GB-IDS: An Intrusion Detection System for CAN Bus Based on Graph Analysis

Yan Meng
Shanghai Jiao Tong University
Shanghai, China
yan_meng@sjtu.edu.cn

Jiachun Li
Shanghai Jiao Tong University
Shanghai, China
jiachunli@sjtu.edu.cn

Fazhong Liu
Shanghai Jiao Tong University
Shanghai, China
liufazhong@sjtu.edu.cn

Shaofeng Li
Frontier Research Center, Peng Cheng Laboratory
Shenzhen, China
lishf@pcl.ac.cn

Haotian Hu
Intel
Shanghai, China
haotian.hu@intel.com

Haojin Zhu
Shanghai Jiao Tong University
Shanghai, China
zhu-hj@sjtu.edu.cn

***Abstract*—In smart connected vehicles, the controller area network (CAN) bus provides a platform for communication and interaction among various heterogeneous electronic control units (ECUs), enabling intelligent travel services for users. However, existing researches show that the CAN bus is vulnerable to various injection attacks (e.g., denial of service (DoS) attack, fuzzing attack, impersonation attack), which not only threaten the operation of vehicles but also jeopardize user safety. Traditional intrusion detection systems (IDSs) are limited in their practicality, as they either require parsing the CAN communication protocol of the vehicle or rely on massive amounts of training data. In this paper, we propose GB-IDS, a graph-based CAN bus detection system. GB-IDS leverages a novel graph structure that characterizes the CAN ID time series, which overcomes the protocol parsing defect and achieves more accurate characterization than previous works. Meanwhile, the variational autoencoder (VAE) is exploited to train classifiers without negative samples. Our experimental results on the public dataset, called OTIDS, demonstrate that GB-IDS can achieve high detection success rates, especially in the absence of negative samples.**

***Index Terms*—in-vehicle network, intrusion detection, graph analysis, CAN bus**

I. Introduction

With the equipment of various electronic control units (ECUs), vehicles are becoming more intelligent and providing users with diversified services (e.g., autonomous driving, in-vehicle entertainments) beyond traditional travel functions [1]. From the in-vehicle perspective, various heterogeneous ECUs (e.g., accelerometers, fuel gauges, and GPS sensors) communicate with each other through the controller area network (CAN) bus [2]. A message in the CAN bus contains elements such as timestamp, CAN ID, and CAN Data, representing the arrival time of the message, the type of the message, and the effective payload of the message, respectively. The message sequence in the CAN bus supports the implementation of complex functions in vehicles and promotes the development of intelligent vehicles.

Although the CAN bus [2] is an efficient and flexible in-vehicle network, it was designed without considering security factors such as authentication and authorization mechanisms. More seriously, most CAN messages are plain text without encryption. Thus, attackers can readily inject malicious messages into CAN bus to hijack ECUs, leading to severe consequences such as brake failure and sudden flameout. To remedy these security issues existing in CAN bus, numerous intrusion detection systems (IDSs) based on diverse features have been proposed [3]–[6].

Existing intrusion detection systems (IDSs) face two significant challenges. Firstly, some solutions [6], [7] require knowledge of the details of the CAN protocol or even modification of the CAN protocol (e.g., adding an additional authentication field). However, these options are not practical for a significant portion of smart vehicles with closed-source protocols. Secondly, some IDSs (e.g., graph-based IDSs [8]–[10]) rely on advanced machine learning models to differentiate CAN messages during intrusion attacks from those in attack-free cases. However, these models require an extensive amount of carefully labeled CAN messages (i.e., intrusion and attack-free messages) to train an effective classifier. It is not always easy to collect a universal set of various attack patterns for a given arbitrary smart vehicle, making it difficult to generate an effective classifier. Therefore, there is a need to propose a novel IDS for the CAN bus that is protocol-independent, efficient, and does not rely on negative samples.

