Intrusion detection method using bi-directional GPT for in-vehicle controller area networks," *IEEE Access*, vol. 9, pp. 124931–124944, 2021.
[31] K.-T. Cho and K. G. Shin, "Fingerprinting electronic control units for vehicle intrusion detection," in *Proc. 25th USENIX Secur. Symp.*, 2016, pp. 911–927.
[32] S. Halder, M. Conti, and S. K. Das, "COIDS: A clock offset based intrusion detection system for controller area networks," in *Proc. 21st Int. Conf. Distrib. Comput. Netw.*, Jan. 2020, pp. 1–10.
[33] A. Taylor, N. Japkowicz, and S. Leblanc, "Frequency-based anomaly detection for the automotive CAN bus," in *Proc. World Congr. Ind. Control Syst. Secur.*, Dec. 2015, pp. 45–49.
[34] F. Martinelli, F. Mercaldo, V. Nardone, and A. Santone, "Car hacking identification through fuzzy logic algorithms," in *Proc. IEEE Int. Conf. Fuzzy Syst. (FUZZ-IEEE)*, Jul. 2017, pp. 1–7.
[35] R. Islam, R. U. D. Refat, S. M. Yerram, and H. Malik, "Graph-based intrusion detection system for controller area networks," *IEEE Trans. Intell. Transp. Syst.*, early access, Oct. 1, 2020, doi: 10.1109/TITS.2020.3025685.
[36] M. Weber, G. Wolf, B. Zimmer, and E. Sax, "Online detection of anomalies in vehicle signals using replicator neural networks," in *Proc. 6th ESCAR Conf.*, Ypsilanti, MI, USA, Jun. 2018, p. 14.
[37] J. Ashraf, A. D. Bakhshi, N. Moustafa, H. Khurshid, A. Javed, and A. Beheshti, "Novel deep learning-enabled LSTM autoencoder architecture for discovering anomalous events from intelligent transportation systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4507–4518, Jul. 2021.
[38] E. Seo, H. M. Song, and H. K. Kim, "GIDS: GAN based intrusion detection system for in-vehicle network," in *Proc. 16th Annu. Conf. Privacy, Secur. Trust (PST)*, Aug. 2018, pp. 1–6.
[39] H. M. Song, J. Woo, and H. K. Kim, "In-vehicle network intrusion detection using deep convolutional neural network," *Veh. Commun.*, vol. 21, Jan. 2020, Art. no. 100198.
[40] L. Yang, A. Moubayed, and A. Shami, "MTH-IDS: A multi-tiered hybrid intrusion detection system for Internet of Vehicles," *IEEE Internet Things J.*, early access, May 28, 2021, doi: 10.1109/JIOT.2021.3084796.
[41] S. Woo, H. J. Jo, I. S. Kim, and D. H. Lee, "A practical security architecture for in-vehicle CAN-FD," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 8, pp. 2248–2261, Aug. 2016.
[42] M. Çakir, T. Häckel, S. Reider, P. Meyer, F. Korf, and T. C. Schmidt, "A QoS aware approach to service-oriented communication in future automotive networks," in *Proc. IEEE Veh. Netw. Conf. (VNC)*, Dec. 2019, pp. 1–8.
[43] R. Bosch, "CAN specification version 2.0," Robert Bosch GmbH, Postfach, Germany, Tech. Rep. 300240, 1991, p. 72.
[44] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," *Black Hat USA*, vol. 2015, no. S91, pp. 1–92, 2015.
[45] S. Nie, L. Liu, and Y. Du, "Free-fall: Hacking Tesla from wireless to CAN bus," *Briefing, Black Hat USA*, vol. 25, pp. 1–16, Jul. 2017.
[46] S. Nie, L. Liu, Y. Du, and W. Zhang, "Over-the-air: How we remotely compromised the gateway, BCM, and autopilot ECUs of Tesla cars," in *Proc. Defcon*, vol. 1, 2018, pp. 1–19.
[47] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage, "Experimental security analysis of a modern automobile," in *Proc. IEEE Symp. Secur. Privacy*, May 2010, pp. 447–462.
[48] Microchip. *High-Speed CAN Transceiver*. Accessed: Oct. 1, 2021. [Online]. Available: https://ww1.microchip.com/downloads/en/DeviceDoc/20001667G.pdf
[49] P. F. de Araujo-Filho, G. Kaddoum, D. R. Campelo, A. G. Santos, D. Macêdo, and C. Zanchettin, "Intrusion detection for cyber-physical systems using generative adversarial networks in fog environment," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6247–6256, Apr. 2021.
[50] H.-J. Xing and M. Ji, "Robust one-class support vector machine with rescaled hinge loss function," *Pattern Recognit.*, vol. 84, pp. 152–164, Dec. 2018.
[51] S. Yin, X. Zhu, and C. Jing, "Fault detection based on a robust one class support vector machine," *Neurocomputing*, vol. 145, pp. 263–268, Dec. 2014.
[52] D. Xu, Y. Wang, Y. Meng, and Z. Zhang, "An improved data anomaly detection method based on isolation forest," in *Proc. 10th Int. Symp. Comput. Intell. Design (ISCID)*, vol. 2, Dec. 2017, pp. 287–291.
[53] Hacking and Countermeasure Research Lab. *Car-Hacking Dataset*. Accessed: Oct. 1, 2021. [Online]. Available: https://sites.google.com/a/hksecurity.net/ocslab/Datasets/CAN-intrusion-dataset
[54] Intrepid Control Systems. *neoVI FIRE 2*. Accessed: Oct. 1, 2021. [Online]. Available: https://intrepidcs.com/products/vehicle-network-adapters/neovi-fire-2/
[55] Microchip. *Stand-Alone CAN Controller With SPI Interface*. Accessed: Oct. 1, 2021. [Online]. Available: https://ww1.microchip.com/downloads/en/DeviceDoc/MCP2515-Family-Data-Sheet-DS20001801K.pdf
[56] AUTOSAR. *Specification of Secure Onboard Communication*. Accessed: Nov. 1, 2021. [Online]. Available: https://www.autosar.org/fileadmin/user_upload/standards/classic/4-3/AUTOSAR_SWS_SecureOnboardCommunication.pdf
[57] A. Kohn, R. Schneider, A. Vilela, A. Roger, and U. Dannebaum, "Architectural concepts for fail-operational automotive systems," SAE Tech. Paper 2016-01-0131, 2016.
[58] M. Boehner, "Security for connected vehicles throughout the entire life cycle," *ATZelectronics Worldwide*, vol. 14, nos. 1–2, pp. 16–21, Feb. 2019.
[59] AUTOSAR. *AUTOSAR Security Adaptive Platform Must Focus on Holistic Vehicle Protection*. Accessed: Nov. 1, 2021. [Online]. Available: https://www.etas.com/download-center-files/DLC_realtimes/RT_2019_2020_en_54_rgb_ESCRYPT.pdf