In this study, we propose GB-IDS, a graph-based intrusion detection system for detecting intrusion attacks in the CAN bus. GB-IDS is based on the key observation that the pattern of CAN messages during intrusion attacks is significantly different from that in attack-free cases. First, to achieve protocol independence, GB-IDS only uses the CAN ID and ignores the payload information. Specifically, GB-IDS converts the CAN messages during the detecting time slot into a graph, where the CAN IDs are vertices and CAN ID transitions are edges. Then, unlike existing graph-based IDSs [10] that focus only on the CAN ID transition relationships, GB-IDS preserves the time sequence information for each edge in the generated graph, which effectively improves detection efficiency. Finally,

Haojin Zhu is the corresponding author.

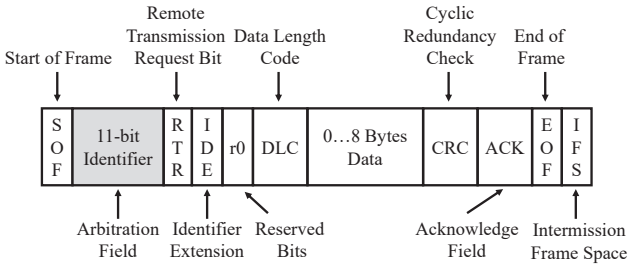
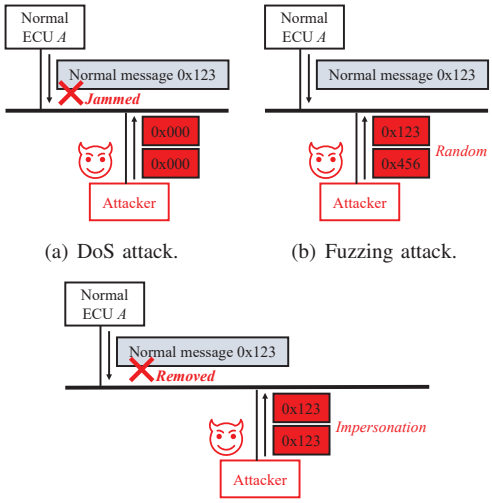


Fig. 1. The format of a CAN frame.



(a) DoS attack. (b) Fuzzing attack.

(c) Impersonation attack.

Fig. 2. Illustration of intrusion attacks in CAN bus.

to generate an effective classifier without negative samples, GB-IDS converts the graphs collected in the attack-free cases into features based on the PageRank algorithm and trains the classifier using a variational auto-encoder (VAE). Since the VAE can effectively characterize the abnormal level of a given graph compared to graphs used for training, GB-IDS can effectively identify intrusion behaviors based on the reconstruction loss calculated by VAE.

We evaluated the performance of GB-IDS on the publicly available OTIDS dataset, which comprises 236.9 million attack-free CAN messages and 224.4 million intrusion attack CAN messages. The detection accuracy for denial-of-service (DoS), fuzzing, and impersonation attacks were found to be 99.17%, 99.72%, and 99.69%, respectively, with a detection time of only 0.08 seconds. In summary, the main contributions of this paper are:

- We propose GB-IDS, a graph-based IDS for the in-vehicle CAN bus that can detect intrusion attacks without prior knowledge of CAN protocols and without relying on negative training samples.
- To improve detection performance, we introduce a novel graph structure that preserves both CAN ID transition and time interval information. Additionally, we propose a VAE-based classifier to overcome the limitations of requiring negative samples.
- We evaluate GB-IDS on a third-party public dataset and demonstrate its effectiveness in detecting DoS, fuzzing, and impersonation attacks. Furthermore, GB-IDS exhibits acceptable time costs for detection.

We have organized the remainder of this paper as follows: In Section II, we provide a brief overview of related works on IDSs for CAN bus. In Section III, we present the details of GB-IDS's system design. The implementation and evaluation of GB-IDS are described in Section IV, followed by a discussion of our findings in Section V. Finally, we conclude this paper in Section VI.

II. Preliminaries and Related Work

In this section, we provide an introduction to the concept of the CAN bus, followed by a review of the intrusion attacks targeting the CAN bus. We then provide a survey of the existing IDSs designed to detect these attacks.

A. Controller Area Network

To ensure efficient communication between ECUs in vehicles, the CAN was proposed as a multifunctional bus standard that is superior to other standards. Unlike other protocols that use station addresses to deliver messages, CAN bus uses unique message identifiers to exclusively identify messages and stipulate their priorities. This makes the CAN bus flexible and unaffected by the addition or removal of ECU nodes in the system.

In this section, we introduce the message formats in the CAN bus. The basic CAN bus supports two different message formats or frame formats: the standard frame format (described in CAN2.0 and CAN2.0B) and the extended frame format (described only in CAN 2.0B). The only difference between them is their lengths. Figure 1 illustrates the format of the standard CAN message (frame). In our study, we focus on the CAN ID, which represents the priority of the message. The smaller the CAN ID value is, the higher the priority of the message. For instance, the CAN ID value in the braking-related CAN message is smaller than that in an entertainment-related CAN message.

However, in modern vehicle manufacturing, manufacturers usually modify CAN protocols for specific usage, making CAN bus protocols diverse and difficult to reverse engineer. Therefore, messages sent over the CAN bus are not standard. Due to these complex protocols, it is time-consuming and costly to reverse engineer the protocols to identify each message in real-world situations. This motivated our work, which aims to provide a universal mechanism to detect intrusion behaviors in the CAN bus without prior knowledge of the details of CAN protocols.

B. Intrusion Attacks for In-vehicle CAN Bus

In this subsection, we will discuss different types of attacks that can be performed on the CAN bus, including denial

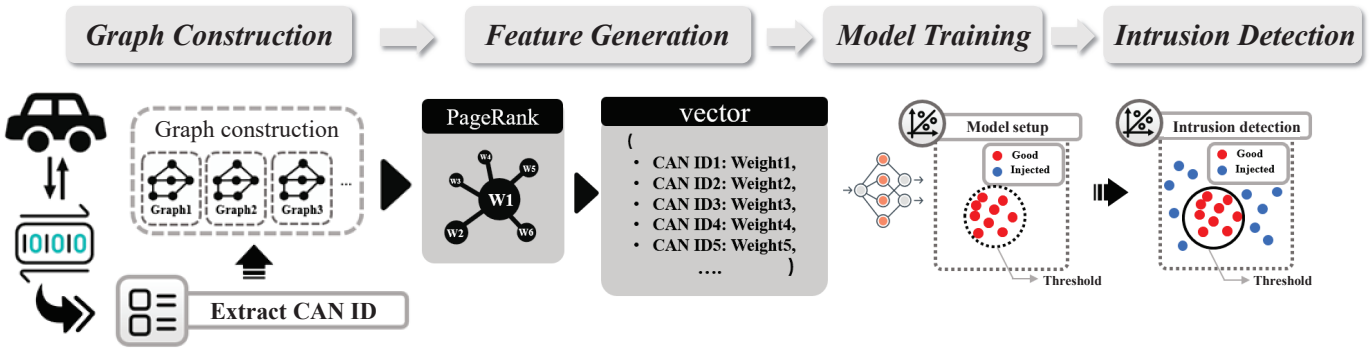


Fig. 3. Workflow of GB-IDS.

of service (DoS) attacks, fuzzing attacks, and impersonation attacks.