PAULO FREITAS DE ARAUJO-FILHO (Graduate Student Member, IEEE) received the bachelor's degree (Hons.) in electrical and electronic engineering and the M.S. degree in computer science from the Universidade Federal de Pernambuco (UFPE), Recife, Brazil, in 2016 and 2018, respectively. He is currently pursuing the Ph.D. degree in computer science with UFPE and the Ph.D. degree in electrical engineering with the École de Technologie Supérieure, Université du Québec, Montreal, QC, Canada. His current research interests include network security, intrusion detection systems, artificial intelligence, and the Internet of Things.



ANTÔNIO J. PINHEIRO received the degree in computer networks from Universidade Federal do Ceará, in 2013, and the master's and Ph.D. degrees in computer science from the Universidade Federal de Pernambuco, Brazil, in 2016 and 2020, respectively. He is currently a Software Engineer with the Centro de Estudos e Sistemas Avançados do Recife (CESAR). He has experience in computer science, with focus on computer networks, machine learning, the Internet of Things, privacy, and network traffic classification.

GEORGES KADDOUM (Senior Member, IEEE) received the bachelor's degree in electrical engineering from the École Nationale Supérieure de Techniques Avancées (ENSTA Bretagne), Brest, France, in 2005, the M.S. degree in telecommunications and signal processing (circuits, systems, and signal processing) from the Université de Bretagne Occidentale and Telecom Bretagne (ENSTB), Brest, in 2005, and the Ph.D. degree (Hons.) in signal processing and telecommunications from the National Institute of Applied Sciences (INSA), University of Toulouse, Toulouse, France, in 2009. He is currently an Associate Professor and the Tier 2 Canada Research Chair with the École de Technologie Supérieure (ÉTS), Université du Québec, Montreal, Canada. In 2014, he was awarded the ÉTS Research Chair in physical-layer security for wireless networks. Since 2010, he has been a Scientific Consultant in the field of space and wireless telecommunications for several U.S. and Canadian companies. He has published over 200 journal and conference papers and has two pending patents. His recent research interests include mobile communication systems, modulations, security, and space communications and navigation. He received the Best Papers Award at the 2014 IEEE International Conference on Wireless and Mobile Computing, Networking, Communications (WIMOB) with three coauthors and the 2017 IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC) with four coauthors. Moreover, he received the IEEE TRANSACTIONS ON COMMUNICATIONS Exemplary Reviewer Award in 2015, 2017, and 2019. In addition, he received the Research Excellence Award of the Université du Québec in 2018. In 2019, he received the Research Excellence Award from ÉTS in recognition of his outstanding research outcomes. He is also serving as an Associate Editor for IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY and IEEE COMMUNICATIONS LETTERS.

DIVANILSON R. CAMPELO (Member, IEEE) received the degree in electrical engineering from the Universidade Federal de Pernambuco (UFPE), Recife, Brazil, in 1998, and the M.Sc. and Doctoral degrees in electrical engineering from the Universidade Estadual de Campinas, Campinas, Brazil, in 2001 and 2006, respectively. He was a Visiting Assistant Professor with the Electrical Engineering Department, Stanford University, Stanford, CA, USA, from September 2008 to December 2009. He is currently an Associate Professor with the Centro de Informática, UFPE. One of his prior investigations with other four collaborators, a new ONU power management mode for TDM PONs, named the Watchful Sleep mode, is now a world standard in the ITU-T [see recommendations G.989.3 (10/15), G.987.3 (01/2014), and G.984.3 (01/2014)]. He has published more than 60 scientific papers in journals and conferences. His recent research interests include machine learning for systems, networks and security, automotive and vehicular networking, and time-sensitive networking. He received the Best Paper Award at the Second International Workshop on Green Communications, held in Conjunction with IEEE GLOBECOM 2009, with five coauthors. He has been the principal investigator (PI) or the co-PI of several industry–academia and state-funded projects. He is also a member of ACM and the Sociedade Brasileira de Computação.

FABIO L. SOARES received the degree in computer engineering and the M.Sc. degree in computer science from the Universidade Federal de Pernambuco (UFPE), Recife, Brazil, in 2012 and 2015, respectively. Since 2014, he has been working on in-vehicle networking communication protocols. For the last five years, he focused on integrating and simulating various automotive protocols for Ethernet-based architectures. He currently works with Intrepid Control Systems as the Engineering Manager and as the Automotive Ethernet Specialist for the entire European region.