- **DoS attack.** As shown in Fig. 2(a), the attacker injects meaningless message frames into the CAN bus, interrupting the communication of the ECUs. As a result, legitimate messages are blocked until the resources can be recovered.
- **Fuzzing attack.** As shown in Fig. 2(b), fuzzing attacks involve the injection of invalid, unexpected, or random message frames into the CAN bus, similar to a fuzzing test. These attacks can cause accidents, damage to the CAN bus, and even pose a threat to the security of the vehicle.
- **Impersonation attack.** As shown in Fig. 2(c), the attacker impersonates a specific ECU by sending corresponding signals to achieve their goals. They first disconnect a particular ECU and then replace it by sending a message frame with the corresponding CAN ID. For instance, the attacker may disconnect the ECU component that sends the CAN ID 0x123 and replace it by sending a message frame with the same ID. These attacks are purposeful and destructive, aimed at attacking the system from a specific ECU.

C. Existing Intrusion Detection Systems

Existing research has investigated CAN IDS and can be classified into two categories: detection based on raw data and detection based on graphs.

For detection based on raw data, Choi et al. proposed a voltage-based intrusion detection system [11]. Their system uses the unique electrical characteristics of a CAN signal as a fingerprint for electronic control units. However, the system has three drawbacks. Firstly, it cannot ensure the scalability of detection. Secondly, it is challenging to establish specific patterns for each attack. Finally, the side-channel mechanism requires prior knowledge of specific attacks and can be easily disturbed by environmental factors such as magnetic and electric fields. Hossain et al. proposed an LSTM-based anomaly detection mechanism on CAN messages [7]. However, these methods require negative samples for model training, which can be difficult to obtain in real-world detection scenarios.

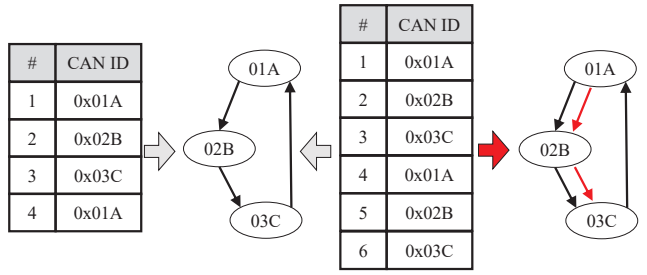
Apart from raw data-based mechanisms, researchers have also employed graph-based mechanisms. Graph-based mechanisms capture relationships between CAN messages and provide more information. Firstly, Islam et al. proposed a graph-based mechanism that uses chi-square testing to measure the difference between message graphs [8]. Jedh et al. proposed a CAN bus IDS based on the similarities of successive message-sequence graphs [9]. Furthermore, Islam et al. [10] proposed GGNB to verify the conclusion that entropy influences the strength of the anomaly detection system. However, these methods also require negative samples for effective classifier training, which can be challenging to obtain in practice.

III. System Design

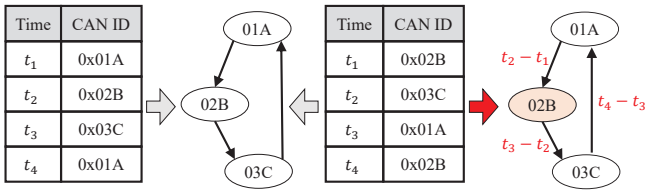
In this section, we propose a graph-based in-vehicle CAN bus IDS named GB-IDS to detect intrusion attacks. As illustrated in Fig. 3, GB-IDS consists of four modules: the *Graph Construction Module*, the *Feature Generation Module*, the *Model Training Module*, and the *Intrusion Detection Module*. In the *Graph Construction Module*, GB-IDS obtains initial data from the CAN bus in vehicles and generates directed and weighted graphs with extracted CAN IDs. In the *Feature Generation Module*, GB-IDS assigns a specific value to each vertex existing in the graphs to denote its priority and calculate the feature vector. In the *Model Training Module*, GB-IDS utilizes the VAE to train a classifier based on the features extracted from graphs generated from attack-free CAN messages. Finally, for a given time slot, the *Intrusion Detection Module* uses the trained classifier to determine whether an intrusion attack is launched, alerts the users, and marks the timestamp when the intrusion starts. We elaborate on the details of each module as follows.

A. Graph Construction Module

In this subsection, we describe how we convert the CAN messages into graphs that can be used to identify intrusion behaviors in a protocol-unrelated manner. As the protocols used in communication networks of different vehicles can vary, even between different types of the same manufacturer, we select two common fields in messages, namely, timestamps and CAN ID, as the inputs for graph construction. To achieve



(a) From simple graph to complex graph.



(b) Considering time sequence information.

Fig. 4. Design of our proposed graph structure.

this, we collect CAN IDs in the order of timestamps over a fixed period of time as a time window or a fixed number of messages as a message window δT. We take adjacent CAN IDs as directed edges and construct a graph. For instance, in the left part of Fig. 4(a), we can convert the messages in the window as $01A \rightarrow 02B$, $02B \rightarrow 03C$, and $03C \rightarrow 01A$, resulting in 3 directed edges. By default, we use 100 messages as a fixed message window δT and generate a graph G_t as below:

$$G_t = \bigcup_{t_i}(ID_{t_i} \rightarrow ID_{t_{i+1}}), \\ t_i \in (t_0 + k\delta T, t_0 + (k+1)\delta T), k \in \mathbb{N}. \tag{1}$$

However, as shown in Fig. 4(a), directly using the simple graphs utilized by existing works [8] cannot achieve optimal performance. The reason is that simple graphs treat duplicated edges as one. For example, the two message sequences containing 4 and 6 messages illustrated in Fig. 4(a) will be converted to the same graph. Thus, we define multiple edges between two vertices (i.e., CAN IDs) as long as there are multiple transitions.

Besides, we also define a weight for each edge. As shown in Fig. 4(b), simple graphs cannot distinguish graphs with a circular structure. Sequences $01A \rightarrow 02B \rightarrow 03C \rightarrow 01A$ and $02B \rightarrow 03C \rightarrow 01A \rightarrow 02B$ are converted to the same graph. Thus, for CAN ID ID_i and ID_j with timestamps TS_i and TS_j, we define the weight of edge $ID_i \rightarrow ID_j$ as $TS_j - TS_i$. We also define an edge for the start CAN ID with a weight of 0 (e.g., $02B$ in Fig. 4(b)). Thus, the constructed graph will contain both transition details and time sequence information.

Therefore, the final generated graph $G'_t = (V, E)$ contains N_V vertices $\{v_1, v_2, \cdots, v_{N_V}\}$ and N_E edges $\{E_1, E_2, \cdots, E_{N_E}\}$. The i-th edge E_i can be written as:

$$E_i = (v_{e(i)} \rightarrow v_{e(i+1)}, TS_{i+1} - TS_i), \tag{2}$$

where $v_{e(i)}$ and $v_{e(i+1)}$ are two CAN IDs with timestamps TS_i and TS_{i+1}.

B. Feature Generation Module

For the sequence of directed graphs G'_t formed by CAN messages, we need a method to measure the importance of vertices so that they can be transformed into mathematical vectors. We propose a feature generation method based on the PageRank algorithm [12] to calculate the priority of each vertex based on its edges.

To measure vertex priorities, we first review the basic idea of the PageRank algorithm and then introduce our optimization.

1) PageRank Algorithm: The PageRank algorithm was first proposed by Google for ranking websites in search engine results. It originally used the weight of websites and out-degrees to analyze the significance of each web page. In other words, a page that is referred to more often counts more. In general, it assigns a specific weight to every vertex in a directed graph to measure its importance.

Assume there are three nodes $V_1 = 01A$, $V_2 = 02B$, and $V_3 = 03C$ as shown in Fig. 4(a). To ensure that probabilities range from 0 to 1, we hypothesize initial values for every node as $\frac{1}{3}$, which means an even division. We use $L(V_i)$ to denote the sum of outgoing edges of vertex V_i and let $Pri(V_i)$ denote the PageRank value of vertex V_i. In every iteration, we recalculate the weights $Pri(V_i)$ by adding those that point to V_i. For example, in the next iteration, we compute the weight of page V_1 as:

$$Pri(V_1) = \frac{Pri(V_2)}{L(V_2)} + \frac{Pri(V_3)}{L(V_3)}. \tag{3}$$

In case there is an isolated vertex whose in-degree is 0, a minimum value is set so that every web page can be visited with a minimal probability. We use q to denote a damping factor and N to denote the sum of vertices. The general formula can be concluded as:

$$Pri(V_1) = \frac{1-q}{N_V} + q \cdot (\frac{Pri(V_2)}{L(V_2)} + \frac{Pri(V_3)}{L(V_3)}), \tag{4}$$

where N_V represents the total number of vertices, which is 3 in Fig. 4(a).

2) Optimization of PageRank: The PageRank algorithm mentioned above does not take into account multiple edges between two vertices and the weights of edges. To address this, we use the weights of edges related to vertex p_i when calculating its priority. Thus, the equation 4 for calculating the priority of a vertex p_i, denoted as $Pri(p_i)$, can be optimized as:

$$Pri(p_i) = \frac{1-q}{N_V} + q \sum_{p_j} \frac{Pri(p_j)}{C(p_j)}, \tag{5}$$

where $C(p_j)$ represents the sum of edge weights between vertex p_j and p_i, and N_V represents the total number of vertices. q is the damping coefficient set for isolated vertices, which is usually set to 0.85.

C. Model Training Module

The Model Training Module uses the vectors generated by the *Feature Generation Module* during attack-free scenarios as inputs in a VAE model. The VAE (Variational Autoencoder) is a generative model used for generating samples similar to the inputs based on the distribution of input properties, consisting of an encoder and a decoder. The VAE model can be implemented in two stages: the training stage and the generating stage.

1) Training Stage: During the training stage, the VAE learns to encode input samples into a latent space, which is a lower-dimensional space that represents the distribution of input features. The VAE model then fetches the features from the distribution in the latent space and decodes them to generate outputs.

2) Generating Stage: In the generating stage, any input defined as x is encoded by the encoder and sent into the latent space as $x_{encoded}$. After this procedure, the VAE model reconstructs the encoded input $x_{encoded}$ into another sample x_{rc} based on the distribution of positive features in the latent space.

For each input x and the reconstructed sample x_{rc}, we can calculate the reconstruction loss $loss_{rc}$ as follows:

$$loss_{rc} = -(\bar{x}\log(1e^{-10} + \bar{x}_{rc}) + \log(1 - \bar{x}_{rc})), \quad (6)$$

where $\bar{x}$ and $\bar{x}_{rc}$ represent the mean value of input vectors and output vectors.

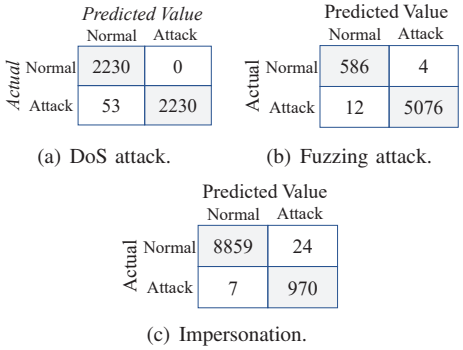
If the input is positive, the corresponding output x_{rc} will be similar to x, and the reconstruction loss $loss_{rc}$ will be below the pre-defined threshold. However, if the input x is negative, x_{rc}'s reconstruction depends on the knowledge of positive features, and the corresponding output x_{rc} will differ from x. In this case, the reconstruction loss $loss_{rc}$ will be above the pre-defined threshold. By analyzing the reconstruction loss, we can detect intrusion behavior and realize the function of intrusion behavior detection.

D. Intrusion Detection

As described in Section III-C, the GB-IDS measures the difference between inputs and outputs and determines the range of a positive input. We assume the upper bound is Thr_U and the lower bound is Thr_L, so the raw threshold is $Thr_{raw} = (Thr_L, Thr_U)$. In a real-world situation, we suppose that the recall rate of GB-IDS should be high, thus we introduce an *error coefficient* θ and adjust the threshold as:

$$Thr = ((1+\theta)\cdot Thr_L, (1-\theta)\cdot Thr_U). \quad (7)$$

The threshold Thr is generated by the vehicle in a secure environment, such as after production, and represents the traffic pattern in the CAN bus. If the *reconstruction error* of some input is not in the range of Thr, the GB-IDS will report an intrusion event.



(a) DoS attack. (b) Fuzzing attack.

(c) Impersonation.

Fig. 5. Confusion matrixs when detecting intrusion attacks.

IV. Evaluation

In this section, we present the evaluation of our system on an existing public dataset and measure its ability to detect DoS attacks, fuzzing attacks, and impersonation attacks. We introduce the dataset and evaluation results as follows.

A. Dataset

We evaluated our system on the public OTIDS dataset [4], which includes 656,579, 591,990, 995,472, and 2,369,868 CAN messages collected in DoS attack, fuzzing attack, impersonation attack, and attack-free states, respectively. The messages were generated by logging CAN traffic via the OBD-II port from a real vehicle while message injection attacks were being performed.

B. Performance Analysis

In this section, we evaluate the overall performance of GB-IDS under intrusion attack scenarios, and present the detection performance in Fig. 5.

1) Performance on Detecting DoS Attacks: We first measure the ability of GB-IDS to detect DoS attacks in the OTIDS dataset. As shown in Fig. 5(a), the detection accuracy is 99.71%. Furthermore, if we consider the attack-free cases as negative cases and DoS attacks as positive cases, the precision, recall, and F1-score are 100%, 98.71%, and 99.35%, respectively.

2) Performance on Detecting Fuzzing Attacks: We present the confusion matrix for detecting fuzzing attacks in Fig. 5(b), and report a detection accuracy of 99.72%. Moreover, if we consider fuzzing attacks as positive cases, the precision, recall, and F1-score are 99.92%, 99.76%, and 99.84%, respectively.

3) Performance on Detecting Impersonation Attacks: The confusion matrix for detecting impersonation attacks is illustrated in Fig. 5(c), and the detection accuracy is reported to be 99.69%. If we consider impersonation attacks as positive cases, the precision, recall, and F1-score are 97.59%, 99.28%, and 98.43%, respectively.

4) Time Overhead: Efficiency is crucial for an IDS, especially in moving vehicles. To evaluate GB-IDS's efficiency, we measured its average time cost on the OTIDS dataset. Since the SAE J1939 protocol specifies a communication rate of 250 kbps, we evaluated our system's efficiency based on

this rate. According to ISO 11898, the minimum length of a CAN frame is 46 bits and the maximum length is 110 bits. We assumed an average length of 78 bits for normal CAN bus traffic, which means there are approximately 3,000 CAN messages transmitted per second. We found that our average detection time is about 0.085 seconds with the time window set to 100 messages. This means that we can detect an anomaly in the CAN bus in less than 260 CAN messages after it occurs.

V. Discussion

While the proposed GB-IDS demonstrates good performance and high practicability, it still has certain limitations. In this section, we discuss these limitations and suggest some future research directions.

A. Limitations

1) Traceability: One major limitation of GB-IDS is that it is unable to detect which message is injected. GB-IDS preprocesses several CAN messages in a fixed time window and translates them as a whole to represent the status of the CAN bus. Since the detection consequences of GB-IDS are based on multiple messages, it is difficult to identify which message in the time window is malicious and injected by attackers. GB-IDS can only identify the part or time period of the CAN bus that is attacked, but cannot trace back to the specific message. Thus, GB-IDS needs to be enhanced to identify malicious messages with greater precision.

2) Risk of Model Tampering: Another limitation of GB-IDS is that it runs in an external and untrusted environment, making it vulnerable to attacks. Attackers can gain access to the model and tamper with the classifier, potentially damaging GB-IDS and disrupting the detection process. We assume an environment in which attackers can easily read and modify the model, thus making our system less secure than we propose. Attacks can be implemented at both software and hardware levels, posing a significant risk to GB-IDS.

B. Future Work

1) Trusted Execution Environment: As mentioned earlier, working in an untrusted environment poses a significant security threat to our model. To address this issue, we plan to deploy our model in a Trusted Execution Environment (TEE), such as Intel SGX, to ensure runtime security and prevent attackers from tampering with or stealing our model.

2) Time Window Auto-Adjustment: In our methodology, we set the time window for collecting CAN messages and generating graphs to 100 CAN messages as a default value, and compared the results obtained with different time windows. However, we believe that in some situations, there may be a better time window to use. Therefore, we plan to investigate how to auto-adjust the value of the time window to obtain the best results in every condition.

3) Impact of User Behaviors on CAN Message Patterns: In our methodology, we claim that the distribution of CAN messages is different in attack-free and intrusion conditions. However, different users may have an impact on the patterns of CAN messages. Different driving habits of users may result in different sequences of CAN messages. We plan to investigate users' driving habits and generate fingerprinting by CAN messages for each user. This approach will help us in user authentication and ensure users' privacy.

VI. Conclusion

In this paper, we present GB-IDS, an IDS designed for detecting intrusion attacks such as DoS, fuzzing, and impersonation attacks on in-vehicle CAN buses in real time. GB-IDS extracts CAN messages from vehicles via the OBD-II port and generates graphs based on the relationships between CAN IDs. It uses an optimized PageRank algorithm to assign weights to each CAN ID as a vertex and collects the weights on vertexes as a vector. GB-IDS adopts a VAE model to train with only attack-free data and determines the threshold of the attack-free samples' reconstruction loss. During the detection session, GB-IDS compares the reconstruction loss of the input data with the threshold and alerts when the loss is out of range. GB-IDS achieves an accuracy of 99.71%, 99.72%, and 99.69% under DoS, fuzzing, and impersonation attacks, respectively, in the public dataset OTIDS.

Acknowledgment

This research was supported by National Natural Science Foundation of China under Grants No. 62132013.

References

[1] C. Isidore, "Here's why car prices are so high, and why that matters," https://edition.cnn.com/2021/07/08/business/car-prices-inflation/index.html.

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

[3] G. Dupont, A. Lekidis, J. Den Hartog, and S. Etalle, "Automotive controller area network (can) bus intrusion dataset v2," 2019.

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

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

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

[7] M. D. Hossain, H. Inoue, H. Ochiai, D. Fall, and Y. Kadobayashi, "Lstm-based intrusion detection system for in-vehicle can bus communications," *IEEE Access*, vol. 8, pp. 185 489–185 502, 2020.

[8] R. Islam, R. U. D. Refat, S. M. Yerram, and H. Malik, "Graph-based intrusion detection system for controller area networks," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[9] M. Jedh, L. B. Othmane, N. Ahmed, and B. Bhargava, "Detection of message injection attacks onto the can bus using similarities of successive messages-sequence graphs," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 4133–4146, 2021.

[10] R. Islam, M. K. Devnath, M. D. Samad, and S. M. J. Al Kadry, "Ggnb: Graph-based gaussian naive bayes intrusion detection system for can bus," *Vehicular Communications*, vol. 33, p. 100442, 2022.

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

[12] W. Xing and A. Ghorbani, "Weighted pagerank algorithm," in *Proceedings. Second Annual Conference on Communication Networks and Services Research, 2004.* IEEE, 2004, pp. 305–314.